An Intelligent LLM-Powered Personalized Assistant for Digital Banking Using LangGraph and Chain of Thoughts

1st Arafat Md Easin
Data Science and Engineering Dept.,
Eötvös Loránd University (ELTE)
Budapest, Hungary
arafatmdeasin@inf.elte.hu

2nd Saha Sourav
Dept. of Computer Science and Engineering,
United International University (UIU)
Dhaka, Bangladesh
research.srv.sh@gmail.com

3rd Orosz Tamás
Data Science and Engineering Dept.,
Eötvös Loránd University (ELTE)
Budapest, Hungary
orosztamas@inf.elte.hu

2024 IEEE 22nd Jubilee International Symposium on Intelligent Systems and Informatics (SISY) | 979-8-3503-8560-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/SISY62279.2024.10737601

***Abstract*—Text-based applications and chatbots are increasingly popular for delivering banking services and educational tools, offering convenient and efficient solutions for users. Whereas, personalized assistants have transformed user engagement in the digital banking space by utilizing Large Language Models (LLMs) in conjunction with autonomous agents. This study proposes the development of an intelligent personalized assistant for digital banking, utilizing a multi-agent framework based on the LangGraph and Chain of Thoughts (COT) prompting. While COT guarantees context-aware replies, the LangGraph design maps characteristics to nodes to improve user interactions. The objectives of this system are to enhance task efficiency and elevate the capabilities of digital banking assistants. We present a customizable digital banking system powered by LLM-based models, designed to deliver an interactive and personalized banking experience. The system supports a range of services, including adding money, transferring funds, paying bills, accessing telco services like mobile recharge, managing savings interest rates, DPS schemes, fixed deposits, and answering FAQs related to banking information. Therefore, integrating COT for logical reasoning enhances the effectiveness of multi-agent systems, as each single agent benefits from the structured reasoning process. In addition, LangGraph is employed for structured data management, enabling the assistant to support and accelerate various digital banking processes efficiently. The code implementation of this work is available for public access at: https://github.com/srv-sh/digital_agent.**

***Index Terms*—Large Language Models, Autonomous Agents, Personalized Assistant, LangGraph, Chain of Thoughts**

I. Introduction

In recent years, the development of Large Language Models (LLMs) has created new opportunities for the evolution of intelligent personal assistants (IPAs), demonstrating their ability to tackle growing issues in digital banking and financial services [1], [2]. The field of autonomous problem solvers or agents driven by LLMs is gaining popularity due to its wide range of banking applications. With personalization intelligent systems, LLMs have the potential to revolutionize human-machine interaction. Moreover, from a language agent perspective, unlike standard banking applications, LLMs encourage active user engagement by providing customized and engaging user experiences via real-time replies and natural language interactions [3], [4].

Nowadays, LLMs have also gained significant attention, particularly since OpenAI [5] released its now-popular GPT models. Nowadays, specific LLMs trained for code production have emerged, including Anthropic's Claude, Facebook's CodeLlama [6], and OpenAI's Codex, which powers Github's Copilot [7]. In comparison to traditional methods, LLMs have shown unique capabilities like good generalization even in the absence of explicit training, common sense reasoning, and instruction following. These amazing skills were achieved through unsupervised learning on enormous datasets of over 1.4 trillion words, followed by rigorous fine-tuning based on human feedback. By using these advantages, researchers have successfully incorporated large language models to empower autonomous agents, also known as LLM agents [8]. These agents have been designed to take on complex tasks by using resources such as code interpreters and third-party APIs, and by independently coming up with strategies to solve problems and achieve their objectives effectively.

Artificial Intelligence (AI) applications are now being developed more rapidly than possible due to the interaction with an LLM. In earlier methods, LLMs were given zero-shot or few-shot samples to prompt them [9], [10]. To improve text-based reasoning, new approaches for prompting LLMs systematically have been developed including ReAct [11], Reflexion [12], Tree of Thought (TOT) [13], Graph of Thought (GOT) [14], and Chain of Thought (COT) [15]. In addition, AutoGPT, BabyAGI, LangChain, and Llama-index are examples of single-agent systems [16]–[18] that use LLMs for a variety of functionalities such as tool usage, function calling, and embodied activities. Furthermore, several LLMs take on different functions in multi-agent frameworks [19]–[21], facilitating cooperative problem-solving and natural language communication. This dynamic enables their application across domains such as banking, transactions, finance, telecom, and related fields, which enhances their usefulness and impact.

Alongside this, the development of AI applications using LLMs has been made easier with the recent appearance of the open-source software library LangChain, which provides all-inclusive solutions for all stages involved. Its objective is to make it easier for developers to communicate with other apps and integrate more data sources. LangChain offers components, which are modular abstractions, and chains, which are customized pipelines that use case-specific data. LangChain also includes numerous classes for creating prompts using the specific customized prompt template, which consists of a text string that can include parameters supplied by the user to build a prompt, and is a repeatable technique for creating prompts.

Currently, Langchain AI has introduced LangGraph, an open-source framework for building multi-actor state LLM applications with cyclical operations. In many cases, Retrieval Augmented Generation (RAG) is used to develop single-agent systems, while LangGraph can facilitate the creation of multi-agent applications. However, its practical applicability has not been systematically studied. Undoubtedly, this collection has drawn much interest from the AI community. In this continuum, to further enhance logical reasoning within these multi-agent systems, we incorporate a Chain of Thoughts (COT), making the assistant more effective in handling complex tasks.

In this paper, we propose a multi-agent Chain of Thoughts (COT) architecture, where each node in the graph represents specific features essential for digital banking. These features include adding money, transferring funds, paying bills, accessing telco services like mobile recharge, savings interest rates, DPS schemes, fixed deposits, and providing answers to FAQs related to banking information. This multi-agent hierarchical architecture intends to create a customizable digital banking system that uses language models to deliver a highly interactive and personalized banking experience. The assistant will help numerous financial operations, improve the overall user experience, and speed up services by integrating COT for logical reasoning and issue resolution with LangGraph for structured data management.

II. Related Works

While Large Language Models (LLMs) continue to evolve rapidly, their customization possibilities have received less attention. Despite their limitations, such as occasionally generating incorrect or illogical outputs called hallucinations. Whereas, LLMs have achieved rapid success due to their strong performance and many people have started looking into how to use this technology to personalize decision-making in various fields, such as education, financial development, healthcare, security, and stock trading [11]–[21].

Using cutting-edge AI technologies, such as the LangChain NLP framework, Maldonado et al. [22] create functional requirements. Students, professors, and tutors/assistants are the three key players in the architecture established by this method, which also includes the OpenAI API. The performance of financial advisement systems in the customized finance space was studied by Kausik et al. [23], who emphasized the long-standing objective of financial inclusion by banks. This study evaluates two popular LLM-based chatbots, ChatGPT and Bard, and contrasts their results with that of SafeFinance, a rule-based chatbot developed on the Rasa platform.

Sleiman et al. [24] introduce LLMs as complex systems built from large text-trained models, interfaces, and action agents. They offer high-quality task performance but present risks in costs, accuracy, transparency, privacy, security, and ethics. The digital banking industry's response has been mixed, reflecting both potential and unresolved issues. To leverage these technologies, industry leaders should identify use cases, select simple solutions, design user-friendly experiences, build the right teams, and engage in regulatory discussions. In their discussion of the development of technology in banking, Dhake et al. [25] discuss terms like GenAI and LLM while also pointing out the shortcomings of the current financial system. They highlight how important it is for banks to use these technologies and go into detail about their limitations, problems, and uses. The authors conclude with observations on the revolutionary potential of these technologies in the banking industry. They also examine the future of AI and LLM while taking ethical considerations into account.

To improve efficiency and accuracy in structured finance, Xiangpeng et al. [26] describe the integration of artificial intelligence (AI) with traditional asset assessment methods. They show how AI can efficiently automate the information verification process between bank statements and loan applications using both open-sourced and closed-sourced large language models (LLMs). To provide personalized assistance, Yuanchu et al. [27] focus on Personal LLM Agents, which are agents built on LLMs that are intricately entwined with personal information and gadgets. They predict that shortly, Personal LLM Agents will become a major software paradigm for end users. To bring this vision to reality, they start conversations on several critical Personal LLM Agent areas, including architecture, capabilities, efficiency, and security.

Recently, Xinyi et al. [28] developed a multi-agent AI system powered by LLMs, called StockAgent. The purpose of this approach is to mimic how investors would trade in reaction to changes in the actual stock market. With StockAgent, users can explore how different outside variables affect investor trading and examine how trading behavior affects profitability. Later on, Galitsky et al. [30] address the issues of personalization and potential solutions for how personalization might function on top of LLMs. They explore the evolution and problems of existing personalization systems, the newly emerging capabilities of LLMs, and prospective ways to use LLMs for personalization. More recently, Mahyar et al. [30] presented openCHA, an open-source framework driven by LLM that aims to enable conversational agents to provide personalized answers to users' health-related questions. This framework makes the incorporation of external resources into LLM-based solutions, such as data sources, knowledge bases, and analysis models easier.

Though customization has been recognized as important in many real-world scenarios and has yielded insightful results, more attention must be paid to creating and assessing LLMs designed to provide personalized responses. Thus, a substantial amount of effort needs to be made to automatically develop the structure of these agents and successfully integrate them into an integrated community.

III. Problem Statement

Within the rapidly developing area of digital banking, clients want individualized and effective services catered to their own specific requirements and preferences. This degree of personalization and engagement is frequently beyond the capabilities of traditional banking systems, creating a gap between client expectations and service performance. Moreover, the complexities of financial products combined with the requirement for precise, real-time support presents significant challenges for banks hoping to improve consumer satisfaction and commitment. To address these issues, there is a need for an intelligent, LLM-powered personalized assistant that can effectively understand and respond to individual customer inquiries in real-time. With the use of cutting-edge technologies like Chain of Thoughts and LangGraph, this assistant can comprehend and analyze natural language to deliver accurate and contextually relevant information. The proposed system serves as a foundational approach with some limitations, and it is not yet ready for deployment in real-life applications. Currently, there are no existing state-of-the-art systems directly comparable to our approach. However, this work lays the groundwork for future developments in this domain and can guide further research in building more robust solutions.

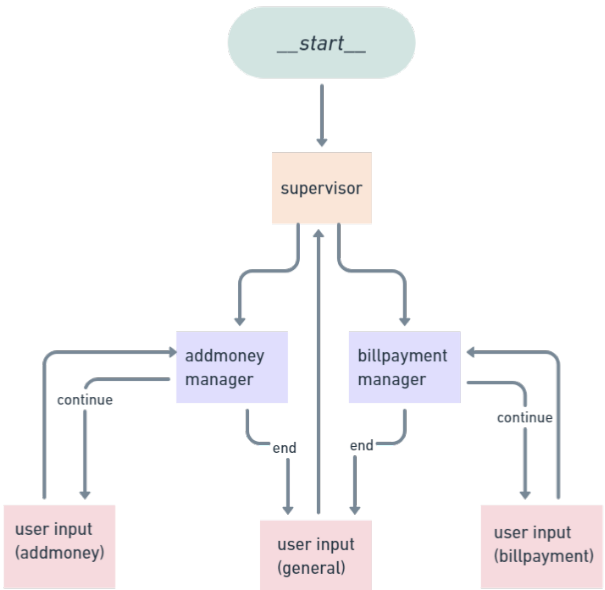

Fig. 1. Machine-Generated Sequence Diagram of the System

In addition to the add money manager, bill payment manager, and user input, Fig. 1 displays the system's machine-generated sequence diagram, which begins with a supervisor. The way the system works with the user to enable them to pay their bills and add money to their account is demonstrated. The subsequent section will explain the comprehensive process, along with several scenarios and relevant examples.

IV. Materials and Method

In this paper, we propose both single and multi-agent hierarchical LangGraph framework in which each node of the graph represents a feature, and each feature corresponds to specific agents of digital banking functionality.

We first developed a personalized single-agent framework (Shown in Fig. 2) designed to assist users with tasks such as adding money, transferring funds, paying bills, and using telco tools for recharges. When a user wants to perform any of the mentioned operations, the assistant receives prompts from the user to carry out the tasks. Based on the query, a JSON script is generated using the LLM-based single-agent chain with LangGraph. This function takes the graph state, formats it into a prompt, and then calls an LLM to generate the best response. If a recursion limit is reached, it means the agent was unable to complete the task within the allocated number of steps, likely due to the assistant becoming confused by the prompt.

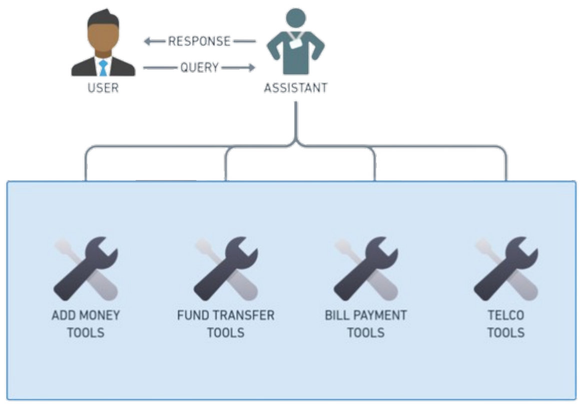

Fig. 2. Single Agent Architecture of the Proposed System

To ensure a comprehensive approach, we merged the GPT-3.5 model into our multi-agent framework. This integration allows the system to handle the entire set of tasks using individual agents, each leveraging the advanced capabilities of GPT-3.5. By distributing the workload among these agents, we significantly reduce the risk of bias when receiving and processing prompts from the user. As shown in Fig. 3, each functionality is managed separately, resulting in an enhanced user experience. The tasks are performed gradually by dedicated assistants, who then pass the results to the main agent, which in turn communicates with the user. Additionally, our multi-agent framework provides a secure tools service, including FAQs, to offer users information on various banking topics such as required savings to earn specific interest amounts, DPS schemes and their benefits over time, optimal closure periods for maximum benefits, and fixed deposit thresholds for worry-free savings. Overall, this multi-agent system is designed to execute tasks more quickly and accurately in real-time, while providing the necessary FAQs.

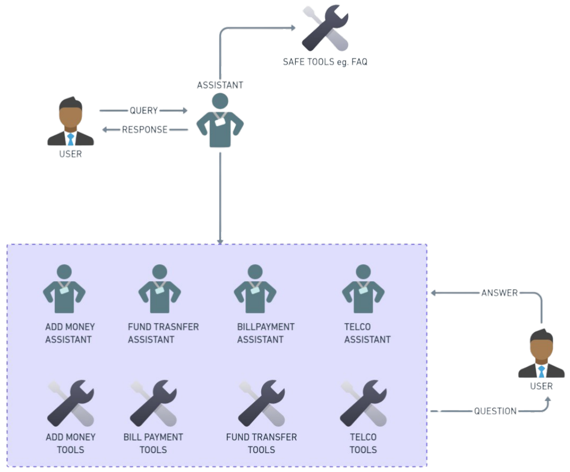

Fig. 3. Multi Agent Architecture of the Proposed System

V. Results and Discussion

In this section, we describe the "add money" and "bill payment" processes in greater detail, as they are similar to the other tasks to some extent. This paper focuses on developing these two tasks with proper functionality and relevant examples. However, we plan to add other functionalities, such as "fund transfer" and "telco tools," more robustly. Since the initial idea is working precisely, we can extend it to address specific gaps in the literature.

As depicted in Fig. 1, according to the machine-generated sequence diagram, the process begins with the supervisor, as named here from a technical perspective. The supervisor sends a message to the user input "add money". The user input "add money" receives the message and prompts the user to enter the amount of money they wish to add.

For instance, executing the "add money" process in the FinBuddy application (a fictional name representing a mobile banking service) from Kori (a digital bank), involves a series of prompts and responses as shown below:

1) The user initiates the "Addmoney" action in the FinBuddy app.
2) The supervisor sends a message to prompt the user for the FinBuddy account number.
3) User provides the FinBuddy account number:

```
Prompt: 'I want to add money from \\
my FinBuddy account to Kori Bank.'
{
    "action": "Addmoney",
    "action_input": {
        "bank_name": "FinBuddy",
        "from_no": "12345891011121",
        "amount": "",
        "kori_no": "",
        "pin": ""
    }
}
```

4) The assistant prompts the user to enter the amount of money they wish to add:

```
Prompt: 'Nice! Tell me FinBuddy's \\
account number.'
{
    "action": "Addmoney",
    "action_input": {
        "bank_name": "Finbuddy",
        "from_no": "12345891011121",
        "amount": "",
        "kori_no": "",
        "pin": ""
    }
}
```

5) User specifies the amount to be added (e.g., 1000 euro):

```
Prompt: 'Now tell me how much money \\
you want to add to your Kori account?'
{
    "action": "Addmoney",
    "action_input": {
        "bank_name": "Finbuddy",
        "from_no": "12345891011121",
        "amount": "1000 euro",
        "kori_no": "",
        "pin": ""
    }
}
```

6) The assistant presents options for selecting the destination account or entering a full account number:

```
Prompt: 'Now, select the destination \\
account from the options below or \\
type the full account number:
1> 000000008
2> 000000009
3> 000000021
4> 000000069
5> 000000080
{
    "action": "Addmoney",
    "action_input": {
        "bank_name": "Finbuddy",
        "from_no": "12345891011121",
        "amount": "1000",
        "kori_no": "000000080",
        "pin": ""
    }
}
```

7) User provides the PIN for authentication:

```
{
    "action": "Addmoney",
    "action_input": {
```

```
        "bank_name": "Finbuddy",
        "from_no": "12345891011121",
        "amount": "1000",
        "kori_no": "000000080",
        "pin": "12345"
    }
}
Response: 'Congratulations! Your transaction
has been completed.'
```

To summarize, the process begins with the user entering an amount, which is then sent to the addMoney function for processing. The add money function, managed by the add money manager, receives this amount and credits it to the user's account. Subsequently, the manager sends a confirmation message to the supervisor, signaling the completion of the transaction. Upon receiving this message, the supervisor concludes the process.

The same process is followed for bill payment (user inputs bill payment and the bill payment manager handles it). Here, we pay the water bill directly to the WASA (Water Supply and Sewerage Authority) from the Kori digital bank account. Here are some examples of how these tasks are performed, along with appropriate examples.

1) The user initiates the "Billpayment" action in the FinBuddy app.
2) The supervisor prompts the user to enter the bill number ('bll no'):

```
{
    "action": "Billpayment",
    "action_input": {
        "service_name": "water",
        "biller_name": "wasa",
        "bllr_accno": "",
        "bill_mobno": "",
        "bll_period": "",
        "bll_no": "",
        "amount": "",
        "bll_typ": "",
        "meter_no": "",
        "kori_account": "",
        "pin": ""
    }
}
```

3) User provides the bill number ('bll no'):

```
{
    "action": "Billpayment",
    "action_input": {
        "service_name": "water",
        "biller_name": "wasa",
        "bllr_accno": "",
        "bill_mobno": "",
        "bll_period": "",
        "bll_no": "T23060042982",
        "amount": "",
        "bll_typ": "",
        "meter_no": "",
        "kori_account": "",
        "pin": ""
    }
}
```

4) The assistant presents options for selecting the destination account or entering a full account number:

```
Prompt: 'Choose the account you \\
want to use to pay your bill from \\
the following options.
1> 000000008
2> 000000009
3> 000000021
4> 000000069
5> 000000080
{
    "action": "Billpayment",
    "action_input": {
        "service_name": "water",
        "biller_name": "wasa",
        "bllr_accno": "",
        "bill_mobno": "",
        "bll_period": "",
        "bll_no": "T23060042982",
        "amount": "",
        "bll_typ": "",
        "meter_no": "",
        "kori_account": "000000080",
        "pin": ""
    }
}
```

5) User provides the PIN for authentication:

```
{
    "action": "Billpayment",
    "action_input": {
        "service_name": "water",
        "biller_name": "wasa",
        "bllr_accno": "",
        "bill_mobno": "",
        "bll_period": "",
        "bll_no": "T23060042982",
        "amount": "",
        "bll_typ": "",
        "meter_no": "",
        "kori_account": "000000080",
        "pin": "12345"
    }
}
```

VI. Conclusion

This paper presents a novel approach to digital banking by integrating Large Language Models (LLMs), LangGraph, and Chain of Thoughts (COT) to create a highly interactive and customized assistant. By mapping features to LangGraph nodes and using a Chain of Thoughts for logical reasoning, our system increases user interactions and the efficiency of banking activities including adding money, fund transfer, bill payment, cell recharge, and customer FAQs. The multi-agent, hierarchical design enables adaptation across different financial contexts, with the potential to improve customer experience, expedite service delivery, and assure resilience and security in digital banking operations. In the future, we hope to improve the system even further by adding a web interface and adding more features for greater endurance to handle tasks such as making reservations for hotels, flights, cars, and trips on behalf of users. We will also focus on enhancing the ethical deployment of the LLM-powered personalized assistant for digital banking. This includes developing advanced privacy-preserving techniques, integrating robust encryption protocols, and refining bias-mitigation algorithms. Additionally, we plan to implement more sophisticated explainable AI (XAI) methods and establish rigorous frameworks for transparency, accountability, and regulatory compliance. These advancements will be essential to ensure secure, fair, and trustworthy digital banking experiences.

References

[1] K. Lakkaraju, S. Jones, S. Vuruma, V. Pallagani, B. Muppasani, and B. Srivastava, "LLMs for Financial Advisement: A Fairness and Efficacy Study in Personal Decision Making," Proceedings Of The Fourth ACM International Conference On AI In Finance, pp. 100-107, 2023.

[2] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, et al., "A survey on large language model based autonomous agents," Frontiers of Computer Science, vol. 18, pp. 1-26, 2024.

[3] S. Dhake, L. Lassi, A. Hippalgaonkar, R. Gaidhani, and J. NM, "Impacts and Implications of Generative AI and Large Language Models: Redefining Banking Sector," Journal Of Informatics Education And Research, vol. 4, 2024.

[4] J. Sleiman, "Generative artificial intelligence and large language models for digital banking: First outlook and perspectives," Journal of Digital Banking, vol. 8, pp. 102–117, 2023.

[5] P. Ray, "ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," Internet of Things and Cyber-Physical Systems, vol. 3, pp. 121-154, 2023. [Online]. Available: https://doi.org/10.1016/j.iotcps

[6] Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X., Adi, Y., Liu, J., Remez, T., Rapin, J., et al., "Code llama: Open foundation models for code," arXiv preprint arXiv:2308.12950, 2023.

[7] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, et al., "Evaluating large language models trained on code," ArXiv preprint ArXiv:2107.03374, 2021.

[8] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou, et al., "The rise and potential of large language model based agents: A survey," ArXiv Preprint ArXiv:2309.07864, 2023.

[9] T. Kojima, S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," Advances in Neural Information Processing Systems, vol. 35, pp. 22199-22213, 2022.

[10] K. Deng, G. Sun, and P. Woodland, "Wav2Prompt: End-to-End Speech Prompt Generation and Tuning for LLM in Zero and Few-shot Learning," ArXiv Preprint ArXiv:2406.00522, 2024.

[11] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," ArXiv Preprint ArXiv:2210.03629, 2022.

[12] N. Shinn, B. Labash, and A. Gopinath, "Reflexion: an autonomous agent with dynamic memory and self-reflection," ArXiv Preprint ArXiv:2303.11366, 2023.

[13] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," Advances in Neural Information Processing Systems, vol. 36, 2024.

[14] M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk, et al., "Graph of thoughts: Solving elaborate problems with large language models," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, pp. 17682-17690, 2024.

[15] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. Le, D. Zhou, et al., "Chain-of-thought prompting elicits reasoning in large language models," Advances in Neural Information Processing Systems, vol. 35, pp. 24824-24837, 2022.

[16] K. Mei, Z. Li, S. Xu, R. Ye, Y. Ge, and Y. Zhang, "Aios: LLM agent operating system," ArXiv E-prints, pp. ArXiv-2403, 2024.

[17] Z. Liu, W. Yao, J. Zhang, L. Xue, S. Heinecke, R. Murthy, Y. Feng, Z. Chen, J. Niebles, D. Arpit, et al., "Bolaa: Benchmarking and orchestrating llm-augmented autonomous agents," ArXiv Preprint ArXiv:2308.05960, 2023.

[18] Z. Wu, S. Zheng, Q. Liu, X. Han, B. Kwon, M. Onizuka, S. Tang, R. Peng, and C. Xiao, "Shall We Talk: Exploring Spontaneous Collaborations of Competing LLM Agents," ArXiv Preprint ArXiv:2402.12327, 2024.

[19] Q. Wu, G. Bansal, J. Zhang, Y. Wu, S. Zhang, E. Zhu, B. Li, L. Jiang, X. Zhang, and C. Wang, "Autogen: Enabling next-gen LLM applications via multi-agent conversation framework," ArXiv Preprint ArXiv:2308.08155, 2023.

[20] Y. Dong, S. Ding, and T. Ito, "An Automated Multi-Phase Facilitation Agent Based on LLM," IEICE Transactions on Information and Systems, vol. 107, pp. 426-433, 2024.

[21] S. Hong, X. Zheng, J. Chen, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. Yau, Z. Lin, L. Zhou, et al., "Metagpt: Meta programming for multi-agent collaborative framework," ArXiv Preprint ArXiv:2308.00352, 2023.

[22] P. Neira-Maldonado, D. Quisi-Peralta, J. Salgado-Guerrero, J. Murillo-Valarezo, T. Cárdenas-Arichábala, J. Galan-Mena, and D. Pulla-Sanchez, "Intelligent educational agent for education support using long language models through Langchain," International Conference On Information Technology & Systems, pp. 258-268, 2024.

[23] K. Lakkaraju, S. Jones, S. Vuruma, V. Pallagani, B. Muppasani, & B. Srivastava. "LLMs for Financial Advisement: A Fairness and Efficacy Study in Personal Decision Making." Proceedings Of The Fourth ACM International Conference On AI In Finance, pp. 100-107 (2023).

[24] J. Sleiman. "Generative artificial intelligence and large language models for digital banking: First outlook and perspectives." Journal Of Digital Banking **8**, 102-117 (2023).

[25] S. Dhake, L. Lassi, A. Hippalgaonkar, R. Gaidhani, & J. NM. "Impacts and Implications of Generative AI and Large Language Models: Redefining Banking Sector." Journal Of Informatics Education And Research **4** (2024).

[26] X. Wan, H. Deng, K. Zou, & S. Xu. "Enhancing the Efficiency and Accuracy of Underlying Asset Reviews in Structured Finance: The Application of Multi-agent Framework." ArXiv Preprint ArXiv:2405.04294 (2024).

[27] Y. Li, H. Wen, W. Wang, X. Li, Y. Yuan, G. Liu, J. Liu, W. Xu, X. Wang, Y. Sun, et al. "Personal llm agents: Insights and survey about the capability, efficiency and security." ArXiv Preprint ArXiv:2401.05459 (2024).

[28] X. Liu, C. Zhang, M. Jin, Z. Zhang, Z. Wang, D. Shu, S. Zhu, S. Li, M. Du, & Y. Zhang. 11When AI Meets Finance (StockAgent): Large Language Model-based Stock Trading in Simulated Real-world Environments." Preprints (2024).

[29] B. Galitsky. "LLM-Based Personalized Recommendations in Health." Preprints (2024).

[30] M. Abbasian, I. Azimi, A. Rahmani, & R. Jain. "Conversational health agents: A personalized LLM-powered agent framework." ArXiv Preprint ArXiv:2310.02374 (2023).