Scientific Research Publishing

Open Journal of Social Sciences, 2025, 13(12), 336-348
https://www.scirp.org/journal/jss
ISSN Online: 2327-5960
ISSN Print: 2327-5952

Exploring Interdisciplinary Project-Based Teaching Reform of "AI Meets HarmonyOS" in Higher-Vocational IoT Programs: A Case Study on Smart-Home Development

Jianqing Tang

School of Internet of Things, Guangdong Open University (Guangdong Polytechnic Institute), Guangzhou, China
Email: 514300091@qq.com

How to cite this paper: Tang, J. Q. (2025). Exploring Interdisciplinary Project-Based Teaching Reform of "AI Meets HarmonyOS" in Higher-Vocational IoT Programs: A Case Study on Smart-Home Development. *Open Journal of Social Sciences, 13,* 336-348.
https://doi.org/10.4236/jss.2025.1312026

Received: December 13, 2025
Accepted: December 23, 2025
Published: December 26, 2025

Abstract

The mass deployment of HarmonyOS and the convergence of generative AI have created a twofold dilemma for higher-vocational IoT education: curricula lag behind rapid technological change, and abstract concepts remain disconnected from practice. Grounded in a competence-based approach, we redesigned a course around a smart-home application that deeply integrates GitHub Copilot and Tongyi Lingma into the complete workflow of project decomposition, code generation, debugging, and integration testing. The resulting three-dimensional framework—project-driven, AI-empowered, and hierarchically scaffolded—enables students to advance through basic functions, core technical challenges, and innovative extensions within 72 class hours, using a light-control subsystem as the illustrative thread. Empirical results demonstrate significant gains in students' ability to solve complex engineering problems and to collaborate effectively with AI tools. The study offers a replicable model for interdisciplinary, project-based teaching in higher vocational colleges navigating the new-technology wave.

Keywords

Artificial Intelligence, HarmonyOS, Project-Based Teaching, Internet of Things, Smart Home, Higher Vocational Education

1. Introduction

Against the backdrop of the deep integration between the digital economy and the real economy, the Internet of Things (IoT), as a core component of the new generation of information technology, is accelerating industrial transformation and

upgrading. Leveraging its distributed architecture and open-source ecosystem, the HarmonyOS operating system has become a fundamental technological platform for IoT terminal development. Meanwhile, generative AI technology is profoundly reshaping software development paradigms through code generation and intelligent assistance (Hu, 2025; Huang & Wang, 2025), imposing new demands on technical talent for their ability to synergize "intelligent tools with professional expertise." (Kong et al., 2025; Long et al., 2025)

As primary providers of technical and skilled talent, higher vocational colleges have long faced contradictions in their IoT major teaching, such as "rapid technological iteration versus outdated teaching content" and "students' weak theoretical foundations versus high demands of complex projects." Most courses still focus on traditional programming languages and single hardware modules, lacking systematic coverage of new systems like HarmonyOS and failing to effectively integrate AI tools, resulting in a disconnect between talent cultivation and industry needs. Therefore, exploring a teaching model tailored to the cognitive characteristics of vocational college students and integrating "AI + HarmonyOS" technologies has become an urgent requirement for professional teaching reform. Smart home, as a typical IoT scenario with modular functionality and specific demands, provides an ideal vehicle for integrating cross-disciplinary knowledge.

2. "AI + HarmonyOS" Interdisciplinary Project-Based Teaching Plan Design

2.1. Design Philosophy and Objectives

Designed to address the characteristics of vocational college students, who typically exhibit "weak theoretical foundations, strong practical motivation, and a preference for concrete thinking." Guided by the philosophy of "industry demand-driven, project-based learning at its core, AI tools as enablers, and skill progression as the goal" (Li et al., 2025; Zhao & Zhang, 2025), it transforms abstract technologies into actionable tasks through real-world projects. By leveraging AI tools to lower technical barriers and implementing tiered task design, the program facilitates a step-by-step progression from "foundational skills → integrated application → innovative expansion." The teaching objectives include: mastering knowledge of HarmonyOS distributed architecture, UI development, and device interconnectivity (knowledge objectives); developing the practical ability to utilize AI tools to assist in HarmonyOS application development, debugging, and hardware interconnection (skill objectives); and fostering rigorous engineering ethics, an innovative mindset enabled by tools, and professional adaptability to technological changes (literacy objectives).

2.2. Maintaining the Integrity of the Specifications

With the "Multi-terminal Collaborative Smart Home Control System" as the core project, it covers four functional modules: terminal control, status monitoring, scenario linkage, and AI-assisted (voice control) functions. The teaching content

is divided into three levels based on 72 class hours:

1) Basic Level (18 class hours): Focuses on the setup of HarmonyOS development environment and basic UI development. AI tools are utilized to generate page prototypes and interpret API documents.

2) Advanced Level (40 class hours): Delves into distributed capability development, device interconnection protocols, and data collection and processing. AI is employed to generate code frameworks, assist in debugging, and optimize logic.

3) Innovation Level (14 class hours): Completes complex functions such as scenario linkage and voice control integration. AI tools provide support for scheme verification, code review, and performance optimization.

3. Integrating AI Tools into the Full Teaching Process

3.1. Requirement Translation & Framework Generation (8 Class Hours)

Students are guided to describe functional requirements in natural language (e.g., the interaction logic of a lighting-control page). Using Tongyi Lingma's "natural-language-to-code" capability, an initial page-layout and module-division framework is automatically produced. The instructor focuses on standardising the requirement descriptions to ensure the accuracy of the AI-generated code. For example, after a student's description, the AI can output a well-structured ArkTS code skeleton that includes state management, layout, and interaction modules, laying a solid foundation for subsequent development.

3.2. Code Generation and Problem-Solving (56 Class Hours)

1) Basic-level Empowerment:

For tasks such as environment configuration and basic UI development, AI is used to generate operational guides and sample code, helping students get started quickly. For example, when students input layout requirements, AI generates the corresponding code. The teacher then guides students in analyzing the structure to understand component properties and logic.

2) Intermediate-level Empowerment:

The "Autonomous Thinking + AI Completion" model is adopted. Students first independently draft the core logic code for tasks such as distributed communication or hardware data collection. They then utilize GitHub Copilot's code completion feature to refine the details and compare the differences to deepen their understanding. During debugging, students are guided to input error information into AI tools to obtain solutions, thereby cultivating their problem-solving skills.

3) Innovation-level Empowerment:

Students are encouraged to autonomously design solutions for scene linkage and voice control integration. They then use AI tools to generate core logic code for feasibility verification and optimization. For example, after designing the linkage rules for a "Wake-Up Mode," AI generates the specific trigger condition and device action class code.

3.3. Code Review and Performance Optimization (8 Class Hours)

1) Code Review

Leverage the "code review" functionality of AI tools to detect normative issues and potential risks, generating review reports. This is combined with manual peer review to identify scenario-specific logical problems (such as device offline handling) that AI may overlook.

2) Performance Optimization

Utilize AI to analyze application runtime data (e.g., rendering time, communication latency), obtain optimization suggestions (such as reducing component re-rendering, reusing hardware connections), and guide students in code refactoring to enhance operational efficiency.

4. Teaching Case: AI-Assisted Development of a Smart-Home Lighting Control System

This section selects the core sub-project "Smart-Home Lighting Control System" as the teaching case. It presents the complete implementation of the "AI + HarmonyOS" project-based approach, covering requirement decomposition, technical realization, AI-tool usage, and instructional organization. The case illustrates how stepped task design advances higher-vocational students' competences in a concrete scenario.

4.1. Case Background and Teaching Objectives

4.1.1. Case Background

This case is selected from the core module of the "progressive level" in the teaching plan and is implemented after students have completed the foundational level of learning (mastering the basics of HarmonyOS UI development). Lighting control is one of the most intuitive and moderately complex functional modules in smart home systems, involving multiple core HarmonyOS technical points such as UI interaction, state management, distributed communication, and hardware control. It serves as an ideal transitional task for students moving from basic development to comprehensive application.

4.1.2. Learning Objectives

1) Knowledge:

- Master the APIs of HarmonyOS Distributed Data Manager.
- Understand how state variables (@State, @Prop) enable cross-page data synchronization.
- Grasp the mapping logic between UI components (Toggle, Slider) and hardware-control commands.

2) Skills:

- Use AI tools (GitHub Copilot) to interpret distributed-API documentation and generate core communication code.
- Debug typical distributed-sync problems (device offline, data conflicts).

- Independently develop a complete module supporting multi-terminal control.

3) Competencies:

- Cultivate rigorous logical thinking for hardware communication.
- Establish a collaborative habit of "AI-generated framework + human logic optimization".
- Strengthen communication and stress-resistance when solving technical problems in team work.

4.2. Case Implementation Process (14 Class Hours)

To provide a clearer understanding of the case implementation process, the project has devised a closed-loop teaching process featuring "two main threads and four phases", with the specific process illustrated in Figure 1.

As shown in Figure 1, the diagram clearly illustrates:

1) Dual-track drivers:

Three distinct yet intertwined interaction threads—student activities (hands-on work and reflection), AI-tool application (generation and suggestions), and teacher guidance (explanation and elevation)—embodying the modern instructional model of "students as the subject, AI as the tool, and teacher as the guide."

2) Four-stage progression:

The figure strictly aligns with the four phases described. Each phase forms a micro-cycle of "student initiation → AI response → teacher intervention," ensuring continuous support and scaffolding.

3) Outcomes-and-reflection closed loop:

The final stage evaluates not only knowledge and skills (efficiency, technical mastery, AI literacy) but also delves into pedagogical reflection—role transformation, tiered design, and potential pitfalls such as AI overreliance—thereby creating a complete loop for instructional improvement.

4.2.1. Requirements Analysis & AI Framework Generation (3 Class Hours)

Develop a mobile-controlled lighting page that synchronizes switch status, brightness value, and last-operation time in real time; the expected interface is shown in Figure 2.

1) Natural-Language Requirement

In small groups, students describe their needs to AI in structured language:

Create a HarmonyOS ArkTS page that:

- provides a circular switch button to control the lamp;
- shows a horizontal brightness slider (0 - 100) enabled only when the lamp is on;
- displays the brightness percentage in real time;
- shows the device name and its status (on/off);
- shows the last-operation time;
- synchronises all state changes instantly on the phone.

2) AI-Generated Code Frame

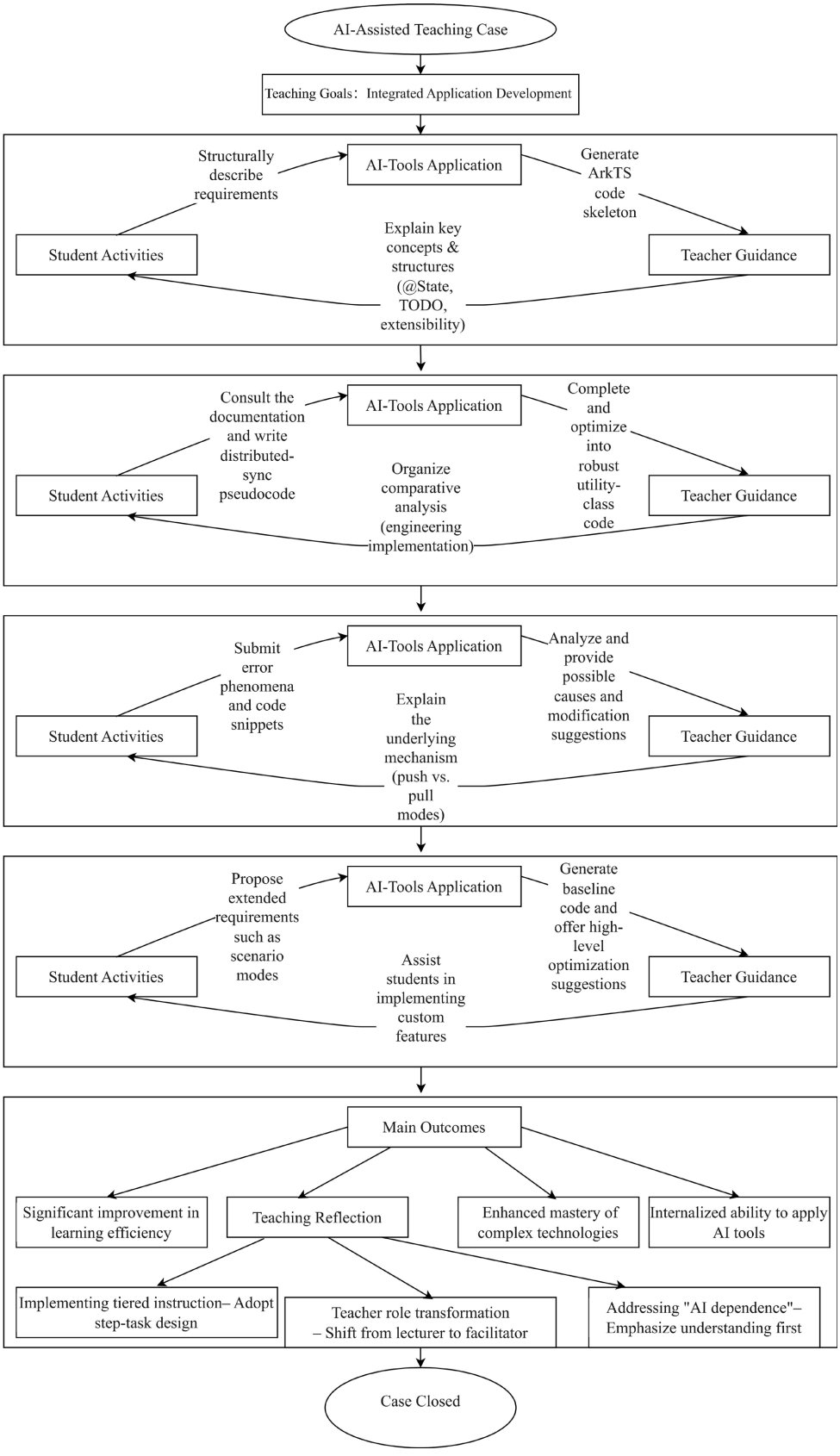

Figure 1. "Two main threads and four phases" teaching process.

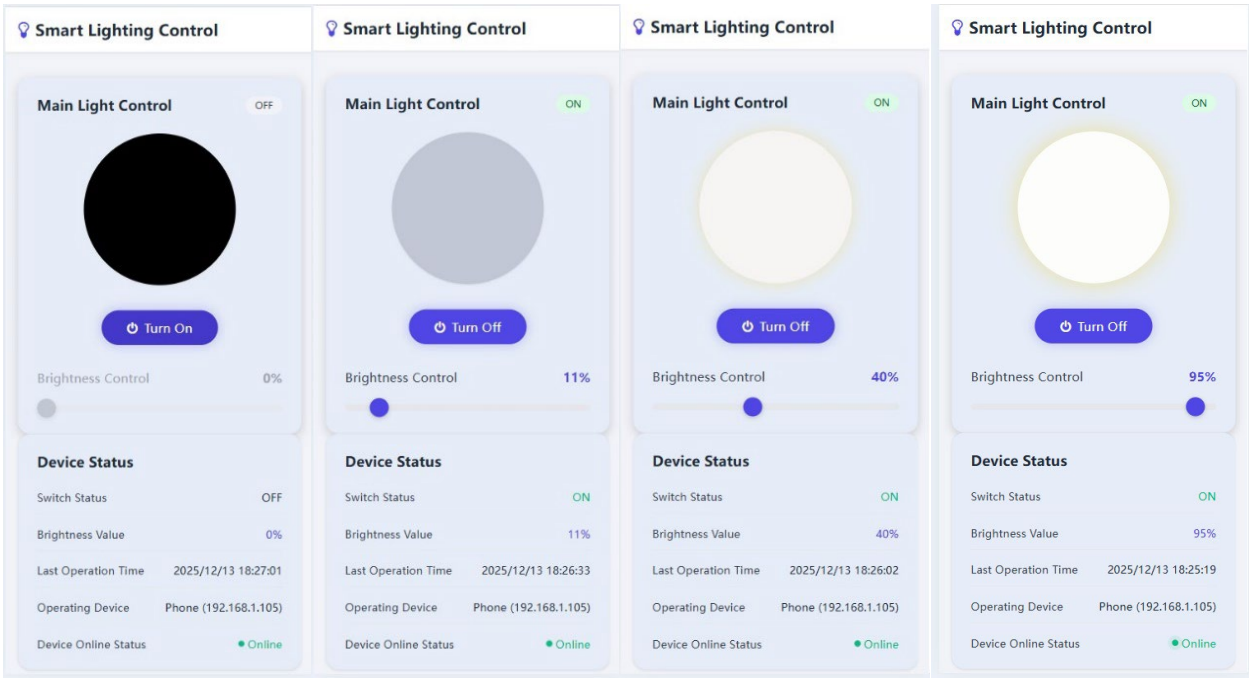

Figure 2. Functional Interface Mock-up.

AI returns an ArkTS page skeleton with state variables, basic UI layout, and function stubs, clearly organised within a LightControlPage component. The project structure diagram is shown in Figure 3.

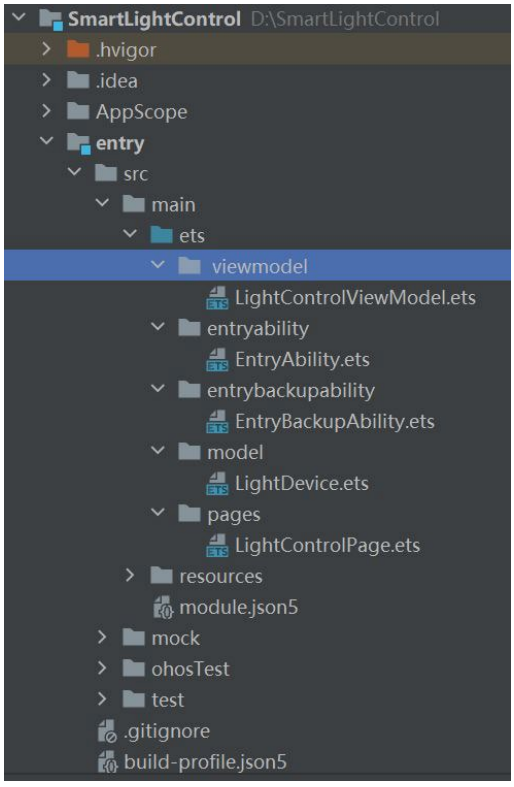

Figure 3. The project structure diagram.

3) Teacher Guidance

The instructor guides students to analyze the logic of the AI-generated smart lighting control code, focusing on two core points for in-depth explanation:

By examining real-time response scenarios such as light switch status and brightness value adjustments, analyze the core role of the @State decorator in state management within components (e.g., the animation linkage of scaleValue during switch clicks and the immediate update of the brightness slider value). Compare the interaction flaws in the absence of state management and clarify its necessity in achieving two-way binding between UI and data.

Based on the existing code architecture, analyze the feasibility of functional expansion (such as adding light color control or color temperature adjustment modules). Break down the decoupled design of the LightDevice model, the LightControlViewModel business layer, and the LightControlPage view layer. Identify structural areas in the current code that require optimization for future extensions (e.g., abstracting brightness control logic into a generic device parameter control method and reserving extension interfaces for color attributes).

4.2.2. Technical Breakthroughs & AI-Assisted Coding (6 Class Hours)

Traditional instruction struggles to reconcile complex business logic with rapid-development demands in HarmonyOS courses. By embedding AI pair-programming, we boost students' ability to tackle technical bottlenecks and shorten iteration cycles. Guided by AI, learners practice code review, state management, component encapsulation, business-logic decoupling, and feature extension—acquiring the full-stack skills demanded by industry and graduating as versatile, market-ready developers.

1) Instructional Content Design

The instructional content design is shown in Table 1.

Table 1. Technical content of the breakthroughs.

Session	Topic	Core Content
1	Structured Review of AI-Generated Code	Review methods, initial screening dimensions, practical review, summary of advantages and limitations
2	State Management Breakthrough — In-depth Application of @State Decorator	Nature of decorators, management pain points, AI-assisted practice, best practice expansion
3	Component Encapsulation and Reuse Breakthrough	Component problem analysis, AI-assisted refactoring, practical reuse testing
4	Business Logic Decoupling Breakthrough	Logic coupling issues, AI-assisted decoupling and refactoring, data flow controllability
5	Function Expansion Breakthrough — Color Control Module Development	Requirement analysis, AI-assisted expansion development, debugging and optimization
6	Full-Process Review and Optimization of AI-Assisted Coding	Full-process review, AI coding optimization strategies, final acceptance

2) Instructional Implementation

Following the content design in Table 1, the teaching process is delivered as follows.

1) Structured Review of AI-Generated Code

- Review method: Teach students to systematically inspect AI-generated code to identify defects.
- Screening dimensions: syntax compliance (e.g., misuse of @State), architecture rationality (clear MVVM layers, no UI-logic coupling), and functional completeness (coverage of core interactions).
- Hands-on review: In groups, students mark issues such as invalid assignments in BrightnessControl or thread-safety risks in the DeviceManager singleton; cross-check with DevEco Studio AI inspector.
- Strengths & limits: Summarise AI's speed in scaffolding and styling versus its shallow business fit, missing extension hooks, and ignorance of HarmonyOS-specific rules.

2) State-Management Breakthrough - Deep Use of @State

- Essence: @State is component-private; value change triggers UI re-render—ideal for local data like switch scale or slider brightness.
- Pain points: AI code often fails to separate "component-local" from "cross-component" state, causing redundant global re-renders.
- AI-aided practice: Refactor LightControlViewModel's device to @Observable; pages link via @Link to avoid redundant paints; AI helps trace update delays.
- Best-practice extension: Clarify when to use @State (local), @Link (parent-child), @Provide/@Consume (cross-level); validate AI-generated decorator choices.

3) Component Encapsulation & Re-use

- Problem analysis: AI-built CircleSwitchButton or BrightnessControl is tightly coupled to business, preventing re-use; parameters lack type checks or defaults.
- AI-assisted refactor: Define param interfaces (e.g., BrightnessControlParam); AI generates types and validation logic; extract business logic to create a reusable component library.
- Re-use test: Add a "bed-room light" page that re-uses the encapsulated components; AI helps resolve style clashes during parameter/event callback verification.

4) Decoupling Business Logic

- Coupling issues: toggleSwitch/updateBrightness directly call viewModel without exception handling; DeviceManager is overly intimate with LightControlViewModel; sync logic is scattered.
- AI-aided decoupling: AI produces a generic updateDeviceParam<T>
- Data-flow mapping: Draw the "Page → ViewModel → Model → DeviceManager" diagram; AI verifies unidirectionality to avoid circular dependencies.

5) Feature Extension - Color-Control Module

- Requirement analysis: Add RGB & colour-temperature control; identify changes in Model, ViewModel, and View layers.
- AI-assisted development: Feed AI the spec; receive baseline ColorControl component; integrate colour logic into existing ViewModel, ensure sync to DeviceManager and history module.
- Debug & optimise: Test colour-light linkage; AI pinpoints format errors or sync delays; refine colour-parameter validation.

6) Full-Cycle Review & Optimisation

- Whole-process review: Recap key milestones—requirements, AI scaffolding, manual review, tech breakthroughs, extension, tuning; quantify AI gains (60% faster component generation, 40% quicker state-debug).
- Optimisation strategy: Distil high-leverage prompt templates; let AI suggest refactors; manually screen and adopt viable ones.
- Final acceptance: Integrate all features; verify code quality, functional completeness, and extensibility; publish "AI-Assisted HarmonyOS Development Guidelines" that specify review standards, state-management rules, and component-encapsulation principles.

4.2.3. Integrated Debugging & AI-Assisted Troubleshooting (2 Class Hours)

Teaching zeros in on the two core debugging links—"problem finding" and "problem fixing"—while letting AI tools empower every step, forming a compact and practical learning loop.

1) Phase 1: Problem Finding & Environment Validation (1 class hour)

- Tutor demo: one-click AI diagnosis of SDK/dependencies and automatic generation of structured log templates.
- Group practice: "three-dimension health check"
- Function: verify switch & slider interaction.
- Data flow: trace one brightness change from ViewModel to Model.
- Resource/layout: ensure icons load and UI is aligned.

2) Phase 2: AI-Aided Diagnosis & Repair (1 class hour)

- Students interact with AI: describe the bug, supply code snippets, use prompts like "analyse the root cause of state desync" or "give a fix for the unresponsive UI".
- They evaluate AI's explanation and patch, verify the fix, and complete a short Project Initial-State Report listing 1 - 2 visible issues plus the repair process.

4.2.4. Performance Optimization and AI-Assisted Diagnosis (3 Class Hours)

This phase guides students to shift from "functional implementation" to "performance excellence," helping them develop an awareness of quantitative performance evaluation and learn to use professional tools and AI for performance bottleneck diagnosis and precise optimization. The designed tasks are shown in **Table 2**.

Table 2. Performance optimization tasks.

Class Hour	Task	AI Empowerment	Student Deliverables
1	Preliminary Performance Diagnosis	Interpreting flame graphs, identifying high-frequency onChange calls, memory leaks	Locate 1 bottleneck, complete the "Performance Snapshot" form
1	Immediate Optimization	One-click generation of debounce	Frame rate ↑10 fps, CPU usage ↓20%
1	Creative Expansion	AI suggests a "scenario mode persistence" solution	Submit core code for Preferences data storage

Based on the design in Table 2, the process begins with a teacher demonstration: AI reads the flame graph and highlights the Slider's 247 calls/s in seconds → suggests using on ChangeEnd + 200 ms debouncing. Students then verify the changes: after replacement and re-sampling, the frame rate improves from 45 to 55 fps, and latency drops from 150 to 70 ms. A screenshot comparison completes this step. Finally, students implement the creative solution: AI provides a template for "saving the current scenario to Preferences," and students supplement the key-value pairs and toast prompts based on the AI-generated code, achieving a runnable demonstration within 10 minutes.

4.2.5. Teaching Reflection

1) Effectiveness Analysis

a) Significant Improvement in Learning Efficiency:

Two parallel classes of the same grade taught by the author in the autumn semester of 2025 were selected as the research objects. Among them, Class 1 of Internet of Things (IoT) Application Technology adopted the "AI + HarmonyOS" teaching model, while Class 2 of IoT Application Technology adopted the traditional teaching model (only using HarmonyOS development tools without AI assistance). Students in Class 2 needed an average of more than 8 class hours to understand and implement the distributed synchronization function, and they encountered great difficulties in debugging. By virtue of AI tools' capabilities in document interpretation and code generation, students in Class 1 shortened the development time of core functions to 6 class hours, enabling them to focus more energy on logical understanding and problem troubleshooting.

b) Enhanced Proficiency in Complex Technologies:

Through the hands-on, contrastive approach of "independent design followed by AI-assisted completion," students' understanding of abstract concepts like distributed communication evolved from merely "being aware of its existence" to "knowing exactly how to implement it and handle exceptions." Testing and practical project defenses showed that over 85% of students could clearly articulate the synchronization logic and fault-tolerance mechanisms in their code.

c) Internalization of AI Tool Application Capabilities:

Students have transformed from initially "blindly trusting AI outputs" to "using AI purposefully for inquiry, verification and optimization". At the end of the case,

students can skillfully use natural language to describe technical problems to AI and critically evaluate the rationality of AI-generated codes.

2) Teaching Reflection

a) The "Double-Edged Sword" Effect of AI Tools

AI has greatly improved learning efficiency, but it has also been observed that some students have developed a "copy-paste dependence". In teaching, the principle of "understanding takes priority over application" must be emphasized, and a three-step principle should be adopted: Pre-control—Students must first submit pseudocode and flowcharts, which can only be used with AI tools after being reviewed and approved by teachers; Process review—Establish a "code traceability form" that requires students to mark and explain the reasons for selecting AI-generated code and the parts modified by themselves; Post-review—Analyze typical error cases in class to help students understand the limitations of AI and systematically cultivate their ability to use AI critically.

b) Profound Transformation of Teachers' Roles:

Teachers have transformed from "knowledge instructors" to project "architects" and "guides". Their core responsibilities include designing challenging tasks, intervening when there are deviations in students' interaction with AI (such as correcting vague demand descriptions), and refining and elevating engineering experience and design ideas that AI cannot impart.

c) Accuracy of Hierarchical Teaching:

The success of this case confirms the necessity of hierarchical tasks. By decomposing the complex lighting control system into a ladder of "UI construction - standalone logic - distributed synchronization - function expansion", students with different foundations can find entry points, obtain positive incentives after completing each step, and finally work together to complete the seemingly arduous comprehensive project.

5. Conclusion

This study addresses the pain points of IoT education in higher vocational colleges by constructing and practicing an interdisciplinary, project-based teaching model that integrates "AI + HarmonyOS." Using a real-world smart-home project as the carrier, AI tools are embedded throughout the entire teaching process, effectively bridging the gap between cutting-edge industry technologies and students' cognitive patterns. The model provides a feasible reference for reforming professional education and cultivating highly skilled talents suited to intelligent terminal development in the era of artificial intelligence. In the future, it will be necessary to continuously track technological evolution, dynamically update the project repository, and further explore the deep application of AI tools in teaching evaluation and personalized learning support

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

Hu, W. X. (2025). AI-Driven Educational Transformation: Application Fields, Real Challenges and Future Prospects. *Journal of Chinese Education, (S1):* 4-6.

Huang, J. C., Wang, Y. D. (2025). From "AI Exclusion" to "AI Enablement": Deepening the Application of Artificial Intelligence in University Teaching. *China Higher Education Research, 4,* 34-41.

Kong, L. S., Hou, J., Qian, L. (2025). Actions and Reflections on AI-Based Instructional Applications in U.S. K-12 Education. *Open Education Research, 31,* 65-73.

Li, J. Y., Lin, B. T., Hu, X. D., et al. (2025). Exploring a Dual-Empowerment Model for Cultivating Interdisciplinary Talent in Oil-Gas AI. *Academic Degrees & Graduate Education, 7,* 39-46.

Long, B. X., Gao, Y. F., Cao, Y. W. (2025). Trends, Risks and Countermeasures of AI Application in Education. *Journal of Soochow University (Educational Science Edition), 13,* 51-61.

Zhao, J., Zhang, J. Y. (2025). Construction and Application of Teaching-Resource Libraries Based on Large AI Models. *China University Teaching, 7,* 37-43+79.