CodePulse：資料結構與演算法視覺化教學平台

專題編號：114-2-CSIE-S017
執行期限：114 年第 1 學期至 114 年第 2 學期
指導教授：陳香君
專題參與人員： 112590007 鄭安琋
112590035 葉凱成
112590042 鄭錦鑫

一、摘要

This study develops CodePulse, a visual learning platform for data structures and algorithms, aimed at improving learners' understanding of algorithms and program execution. The platform consists of two main functional areas: Tutorial Mode and Explore Mode. Tutorial Mode provides theory introduction, interactive visualizations, and practice exercises for foundational learning. Explore Mode includes Lab for algorithm comparison and Playground for code analysis, flowchart generation, and animation application. AI techniques are further integrated for algorithm identification, code analysis, and error feedback to assist learners in program comprehension and debugging. Through these features, the platform provides a learning environment applicable to introductory programming and algorithm learning contexts, thereby helping learners construct clearer mental models of program execution.
關鍵詞：program visualization, algorithm education, interactive learning, debugging assistance

二、緣由與目的

Current programming education broadly faces challenges of insufficient visualization tools and poor interactivity. Most beginners rely on text descriptions or static diagrams to understand program behavior, making it difficult to grasp data state changes and the dynamic characteristics of algorithm execution, leading to significantly increased cognitive load[1]. Research shows that lack of immediate feedback is a key factor in high CS1 dropout rates [2]. Existing tools exhibit notable limitations: traditional debuggers such as GDB and pdb present high conceptual barriers, whereas lightweight visualization tools such as Python Tutor and VisuAlgo provide accessibility but lack analytical depth, making it difficult to support in-depth exploration of complex data structures and algorithms [3], [4], thereby limiting learners' ability to construct accurate mental models of program execution. [5].

Based on this, the CodePulse platform integrates visualization, interactivity, and AI analysis to fill the above gap. Tutorial Mode provides dynamic visualization of data structures and algorithms, adjustable interactive operations, and practice exercises, allowing learners to simultaneously observe source code and execution behavior. Lab allows users to intuitively compare behavioral differences between algorithms; Playground accepts user-submitted code and automatically generates visual analysis with AI feedback for program comprehension and debugging, achieving a learning environment that is both systematic and exploratory.

三、研究範圍與系統架構

This research covers dynamic code tracing, algorithm visualization, and AI-assisted analysis, with Python as the primary supported language, offering three functional modes: Tutorial, Lab, and Playground.

The system adopts a four-layer architecture consisting of presentation, API gateway, application, and infrastructure

layers. The application layer includes modules for authentication, user management, code execution, AI analysis, and learning progress management. The infrastructure layer integrates Celery, Redis, Docker-based sandbox execution, and PostgreSQL, with external services including Gemini API, Cloudinary, and SMTP Email.

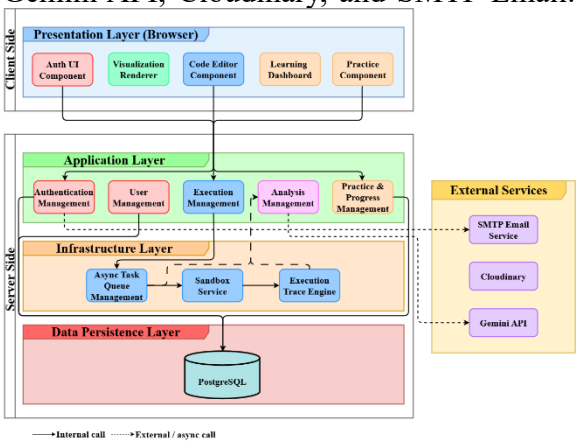

Figure 1 The architecture of the system

四、使用技術與工具

The frontend is developed using React and TypeScript, integrating Monaco Editor, D3.js, and Cytoscape.js for code editing, algorithm animation, and program flow visualization.

The backend is built with Flask and incorporates asynchronous task processing, sandboxed code execution, and execution tracing mechanisms for secure program analysis.

Algorithm identification is performed through embedding-based similarity analysis, while large language models are integrated for AI-assisted feedback generation.

五、實作成果與結論

The platform has completed core features including interactive tutorials, algorithm comparison, and code analysis with visualization support.

This research addresses the gap between highly complex debugging tools and overly simplified visualization systems by providing a platform that balances structured learning with exploratory analysis. The implementation demonstrates the feasibility of integrating program visualization, dynamic code tracing, and AI-assisted analysis within an interactive learning environment.

參考文獻

[1] E. Lahtinen, K. Ala-Mutka and H. M. Järvinen, "A study of the difficulties of novice programmers," *ACM SIGCSE Bulletin,* vol. 37, no. 3, pp. 14-18, 2005.

[2] A. Petersen, M. Craig, J. Campbell and A. Tafliovich, "Revisiting why students drop CS1," in *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, Koli, 2016.

[3] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B.-D. Kolikant, C. Laxer, L. Thomas, I. Utting and T. Wilusz, "A multi-national, multi-institutional study of assessment of programming skills of first-year CS students," *ACM SIGCSE Bulletin,* vol. 33, no. 4, pp. 125-180, 2001.

[4] T. L. Naps, R. Fleischer, M. McNally, G. Rößling, C. Hundhausen, S. Rodger, V. Almstrum, A. Korhonen, J. Á. Velázquez-Iturbide, W. Dann and L. Malmi, "Exploring the role of visualization and engagement in computer science education," *ACM SIGCSE Bulletin,* vol. 5, no. 2, pp. 131-152, 2002.

[5] J. Sorva, "Visual program simulation in introductory programming education," Aalto University, Espoo, 2012.