*Grenze International Journal of Engineering and Technology, January Issue*

# Optimised Intelligent Software Company Management System using Multi-Agent Framework

Purva Sharma[1] and Jayakumar Kaliappan[2]
[1-2]Vellore Institute of Technology, Vellore, India
Email: purvasharma3008@gmail.com, jayakumar.k@vit.ac.in

***Abstract*— The software development industry faces ongoing challenges in managing complex projects and resources effectively. The Intelligent Software Company Management System (ISCMS) is an innovative approach designed to enhance the efficiency of software company operations through a multi-agent framework. This system automates key roles within a software company, including Client, Architect, Developer, Engineer, Tester, and Project Manager, with each agent responsible for specific operational tasks. These agents work together to manage projects, allocate resources, and handle client interactions, resulting in greater efficiency and streamlined project delivery. ISCMS is built on a powerful tech stack comprising Autogen, CREW AI, LangGraph, Python, and advanced Language Learning Models (LLMs) like Gemini and OpenAI, ensuring it can scale and adapt to various project needs and organizational structures. This research explores the performance of ISCMS by comparing the efficiency of three agent frameworks—Autogen, CREW AI, and LangGraph—under the same project conditions. The findings reveal notable differences in performance, shedding light on the strengths and weaknesses of each framework in managing software operations. By clearly defining agent roles and leveraging cutting-edge technologies, ISCMS offers a versatile platform that not only automates management processes but also improves communication and productivity within development teams. This research contributes valuable insights into the practical application of multi-agent systems in corporate management and highlights the most effective frameworks for optimizing software company operations.**

***Index Terms*—Language Learning Models (LLMs), Multi-Agent Framework, Autogen, CrewAI, LangGraph, Python.**

## I. INTRODUCTION

In today's rapidly evolving software industry, managing increasingly complex projects and resources efficiently has become a pressing challenge, especially as companies scale and diversify their operations. Traditional management methods often fail to meet the demands for automation, seamless coordination, and adaptability across diverse teams and projects, creating a need for more intelligent and scalable solutions. This has motivated the exploration of multi-agent systems, which have the potential to streamline processes by automating key roles within a software company, such as Client, Architect, Developer, Engineer, Tester, and Project Manager. These systems not only enhance efficiency in project execution and resource allocation but also improve communication and client management. To address this need, the Intelligent Software Company Management System (ISCMS) was developed, leveraging advanced technologies like Autogen, CREW AI, LangGraph, and cutting-edge Language Learning Models (LLMs) including Gemini and OpenAI.

*Grenze ID: 01.GIJET.11.1.1*

The objective of this research is to compare the performance of three agent frameworks—Autogen, CREW AI, and LangGraph—under identical project conditions, assessing their strengths and limitations in managing operations within software companies. This study aims to provide insights into the optimal framework for improving project delivery, scalability, and overall operational efficiency in the software development process.

## II. Literature Survey

The literature on multi-agent systems (MAS) highlights their growing role in enhancing software development through automation and collaboration. Park and Sugumaran [1] introduce a framework addressing software complexity by emphasizing role differentiation among agents. Dibia et al. [2] expand accessibility with AutoGen Studio, a no-code platform for creating agent-based applications.

Peltomaa Åström and Winoy [3] show how integrating large language models (LLMs) with MAS improves automation, while Tufano et al. [4] emphasize the flexibility of AI-driven methodologies like AutoDev in adapting to technological changes. Testing is explored by Bokil et al. [5], who use MAS for automatic test data generation, and Fisher et al. [6], who demonstrate MAS efficiency in managing complex command sequences. Wu et al. [7] present Autogen for multi-agent communication, enhancing task management, while White [8] discusses autonomous behaviors in MAS design.

Recent advancements in LLMs have driven MAS innovation. Zhu et al. [10] introduce ReDel, a toolkit combining recursive MAS with LLMs, while Bersenev et al. [9] and Zhuge et al. [13] explore their application in research and mathematical optimization. Educational uses are highlighted by Kasneci et al. [15] and Askarbekuly and Aničić [16], who show LLMs' adaptability in learning environments. Ma [17] illustrates their potential in self-evolving AI systems for chemical research.

This body of work underscores MAS and LLMs' transformative potential across domains. By comparing Autogen, CREW AI, and LangGraph, this research aims to identify their strengths and limitations, contributing to intelligent software management systems and advancing automation in project management.

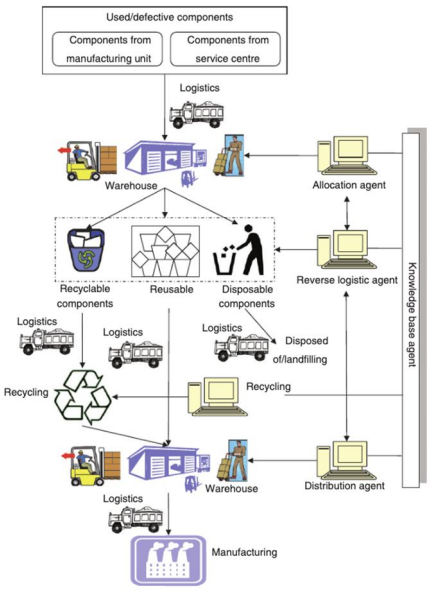


Figure 1. Workflow of sample multi-agent framework for logistics in green supply chain [18]

A multi-agent framework in green supply chain logistics optimizes efficiency and sustainability, informing comparisons of Autogen, CREW AI, and LangGraph in software project management. It highlights the role of intelligent systems in promoting sustainability and modern research.

## III. Proposed Methodology

Figure 2 outlines the experiment steps. This research compares Autogen, CREW AI, and LangGraph for automating software project management. A standardized environment tests each framework using the same inputs, tasks, and KPIs like execution time and error rates. Findings, analyzed statistically and through case studies, provide recommendations for selecting the best framework.

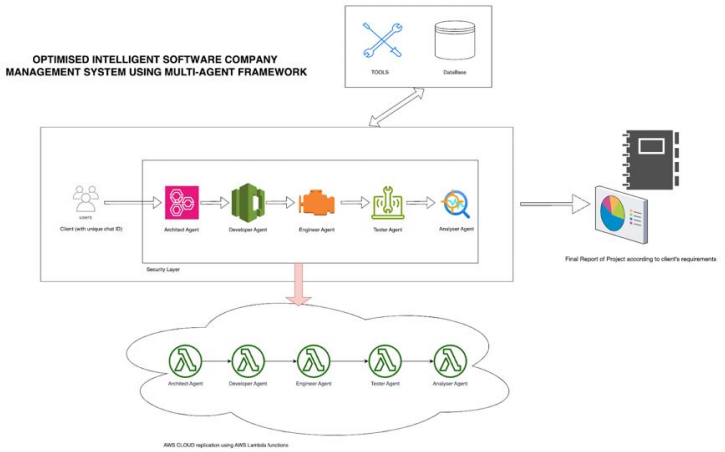


Figure 2. A Flowchart of Our Proposed Methodology

### *A. Framework Selection*

The research focuses on three multi-agent frameworks—Autogen, CREW AI, and LangGraph—selected for their features in automating software project management. Autogen excels in enabling human-like agent conversations for efficient task management.

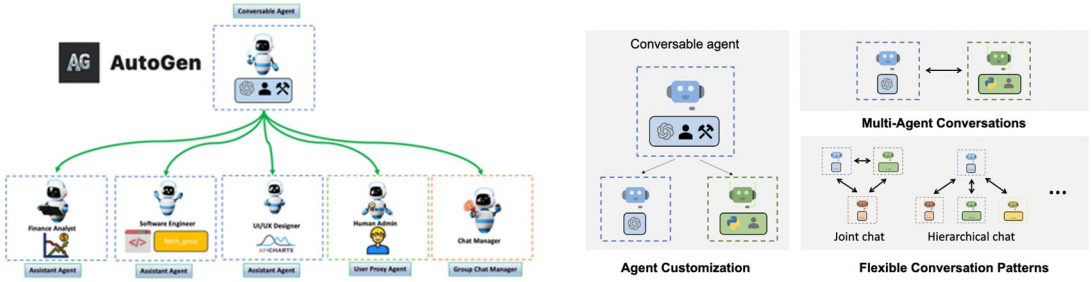


Figure 3. Autogen agents

CREW AI, on the other hand, excels in its integration with large language models, enabling advanced natural language processing capabilities. This integration allows CREW AI to understand and interpret user inputs more effectively, leading to improved responses and interactions tailored to specific project needs.

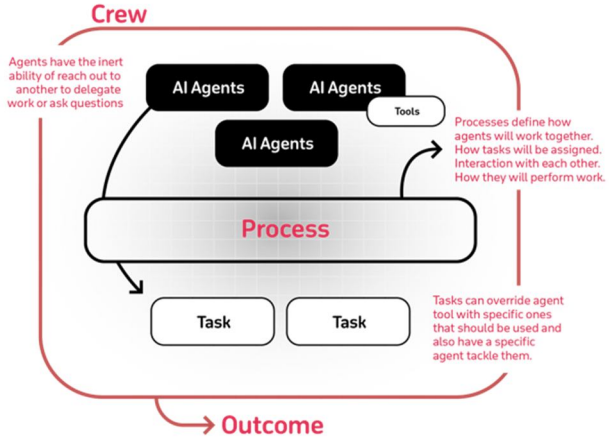


Figure 4. CrewAI workflow

LangGraph stands out for its innovative approach to visualizing agent workflows and interactions. This framework enables users to design and manage multi-agent systems through a graphical interface, simplifying the process of setting up and modifying agent behaviors as project requirements evolve.

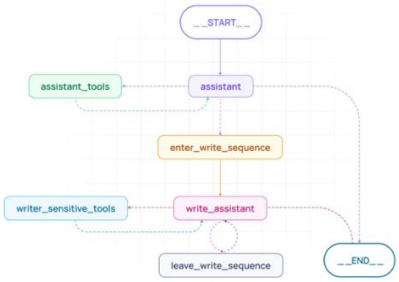


Figure 5. LangGraph working

*B. Implementation Environment*

The implementation environment is designed to develop and test the multi-agent frameworks: Autogen, CREW AI, and LangGraph. Python is the primary language due to its robust libraries and API integration capabilities. A virtual environment (using venv or conda) ensures dependency isolation, with essential libraries like requests, Flask, and pandas installed for API calls, web interfaces, and data handling.

IDE tools such as PyCharm or VS Code streamline coding, debugging, and testing with features like file navigation, real-time analysis, and an integrated terminal. Configuration files securely manage API keys and framework settings. This setup supports iterative development, enabling adjustments based on framework performance evaluations.

*C. Input Requirements*

The input requirements for the system are categorized into hardware and software specifications to ensure optimal performance and scalability for the multi-agent framework.

*Hardware and Software Requirements*

- Processor: A multi-core CPU (e.g., Intel i7 or AMD Ryzen 7) for efficient parallel task handling, especially with simultaneous agent execution.
- Operating System: Windows 10/11 or Ubuntu 20.04 LTS+ (preferred for tool compatibility and package management).
- Languages: Python for the primary framework, with optional JavaScript for front-end development.
- Backend Frameworks: Flask or Django for web interfaces; Pandas and NumPy for data processing within the framework.
- Database: PostgreSQL or MongoDB, selected based on scalability and query needs.
- Security: OpenSSL for encryption, firewalls for network security, and access controls to secure sensitive data and communication pathways.

## IV. Results and Discussion

The user provided a sample project to all three frameworks—Autogen, LangGraph, and CREW AI—to compare their performance in managing roles like Architect, Developer, Engineer, and Tester.

| | |
|---|---|
| Project Requirements | 1. Develop a web-based project management tool.<br>2. Features:<br>- User authentication and authorization<br>- Task tracking and management<br>- Project timelines and Gantt charts<br>- Team collaboration and messaging<br>- File sharing and document management<br>- Integration with third-party tools (e.g., Slack, GitHub) |
| Constraints and Limitations | 1. The system must be scalable to handle up to 1,000 concurrent users.<br>2. Must comply with GDPR and data protection regulations.<br>3. Limited budget of $50,000.<br>4. Project deadline: 6 months from the start date. |
| System Integration Requirements | 1. Integration with existing CRM systems.<br>2. APIs for data import/export.<br>3. Support for single sign-on (SSO) with OAuth2. |
| Testing Requirements | 1. Functional testing for all features.<br>2. Performance testing to ensure system scalability.<br>3. Security testing to identify and fix vulnerabilities. |
| Project Timelines and Budget | 1. Initial planning and design: 1 month<br>2. Development and integration: 4 months<br>3. Testing and quality assurance: 1 month<br>4. Total budget: $50,000 |

Figure 6. Sample input given by user (client)

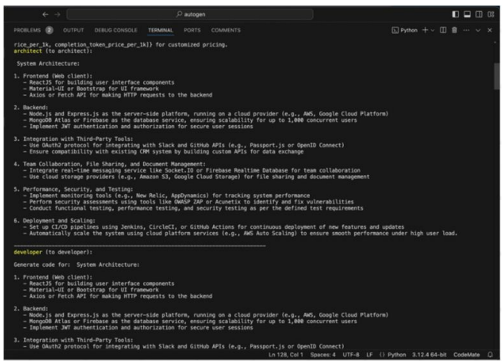

Figure 7. Sample input given by user (client)

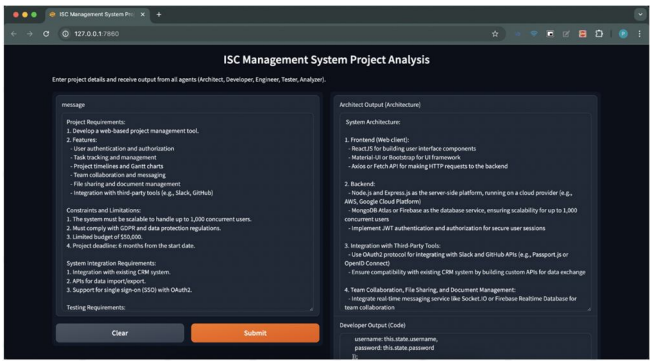


Figure 8. Sample Output (UI using Gradio)

Figure (9) compares the response times of agents across the three frameworks, with LangGraph showing the fastest response times, followed by Autogen and CREW AI. Figure (10) analyzes the number of bugs caught by human testers versus automated agents. While human testers were more effective at identifying complex issues, automated agents excelled at detecting routine bugs and discrepancies, offering greater speed and accuracy in handling repetitive tasks.

Charts examined resource use (CPU, memory, storage) and agent communication efficiency, showing CREW AI slightly excelled in scalability with larger inputs (Fig. 8).

Overall, multi-agent systems reduced task times and improved bug detection, with each framework showing unique strengths for optimization.

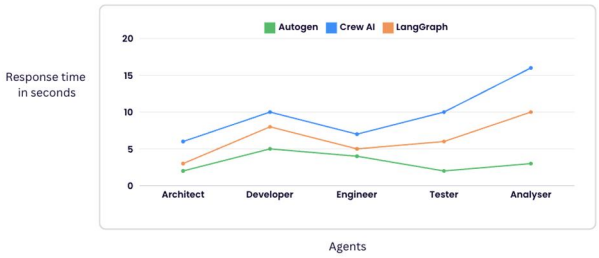


Figure 9. Response times comparison

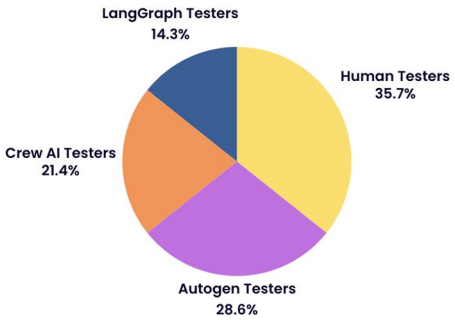


Figure 10. Bugs caught by human testers versus the automated tester agents within each framework using five identical sample inputs


Figure 11. Efficiency based on several metrics

## V. Conclusions

The experimental comparison of Autogen, LangGraph, and CREW AI highlighted unique strengths and weaknesses in software company simulations. Autogen emerged as the top performer due to its advanced features, robustness, and comprehensive documentation, simplifying implementation and customization. It excels in dynamic, conversation-driven tasks, offers built-in termination options to prevent infinite loops, and includes a "human in the loop" feature for real-time error mitigation—features lacking in the other frameworks.
LangGraph performed well in structured, finite state machine workflows but lacks the flexibility of Autogen, limiting its suitability for dynamic scenarios. CREW AI showed promise but struggled with agent loops and frequent hallucinations, reducing its reliability for complex projects.
In conclusion, while LangGraph and CREW AI have potential, Autogen is currently the best choice for projects needing conversational flexibility, reliable agent termination, and human oversight.

## Acknowledgment

I am deeply grateful to the management of Vellore Institute of Technology (VIT) for providing me with the opportunity and resources to undertake this project. Their commitment to fostering a conducive learning environment has been instrumental in my academic journey. The support and infrastructure provided by VIT have enabled me to explore and develop my ideas to their fullest potential.

## References

[1] Park, Sooyong, and Vijayan Sugumaran. "Designing multi-agent systems: a framework and application." Expert Systems with Applications 28.2 (2005): 259-271.
[2] Dibia, Victor, et al. "AutoGen Studio: A No-Code Developer Tool for Building and Debugging Multi-Agent Systems." arXiv preprint arXiv:2408.15247 (2024).
[3] Peltomaa Åström, Samuel, and Simon Winoy. "Automating Software Development Processes Through Multi-Agent Systems: A Study in LLM-based Software Engineering." (2024).
[4] Tufano, Michele, et al. "AutoDev: Automated AI-Driven Development." arXiv preprint arXiv:2403.08299 (2024).
[5] Bokil, Prasad, et al. "Automatic test data generation for c programs." 2009 Third IEEE International Conference on Secure Software Integration and Reliability Improvement. IEEE, 2009.
[6] Fisher, Forest, Roy Gladded, and Teerapat Khanampompan. Automatic Command Sequence Generation. No. NPO-43638. 2007.
[7] Wu, Qingyun, et al. "Autogen: Enabling next-gen llm applications via multi-agent conversation framework." arXiv preprint arXiv:2308.08155 (2023).
[8] White, David H. "A theory for the origin of a self-replicating chemical system. I: Natural selection of the autogen from short, random oligomers." Journal of molecular evolution 16 (1980): 121-147.
[9] Bersenev, Dennis, Ayako Yachie, and Sucheendra K. Palaniappan. "Replicating a High-Impact Scientific Publication Using Systems of Large Language Models." bioRxiv (2024): 2024-04.
[10] Zhu, Andrew, Liam Dugan, and Chris Callison-Burch. "ReDel: A Toolkit for LLM-Powered Recursive Multi-Agent Systems." arXiv preprint arXiv:2408.02248 (2024).
[11] Berti, Alessandro, et al. "Re-Thinking Process Mining in the AI-Based Agents Era." arXiv preprint arXiv:2408.07720 (2024).
[12] Kumar, Aathira Anil. A comparative study between humans and GPT for TradeSpace analysis. MS thesis. University of Georgia, 2024.
[13] Zhuge, Mingchen, et al. "Language agents as optimizable graphs." arXiv preprint arXiv:2402.16823 (2024).
[14] Singh, Aditi, et al. "Enhancing AI Systems with Agentic Workflows Patterns in Large Language Model." 2024 IEEE World AI IoT Congress (AIIoT). IEEE, 2024.
[15] Gimeno-Ballester, Vicente, and Cristina Trigo-Vicente. "[Translated article] The role of artificial intelligence in scientific publishing: perspectives from hospital pharmacy." Farmacia Hospitalaria (2024).
[16] Pichlmair, Martin, Riddhi Raj, and Charlene Putney. "Drama Engine: A Framework for Narrative Agents." arXiv preprint arXiv:2408.11574 (2024).
[17] Ma, Kangyong. "Application of Self-Evolving AI Agents in Chemical Research: A Novel Intelligent Assistance System." (2024).
[18] Mishra, Nishikant & Kumar, Vikas & Chan, Felix. (2012). A Multi-Agent Architecture for Reverse Logistics in Green Supply Chain. International Journal of Production Research. 50. 2396-2406. 10.1080/00207543.2011.581003.
[19] Kasneci, Enkelejda, et al. "ChatGPT for good? On opportunities and challenges of large language models for education." Learning and individual differences 103 (2023): 102274.
[20] Chang, Yupeng, et al. "A survey on evaluation of large language models." ACM Transactions on Intelligent Systems and Technology 15.3 (2024): 1-45.
[21] Zhao, Wayne Xin, et al. "A survey of large language models." arXiv preprint arXiv:2303.18223 (2023).

[22] Teubner, Timm, et al. "Welcome to the era of chatgpt et al. the prospects of large language models." Business & Information Systems Engineering 65.2 (2023): 95-101.
[23] Chen, Mark, et al. "Evaluating large language models trained on code." arXiv preprint arXiv:2107.03374 (2021).
[24] Hoffmann, Jordan, et al. "Training compute-optimal large language models." arXiv preprint arXiv:2203.15556 (2022).
[25] Wei, Jason, et al. "Emergent abilities of large language models." arXiv preprint arXiv:2206.07682 (2022).
[26] Wen, Hao, et al. "Empowering llm to use smartphone for intelligent task automation." arXiv e-prints (2023): arXiv-2308.
[27] Ethape, Priya, et al. "Smart automation using llm." International Research Journal of Innovations in Engineering and Technology 7.11 (2023): 603.
[28] Morris, Clint, Michael Jurado, and Jason Zutty. "Llm guided evolution-the automation of models advancing models." Proceedings of the Genetic and Evolutionary Computation Conference. 2024.
[29] Guan, Yanchu, et al. "Intelligent Agents with LLM-based Process Automation." Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2024.
[30] Schwartz, Sivan, Avi Yaeli, and Segev Shlomov. "Enhancing trust in LLM-based AI automation agents: New considerations and future challenges." arXiv preprint arXiv:2308.05391 (2023).
[31] Pu, Hongxu, et al. "AutoRepo: A general framework for multimodal LLM-based automated construction reporting." Expert Systems with Applications 255 (2024): 124601.
[32] Ma, Pingchuan, et al. "InsightPilot: An LLM-empowered automated data exploration system." Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. 2023.
[33] Askarbekuly, Nursultan, and Nenad Aničić. "LLM examiner: automating assessment in informal self-directed e-learning using ChatGPT." Knowledge and Information Systems (2024): 1-18.
[34] Ma, Xinbei, Zhuosheng Zhang, and Hai Zhao. "Comprehensive Cognitive LLM Agent for Smartphone GUI Automation." arXiv preprint arXiv:2402.11941 (2024).
[35] Zhang, Zheying, et al. "LLM-based agents for automating the enhancement of user story quality: An early report." International Conference on Agile Software Development. Cham: Springer Nature Switzerland, 2024.
[36] Mawela, Chamith. A web-based solution for federated learning with LLM based automation. MS thesis. C. Mudiyanselage, 2024.
[37] Azzouzi, El Mehdi, Jeremy Bourdon, and Robert Plana. "Model-Driven Engineering for Optimal Project Delivery: Introducing the Project Delivery Model (PDM)." Saudi International Conference On Nuclear Power Engineering. Cham: Springer Nature Switzerland, 2023.
[38] Guo, Kai, and Limao Zhang. "Multi-objective optimization for improved project management: Current status and future directions." Automation in Construction 139 (2022): 104256.
[39] Niemeier, Debbie A., et al. "Optimization models for transportation project programming process." Journal of transportation Engineering 121.1 (1995): 14-26.
[40] Yarlagadda, Ravi Teja. "Python engineering automation to advance artificial intelligence and machine learning systems." International Journal of Innovations in Engineering Research and Technology [Ijiert] (2018).
[41] Heidrich, Benedikt, et al. "pyWATTS: Python workflow automation tool for time series." arXiv preprint arXiv:2106.10157 (2021).
[42] Johnson, Jami L., Henrik tom Wörden, and Kasper van Wijk. "PLACE: an open-source python package for laboratory automation, control, and experimentation." Journal of laboratory automation 20.1 (2015): 10-16.
[43] Ramachandran, Prabhu. "automan: A python-based automation framework for numerical computing." Computing in Science & Engineering 20.5 (2018): 81-97.
[44] Miller, Clayton, Christian Hersberger, and Marcus Jones. "Automation of common building energy simulation workflows using Python." (2013).
[45] Pajankar, Ashwin. Python Unit Test Automation: Practical Techniques for Python Developers and Testers. Apress, 2017.
[46] Choi, Brendan. "Introduction to python network automation." Introduction to Python Network Automation Volume I-Laying the Groundwork: The Essential Skills for Growth. Berkeley, CA: Apress, 2024. 1-44.
[47] Khalid, Muhammad Usman, et al. Python based power system automation in PSS/E. Lulu. com, 2014.
[48] Chou, Eric, Michael Kennedy, and Mandy Whaley. Mastering Python Networking: Your one-stop solution to using Python for network automation, programmability, and DevOps. Packt Publishing Ltd, 2020.
[49] Walker, Clinton, et al. "Forensic Analysis of Artifacts from Microsoft's Multi-Agent LLM Platform AutoGen." Proceedings of the 19th International Conference on Availability, Reliability and Security. 2024.
[50] Barbarroxa, Rafael, et al. "Benchmarking AutoGen with different large language models." 2024 IEEE Conference on Artificial Intelligence (CAI). IEEE, 2024.