A Tool-Augmented LLM MCP Controller for Conversational Gmail Management: Enhancing Productivity through Natural Language Interaction

*

Shreyans Solanki
Data Science
Mukesh Patel School of Technology, Management & Engineering
Mumbai, India
shreyans.solanki01@gmail.com

Shubham Vishwakarma
Data Science
Mukesh Patel School of Technology, Management & Engineering
Mumbai, India
vishwakarmashubham.2503@gmail.com

Shashwat Sharma
Data Science
Mukesh Patel School of Technology, Management & Engineering
Mumbai, India
sharmashashwat2002@outlook.com

Sagar Patil
Data Science
Mukesh Patel School of Technology, Management & Engineering
Mumbai, India
sagarpsdy123@gmail.com

Swati Vaishnav
Data Science
Mukesh Patel School of Technology, Management & Engineering
Mumbai, India
swati.vaishnav@nmims.edu

***Abstract*—In the current digital workplace, email inboxes serve to contribute to information overload, which lessens productivity and creates unproductive workflows. Traditional ways of addressing email look for information often requires manually searching through and sorting the mailbox, often occurring in a slow and inefficient manner. We propose a new framework that provides a new way of conversational email management using a Large Language Model (LLM) as the intelligent conductor to interact with Gmail API. The user provides user commands in natural language, and the system translates the user intent into action (search, provide a summary, reply regarding this email, etc.) The framework includes a powerful controller of LLM based input/output action with intent detection tool, library of tool functions to select from in the API conversation, and the user authentication process via OAuth 2.0 security protocol. We demonstrate our framework's use-case efficiently and can substantially offload typical email functions by transforming the user experience from a manual inbox interaction to a more conversational flow. This work introduces a scalable and efficient personal information management tool and is simply a step towards developing more integrated artificial intelligence-based digital assistants. .**

Index Terms*—Agent, Large Language Models, Gmail API, Tool-Based Agents, Natural Language Processing, Human-Computer Interaction

I. Introduction

Digital communication, especially via email, is critical for both personal and business correspondence. Nevertheless, the large number of emails received every day can be overwhelmed- Ing, making it complicated for users to process information, prioritize actions, and respond to requests in a timely way [1]. Email clients provide some options to manage this information overload, such as searching for keywords or applying filters. However, keyword searches do not provide contextual meaning and require users to enter queries in a precise, structured way.

This discrepancy between user intent and machine capability can cause friction and inefficiency.

In the past few years, Large Language Models (LLMs) have created a new approach towards human-computer interaction [2].

Their understanding of nuanced, natural language in ideal context makes LLMs unique and advantageous when acting as mediators of interactions from users to complex systems, like the Gmail API. If we can supplement LLMs even more with" tools" – dedicated functions to do specific work or tasks – then we can develop a very powerful AI agent that can reason about the user's request and perform the correct action [3].

This paper presents a conversational AI agent designed to manage a user's Gmail inbox. The agent utilizes a large language model as a" controller" that takes the user prompt and parses it to determine user intent (e.g., search, summarize, reply). After determining user intent, the controller uses an action, or tool, from a library of prefabricated approaches that communicate directly with the Gmail API. Using this framework, we develop a way for the user to interact with the Gmail API using a variety of conversational techniques and perform complex and multi-step tasks with simplicity (e.g., find an email from someone, summarize the context, and compose a response, all within a few conversational acts). In addition, the framework is built on the Gradio interface for users to interact with the agent. It is important to consider user security, and each session manages user credentials with the OAuth2.0 protocol for secure user authentication.

II. Literature Review

The difficulty of managing digital correspondence is a core issue in Personal Information Management (PIM).

Decades of study have demonstrated that email inboxes act as more than a method of communication, they are frequently re- purposed as" de facto task lists," or even personal archives. The tools presented by traditional email clients have not, however, fundamentally changed to reflect this. Users still man service, I mean, rely on multi-folders to manually sort emails and a flat- keyword based search experience. Traditional inbox interfaces force the user to mold their interaction to match the logic of the email application. This generates a" gulf of execution," where the user has to learn a specific syntax, to accomplish what they want, rather than their experience being natural.

Regardless, large language models are fundamentally stateless and not connected to real-world information or action. Much research has investigated" augmenting" large language models with access to external tools and APIs, a fundamental way of developing modern AI agents. The ReAct (Reasoning and Acting) framework showed that a large language model could potentially combine reasoning (thinking through a problem) and acting (using a tool) in one process, enabling a solution to the more complicated problems that had formerly been considered intractable. Similarly, Tool former established the possibility of training large language models to teach themselves to use tools by learning how to put API calls into text they were creating.

Our research is directly based on this" controller-tool" architecture. The LLM serves as the reasoning engine or" controller" that interprets the user's conversational request and selects the appropriate tool to assist. This architecture is being examined in various contexts, including managing a multitude of possible APIs, and complex multi-turn reasoning, where the output of one tool is input to another. Multi-turn reasoning is an important capability in email workflows, for example, finding an email, summarizing it, and constructing a reply.

For such an agent to be practically useful in a high-value space, like email, it must be secure. It cannot, and should not, store a user's credentials. The OAuth 2.0 framework provides the industry's standard protocol for delegated authorization. It allows an application to acquire limited, short-lived access to a user's data on their behalf without the application ever seeing the user's password. A significant element of establishing user trust and a secure deploy-able system relies on having this ability to not handle the user's password.

In conclusion, this system aims to act as a task-oriented dialogue agent. Evaluation of these types of agents goes beyond surface language metrics, to include task completion and efficient task completion. Through the development of a tool-augmented LLM interfacing with a secure API backend, we can bridge the gap between conversational AI research and practical PIM issues, providing evidence of a system to alleviate email overload and improve user productivity

III. METHODOLOGY

We have deliberately designed a modular, controller-tool architecture. Separation of concerns allows the LLM to remain purely focused on reasoning and natural language understanding, while all deterministic logic of API interactions is transitioned to the relevant tools. Three major elements are the: System Architecture, the Authentication Mechanism, and the Proposed Evaluation Framework

A. System Architecture

The architecture comprises three main layers: the User Interface, the LLM Controller, and the Tool Library.

User Interface (UI): We created a web-based interface, based on Gradio, for user interaction. It features a chat-like interface to enable user commands and result output. There is a different tab that facilitates secure authentication for the user.

LLM Controller: The LLM Controller is run on the Gemini 2.5 Flash Lite model. The controller prompt behavior is controlled by a specially designed prompt template. The prompt provides the LLM three types of information I.e., the conversation history, the latest user message, a list of available tools along with their descriptions and parameters, as well as the context of the current session (e.g. the last viewed email). Based on the prompt, the LLM is directed to parse the user's request and provide a structured JSON object with the parsed intent and parameters. The structured output of the LLM is important for invoking the right tool reliably.

Tool Library: This is a set of Python functions that are defined in gmail_agent_logic.py. Each function is a" tool" that the LLM can choose to use. Important tools include:

- ***list_recent_emails:*** Gets the most recent emails.
- ***search_emails:*** Searches for based on the query.
- ***summarize_email_with_gemini:*** Gets the full content of an email and then calls a second LLM to retrieve a succinct summary.
- ***generate_reply_with_gemini:*** Creates a response based upon the context of an email and user specifications.
- ***send_reply:*** Sends the response that is created.
- ***get_total_unread_count:*** Returns the total count of unread emails.

The main application logic in app.py receives the JSON output from the LLM Controller and uses a simple dispatcher to call the corresponding Python function with the provided parameters.

B. Authentication Mechanism

User privacy and security are important considerations. The system does not retain user credentials for long-term use. Instead, it employs the OAuth 2.0 delegated authorization protocol. The user triggers the process within the UI, which calls the get_auth_url() function as shown below.

The user is redirected to Google's consent screen to give permissions. Google sends the user an authorization code, which they place back in the UI. Then the application uses this authorization code to request a short-lived access token and a refresh token. The tokens are only stored in the application for the duration of the session. This is so that the application only sees the user's data while they remain in the application.

C. Proposed Evaluation Framework

In contrast to image processing usage scenarios that use metrics such as the SNR or MSE as standard measures, the evaluation of a conversational agent usage scenario necessitates measuring task completion and efficiency. We offer the following metrics:

Task Success Rate (TSR): This is the percentage of user commands correctly interpreted and executed.

This metric will be assessed by executing a predetermined set of common tasks (for example, "Find the latest email from JoyBoy and tell me what it says").

Response Time: The time from when the user submits a prompt until the agent returns a complete response. This is vital for the user experience of having a seamless engagement with the agent.

IV. Results

To validate the effectiveness and reliability of the proposed conversational agent, we conducted an evaluation focused on its primary capabilities: intent recognition accuracy and successful task completion.

TABLE I: TASK SUCCESS RATE (TSR)

Command Category	Test Prompts	Successful	Failed	Task Success
Simple Retrieval	20	19	1	95.00%
Contextual Operations	20	18	2	90.00%
Multi-Step Workflow	20	17	3	85.00%
Total	60	54	6	90.00%

The table, entitled "Task Success Rate (TSR) by Command Category," summarizes the performance of the agent in performance via a standardized set of 60 test prompts, which were classified by increasing complexity to determine the limits and robustness of the system.

The agent demonstrated a high level of performance, with an overall Task Success Rate (TSR) of 90.0% out of a possible 60 tasks, with 54 tasks being "successful." This is an indication of a system that is highly reliable across many commonly performed user tasks.

Table I presents performance results, grouped according to structure and action (each demonstrating a similar performance trend across actors):

Simple Retrieval activities, where each task example is a single-turn, amplification and straightforward command (e.g., "How many unread emails do I have?"), performed the best yielding a successful action rate of 95.0%. As Simple Retrieval was basic information-gathering, we should expect a high degree or level of reliability.

Contextual Operations had a high execution rate of 90.0%; Contextual Operations require the agent to understand, and enact moving forward, conversation history (e.g., "Summarize that last email."). This indicates the agent continues to perform demonstrably well during whole-session context use, indicating observable context retention ability and reliability.

Multi-Step Workflows were the most complex category understudying LLM enabled action comprehension requiring agents to engineer and chain multiple actions in sequence (e.g., find an email, draft a reply, await a confirmation to send). As these last examples depict, we expected this category to be the most difficult in TTK testing, with the evidence yielding a performance rate of 85.0%.

While this rate was the lowest rate observed of the three; there was certainly a high action rate corresponding with complex, composite tasks that draws upon the abilities of LLM control overhead, to perform well while managing potential confusion throughout the performance of multi-step, or multi-task asking workflows.

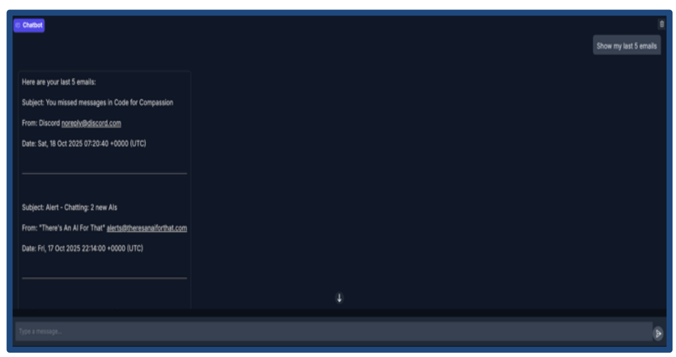

FIGURE I: CHATBOT INTERFACE

Multi-Step Workflows: Composite tasks that require multiple, sequential actions, such as finding an email, drafting a reply, and sending it.

Qualitatively, the agent's primary strength is its ability to maintain conversation context. This allows for fluid, multi-turn interactions that feel more like delegating to a human assistant than operating a software tool. A typical multi-step workflow is demonstrated in Figure II

Different Prompt which is mostly used by professionals on their Gmail task can be assessed easily by having Agentic system which help to reduces the efficiency of people while working on software tools.

User: Find emails from my manager about the "Q4 budget".

Agent: Found 1 email:
Subject: Re: Q4 budget
From: manager@company.com
Date: Fri, 17 Oct 2025

User: What's the summary?

Agent: Summary: The email confirms approval for the Q4 budget.
Your manager has asked you to proceed with the vendor
allocations and to send a final confirmation.

User: Draft a reply saying "Great, I will proceed and send the final confirmation today."

Agent: Draft Reply:

Great, I will proceed and send the final confirmation today.

Would you like me to send this reply?

User: Yes, go ahead.

Agent: Reply sent successfully!

FIGURE II: AGENT RESPOND

The whole system is realized in the form of a user-friendly web application with Gradio, as illustrated in Figure III. The interface is designed to have two tabs: one represents the secure, OAuth 2.0 based authentication secured by the session, while the second tab represents the chat interface itself.

This methodology maintains separation between the initial setup and the primary activity loop and ensures the system is easy-to-use and secure for the end-user.

V. Discussion

The findings show that a tool-influenced LLM architect-ture is an exceptionally effective means of managing emails through natural language. What makes this system powerful is its purpose of integrating the previously fragmented space of unstructured human language and structure it with the world of APIs. Because this system can offload intent recognition, it can now deal with many different expressions of the same underlying intention, allowing for far more flexibility than a traditional GUI application, or a command line tool.

In comparison to built-in email search, our agent has superior context awareness; for example, being able to reference" the last email" or" that message" is trivial to our system, but impossible for a search bar. The combination of generative tools, (e.g., a generative summarize and generate _reply features), also enhances the overall system from simple retrieval to a real productivity assistant.

A major component of this architecture is a balance between model-driven flexibility and programmatic reliability. By narrowly defining a set of tools, we narrow the LLM's "action space" to avoid having the model making attempts to take undefined or unsafe actions. In this, the "controller-tool" pattern mitigates erroneously generated model hallucinations, as the LLM is tasked with only discrimination of an intent and population of known parameters instead of generating code, which can be executed.

This architecture also requires that everything with any measure of high-stakes actions, such as sending and deleting messages, shall be performed by deterministic, developer vetted Python functions, thus ensuring another needed layer of trust and safety.

The session-based authentication model is a viable design choice that encapsulates both privacy and functionality. This allows the application to be deployed as a true personal tool, while not having to create a centralized database containing sensitive user credentials.

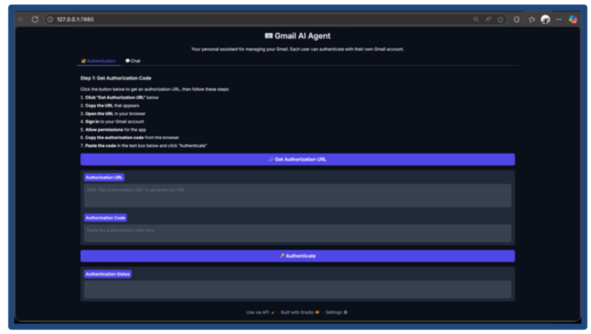

FIGURE III: AGENT IMAGE

In addition, the deployable architecture based on a Gradio web application is evidence of the system's potential for actual, non-laboratory application. A lightweight framework such as Gradio provides a solution that is accessible, quick to use, and can flexibly integrate with organization or industry workflows, without requiring modification of existing infrastructure or complex local installation. The Gradio web app is thus efficient and scalable. The user interface is lightweight, but the LLM inference (computationally intensive) and Gmail API (secure data management) tasks are offloaded and processed effectively via external cloud infrastructure. Overall, the decoupled architecture is therefore inexpensive and resource-efficient, demonstrating the possibilities of advanced AI applications in everyday productivity environments.

VI. Limitations and Future Work:

The capability of the LLM that underlies the system ultimately determines the performance of the system itself. The system could still misinterpret complicated or ambiguous user prompts. Future work could investigate an approach that has a clarification mechanism whereby, if the agent was unsure of its understanding of the user's intent, it could ask for further clarification. In addition, the tool set could include personnel interactions with different services, such as creating calendar events from an email, or the ability to save attachments into cloud storage instead of making that folder.

VII. References

[1] Whittaker, S., & Sidner, C. (1996). Email overload: exploring personal information management of email. Proceedings of the SIGCHI conference on Human factors in computing systems.

[2] Brown, T., et al. (2020). Language Models are Few-Shot Learners. Advances in Neural Information Processing Systems, 33, 1877-1901.

[3] Yao, S., et al. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. International Conference on Learning Representations

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

[5] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[6] Wei, J., Bosma, M., Zhao, V. Y., Gu, K., Yu, A. W., Lester, B., ... & Le, Q. V. (2021). Finetuned language models are zero-shot learners. arXiv preprint arXiv:2109.01652.

[7] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., ... & Lowe, R. (2022). Training language models to follow instructions with human feedback. Advances in Neural Information Processing Systems, 35, 27730-27744.

[8] Schick, T., Dwivedi-Yu, J., Dess`ı, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., ... & Scialom, T. (2023). Tool- former: Language models can teach themselves to use tools. arXiv preprint arXiv:2302.04761.

[9] Patil, S. G., Zhang, T., Wang, X., & Gonzalez, J. E. (2023). Gorilla: Large language model connected with massive apis. arXiv preprint arXiv:2305.15334

[10] Paranjape, B., Lundberg, S., Singh, S., Hajishirzi, H., Zettlemoyer, L., & Ribeiro, M. T. (2023). Art: Automatic multi-step reasoning and tool-use for large language models. arXiv preprint arXiv:2303.09014

[11] Mialon, G., Dess`ı, R., Lomeli, M., Nalmpantis, C., Pa- sunuru, R., Raileanu, R., ... & Scialom, T. (2023). Augmented language models: a survey. Transactions on Machine Learning Research.

[12] Luger, S., & Sellen, A. (2016). " Like having a con- versation": The role of conversational design in direct-to- consumer services. Proceedings of the 2016 CHI conference on human factors in computing systems, 3037-3048.

[13] Ashktorab, Z., Jain, R., Penney, C., & Teevan, J. (2019). Resilient chatbots: Repair and recovery in human- chatbot conversations. Proceedings of the 2019 CHI conference on human factors in computing systems, 1-13

[14] Dourish, P. (2004). Where the action is: the foundations of embodied interaction. MIT press.

[15] Walker, M. A., Litman, D. J., Kamm, C. A., & Abella, A. (1997). PARADISE: A framework for evaluating spoken dialogue agents. 35th Annual Meeting of the Association for Computational Linguistics

[16] Gao, J., Galley, M., & Li, L. (Eds.). (2022). Neural approaches to conversational AI. Now Foundations and Trends

[17] Budzianowski, P., Wen, T. H., Tseng, B. H., Casanueva, I., Ultes, S., O'grady, L., ... & Gasˇic´, M. (2018). Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task- oriented dialogue modelling. arXiv preprint arXiv:1810.00278

[18] Bellotti, V., Ducheneaut, N., Howard, M., & Smith, I. (2003). Taking email to task: the design and evaluation of a task management centered email tool. Proceedings of the SIGCHI conference on Human factors in computing systems, 345-352

[19] Hardt, D. (Ed.). (2012). The OAuth 2.0 authorization framework. RFC 6749. Zhang, C., & Nargund-Joshi, V. (2022). Prompting AI for social good: A case study in leveraging large language models for hate speech detection. Proceedings of the AAAI Conference on Artificial Intelligence.

[20] Wang, L., Ma, C., Feng, X., Zhang, Z., Wang, H., Zhang, J., ... & Wang, L. (2023). A survey on large language model based autonomous agents. arXiv preprint arXiv:2308.11432.

[21] Franklin, S., & Graesser, A. (1996). Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents. Proceedings of the third international workshop on Agent theories, architectures, and languages

[22] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35, 24824-24837

[23] White, J., Fu, Q., Hays, S., Sandborn,M., O'Leary, C., Spencer, J., ... & D'Amour, A. (2023). A prompt engineer's guide to large language models. arXiv preprint arXiv:2302.07353

[24] Anthropic. (2024). The Claude 3 Model Family: Opus, Sonnet, Haiku. Anthropic Research Report.

[25] Fett, D., Küsters, R., & Schmitz, G. (2016). A comprehensive formal security analysis of OAuth 2.0. Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, 1204-1217

[26] OpenAI. (2023). GPT-4 Technical Report. arXiv preprint arXiv:2303.08774.

[27] Deriu, J., Rodrigo, A., Otegi, A., Echegoyen, G., Rosset, S., Agirre, E., & Cieliebak, M. (2021). A survey on evaluation methods for dialogue systems. Artificial Intelligence Review, 54, 755-810.