

Design and Implementation of a Telegram-Based Conversational AI Agent for Automated Personal Expense Management Using Large Language Models and Google Sheets

E Channaveeramma¹

¹Department of Electronics and Communication Engineering, Navodaya Institute of Technology, Raichur, Karnataka, India

Emails: channaveeramma.e@navodaya.edu.in¹

Abstract

Manual logging of day-to-day personal expenses is often abandoned by users because it requires them to open a dedicated spreadsheet or application and enter data in a rigid, structured format. This paper presents the design and implementation of a conversational Artificial Intelligence (AI) agent that allows a user to record and retrieve personal expenses simply by chatting in natural language through the Telegram messaging platform. The system is built using the n8n low-code automation platform and combines a Telegram Trigger node, an AI Agent powered by an OpenAI Chat Model, a Simple Memory module for conversational context, and two Google Sheets tools, namely Read Expenses and Add Expense, that allow the agent to write new expense records to, and retrieve existing records from, a connected Google Sheet. When a user sends a message such as "I spent Rs.231 for Medicine today," the agent extracts the expense description, amount, and date, appends the entry to the spreadsheet, and returns a natural-language confirmation. The agent can also generate an on-demand categorized daily summary. The proposed system was implemented and tested with real conversational data, and the results demonstrate that the agent accurately parses free-form expense statements, updates the backend spreadsheet correctly, and produces coherent, human-readable summaries, thereby lowering the friction traditionally associated with manual expense tracking.

Keywords: Conversational AI Agent; Telegram Bot; Large Language Model; Workflow Automation; n8n; Google Sheets; Personal Finance Management

1. Introduction

Personal financial well-being begins with awareness of daily spending, yet a large fraction of individuals abandon manual expense-tracking habits within a few weeks because the process interrupts their primary task, requires switching applications, and demands rigid data entry [7], [9]. Existing mobile expense-tracker applications address part of this problem through reminders, visual dashboards, and gamification, but they still require the user to open a dedicated interface every time an expense occurs [7]. Messaging platforms such as Telegram are already open on a user's device throughout the day and support programmable bots through the Telegram Bot API [8], [9]. When combined with a Large Language Model (LLM), a Telegram bot can move beyond rigid command syntax and understand free-form natural-language statements such as "I spent Rs.231 for Medicine today." Recent advances in LLM-based conversational agents have shown that

such systems can reliably interpret user intent, extract structured information from unstructured text, and invoke external tools to complete a task [1], [2]. This paper proposes a Telegram-based AI expense-management agent that is orchestrated using the open-source, low-code workflow automation platform n8n [3]–[5]. The workflow receives a message from the user, allows an AI Agent to interpret it with the help of an OpenAI Chat Model and a Simple Memory module, and lets the agent invoke one of two Google Sheets tools — Read Expenses or Add Expense — to interact with a spreadsheet acting as the persistent data store. The main contributions of this work are: (i) a working architecture that couples an LLM-driven conversational agent with a no-code automation platform and a spreadsheet backend for personal expense tracking; (ii) a demonstration, using real conversational logs, of accurate natural-language

parsing and automatic categorized summarization; and (iii) a discussion of the advantages and limitations of such chat-first automation for everyday personal-finance use cases. The related work is discussed in Literature Review section II, Proposed system architecture is discussed in section III, Implementation details in section IV, Results and discussion in section V and Conclusions and Futurescope in section VI.

2. Literature Review

LLM-based conversational agents have rapidly evolved from simple rule-based chatbots into sophisticated systems capable of understanding context, extracting structured information, and invoking external tools; Dam et al. provide a comprehensive survey of this evolution and its application across multiple industries [1], while Zhu et al. distinguish LLM chatbots from more capable LLM-based AI agents that can plan and act using external tools, which is the category the proposed expense agent belongs to [2]. On the automation side, n8n has emerged as a widely used open-source, low-code platform for connecting messaging services, spreadsheets, and AI models into a single workflow. Recent studies report that n8n workflows substantially reduce execution time and error rate compared with manual, human-driven processes, and that the platform is well suited to orchestrating LLM-based agentic pipelines that combine triggers, memory, and third-party tool nodes [3] – [5]. Prior work on Telegram bots has largely focused on customer support, academic notification, and file-handling use cases, using either rule-based logic or classical natural-language-processing pipelines to interpret user messages [6], [8], [9]. Separately, research on mobile personal-finance-management applications shows that ease of use and low data-entry friction are the strongest predictors of sustained user adoption [7]. The present work differs from these literature works by combining an LLM-based conversational agent, rather than a rule-based or NLP-pipeline chatbot, with a no-code orchestration platform and a spreadsheet backend, specifically to minimize the friction of everyday expense logging.

3. Proposed System Architecture

3.1. System Overview

The proposed workflow, illustrated in Fig. 1, follows

a linear trigger–process–respond pattern that is typical of n8n automations. A user sends a message through the Telegram application; the Telegram Trigger node captures the incoming update and forwards it to the AI Agent; the AI Agent consults the OpenAI Chat Model to interpret the request and the Simple Memory module to retain conversational context; depending on the interpreted intent, the AI Agent invokes either the Read Expenses tool or the Add Expense tool, both of which operate on a connected Google Sheet; and finally, the Send a Text Message node returns the AI Agent's response to the user on Telegram.

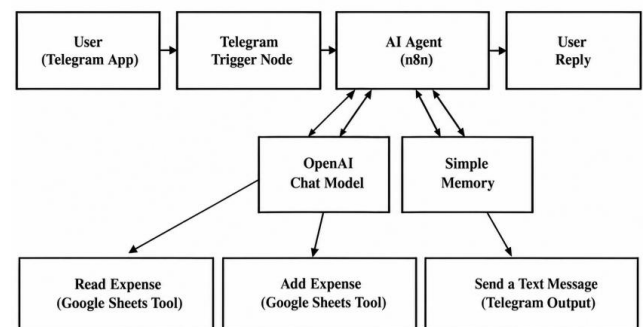


Figure 1 Architecture of the Telegram–AI Agent expense-management workflow

3.2. Component Description

Telegram Trigger: the entry point of the workflow, which continuously listens for incoming Telegram messages and passes each captured update to the AI Agent. **AI Agent:** the central decision-making node of the workflow. It interprets the user's natural-language message and decides which downstream tool, if any, should be invoked — for example, an Add Expense action for a statement describing new spending, or a Read Expenses action for a request to view past records. **OpenAI Chat Model:** supplies the language-understanding capability that the AI Agent relies on to interpret free-form requests, extract structured fields such as description, amount, and date, and compose natural responses. **Simple Memory:** retains recent conversational context so that the agent can correctly resolve follow-up messages, such as a user replying "Yes please" to a previous prompt asking whether a summary is wanted. **Read Expenses (Google Sheets tool):** retrieves expense

rows from the connected spreadsheet whenever the user asks to view, search, or summarize previously recorded expenses. Add Expense (Google Sheets tool): appends a new row containing the expense type, amount, and date to the spreadsheet whenever the AI Agent determines that the user is reporting a new expense. Send a Text Message: the output node that delivers the AI Agent's final natural-language response back to the user through Telegram, completing the interaction loop.

4. Implementation

4.1.Expense Recording Module

When a user sends a message describing a new expense, for instance "I spent Rs.231 for Medicine today" or "I spent Rs 70 for coconut water today," the AI Agent extracts the item description, the numeric amount, and the transaction date, and invokes the Add Expense tool to append the corresponding row to the Google Sheet. The agent then replies with a short natural-language confirmation, such as reporting that the expense has been added, and asks whether the user wants to record anything else. Fig. 2 shows a representative sequence of such interactions captured from the deployed Telegram bot.

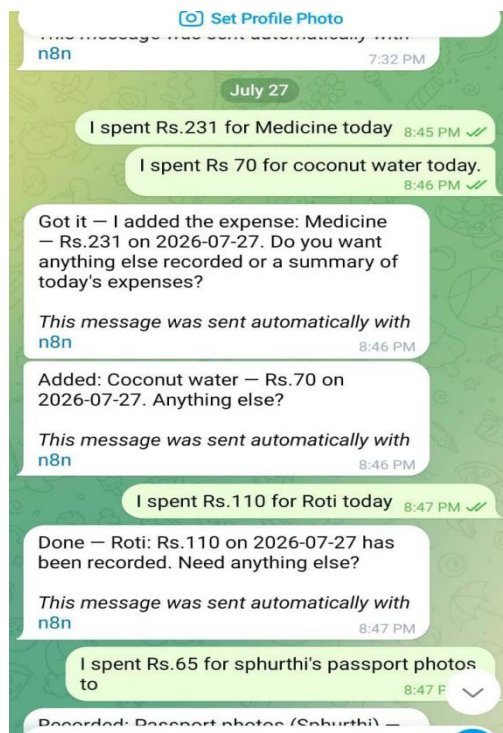


Figure 2 Sample conversation showing the agent recording multiple expenses in natural language

4.2.Expense Retrieval and Summarization Module

When a user asks for a summary, the AI Agent invokes the Read Expenses tool to retrieve the day's records from the spreadsheet, aggregates the amounts, and formats a categorized, itemized response together with a running total. Fig. 3 shows the agent generating a complete daily summary after a sequence of expenses were logged, including a computed total and a follow-up offer of further options such as a weekly summary or data export.

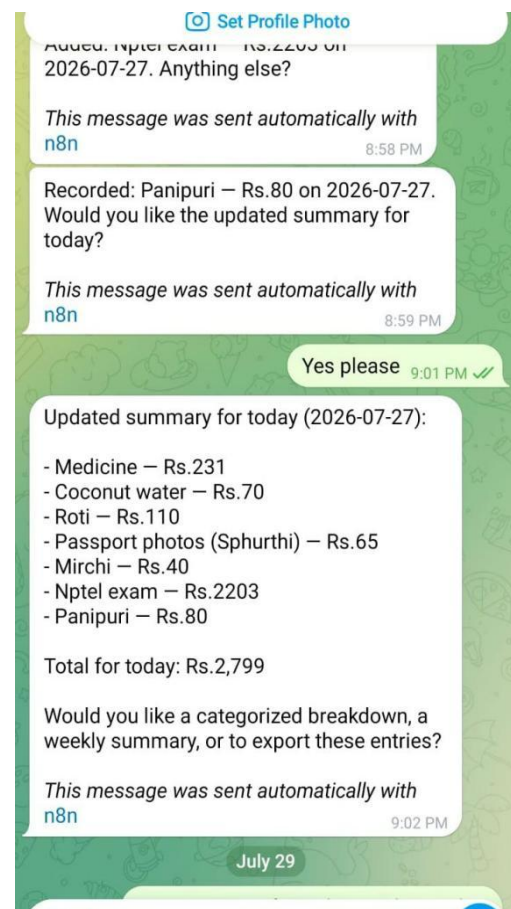
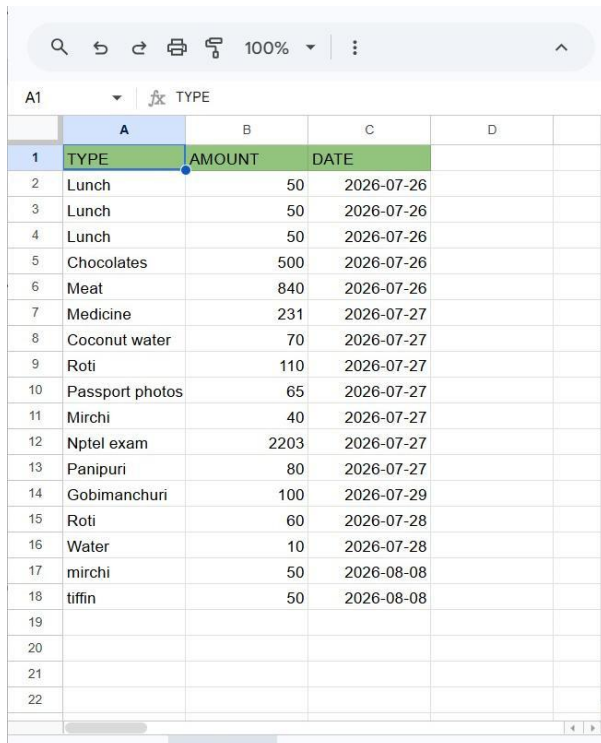


Figure 3 The AI Agent generating an on-demand categorized daily expense summary

Every entry recorded through the conversational interface is persisted as a row in a Google Sheet with the columns TYPE, AMOUNT, and DATE, which serves as the durable backend data store for the system and can be independently opened, filtered, or charted by the user at any time. Fig. 4 shows the resulting spreadsheet after several days of use.



	A	B	C	D
1	TYPE	AMOUNT	DATE	
2	Lunch	50	2026-07-26	
3	Lunch	50	2026-07-26	
4	Lunch	50	2026-07-26	
5	Chocolates	500	2026-07-26	
6	Meat	840	2026-07-26	
7	Medicine	231	2026-07-27	
8	Coconut water	70	2026-07-27	
9	Roti	110	2026-07-27	
10	Passport photos	65	2026-07-27	
11	Mirchi	40	2026-07-27	
12	Nptel exam	2203	2026-07-27	
13	Panipuri	80	2026-07-27	
14	Gobimanchuri	100	2026-07-29	
15	Roti	60	2026-07-28	
16	Water	10	2026-07-28	
17	mirchi	50	2026-08-08	
18	tiffin	50	2026-08-08	
19				
20				
21				
22				

Figure 4 Google Sheet backend showing recorded expense entries with TYPE, AMOUNT, and DATE columns

5. Results and Discussion

The system was exercised over several days of real usage. Table I reproduces a representative subset of the entries recorded in the backend Google Sheet, spanning multiple categories logged purely through natural-language Telegram messages.

Type	Amount (Rs.)	Date
Medicine	231	2026-07-27
Coconut water	70	2026-07-27
Roti	110	2026-07-27
Passport photos	65	2026-07-27
Mirchi	40	2026-07-27
Nptel exam	2203	2026-07-27
Panipuri	80	2026-07-27

Table 2 Sample Expense Records Retrieved from the Google Sheets Backend

Across the observed interactions, the AI Agent correctly parsed the expense description, amount, and date in every recorded message, including messages

phrased differently from one another (e.g., "I spent Rs.231 for Medicine today" versus "I spent Rs 70 for coconut water today"), which indicates a reasonable tolerance to natural variation in phrasing rather than a dependence on a single fixed sentence template. The Simple Memory module allowed short follow-up replies, such as "Yes please," to be correctly interpreted as a continuation of the immediately preceding request for a summary, rather than being treated as an unrelated new message. The generated daily summary correctly aggregated seven distinct entries into a total of Rs. 2,799, matching the sum of the individual amounts, and the corresponding rows were verified to be present with the correct values in the underlying Google Sheet. These observations suggest that coupling an LLM-based AI Agent with a no-code orchestration platform and a spreadsheet backend is a practical and low-friction approach to everyday personal expense tracking, since the user interacts entirely through ordinary chat messages without needing to learn a command syntax or open a separate application. A limitation of the current implementation is that expense categorization relies on the descriptive text supplied by the user rather than a fixed taxonomy, which can lead to inconsistent category labels (e.g., "Mirchi" appearing as a separate entry rather than being grouped under "Groceries"). Reliability is also dependent on the uptime and rate limits of the underlying Telegram, OpenAI, and Google Sheets APIs.

Conclusion And Future Scope

This paper presented the design and implementation of a Telegram-based conversational AI agent for automated personal expense tracking, built using n8n, an OpenAI Chat Model, a Simple Memory module, and Google Sheets Read Expenses and Add Expense tools. Testing with real conversational data showed that the agent could accurately record free-form expense statements and generate correct, coherent daily summaries, demonstrating that chat-first automation can meaningfully lower the friction of routine expense logging compared with traditional spreadsheet or app-based entry. Future work includes extending the agent with automatic category normalization, monthly and weekly summary generation, configurable budget alerts, category-wise spending analysis and visualization, multi-user

support with per-user sheets, and explicit write-verification and error-handling logic to improve reliability under API failures.

References

- [1].S. K. Dam, C. S. Hong, Y. Qiao, and C. Zhang, "A Complete Survey on LLM-based AI Chatbots," arXiv preprint arXiv:2406.16937, 2024.
- [2].J. Zhu, M. Zhu, R. Rui, R. Shan, C. Zheng, B. Chen, Y. Xi, J. Lin, W. Liu, R. Tang, Y. Yu, and W. Zhang, "Evolutionary Perspectives on the Evaluation of LLM-Based AI Agents: A Comprehensive Survey," arXiv preprint arXiv:2506.11102, 2025.
- [3].P. Venkiteela, "n8n: An Open-Source Workflow Automation Platform for Enterprise Integration and AI-Driven Orchestration," International Journal of Computer Applications, vol. 187, no. 63, pp. 1–11, Dec. 2025.
- [4].A. R. Amir et al., "Evaluating Workflow Automation Efficiency Using n8n: A Small-Scale Business Case Study," arXiv preprint arXiv:2602.01311, 2026.
- [5]. "Characterizing Large Language Model Agentic Workflows: A Study on the n8n Ecosystem," arXiv preprint arXiv:2606.29116.
- [6].S. Raj, Building Chatbots with Python: Using Natural Language Processing and Machine Learning, Apress, 2019.
- [7]. "Personal Finance Management Application," ResearchGate preprint, 2024.
- [8].IJRASET, "Developing a Telegram Chatbot using Python: A Comprehensive Study," International Journal for Research in Applied Science & Engineering Technology, 2025.
- [9]. "Development of Intelligent Telegram Chatbot Using Natural Language Processing," in Proc. IEEE Conference, 2021.