MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages 4691-4695

Multi AI Agent Based Interactive Programming Learning Platform

Kalaivani K, Associate Professor,
Department of CST, SNS COLLEGE OF ENGINEERING, Coimbatore, Tamil Nadu, India, kalaivanisns2@gmail.com
Abinauw B
Department of CST, SNS COLLEGE OF ENGINEERING, Coimbatore, Tamil Nadu, India, reachabinauwbalaji@gmail.com
Kiran Sagar D S
Department of CST, SNS COLLEGE OF ENGINEERING, Coimbatore, Tamil Nadu, India, kiransagards1105@gmail.com
Prem M
Department of CST, SNS COLLEGE OF ENGINEERING, Coimbatore, Tamil Nadu, India, mprem5032@gmail.com
Sri Muthu Manickam I
Department of CST, SNS COLLEGE OF ENGINEERING, Coimbatore, Tamil Nadu, India, ssrimuthu9@gmail.com
Jeyaseelapandi S
Department of CST, SNS COLLEGE OF ENGINEERING, Coimbatore, Tamil Nadu, India, jeyaseelapandi9626@gmail.com

***Abstract*—** The rapid growth of digital education has transformed the way programming skills are taught and learned. However, traditional learning platforms often lack real-time feedback, personalized guidance, and secure coding environments, which limits effective skill development. This project proposes a Multi-Agent AI-Based Programming Learning Platform that provides interactive tutoring, intelligent code evaluation, secure code execution, and personalized learning pathways. The system integrates multiple AI agents, where each agent performs a specific role such as tutoring assistance, automated code analysis, content recommendation, and performance tracking. The platform enables students to learn programming concepts, write and execute code securely within a sandbox environment, receive instant feedback, and monitor their progress through analytics. Teachers can upload study materials, track student performance, and analyse learning trends, while administrators manage system access and security. The application is designed using modern web technologies with a scalable backend architecture and AI model integration to ensure real-time response and adaptability. By combining intelligent tutoring, automated assessment, and personalized content delivery, the proposed system enhances learner engagement, reduces dependency on manual instruction, and creates a safe and adaptive programming education environment.

Keywords— Multi-Agent Systems, AI Tutor, Programming Education, Intelligent Code Evaluation, Secure Code Execution, Personalized Learning.

I. INTRODUCTION

The rapid advancement of digital technologies has significantly transformed the landscape of education, particularly in the field of computer science and programming. In recent years, online learning platforms, coding practice websites, and digital classrooms have become essential tools for students seeking to develop programming skills. While these platforms provide accessibility and convenience, many of them still follow traditional content-delivery methods that lack personalization, intelligent feedback, and adaptive support. As a result, learners often face challenges in understanding complex programming concepts, debugging errors, and progressing efficiently according to their individual learning pace.Programming education requires continuous practice, logical reasoning, and timely feedback. However, in conventional classroom settings, instructors may not be able to provide personalized attention to every student due to limited time and large class sizes. Similarly, many online platforms offer static tutorials and automated test cases without explaining why errors occur or how solutions can be improved. This lack of contextual assistance and interactive guidance can lead to confusion, reduced motivation, and slower skill development among learners.With the emergence of Artificial Intelligence (AI), Machine Learning (ML), and Natural Language Processing (NLP), there is a growing opportunity to enhance programming education through intelligent and adaptive systems. AI-powered tutoring systems can understand user queries, analyze code submissions, and provide real-time feedback in a conversational manner. Multi-agent systems, in particular, offer a structured approach where multiple intelligent agents collaborate to perform specialized tasks such as tutoring, evaluation, personalization, and monitoring. This distributed intelligence improves system efficiency, scalability, and responsiveness.

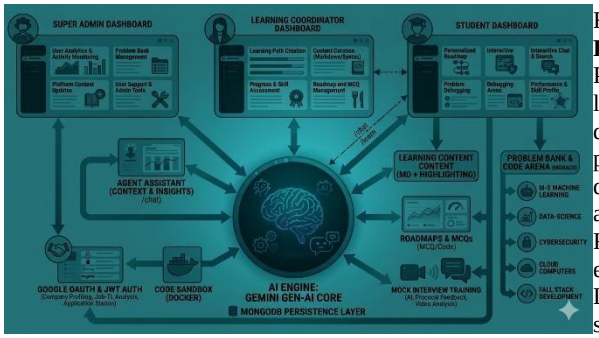


Fig 1. AIGENT LEARN: Integrated Modular Architecture

Figure 1 presents the overall architecture of the AIGENT LEARN – AI- Powered Programming Learning Platform, where the central AI Engine acts as an intelligent learning assistant that supports users in understanding programming concepts, debugging code, and receiving personalized learning guidance. The AI engine processes user queries, learning topics, and code inputs to generate explanations, debugging suggestions, and structured learning content. The platform is designed around a user-centric interface consisting of key modules such as Login and Registration, User Dashboard, Learn Module, Chat Module, and Debug Module, each providing specific functionality to enhance the learning experience.The Login and Registration page ensures secure access and user authentication, allowing learners to create accounts and access the platform safely. After authentication, users are directed to the User Dashboard, which serves as the central hub displaying learning progress, recently accessed topics, and navigation to different modules. The Learn Module allows users to explore programming topics with AI-generated explanations, examples, and structured guidance that simplify complex concepts. The Chat Module functions as an interactive AI assistant where learners can ask programming- related questions and receive instant explanations and support. The Debug Module enables users to analyze their code by detecting errors and providing intelligent debugging suggestions to help them understand and correct mistakes.

II. RELATED WORKS

[1] M. Wooldridge (2009) discussed the concept of multi-agent systems where multiple intelligent agents cooperate to solve complex tasks. This idea forms the foundation for modern AI systems that distribute responsibilities across different agents, which is relevant for platforms that combine tutoring, assistance, and analysis features.

[2] B. Woolf (2010) explored intelligent tutoring systems designed to assist learners by adapting explanations based on student performance. The study highlighted how AI can guide learners through difficult topics and provide feedback that supports independent learning.

[3] G. Romero and S. Ventura (2013) presented educational data mining techniques used to analyze student learning behavior. Their work showed that analyzing learner interactions with digital platforms can help identify learning patterns and improve educational systems.

[4] J. Piech et al. (2015) introduced systems that use machine learning to analyze student programming submissions and provide automated feedback. This research demonstrated that AI can assist students in understanding programming errors and improving their coding skills.

[5] C. Guo (2018) studied online programming learning platforms and highlighted the challenges students face when learning coding independently. The research emphasized the need for interactive learning tools that can guide students when they encounter difficulties.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages 4691-4695

[6] T. Brown et al. (2020) demonstrated the capabilities of large language models in understanding natural language queries and generating meaningful responses. These models enable conversational AI systems that can assist users with explanations and problem solving.
[7] J. Devlin et al. (2019) introduced transformer-based models for natural language understanding, showing how AI systems can interpret questions and generate relevant responses. Such models are widely used in AI assistants and educational support tools.
[8] A. Ko and B. Myers (2010) studied how novice programmers struggle with debugging and error understanding. Their research showed that beginners often require guided explanations to understand programming mistakes effectively.
[9] K. Ala-Mutka (2005) examined automated assessment systems for programming education. The study explained how automated tools can evaluate code and provide feedback, helping learners improve their programming skills.
[10] M. Lister et al. (2004) analyzed how students learn programming concepts and found that beginners benefit from step-by-step explanations and interactive guidance rather than static learning materials.

III. ARCHITECTURE AND DESIGN

The proposed **AIGENT LEARN – AI-Powered Programming Learning Platform** is designed using a modular architecture that supports interactive learning, intelligent assistance, and automated code analysis. The system integrates multiple components that work together to help users understand programming concepts, resolve coding errors, and receive personalized guidance during the learning process. The architecture mainly consists of five major components: User Interaction Interface, Query Processing and Validation Module, AI Learning and Assistance Engine, Code Analysis and Debugging Module, and Learning Progress and Recommendation Layer. These components collectively transform user inputs such as questions or code snippets into meaningful explanations, debugging suggestions, and structured learning content.

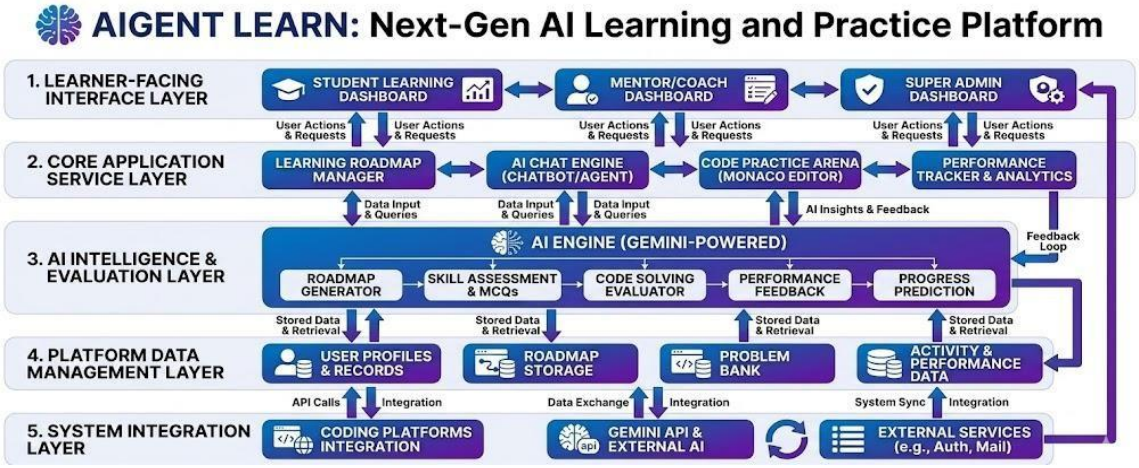


Fig 2. System architecture

A. User Interaction Interface

The User Interaction Interface acts as the entry point through which learners interact with the AIGENT LEARN platform. It includes the Login and Registration page, User Dashboard, Learn module, Chat interface, and Debug page. Through this interface, users can access learning resources, ask programming-related questions, and submit code for analysis. The interface is designed to be simple and responsive so that learners can easily navigate between modules. Authentication mechanisms ensure that only registered users can access the platform. This layer focuses on usability and accessibility so that beginners can interact with the system without technical complexity.

B. Query Processing and Validation Module

Once the user submits a question or code snippet, the system processes the input through the Query Processing and Validation module. This component verifies the input format, removes unnecessary characters, and organizes the information so that it can be analyzed effectively. For example, programming code is formatted and checked for basic structure before further analysis. Similarly, user queries are interpreted to understand the intent of the question. By cleaning and structuring the input data, this module ensures that the AI components receive clear and consistent information, improving the reliability of the system's responses.

C. AI Learning and Assistance Engine

The AI Learning and Assistance Engine forms the intelligence core of the AIGENT LEARN platform. This component interprets user questions and generates explanations that help learners understand programming concepts. The engine analyzes the context of the user's request and produces responses that include simplified explanations, examples, and learning guidance. By processing natural language questions, the system acts as an interactive learning assistant that supports users whenever they face difficulties. The AI engine enables the platform to deliver dynamic educational support instead of static learning materials.

D. Code Analysis and Debugging Module

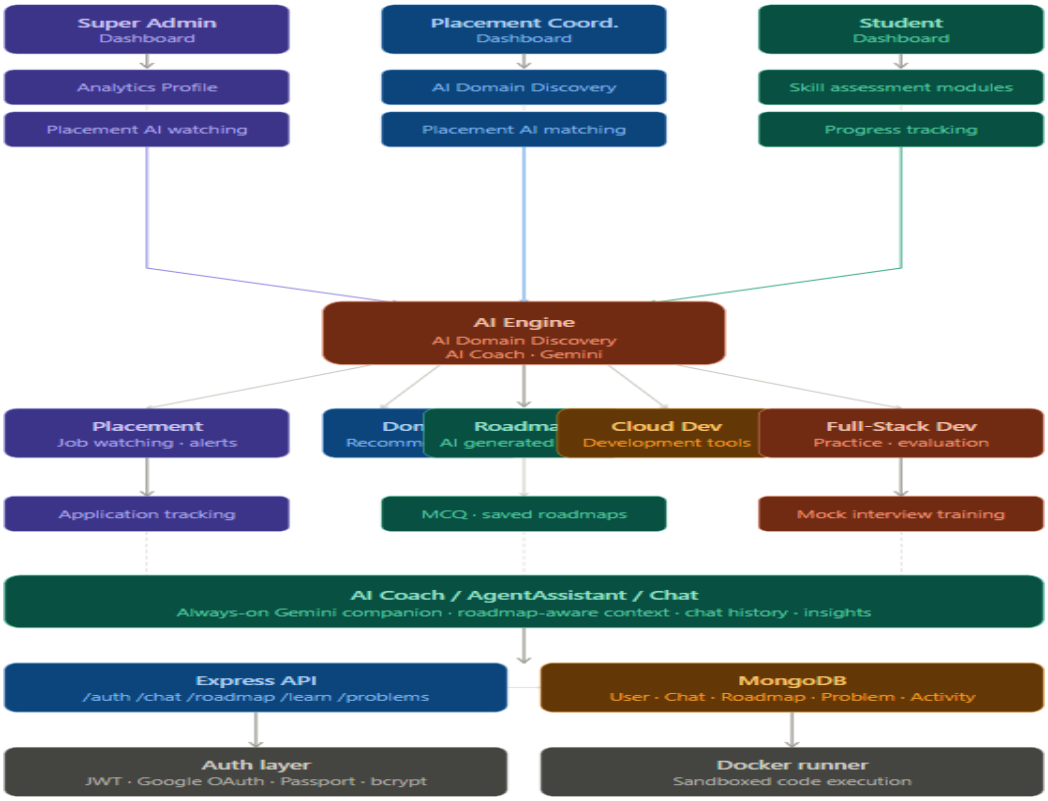
The Code Analysis and Debugging Module focuses on helping users identify and correct errors in their programming code. When users submit code through the Debug page, the system examines it to detect syntax issues, logical mistakes, or inefficient structures. After analyzing the code, the module provides feedback and suggestions that guide the learner toward the correct solution. Instead of simply showing errors, the system explains why the problem occurred and how it can be resolved. This approach helps learners improve their coding skills while developing a deeper understanding of programming logic.

E. Learning Progress and Recommendation Layer

The Learning Progress and Recommendation Layer monitors user activity and tracks learning progress within the platform. It records interactions such as topics explored, questions asked, and debugging attempts. Based on these interactions, the system can suggest additional topics or areas that the learner should focus on. The dashboard presents this information through simple progress indicators that help users understand their improvement over time. By providing personalized learning suggestions, this module encourages continuous learning and helps users strengthen their programming knowledge step by step.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages 4691-4695

Flow chart: System Architecture Workflow



IV. METHODOLOGY

The methodology of the **AIGENT LEARN – AI-Powered Programming Learning Platform** follows a structured workflow that enables users to interact with the system, receive explanations for programming concepts, and debug their code efficiently. The system combines user interaction modules, intelligent query processing, code analysis techniques, and AI-based assistance to create an effective programming learning environment. The methodology ensures that user queries and code inputs are properly processed and converted into meaningful feedback that supports learning and problem solving.The workflow of the system is organized into several phases as described below.

User Data Handling:The platform begins with the collection and management of user information through the Login and Registration module. Users create accounts by providing basic details, which are securely stored in the system database. Authentication mechanisms verify user credentials during login to ensure secure access to the platform. Once authenticated, users are directed to their personal dashboard where their interactions, learning activities, and queries are managed. The system maintains user session information and activity logs so that the platform can track usage and provide a personalized learning experience.

Query Processing and Input Preparation: When users interact with the platform through the Chat or Debug modules, the system first processes the input to understand the user's request. If the user submits a programming question, the system analyzes the text to identify the topic or problem being discussed. If the user submits a code snippet for debugging, the code is formatted and validated to ensure it follows the correct structure. During this stage, unnecessary characters are removed and the input is organized so that the AI engine and code analysis modules can process it efficiently. This step ensures that the system receives clean and structured input for further analysis.

C. AI Assistance and Code Analysis: The core intelligence of the platform operates in this stage. The AI engine interprets user queries and generates explanations for programming concepts, helping learners understand topics more clearly. For debugging requests, the system analyzes the submitted code to identify possible syntax errors, logical mistakes, or inefficient programming patterns. Based on the analysis, the platform provides suggestions and explanations that guide the learner in correcting their code. Instead of simply showing an error message, the system explains the issue and recommends possible improvements so that users can learn from their mistakes.

D. System Deployment and User Interaction: The platform is deployed as a web-based application where users interact with the system through a browser interface. The backend services handle user authentication, query processing, AI responses, and database management. When a user asks a question or submits code, the request is processed through the backend system and the results are returned to the interface in real time. This architecture ensures smooth communication between the frontend interface and the backend intelligence modules. The dashboard provides easy navigation to different features such as learning topics, chat assistance, and debugging tools.

Dataset Type	Description	Number of Records
User Profile Data	Registration details, learning preferences, selected programming languages	4,500
Programming Practice Data	Code submissions, exercise attempts, project implementations	3,800
Debugging & Error Logs	Syntax errors, logical errors, runtime issues identified during submissions	3,200
AI Interaction Data	Chat queries, concept explanations requested, clarification prompts	2,400
Learning Progress Data	Topic completion status, practice frequency, performance trends	1,500
Total		15,400

E. Learning Feedback and Continuous Improvement

The system continuously records user interactions, including questions asked, topics explored, and debugging attempts. This information helps the platform monitor learning progress and suggest areas where the user can improve. Based on previous interactions, the platform can recommend related programming topics or provide additional explanations to strengthen the learner's understanding. This feedback mechanism encourages continuous learning and ensures that users gradually improve their programming skills through guided assistance.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages 4691-4695

RESULTS AND DISCUSSION

A. Experimental Setup: The experimental evaluation of the **AIGENT LEARN – AI-Powered Programming Learning Platform** was conducted by analyzing user interactions, programming queries, and debugging submissions collected during system testing. The evaluation focused on measuring how effectively the system provides programming explanations, detects coding errors, and assists users in resolving problems. User queries and code samples were processed through the AI engine and debugging module to observe system response accuracy and usability. The collected dataset was divided into training and testing sets to evaluate the performance of the AI-assisted learning and code analysis modules. System performance was evaluated based on response relevance, debugging accuracy, and user interaction efficiency. These metrics helped determine how effectively the platform supports programming learning through AI assistance.

Table I. Dataset Distribution

Dataset Type	**Description**	**Number of Records**
User Profile Data	Registration details, learning preferences, activity logs	2,000
Programming Query Data	User-submitted questions and concept clarification requests	5,200
Code Submission Data	Code snippets submitted for debugging and error analysis	4,000
Learning Interaction Data	Chat interactions, topic access patterns, learning sessions	2,700
Total		**13,900**

B. Baseline System Performance: Initial testing focused on evaluating the system's ability to respond to programming queries and analyze code errors. The AI engine successfully generated explanations for common programming topics such as loops, conditional statements, and data structures. The debugging module was able to identify syntax errors and provide correction suggestions in most test cases.

Results showed that combining conversational assistance with code analysis significantly improved the learning experience. Users were able to understand errors more quickly because the system not only identified problems but also explained possible solutions.

C. System Module Evaluation: Different modules of the AIGENT LEARN platform were evaluated to measure their contribution to the learning process. The Chat module was tested for question answering capability, while the Debug module was tested for code error detection and correction suggestions. The Learn module was evaluated for clarity of explanations and topic organization.

The results showed that the integration of these modules provided a more effective learning environment compared to static programming tutorials. The platform allowed users to interact with the system, ask questions, and immediately apply corrections to their code.

Table II. System Feature Comparison

System Type	**Learning Interaction**	**Debugging Support**	**User Effort**
Traditional Programming Books	Low	None	High
Online Video Tutorials	Moderate	None	Medium
Static Programming Learning Websites	Moderate	Limited	Medium
AI-Based Learning Assistant Platforms	High	Moderate	Low
AIGENT LEARN Platform	**High**	**High**	**Low**

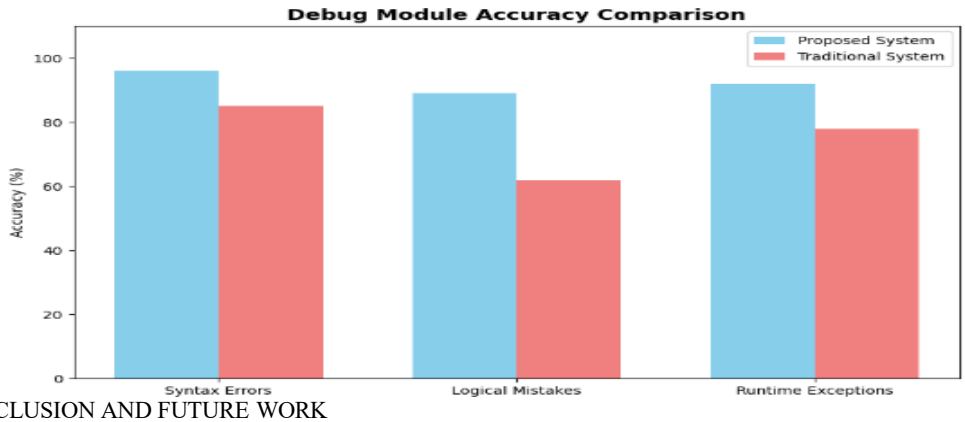
D. Usability and Learning Experience

User testing indicated that the platform provided a convenient and supportive environment for programming learners. The dashboard allowed users to easily navigate between learning resources, AI chat assistance, and debugging tools. Users reported that the chat module helped them clarify programming concepts quickly, while the debugging module helped them identify coding mistakes more effectively.

The integration of AI explanations with interactive debugging made the learning process more engaging. Users were able to experiment with code, identify errors, and improve their understanding of programming logic.

E. Limitations

Although the platform demonstrated effective support for programming learning, some limitations were observed during testing. The accuracy of explanations and debugging suggestions depends on the quality of the training data used by the AI engine. Certain complex logical errors may require deeper analysis beyond basic automated detection.



V. CONCLUSION AND FUTURE WORK

The development of the Multi-Agent AI-Based Programming Learning Platform represents a significant advancement in modern programming education. The system successfully integrates intelligent tutoring, automated code evaluation, secure execution, and personalized learning into a single unified platform. By leveraging multi-agent architecture, the platform ensures efficient task distribution, real-time feedback, and adaptive

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages 4691-4695

content delivery tailored to individual learner needs. The experimental results demonstrate that the system provides fast response times, accurate debugging assistance, and improved learner engagement. The Debug Module enhances conceptual clarity by offering detailed explanations and optimization suggestions rather than simple error messages. The Learn Module supports structured and self-paced learning, while the User Dashboard enables continuous performance tracking and data-driven improvement. The secure sandbox execution environment ensures safe code testing, maintaining system integrity and user data protection. Overall, the proposed platform addresses key limitations of traditional e-learning and coding systems by combining interactivity, intelligence, personalization, and security. It reduces dependency on manual instruction, improves problem-solving skills, and creates a scalable solution suitable for academic institutions and self-learners alike. In future work, the system can be further enhanced by incorporating advanced features such as multilingual support to make the platform accessible to a wider audience. Integration of voice-based interaction and conversational AI can improve user engagement and accessibility.

REFERENCES

[1] R. S. Baker and K. Yacef, "The state of educational data mining in 2009: A review and future visions," *Journal of Educational Data Mining*, vol. 1, no. 1, pp. 3–17, 2009.
[2] B. Woolf, *Building Intelligent Interactive Tutors: Student-Centered Strategies for Revolutionizing E-Learning*, Morgan Kaufmann, 2010.
[3] K. VanLehn, "The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems," *Educational Psychologist*, vol. 46, no. 4, pp. 197–221, 2011.
[4] C. Piech, J. Huang, Z. Chen, C. Do, A. Ng, and D. Koller, "Tuned models of peer assessment in MOOCs," in *Proceedings of the International Conference on Educational Data Mining*, 2013.
[5] S. Gulwani, W. R. Harris, and R. Singh, "Spreadsheet data manipulation using examples," *Communications of the ACM*, vol. 55, no. 8, pp. 97–105, 2012.
[6] M. Ihantola, T. Ahoniemi, V. Karavirta, and O. Seppälä, "Review of recent systems for automatic assessment of programming assignments,"
in *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*, 2010.
[7] E. Keuning, J. Jeuring, and B. Heeren, "A systematic literature review of automated feedback generation for programming exercises," *ACM Transactions on Computing Education*, vol. 19, no. 1, 2018.
[8] P. Brusilovsky and E. Millán, "User models for adaptive hypermedia and adaptive educational systems," in *The Adaptive Web*, Springer, 2007.
[9] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed., Pearson, 2010.
[10] M. Wooldridge, *An Introduction to MultiAgent Systems*, 2nd ed., John Wiley & Sons, 2009.
[11] T. Mikolov et al., "Efficient estimation of word representations in vector space," in *Proceedings of ICLR Workshops*, 2013.
[12] A. Vaswani et al., "Attention is all you need," in *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS)*, 2017.