National College of Ireland

Context Aware Token Optimization for Personalized Financial Advisory with LLMs

MSc Research Project
FinTech

Biswadeep Sarkar
Student ID: 23315652

School of Computing
National College of Ireland

Supervisor: Prof. Brian Byrne

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Biswadeep Sarkar
Student ID:	23315652
Programme:	FinTech
Year:	2024
Module:	MSc Research Project
Supervisor:	Prof. Brian Byrne
Submission Due Date:	11/08/2025
Project Title:	Context Aware Token Optimization for Personalized Financial Advisory with LLMs
Word Count:	4269
Page Count:	15

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	8th August 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Context Aware Token Optimization for Personalized Financial Advisory with LLMs

Biswadeep Sarkar
23315652

Abstract

Generative AI has transformed both industry and academia. The extensive ability of these models to process, analyze, and generate context-aware responses has opened new horizons of possibilities across domains. In this paper, I have explored how Large Language Models (LLMs) can be used as an extended brain for financial advisors, enabling them to generate context-aware responses utilizing information previously shared by customers. While doing so, my goal was to optimize the input token count significantly. Following the work of previous researchers and after several iterations, I have created a multi-LLM financial advisor architecture powered by a multi-layer context extraction engine that generates context-aware responses with more than 92 percent reduction in input token count. The architecture employs both zero-shot and one-shot prompting techniques to balance generalization capabilities with contextual guidance. The system architecture is demonstrated through a Flask-based web application that mimics a WhatsApp-style chat interface using HTML, CSS, and JavaScript, with the backend implemented in Python. This efficiency translates to substantial savings in computational resources, energy consumption, and carbon emissions, highlighting the significant sustainability potential of the approach.

1 Introduction

Artificial Intelligence is one of the biggest discoveries in modern technology. From machine learning to the LLMs, generative AI has been transforming the entire world. This has not only increased the efficiency but also opened doors for new possibilities. Human behavior has also evolved significantly during this course. Initially, what used to be a generic product is now, in today's modern world, expected to be more customized or personalized. To make these possible, researchers and businesses around the world are exploring the possibility of harnessing more data in different formats.

Financial services as a business have always been a personalized solution due to the nature of this business, where the requirements vary person to person as per their needs. For this reason, the industry must depend on human resources. Human agents who act as the bridge between the customer and the company quickly start facing bandwidth limitations to process the customer data, even with modern CRMs, due to the varying and dynamic nature of the customer data.

Financial advisors, hence need a system that can help them record, analyses relevant customer information based on historical interactions and use that to provide highly personalized solutions. This leads to the idea of having an extended mind Clark and Chalmers (1998).

In this work, I have addressed the research question of how can LLMs help financial advisors leverage customer interaction history to enhance personalized service delivery. I have explored how efficiently we can use multi LLM orchestration to serve as an extended brain for the financial advisor. Extending the work by Hou et al. (2024) and Shahin et al. (2024), where they have shown how contextual data can be used to personalize communication using LLMs.

I have further extended this domain to address one of the massive challenges the entire industry is currently facing: optimizing the compute required for these services while maintaining the same quality. In this paper, I have created a multi-LLM orchestration using Gemini 1.5 flash as the base LLM model to extract key contextual information and generate a customer profile as a short-term memory for the LLM, which serves as the context to provide personalized financial advisory for the user. I am able to significantly reduce the number of tokens required to provide as the context from more than 17,000 to around 1200. This means a significant reduction in cost for every single LLM interaction, lower electricity consumption, and hence lower carbon emissions.

To showcase its effectiveness, I have developed a working prototype which demonstrates the readiness and efficacy of my orchestration to be implemented in a real-world scenario.

Going forward, first I have highlighted the related work that have influenced this project. Then I will highlight the design specification required and the implementation summary. And then, finally, I will highlight the evaluation metrics, results, conclusion, and the scope of future work.

2 Related Work

In this literature review we have explored how this space is rapidly evolving. Researchers across the world, as well as big tech firms like Google, OpenAI, and Meta, are pouring billions of dollars into solving this problem. As platforms like ChatGPT, Gemini are becoming more and more popular, the competition to retain market share among those platforms is on the rise. One of the biggest aspiration of these platforms are now is to become the most relevant platfrom for their users. OpenAI has already introduced memory, and other platforms are also following the same. As memory makes the platforms more relatable, the exponential amount of data generated using these platforms makes it even harder and costlier to continuously provide a similar experience going forward.

However, from the user's point of view, this makes people stick to respective platforms as they start using these platforms as their extended memory. We will now see how this work has evolved since 1998 and now becoming a key differentiator.

2.1 Extended Mind

Clark and Chalmers (1998) proposed the idea of the extended mind where they considered notes and computers as an extension of human memory or cognitive processes. They argued that we become dependent of these tools, and while using these tools we no longer use our memory resources to remember those things which we can easily find using these tools. Many other researchers have published similar views on this, like Smart (2013), Clowes (2013), Soares and Storm (2018). These works serve as the base of my project, where, with the modern database and system architectures, we are trying to further enhance this memory outsourcing process, which in many cases, is very difficult for an average human brain to recall.

2.2 Memory Recall

Hou et al. (2024) demonstrated in their work an architecture of memory recall mimicking the way we process context. They have used dynamic memory recall orchestration from a database where they store the historical memory content with temporal context. They used a mathematical model to quantify memory consolidation, considering factors such as contextual relevance, elapsed time, and recall frequency. This process is extremely efficient for storing contextual information, which can be easily accessed in the future. Zheng et al. (2025), Zhang, Dai, Bo, Ma, Li, Chen, Zhu, Dong and Wen (2024), Jiang et al. (2024) suggest similar findings in their recent work. It is also noted that in real-world situations where the data size increases exponentially and a delay in response can significantly impact adoption of the system in many cases for example, voice agents where only even a latency of 500 milliseconds is considered as poor response time, this orchestration poses a challenge.

2.3 Extraction of Actionable Insights

Shah et al. (2023), Shahin et al. (2024), Barravecchia et al. (2022) highlighted how they have used LLM models such as GPT 3.5 to extract actionable insights from customer interaction to use them for feedback and product development. This highlights the ability of advanced models to process more advanced text analysis and extract relevant information for future use cases. This is also a fundamental use case of NLP, which is now largely getting adopted in industry as well as in academia. This work, however limited to only information extraction, clearly signifies the potential of using this information for other purposes.

2.4 Financial Contextual Understanding of LLMs

Kong et al. (2024), Omoseebi et al. (2024), Zhang et al. (2025) emphasize how LLMs can integrate financial contextual knowledge. They have evaluated multiple models, starting from BERT by Google to LLAMA by Meta. They have also focused on finance-specific models like FinBert, InvestLM, FinLlama etc. This clearly demonstrates the ability of these models to understand, extract, and process financial data and provide actionable insights even for financial research. They have also evaluated how multimodality of these models further strengthens their capabilities and future possibilities. We however, on our project, have used a more robust and advanced closed-source model, Gemini 1.5 flash. Gemini 1.5 flash is a multimodal model that is part of Gemini's 1.5 Pro group, which

is trained on 137 billion parameters. As per Google, this model is to be deprecated in September 2025, as they have now released further superior models.

2.5 Customer Preference Using LLM in E-Commerce

Researchers from Amazon AI explored how LLMs can be used to extract preferences from multi-turn conversations in an e-commerce setting. This paper by Malik et al. (2024) used the PEARL methodology, which significantly improved the match performance and reduced the latency by 110 percent. This method is extremely helpful when there is relatively low historical data and a need for an instant recommendation system based on that available data. Fang et al. (2024), Roumeliotis et al. (2024) have also discussed the need for further study of adopting this or similar methods in different domains, which will require customization and further refinement for more independent preference extractions. In this multi-agent approach, it is worth noting that this has significantly improved the extraction precision and latency; however, it is highly dependent on the exemplars, which gives us motivation to explore other methodologies for better performance. The need for a domain-specific solution further justifies our study and why we must explore other domains where researchers are trying to solve this problem differently.

2.6 Twitter Interaction Automation Using LLMs

Customer support involves a lot of repetitive tasks with new and similar interactions. In this paper by Shareef (2024) explored how using the context-aware LLMs one can automate different types of customer queries on Twitter (Now X). In this study, the researchers have categorized different types of historical queries and then used them as context for the LLMs for automating new responses. Similar outcomes can be seen from Nandkumar and Peternel (2024), Jonnala (2024) and other's work. This also aligns with our work and perfectly highlights how domain context makes the interaction more personalized and how LLMs can add value to it.

2.7 Post Call Analytics

Koilakuntla et al. (2024), Karia et al. (2024) Embar et al. (2025) explored the use of LLMs for topic modelling, trend detection, call classification, and FAQ generation. These works have created a methodology for cost-efficient LLM systems that can automate insight extraction from calls in a call center. They have used actual as well as synthetic data for the research, where they have achieved better performance for their fine-tuned model compared to baseline models like ChatGPT 3.5 and Mistral. They were able to quickly identify topic labels, FAQs, and Trends to improve performance while reducing the cost. This study further guides us on how a proper framework can help get a desired result by fine-tuning the underlying base model.

2.8 Multi-Channel Natural Language Processing

Large language models using their sophisticated natural language processing (NLP) techniques can analyze client communications across multiple channels. Takayanagi et al. (2025), Machner et al. (2025) demonstrate that these models can extract semantic meaning from unstructured text in client emails, chat logs, and meeting transcripts; identify

emotional undertones and sentiment patterns. This clearly demonstrates how this can be used to recognize and categorize financial goals, concerns, and priorities and track changes in client preferences and attitudes over time. Following this work we can be confident that with a proper database management system, our architecture can handle customer interaction even from a multi-channel communication space without major changes for the financial advisory domain.

2.9 Temporal Pattern Recognition

The work by Landolsi et al. (2025), Tang et al. (2025) describes how their framework shifted from a rigid rule-based framework to a more human like conversational approach by leveraging the LLMs. In their work we can notice how their models were able to continuously understand temporal patterns and kept improving their performance over time to identify customer preferences. They have also experimented with multi-agent architecture and proved that it is possible to track dynamically changing customer preferences and serve according to their need. As we follow a similar approach this study serves as a key benchmark for our work.

2.10 Multi-modal Capabilities

As the models are getting better day by day and their multi-modal capabilities are getting more mature. We wanted to understand if our architecture can be leveraged for multimodal capabilities in the future. This study by Chen (2025) highlights how conversational AI systems like ChatGPT can process not only text-based interactions but also voice patterns, facial expressions from video meetings, and digital behavior patterns to create a comprehensive understanding of client preferences. This multi-modal analysis capability can allow financial advisors to gain further insights from both verbal and non-verbal cues, like how human advisors pick up on subtle signals during in-person meetings. Similar patterns can be drawn from the study by Zhang, Yu, Dong, Li, Su, Chu and Yu (2024). With this possibility we followed a more robust approach in our architecture, which can seamlessly upgrade to future multimodal capabilities and enable financial advisors to utilize the cutting-edge capabilities without the need for any major system improvements or upgrades.

3 Methodology

In our literature review we have noticed how researchers have previously worked with LLMs to automate customer support. As observed by Zhang et al. (2025), Malik et al. (2024), Machner et al. (2025) LLMs can also be leveraged to extract contextual information from historical data. These combined possibilities motivated us to explore how we can efficiently use LLMs to provide highly personalized financial advise and help financial advisor by providing a tool that can mimic like an extended brain, as mentioned by Clark and Chalmers (1998).

While conducting our ongoing research and literature review, we noticed that the LLMs work based on the context and the prompt. So to make it highly personalized, the amount of context needs to be increased, which results in exponentially increased cost, delayed processing, higher energy requirement, and carbon emission. To address this, we

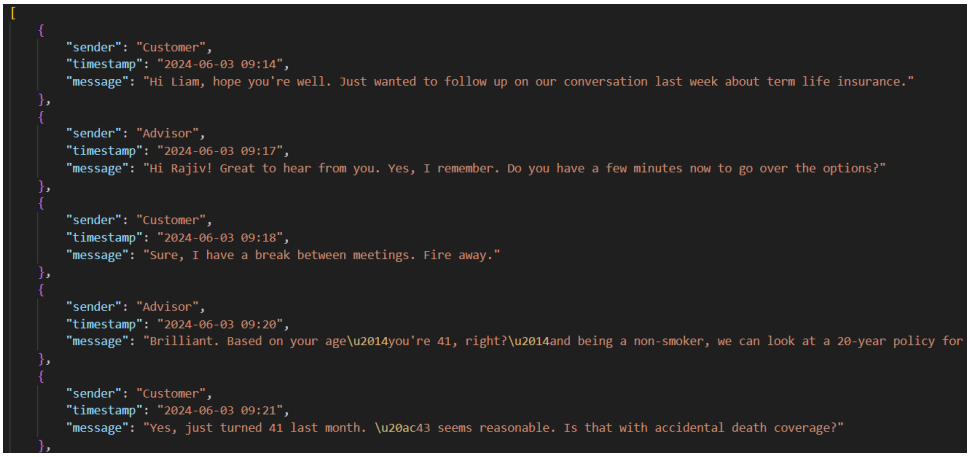

Figure 1: Chat History Dataset

have adopted a multi-LLM architecture that significantly reduces this by optimizing the context window in a time-bound manner.

3.1 Dataset

For our project, we have used a synthetic dataset that mimics a real-life chat between an Indian Engineer Named Rajiv working at Google, having regular conversations over WhatsApp with his financial advisor, Liam Byrne from Dublin. This data initially had 16194 Tokens; however, as we keep using the product and chat as Rajiv the number of Tokens increases. In this conversation, Rajiv occasionally share abut his family, preferences, key dates in a normal conversational manner, where sometime information are not direct and needs to be interpreted based on the current date and time and other factors. This database is saved as json file for the ease of our execution how ever we can use any database for this, like modern vectorized databases and other Structured or unstructured databases.

3.2 System Architecture

We have used a multi-LLM orchestration with Long-term and short-term memory handling to keep our LLM responses highly personalized. The long-term memory serves as the database. We have used a JSON file here as an example; however, this can be any database. This stores all the conversations and is used to retrieve the key contextual detail extraction. We extract the contextual key information to the short-term memory and pass the same to our final LLM model for response generation. In this process we use 180 days of conversation handling on a loop and save the initial extraction in cache memory. Finally we optimize the cache memory and save the final extraction. (See Image 2) This architecture has several advantages over other alternative options. This architecture is fundamentally multimodal, so just by replacing the underlying LLMs, we can process data of that nature as well. On the other hand this same architecture can also be used in line with the Vector database or RAG framework. Since my architecture processes the data at the very source level, we can use the processed data in that architecture, so the system becomes far more efficient. It is also to be understood that we are not processing the database at the time of entry because that limits the extraction context of the LLMs.

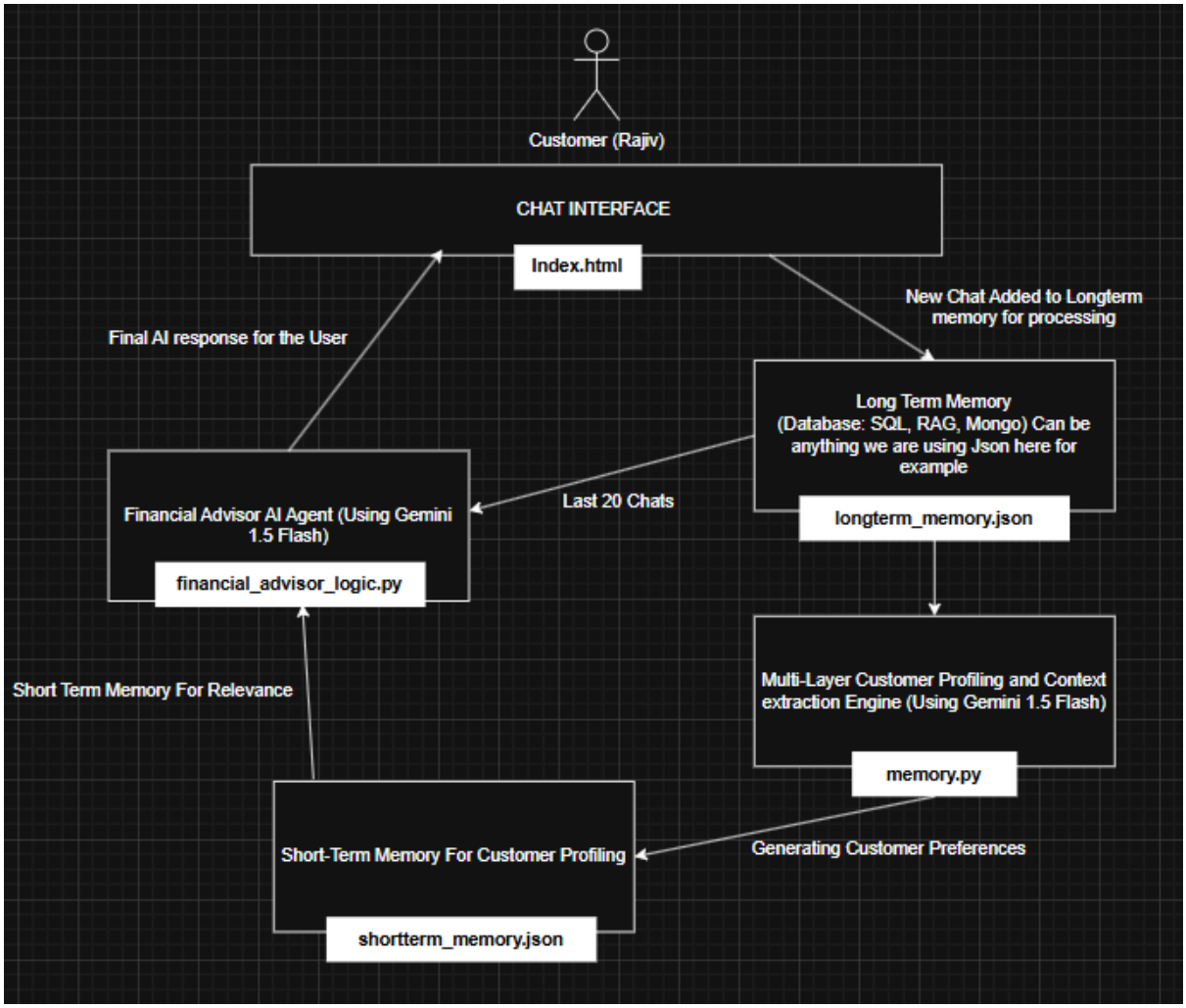

Figure 2: System Architecture

Also, there is a high chance that doing so during the ongoing process will add latency to the system.

3.3 Model Selection

We have evaluated this with multiple open source and closed source models. We have tested using small models like BERT, FinBERT, which is fast however failed to extract all the necessary information. This limitation can be eliminated by fine-tuning this model for our purpose, which will definitely further lower the cost and increase processing time. On the Other hand we have also used a locally hosted Llama-3B model and Mistral. However, we find the best result with Google's Gemini 1.5 Flash model, which is fast, cost-effective, and accurately extracts temporal information as well as other contextual information.

3.4 Prompt Engineering

Our multi-layer extraction engine and financial advisory architecture are configured considering the ongoing rapid improvement in the foundational models. Hence, as models get

more and more sophisticated, we can seamlessly integrate new models into our architecture without any major change. However, to run our architecture, we need to direct our models using prompts. For the same, I have used both Zero-shot and one-shot prompt engineering techniques to balance the generalization and contextual guidance. On our advisory side, where the model has the liberty to be more creative and come up with open-ended conversation, we have used zero-shot prompting means we have not provided any example, so our model follows similar patterns. Although based on performance, we can adopt such practices to maintain a particular tone in the conversation. On the other hand, for extraction, we have used a one-shot prompting technique, ensuring that the model still has the liberty to intelligently understand context and extract information while giving it a hint on how it should balance that approach. It should be noted that, based on performance and the ongoing evolving nature of the data and communication, it is required that we may need to frequently optimize our prompt for the same.

3.5 Evaluation Parameter Selection

As generative AI is evolving there are new parameters are being adopted to evaluate the efficacy of these large language models' performance. There are some major parameters like SWE-bench Verified, Humanity's Last Exam, GPQA Diamond, etc, which these organizations use to evaluate multiple capabilities of the models like reasoning, coding, problem solving, etc. In our case, as our core goal aligns with customer satisfaction on relevance so we have used human evaluation as the parameter to manually check whether our financial advisor can understand the context and answer intelligently with the historical data. Once we ensure that our code artefact can answer contextually, we wanted to measure how efficiently we have optimized the contextual memory.
All large language model charge API users based on the input and output token count. In simple words tokens are considered as the unit of the input and output parameters of LLMs. As per Google's documentation, it has mentioned that Gemini considers 4 character as one token. That means 100 token is about 60-80 english words. Since this is the key parameter for LLM pricing, we have considered tokens as our key parameter to measure the efficiency. We have passed both the raw and processed memory to the token counter to understand the before and after difference using the same public token counter.

4 Design Specification

In our project, we have used multi-LLM orchestration. We have executed the orchestration in our local computer, which has a decent 16GB of RAM and an AMD Ryzen processor. As we have used the closed-source Gemini 1.5 Flash model using the official API, this allows our code to be executed on any decent system with minimal configuration.

We have used a mix of Python, CSS, HTML, and JavaScript for our project. Where we created a simple frontend User Interface mimicking WhatsApp using HTML, CSS and JavaScript, and for backend we have used Flask, a Python-based framework to run the server orchestration.

```
FINAL_PROJECT
└── __pycache__
```

```
├── memory_caches
├── app.py
├── financial_advisor_logic.py
├── index.html
├── longterm_memory.json
├── memory.py
├── requirements.txt
└── shortterm_memory.json
```

We have used Gemini 1.5 Flash API to run our extraction engine and defined the process in the Memory file. Here we chronologically fetch 180 days of conversation from the Long-term Memory file from the first date of interaction to the last date of interaction. So when the Memory file is executed, it continiously processes the conversation of 180 days and extracts key information related to the customer and stores it in the cached memory. Once all the initial data is extracted, it then again processes the cached memory to come up with the final short-term memory or customer profiling context.

We use the App file to run the server and the Financial Advisor Logic file to chat with the user in the frontend or the user interface using the context-aware short-term and long-term memory. Here is the prompt we have used to configure our advisor.

```
system_prompt = (
"You are a friendly, fun and professional financial advisor. You are Irish, and based in Dublin. You speak like a friend with Irish wit."
"Keep your responses concise and limited to 1–2 sentences like a human would over WhatsApp."
"Your name is 'Liam Byrne'. Your client's name is Rajiv."
"Your role is to assist users with financial advice,"
"Use the customer details to refer customer preferences to make the chat more personal and engaging."
"Keep the conversation going and enaging even when its more like a chat and do not push like a sales man."
"answer questions about insurance, investments, and financial planning."
"Keep your responses concise, helpful, and professional."
"Only ask one question at a time, and wait for the user's response before continuing."
"Chat exactly as a human would over WhatsApp, using natural language."
"Do not always end with a question and sound like a robot or sales man."
)
```

5 Implementation

Building and implementing this architecture was an uphill task. Considering the modern and evolving techniques like Vector Search, RAG, etc., my goal was to come up with

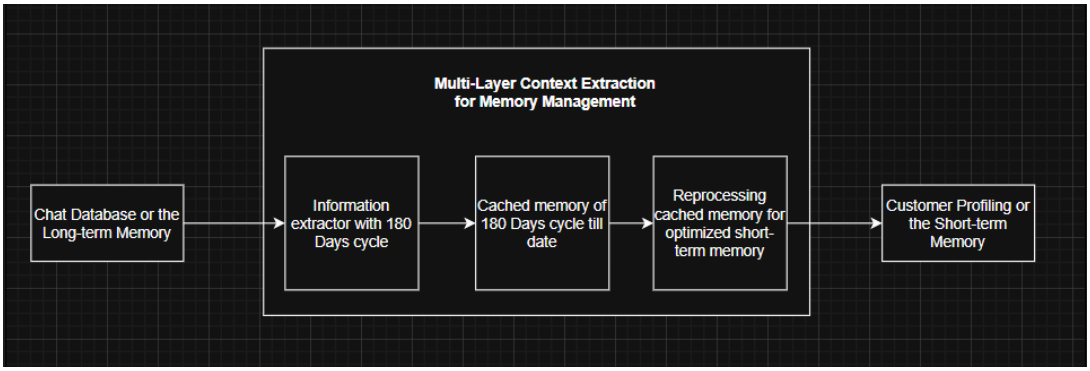

Figure 3: Context Aware Token Optimization

a solution that can be tweaked and implemented in any approach. For this I focused on the fundamental problem that is the key context window, which needs to be passed to the LLM along with the prompt. I considered this particular architecture because it is efficient, and the same model can be implemented in advanced systems like RAG or Vector search, where this architecture will further boost productivity and reduce costs, as it handles the core contextual data issue at the inception.

I have used a mix of Python, HTML, CSS, and JavaScript for our system architecture, where I have a backend engine and a front-end user interface (See image 4). This interface directly mimics the WhatsApp interface and works in a similar manner. This production-ready system can be deployed and made public for immediate use by users. However, for quick development, we have limited this to one dummy user named Rajiv.

On the backend side (see image 3), the memory engine involves mainly three layers. The main memory extraction takes place in three steps before being saved in the short-term memory. This implementation system is also production-ready, and it can be published and run using a cron job in a time-bound manner to keep the short-term memory up-to-date.

To build this architecture I started with the long-term memory for which I used ChatGPT to generate the synthetic chat dataset. After that, I created the memory extraction engine with a multi-layer approach, which processes 180 Days of data at a time to the cache memory, and then finally, the most optimized data is pushed and saved in the short-term memory. My chat engine uses the short-term memory, the last 20 conversations from the long-term memory as the context, and combines it with the user input and passes it to the LLM. The LLM, i.e., Gemini 1.5 Flash, uses and generates a response. Our chat engine also logs these conversations and pushes them back to the long-term memory. This way our system continiously learns about the users and provides up to date contextual response.

6 Evaluation

There are mainly two key evaluation areas in our project to understand its efficacy. First, the relevance of the chat response, and the most important is the reduction of token counts while generating such a response.

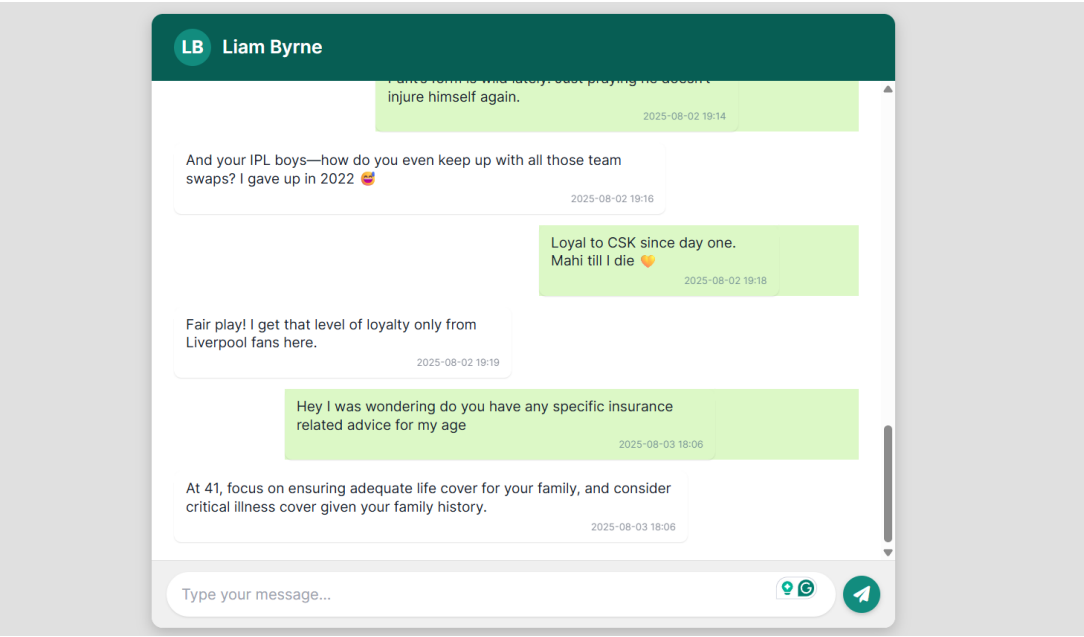

Figure 4: Chat Interface

6.1 Contextual Relevance

Communication is highly personal, and it is technically hard to justify its effectiveness. However to understand the relevance and correctness of our system, we have performed numerous manual human testing, like many major tech firms currently follow. We tried to initiate a conversation without specifically telling our preference and made the advisor think and look for such context from the historical conversation that is in their short-term memory. We have added one screenshot of one such conversation where the LLM-powered Advisor can immediately understand the context and reply back with relevant answers based on user age and favourite drink (See image 5). This is consistent with previous research on the topic. Where they have tested similar experiments in different domains with different methodologies.

6.2 Token Optimization

The major success factor for our work is how much optimization our orchestration has achieved while providing a similar degree of relevance response otherwise be achieved by passing the entire conversation as context. To evaluate this we have considered the token count. Since we have used Google Gemini 1.5 Flash as our core model, we have followed Google documentation (Link [1]). We have used Gemini Token Counter [2]), a publicly available interface, to validate our results in a visually appealing manner. Tokens are largely considered the unit for LLMs; most closed-source LLM providers charge based on the token count for their APIs. Although different model has different way of counting tokens, they generally publish their methodology for the developer community. For

[1] `https://ai.google.dev/gemini-api/docs/tokens?lang=python`
[2] `https://geminitokencounter.dbanswan.com/`

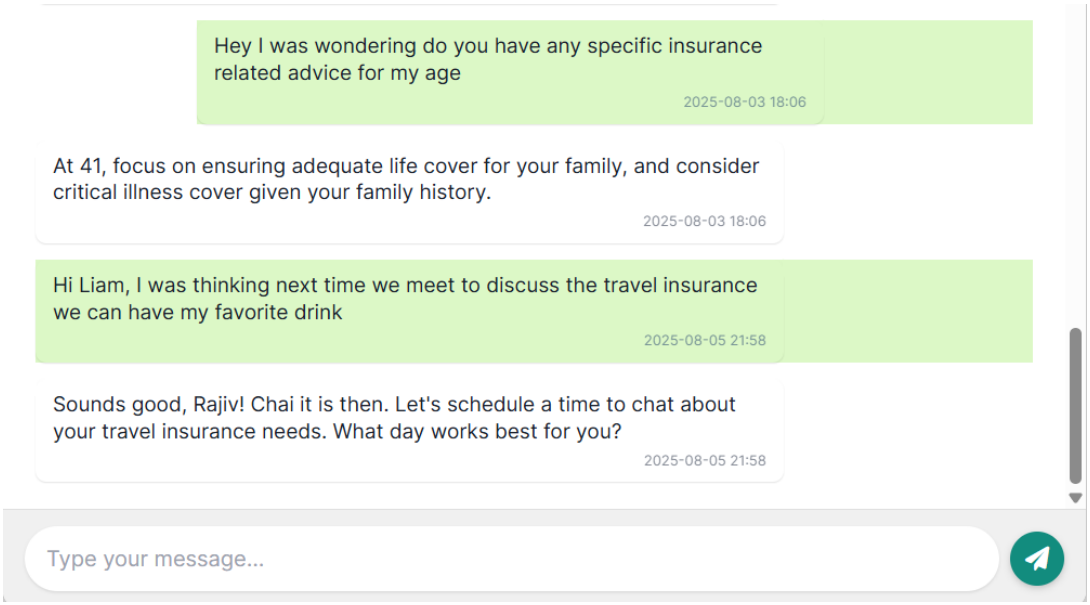

Figure 5: Contextual Relevance

Gemini models, it generally considers 4 characters as 1 token. See the table below for the result, where we got a massive 92.75 percent reduction in token count :

Table 1: Comparison of Original vs. Optimized Token Counts

Description	**Token Count**	**Reduction**
Original Token Count	17106	–
Optimized Token Count	1241	–
Total Reduction	**15865**	**92.75%**

7 Conclusion and Future Work

This work aligns with several global and industrial challenges. As we aim to optimize the token utilized for LLM context, this potentially reduces the cost of LLM compute for the corporates, and on the other side, it reduces the amount of electricity consumed and hence the carbon emission. In our case, we have achieved a massive 92 percent reduction in the token count. And this reduction will cause massive cost saving every time the LLM generates a response, hence creating a massive exponential cost and energy savings.

For our work, we have used the Gemini 1.5 Flash model, which was developed by Google. This is a multi-modal model means it can process different formats of data, which also justifies why our system can process other formats of data than normal text chats. However, for normal text chats, this process can be further optimized by using a smaller open-source model and improving its efficiency by training it on our use case-specific data. The prompt engineering is also an important area for further analysis to optimize the result. Implementing this architecture along with RAG or Vector Databases will further improve the efficiency of the system, and this is a direction I would like to explore as the next step in this work.

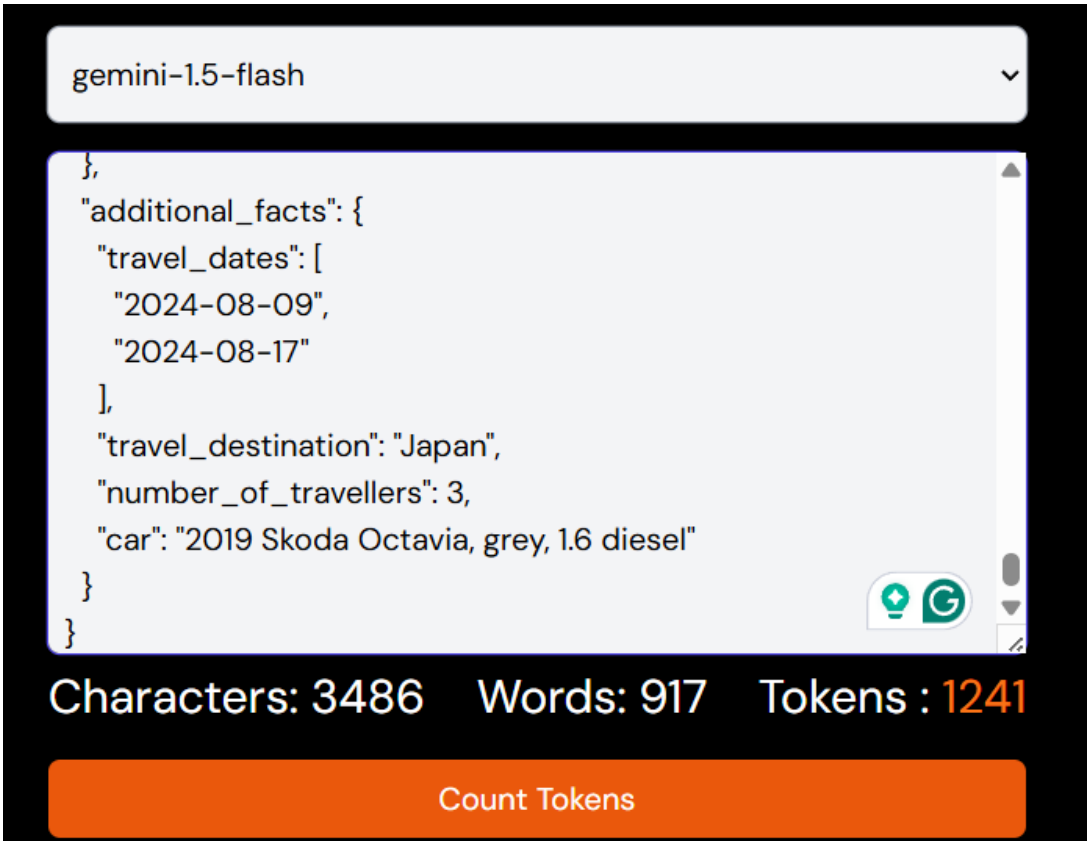

Figure 6: Optimized Token Count

References

Barravecchia, F., Mastrogiacomo, L. and Franceschini, F. (2022). Digital voice-of-customer processing by topic modelling algorithms: insights to validate empirical results, *International Journal of Quality & Reliability Management* **39**(6): 1453–1470.

Chen, Z. (2025). Revolutionizing finance with conversational ai: a focus on chatgpt implementation and challenges, *Humanities and Social Sciences Communications* **12**(1): 1–11.

Clark, A. and Chalmers, D. (1998). The extended mind, *analysis* **58**(1): 7–19.

Clowes, R. W. (2013). The cognitive integration of e-memory, *Review of philosophy and psychology* **4**(1): 107–133.

Embar, V., Shrivastava, R., Damodaran, V., Mehlinger, T., Hsiao, Y.-C. and Raghunathan, K. (2025). Llm-based insight extraction for contact center analytics and cost-efficient deployment, *arXiv preprint arXiv:2503.19090* .

Fang, C., Li, X., Fan, Z., Xu, J., Nag, K., Korpeoglu, E., Kumar, S. and Achan, K. (2024). Llm-ensemble: Optimal large language model ensemble method for e-commerce product attribute value extraction, *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2910–2914.

Hou, Y., Tamoto, H. and Miyashita, H. (2024). " my agent understands me better": Integrating dynamic human-like memory recall and consolidation in llm-based agents,

Extended Abstracts of the CHI Conference on Human Factors in Computing Systems, pp. 1–7.

Jiang, X., Li, F., Zhao, H., Qiu, J., Wang, J., Shao, J., Xu, S., Zhang, S., Chen, W., Tang, X. et al. (2024). Long term memory: The foundation of ai self-evolution, *arXiv preprint arXiv:2410.15665* .

Jonnala, A. (2024). How large language models (llm) help enterprises enhance customer experiences, *Journal Homepage: http://www. ijmra. us* **13**(11).

Karia, K., Mashru, D., Heda, V. and Thoday, V. (2024). Leveraging large language models for evaluating customer service conversations and retrieval-augmented generation for pre-call insights, *2024 International Conference on Communication, Control, and Intelligent Systems (CCIS)*, IEEE, pp. 1–8.

Koilakuntla, B., Rana, P., Ahuja, P., Konjeti, S. and Vepa, J. (2024). Leveraging large language models for post-transcription correction in contact centers, *Proc. Interspeech 2024* pp. 2038–2039.

Kong, Y., Nie, Y., Dong, X., Mulvey, J. M., Poor, H. V., Wen, Q. and Zohren, S. (2024). Large language models for financial and investment management: Models, opportunities, and challenges., *Journal of Portfolio Management* **51**(2).

Landolsi, H., Abdeljaoued-Tej, I. et al. (2025). From rigid robo-advisors to human-like interactions: Revolutionizing financial assistance with llm-powered solutions, *CS & IT Conference Proceedings*, Vol. 15, CS & IT Conference Proceedings.

Machner, N., Ehmanns, C. A., Uludag, Ö., Toreini, P. and Matthes, F. (2025). Leveraging large language models for intelligent customer service channel navigation.

Malik, V., Jagatap, A., Puranik, V. S. and Majumder, A. (2024). PEARL: Preference extraction with exemplar augmentation and retrieval with LLM agents, *in* F. Dernoncourt, D. Preoţiuc-Pietro and A. Shimorina (eds), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, Association for Computational Linguistics, Miami, Florida, US, pp. 1536–1547.
URL: *https://aclanthology.org/2024.emnlp-industry.112/*

Nandkumar, C. and Peternel, L. (2024). Enhancing supermarket robot interaction: A multi-level llm conversational interface for handling diverse customer intents, *arXiv preprint arXiv:2406.11047* .

Omoseebi, A., Jhon, A. and Olabiyi, W. (2024). Large language models (llms) for financial security.

Roumeliotis, K. I., Tselikas, N. D. and Nasiopoulos, D. K. (2024). Llms in e-commerce: A comparative analysis of gpt and llama models in product review evaluation, *Natural Language Processing Journal* **6**: 100056.

Shah, C., White, R., Andersen, R., Buscher, G., Counts, S., Das, S., Montazer, A., Manivannan, S., Neville, J., Rangan, N. et al. (2023). Using large language models to generate, validate, and apply user intent taxonomies, *ACM Transactions on the Web* .

Shahin, M., Chen, F. F. and Hosseinzadeh, A. (2024). Harnessing customized ai to create voice of customer via gpt3. 5, *Advanced Engineering Informatics* **61**: 102462.

Shareef, F. (2024). Enhancing conversational ai with llms for customer support automation, *2024 2nd International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS)*, IEEE, pp. 239–244.

Smart, P. R. (2013). The web-extended mind, *Philosophical Engineering: Toward a Philosophy of the Web* pp. 116–133.

Soares, J. S. and Storm, B. C. (2018). Forget in a flash: A further investigation of the photo-taking-impairment effect, *Journal of Applied Research in Memory and Cognition* **7**(1): 154–160.

Takayanagi, T., Izumi, K., Sanz-Cruzado, J., McCreadie, R. and Ounis, I. (2025). Are generative ai agents effective personalized financial advisors?, *arXiv preprint arXiv:2504.05862* .

Tang, J., Chen, S., Gong, C., Zhang, J. and Tao, D. (2025). Llm-ps: Empowering large language models for time series forecasting with temporal patterns and semantics, *arXiv preprint arXiv:2503.09656* .

Zhang, D., Yu, Y., Dong, J., Li, C., Su, D., Chu, C. and Yu, D. (2024). Mm-llms: Recent advances in multimodal large language models, *arXiv preprint arXiv:2401.13601* .

Zhang, Z., Cao, Y. and Liao, L. (2025). Xfinbench: Benchmarking llms in complex financial problem solving and reasoning, *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8715–8758.

Zhang, Z., Dai, Q., Bo, X., Ma, C., Li, R., Chen, X., Zhu, J., Dong, Z. and Wen, J.-R. (2024). A survey on the memory mechanism of large language model based agents, *ACM Transactions on Information Systems* .

Zheng, J., Shi, C., Cai, X., Li, Q., Zhang, D., Li, C., Yu, D. and Ma, Q. (2025). Lifelong learning of large language model based agents: A roadmap, *arXiv preprint arXiv:2501.07278* .