

MoneyMate: Democratizing Financial Insights via an AI Chatbot on Messaging Platforms

Rafael Eitaró Oshiro

Matheus Tavares Guerson

Samuel Ferraz (Advisor)

¹Faculdade de Computação – Universidade Federal do Mato Grosso do Sul (UFMS)

`rafael.oshiro@ufms.br, matheus.guerson@ufms.br, samuel.ferraz@ufms.br`

Abstract. *This paper presents MoneyMate, an open-source, AI-powered financial chatbot designed to streamline personal finance management through seamless integration with web and Telegram platforms. Leveraging advanced Large Language Models within instant messaging environments, MoneyMate enables intuitive, natural language interactions to deliver personalized financial insights and dynamic reports. The system architecture includes a React-based web dashboard, a Node.js back-end using Prisma ORM, and a PostgreSQL database, all deployed on scalable infrastructure to ensure high reliability and performance. This work contributes by demonstrating a practical integration of LLMs into everyday financial routines and offering a modular, open architecture that can be adapted to similar personal finance applications.*

Keywords: ChatBot, LLM, Telegram

1. Introduction

1.1. Motivation

Managing personal finances effectively has become increasingly important in a world driven by consumerism and complex financial products. A recent study [1] revealed that more than a quarter (26%) of respondents lack the knowledge and skills to make sound financial decisions, being classified as having low financial literacy. Additionally, 60% of adults worldwide have average financial literacy, and only 15% demonstrate high financial literacy. This trend is consistent across all countries surveyed, highlighting the urgent need for accessible financial guidance and tools. Although there is a broad set of financial technology tools, existing applications often fail to meet expectations in the following aspects:

- **User-friendliness.** Users must manually input data and navigate through overwhelming interfaces, which can be intimidating for those unfamiliar with the financial world.
- **Lack of Interactivity.** Most platforms operate with predefined functionalities, leaving little room for customization or dynamic interaction according to individual needs.

- **Limited Usability.** Financial platforms often fail to integrate with communication tools like messaging apps, limiting their accessibility for users who prefer conversational interfaces.

These limitations highlight the need for an innovative solution that bridges the gap between advanced technology and user-centric design. By leveraging conversational AI and integrating with popular messaging platforms, this project aims to redefine the way users interact with their financial data.

1.2. Objective

The primary objective of this project is to address the gaps in existing financial management tools by providing a more user-friendly and accessible solution. The system is designed to overcome the complexity of static and overwhelming financial dashboards by introducing a conversational interface, enabling real-time interactions and insights while reducing the manual effort needed to manage finances. Additionally, it bridges the accessibility gap by integrating financial tools into widely used messaging platforms like *Telegram* [21].

The significance of this project lies in its ability to revolutionize financial management through conversational AI and seamless integration with web systems. By embedding financial tools within familiar interfaces, the project offers several benefits with respect to the state-of-the-art solutions:

- **Simplified Financial Management.** Complex financial tasks were transformed into intuitive conversations, resulting in a tool accessible even to non-experts.
- **Real-Time Interaction.** Through a 24/7 user-friendly ChatBot, users can receive instant updates, generate detailed reports, and analyze spending habits.
- **Financial Literacy Enhancement.** The provisioning of personalized insights and feedback helps users understand and improve their financial behaviors.
- **User Experience.** The combination of a conversational chatbot and a feature-rich web dashboard enables users to manage their finances efficiently without the need to download a dedicated application.
- **Security and Scalability.** The use of a robust back-end architecture enables secure handling of sensitive data while supporting simultaneous user interactions.

This approach ensures that the system caters to a broad audience, including those with limited financial expertise, and promotes better financial decision-making. By integrating AI-driven financial tools into accessible platforms, the project sets a precedent for a more inclusive, efficient, and user-focused approach to personal finance.

1.3. Structure

This paper is organized into the following sections:

- **Chapter 2** discusses related work, highlighting existing financial management tools and chatbot-based solutions.
- **Chapter 3** provides an overview of the technologies and frameworks used in the development of the system.
- **Chapter 4** details the architecture and implementation of the system, including its components and functionalities.
- **Chapter 5** concludes with a discussion of the project's contributions, limitations, and potential future developments.

2. Related Work

The development of AI-driven financial chatbots and personal finance management tools has gained significant traction in recent years. Various research efforts and commercial implementations have explored different approaches to address financial literacy and simplify money management through technology. This section reviews key existing solutions and highlights their key aspects.

2.1. Banking Financial Chatbots

Banking chatbots have become a valuable tool for customers seeking instant support and financial insights. Many banks have incorporated these AI-driven systems into their services, enabling features like balance inquiries, transaction history retrieval, loan information, and more. These chatbots provide enhanced customer service by offering 24/7 assistance and personalized financial advice. Noteworthy applications include:

- **Cleo [5]:** An AI-powered chatbot that connects to users' bank accounts, offering spending analysis, budget tracking, and personalized financial recommendations through a conversational interface. Cleo's approach demonstrates the feasibility of chat-based financial management but lacks deeper integration with external financial systems;
- **Erica (Bank of America) [3]:** A virtual assistant that provides insights on transactions, bill reminders, and budgeting tips. While effective for bank customers, its functionality is limited to Bank of America accounts, restricting its accessibility for users with multiple banking relationships.

While these banking chatbots have boosted user interaction and satisfaction, they are often confined to the bank's own platforms, limiting their ability to connect with third-party applications or offer cross-platform functionality. As a result, the potential for seamless financial management across multiple services remains underutilized.

2.2. Personal Financial Chatbots

Chatbots for personal finance have gained significant traction as a user-friendly solution for managing everyday financial tasks. These AI-powered assistants help individuals with budgeting, expense tracking, savings goals, and even investment advice. Several personal finance apps and platforms now feature chatbots to provide personalized financial recommendations, monitor spending habits, and deliver timely reminders for bill payments. Notable examples include:

- **KAI (Kasisto) [11]:** a conversational AI used by financial institutions to assist customers with banking inquiries and financial planning. However, KAI is primarily designed for enterprise use, limiting its adaptability for personal finance management by individual users.
- **FinZap [7]:** an interesting approach to financial management using WhatsApp as its main interface, following the idea of tracking expenses and income using a well-known App. While it might not yet be the perfect fit for everyone, especially those looking for extensive features out of the box, it reflects a growing shift towards more user-friendly, chat-based financial tools.

- **Meu Assistente [13]:** a Brazilian AI platform that helps users create professional content like text, images, and videos quickly and efficiently. It offers AI-powered tools to simplify and speed up content creation, making the process more accessible. Users register to access the platform's features, which aim to save time and boost productivity.

These chatbots have revolutionized how users engage with their personal finances, offering automated, real-time support. However, many of these tools are limited to specific apps or services, and their integration with broader financial systems or external platforms remains a challenge. As a result, users may find themselves unable to easily access or consolidate all their financial data across various sources.

2.3. Personal Finance Management Tools

Traditional personal finance management tools, such as budgeting apps and financial dashboards, offer structured approaches to track expenses and income. Some widely used platforms include:

- **Mint [10]:** A comprehensive budgeting and expense-tracking app that aggregates financial data from multiple sources. Despite its powerful analytics, Mint requires users to manually interact with its dashboard, which can be overwhelming for non-experts.
- **YNAB (You Need a Budget) [25]:** A goal-oriented budgeting tool that promotes financial discipline through proactive planning. However, its learning curve and subscription cost can be barriers to widespread adoption.
- **Mobills [15]:** A Brazilian mobile-first financial control platform that allows users to categorize expenses, set budgets, and track financial goals. While user-friendly and visually appealing, Mobills offers limited integration with third-party services and automation tools.
- **Organizze [18]:** Another popular Brazilian app known for its simplicity and intuitive interface. It supports multi-account management and visual dashboards but lacks advanced AI-driven insights and chatbot integration.
- **Meu Dinheiro [14]:** A web-based personal finance manager focused on manual input and detailed reports. Although powerful for experienced users, it requires a steeper learning curve and does not offer conversational interfaces.

While these tools provide valuable financial insights, their reliance on traditional dashboard-based interfaces limits accessibility and ease of use. Many users struggle with navigating complex financial data, highlighting the need for a more intuitive and conversational approach.

2.4. Key Differentiators of MoneyMate

Table 1 shows the key aspects of the state-of-the-art solutions for financial management. Note that MoneyMate builds upon these existing solutions by combining the best aspects of financial chatbots and personal finance tools while addressing their limitations.

Table 1. Feature Comparison Across Chatbots, Management Tools, and MoneyMate

Tool	Bank Data Integration	Customizable Interface	Messaging or Chat Integration	LLM-Generated Reports	Open Source
Cleo	✓	✓	Native App	✗	✗
Erica	✓	✗	Native App	✗	✗
KAI	✓	✗	Native App	✗	✗
FinZap	✗	✗	WhatsApp	✓	✗
Mint	✓	✓	✗	✗	✗
YNAB	✓	✓	✗	✗	✗
Mobills	✗	✓	✗	✗	✗
Organizze	✗	✓	✗	✗	✗
Meu assistente	✗	✓	Native App	✓	✗
Meu Dinheiro	✗	✓	✗	✗	✗
MoneyMate	✗	✓	Telegram	✓	✓

- **Conversational AI with Personalized Insights.** Unlike traditional apps, MoneyMate employs advanced AI to offer real-time financial advice through an interactive chat-based interface.
- **Integration with Messaging Platforms.** Unlike most finance apps that require dedicated installations, MoneyMate seamlessly integrates with platforms like Telegram, ensuring accessibility without additional apps.
- **Web Dashboard for Enhanced Control.** Unlike chat-only solutions, MoneyMate complements its chatbot with a feature-rich web interface, giving users the flexibility to manage finances through multiple channels.

3. Relational Model

The relational model (RM) depicted in Figure 1 outlines the relationships among the core entities of the financial management system, emphasizing the interconnectedness of user data, financial transactions, and account management.

Key relationships include:

- **User and Accounts:** Each user can possess multiple bank accounts, enabling efficient financial tracking across different institutions.
- **User and Transactions:** Users are directly associated with their transactions, allowing for the categorization of expenses and incomes. Transactions can be linked to specific accounts, providing granular financial insights.

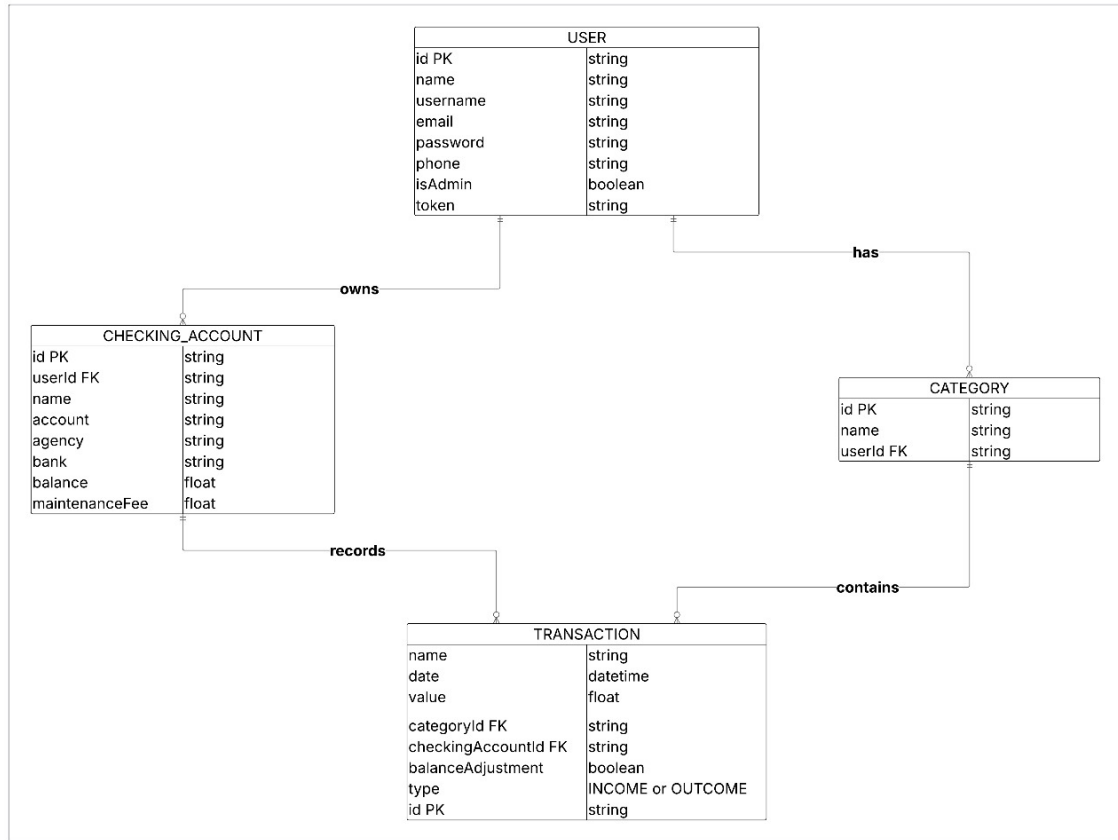


Figure 1. Relational Model.

- *Transactions and Categories:* Every transaction can belong to a category, facilitating structured financial analysis and reporting.

This model ensures that the system maintains a robust structure for managing and analyzing financial data, supporting scalability and user-centric functionality while keeping a simple but powerful data modeling.

4. System Architecture

MoneyMate was developed with the goal of making personal finance management more accessible, modern, and interactive. To achieve this, the system combines a web-based dashboard with a Telegram-integrated chatbot, allowing users to choose how they want to interact with their finances: through natural conversation or a visual interface.

Figure 2 depicts the system architecture, which comprises:

- The front-end layer (1), responsible for web-based user interaction, represented in the `tcc-front` repository [8].
- The back-end API layer (2), handling business logic and backend processing, implemented in the `tcc-api` repository [2].
- The chatbot layer (3) that facilitates real-time user interaction, leveraging large language models (LLMs), and contained in the `tcc-telegram-bot` repository [4].
- The database layer (4) for secure and efficient data management.

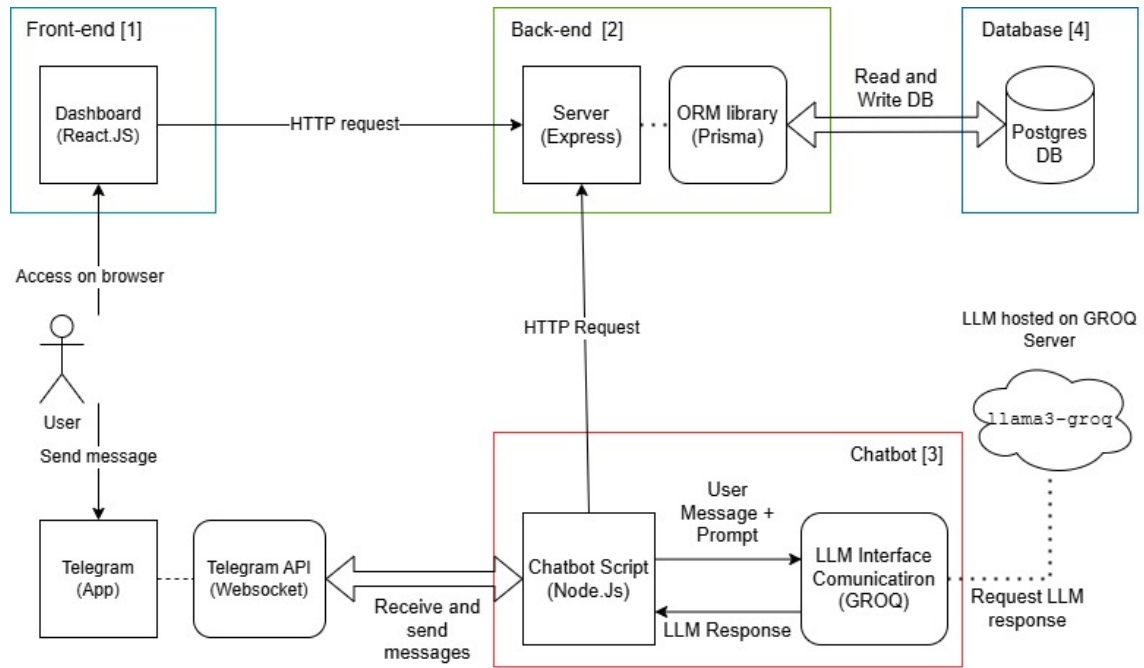


Figure 2. General Architecture of the MoneyMate System.

To summarize, MoneyMate’s architecture revolves around two communication channels—chat and dashboard—both connected to a centralized backend and database. This unified approach ensures data consistency and allows users to manage their finances through whichever interface suits them best.

4.1. Front-End

The front-end application is responsible for providing an intuitive user interface where users can visualize, edit, and manage their financial data. Features include dynamic dashboards that display financial insights and allow for real-time interaction with data.

The front-end was developed with **React** [20], a widely-used JavaScript library for building dynamic, component-based user interfaces. React was chosen due to its ability to deliver a smooth and responsive user experience by utilizing the Virtual DOM, which updates only the parts of the interface that have changed, thereby optimizing performance. Additionally, its component-based architecture promotes code reuse and simplifies maintenance, accelerating development and enabling easier scalability for future enhancements. Another key factor was React’s large community and robust ecosystem of libraries and tools, providing strong support and reliability throughout the development process.

4.2. Back-End

The back-end handles the business logic and API requests. It uses **Prisma** as an ORM to manage the database, enabling efficient and secure data handling. ORM (Object-Relational Mapping) is a technique that allows developers to interact with relational databases using an object-oriented programming language instead of writing raw SQL queries. This simplifies database management and access by abstracting complex SQL operations into objects and methods. It increases productivity by reducing repetitive code

and errors, enhances security by preventing issues like SQL Injection through automatic query handling, and keeps the code cleaner and more maintainable. Prisma is a modern ORM that facilitates create, read, update, and delete operations efficiently and securely.

Additionally, the back-end utilizes **Node.js** [17], a runtime environment that executes JavaScript code on the server side. It uses a non-blocking, event-driven model, making it ideal for real-time applications. In MoneyMate, Node.js is used to run the chatbot server and manage backend services.

Node.js also manages multiple chatbot instances, allowing each user to have an isolated bot process. This architectural decision was made to ensure that the context of each conversation remains isolated, preventing any leakage of data or interaction between users. By running separate bot instances, we enhance security by ensuring that one user's data cannot be accessed by another user. Additionally, this approach improves scalability, as each bot instance operates independently, enabling efficient resource allocation and fault isolation.

We used **Express** [6](a minimalist and flexible web application framework for Node.js) to connect the front-end, database, and chatbot. It simplifies the development of RESTful APIs and provides routing, middleware support, and HTTP request handling.

4.3. Database

We chose PostgreSQL [19] for its reliability, advanced features, and strong community support. Unlike some other relational databases, PostgreSQL offers powerful extensibility, allowing us to customize and optimize the database to fit the specific needs of our system. It supports complex queries and large datasets efficiently, which is crucial for managing user data, transaction history, and chatbot configurations. Additionally, PostgreSQL's strict adherence to SQL standards and robust data integrity mechanisms make it a trustworthy choice for ensuring data consistency and security. These qualities combined made PostgreSQL the best fit for our system compared to other options.

4.4. ChatBot

The chatbot is implemented using the Telegram API [23] and acts as a conversational interface for users to manage their finances. Telegram's API enables the development of bots that interact with users over the Telegram App [21]. It provides methods for receiving messages, sending replies, and managing chat sessions. This API serves as the bridge between the chatbot and the messaging platform.

Using a Large Language Model (LLM) to manage chatbot interactions is important because LLMs provide advanced natural language understanding and generation capabilities. This allows the chatbot to comprehend a wide variety of user inputs, including ambiguous or complex queries, and respond in a more natural, coherent, and context-aware manner. Unlike rule-based systems, LLMs can handle diverse conversation flows, adapt to different user intents, and improve over time with more data. This results in a more engaging and effective user experience, making the chatbot not only a tool for simple commands but also a conversational partner capable of meaningful interactions.

Leveraging Groq's [9] LLM capabilities, the chatbot uses well-designed prompt engineering techniques to ensure accurate understanding of user queries and intelligent responses. These techniques include:

- **Persona definition:** establishes the assistant’s identity with messages like “Você é um chatbot de finanças...”, guiding its tone and behavior.
- **Instruction-based prompting:** uses imperative commands (e.g., “Mostre para ele em forma de lista”) to direct the assistant clearly and unambiguously.
- **Contextual prompting:** dynamically injects user-specific data (such as categories, checking accounts, and transactions) into the prompt, allowing for accurate and relevant responses.
- **Chain-of-thought prompting:** structures the assistant’s workflow through logical steps (e.g., perguntar valor → tipo → categoria → conta → confirmar → registrar), improving coherence during multi-step tasks.
- **Fail-safe prompting:** anticipates missing or invalid inputs and instructs the assistant to respond politely and request clarification.
- **Tool calling:** registers callable functions (e.g., `handelTransaction`) that the LLM can invoke to execute concrete actions.
- **Conversation loop control:** guides the assistant to manage the session by asking whether the user wants to continue or terminate the interaction.

The TypeScript code below illustrates how the assistant’s prompt is constructed in our system. This function generates the prompt messages that guide the chatbot through the process of registering a financial transaction. Each part of the prompt is carefully structured to interact with the user, requesting information such as transaction amount, type, category, and checking account. Additionally, the system ensures that the user follows a specific sequence of steps to complete the transaction registration. We have broken down the main components of the prompt logic below.

Prompt Message Construction

```

1 export function financesBotPrompt({
2   checkingAccounts,
3   categories,
4 }: FinancesBotPromptProps): Groq.Chat.Completions.ChatCompletionMessageParam[] {
5   return [
6     {
7       role: 'assistant',
8       content:
9         'Voce e um chatbot de finanças que ajuda a registrar transacoes financeiras.
10        Voce deve coletar: valor, categoria, conta corrente e tipo (Entrada ou Saida).',
11     },
12     {
13       role: 'assistant',
14       content: `Estas sao suas categorias disponiveis (Nao exiba o ID para o usuario)
15        Mostre para ele em forma de lista:
16        ${categories.map((category) => ` - ${category.name} (ID: ${category.id}) `).join('\n')}
17      `,
18     },
19     {
20       role: 'assistant',
21       content: `Estas sao suas contas correntes disponiveis (Nao exiba o ID para o
22        usuario) Mostre para ele em forma de lista:
23        ${checkingAccounts.map((checkingAccount) => ` - ${checkingAccount.name} (ID: ${
24          checkingAccount.id
25        }) `).join('\n')}
26      `,
27     },
28     {
29       role: 'assistant',
30       content: 'Para registrar uma transacao, siga este processo: 1. Pergunte o valor da
31        transacao 2. Pergunte o tipo (Entrada ou Saida) 3. Mostre as categorias disponiveis
32      '
33     }
34   ]
35 }

```

```

24     e peca para escolher uma 4. Mostre as contas correntes disponiveis e peca para
25     escolher uma 5. Confirme os dados com o usuario 6. Se confirmado, chame a funcao
26     handelTransaction(), o nome da transacao gere com base no contexto a data',
27     },
28     {
29         role: 'assistant',
30         content:
31             'Se faltar qualquer informacao, peca educadamente ao usuario. Se uma categoria
32             ou conta nao existir, informe ao usuario.',
33     },
34     {
35         role: 'assistant',
36         content:
37             'Depois de chamar a funcao handelTransaction(), pergunte ao usuario se ele
38             deseja registrar outra transacao ou encerrar a sessao. Para encerrar a sessao, o
39             usuario deve digitar "sair".',
40     },
41 ]
42 }

```

Listing 1. Function that generates the chatbot prompt using user-specific categories and accounts

The function ‘financesBotPrompt’ constructs a series of messages that the chatbot will use to interact with the user. The assistant will first explain its role, then display a list of available categories and checking accounts to the user. Next, it provides instructions on how to register a transaction, ensuring that the user provides all necessary information. If the user misses any information or selects an invalid option, the assistant will kindly ask for clarification. Finally, the assistant asks whether the user wants to register another transaction or end the session.

Tool Registration

The following function registers the tool that the LLM can invoke to create a financial transaction. This function defines the available parameters for creating a transaction and provides the necessary descriptions to ensure the correct use of each parameter.

```

1  export function financesBotToolsPrompt(): Groq.Chat.Completions.ChatCompletionTool[] {
2      return [
3          {
4              type: 'function',
5              function: {
6                  name: 'handelTransaction',
7                  description: 'Cria um registro de transacao financeira',
8                  parameters: {
9                      type: 'object',
10                     properties: {
11                         name: {
12                             type: 'string',
13                             description: 'Nome ou descricao da transacao',
14                         },
15                         value: {
16                             type: 'number',
17                             description: 'O valor monetario da transacao',
18                         },
19                         balanceAdjustment: {
20                             type: 'boolean',
21                             description: 'Sera sempre falso, pois nao e um ajuste de saldo',
22                         },
23                         type: {
24                             type: 'string',
25                             description: 'Tipo da transacao, ex: INCOME, OUTCOME',
26                         },
27                         checkingAccountId: {
28                             type: 'string',

```

```

29         description: 'ID opcional da conta corrente associada',
30     },
31     categoryId: {
32         type: 'string',
33         description: 'ID opcional da categoria da transacao',
34     },
35 },
36 required: ['name', 'value', 'balanceAdjustment', 'type'],
37 },
38 },
39 },
40 ]
41 }

```

Listing 2. Tool definition available to the LLM for creating financial transactions

The function ‘financesBotToolsPrompt’ registers a tool that allows the chatbot to create a financial transaction. It defines the required parameters, such as the name, value, and type of the transaction, and specifies optional parameters like the checking account and category IDs. The ‘balanceAdjustment’ field is always set to false because it is not used for balance adjustments. The function also provides descriptions for each parameter to guide the correct usage of the tool when creating a transaction.

Groq is a high-performance AI platform designed to support LLMs. It provides low-latency and scalable infrastructure for inferencing tasks. MoneyMate uses the Groq SDK to communicate with LLM models, transforming user messages into prompts and receiving intelligent, context-aware responses. The chatbot also integrates a “tools” object, enabling the LLM to call specific methods dynamically, such as generating reports or performing API calls. This approach enhances the chatbot’s adaptability and ensures a seamless user experience.

MoneyMate uses the LLaMA-3-70B model [12], the language model used by the chatbot to interpret natural language, generate personalized financial reports, and respond to user questions. Hosted on Groq’s infrastructure, it ensures fast and accurate AI-powered interactions. We chose LLaMA 3 because it offers a strong balance between advanced language understanding and generation capabilities, which is crucial for MoneyMate’s needs. While there are lighter models available, they often sacrifice accuracy and the ability to handle complex financial language and personalized reporting. LLaMA 3’s state-of-the-art architecture ensures more precise interpretation of natural language queries and more reliable generation of tailored financial insights.

To better understand entire flow of our system architecture in practice, and how the components interact with each other, the next section provides an illustrative use-case of MoneyMate.

4.5. System Use Case

MoneyMate supports two primary interaction flows: via the **Telegram chatbot** and through the **web dashboard**. Below is an explanation of how each flow works.

1. Chatbot Interaction Flow (Telegram)

1. The user sends a message to the chatbot through the Telegram app (e.g., “How much did I spend in March?”);

2. This message is received by the **Telegram API** [23] and forwarded to the **Chatbot**, which runs on a Node.js server;
3. The chatbot interprets the message intent. If necessary, it sends a prompt to the **GROQ language model (LLM)** to generate a smart and natural response;
4. The chatbot communicates with the **Express API** to query or update user data stored in the **PostgreSQL database**;
5. The final response is sent back to the user via Telegram, providing clear and personalized financial insights.

2. Dashboard Interaction Flow (Web)

1. The user accesses the web-based financial dashboard, developed with **React**;
2. When interacting with the dashboard (e.g., adding a transaction or viewing a report), the browser sends HTTP requests to the **Express backend API**;
3. The API processes the request, performs any necessary operations, and communicates with the **PostgreSQL database**;
4. The updated or retrieved data is sent back and displayed in real time on the dashboard interface.

4.6. Deployment

The deployment of the system is structured to ensure high availability, scalability, and security across all components. The application backend and chatbot services are deployed on Vercel [24], which provides automatic scaling and optimized performance. Vercel's infrastructure allows the system to handle varying loads efficiently without manual intervention, ensuring that users experience fast and reliable responses at all times.

The database is hosted on Neon Tech [16], a cloud platform designed for PostgreSQL with serverless scalability and real-time synchronization. Neon Tech offers robust security features and seamless scaling that are critical for managing structured financial data securely and efficiently.

This separation of concerns between application deployment and database hosting ensures the system is resilient and can grow according to demand. Both platforms support the necessary security and availability requirements for a financial management system.

5. Demo

1. **Clone the Repository and Set Up Locally:** MoneyMate is an open-source project. To get started, clone the repository from our official GitHub page and follow the instructions in the README to install dependencies, configure the environment, and run the application locally.
2. **Set Up Your Checking Accounts:** In order to manage your finances, you need to register at least one checking account. After launching the application, navigate to the **Conta Corrente** section in the sidebar menu (see Figure 3) and click on **Adicionar**. Then, fill in the required details such as account name, bank, and initial balance to complete the setup (Figure 4).

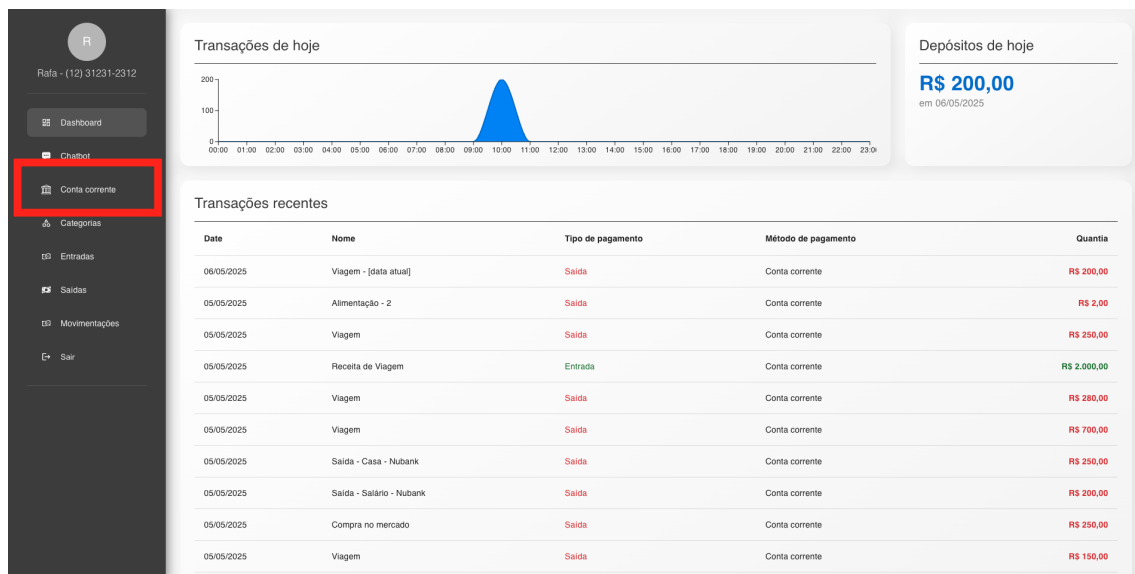


Figure 3. Checking Account on sidebar

The screenshot shows the 'Conta Corrente' management interface. A modal form titled 'Adicionar Conta Corrente' is open, containing input fields for: Nome, Nome do Banco, Número da conta, Agência, Saldo, and Taxa de Manutenção. The 'ADICIONAR' button at the bottom right of the form is highlighted with a red box. The background shows a table with columns for Nome, Taxa de Manutenção, and Ações, with one entry for 'Nubank' and a rate of 'R\$ 12.333,00'.

Figure 4. Creating Checking Account

3. **Create Your Chatbot:** To integrate the chatbot into your system, follow the steps below:

- 3.1. **Initiate Conversation with BotFather:** Start by initiating a conversation with BotFather [22] on Telegram via this link. BotFather is the official bot that helps create new Telegram bots.
- 3.2. **Obtain Your Token:** Once you've initiated the chat, type the command `/newbot` to create a new bot. Follow the instructions to choose a bot name and username. After the bot is created, BotFather will provide you with a unique **access token** (see Figures 5 and 6).

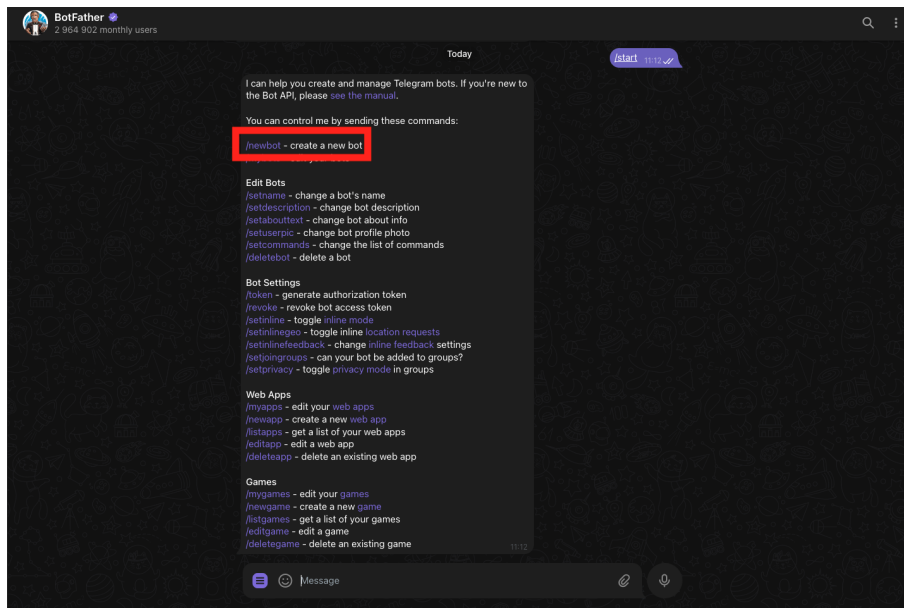


Figure 5. Creating a chatbot using BotFather

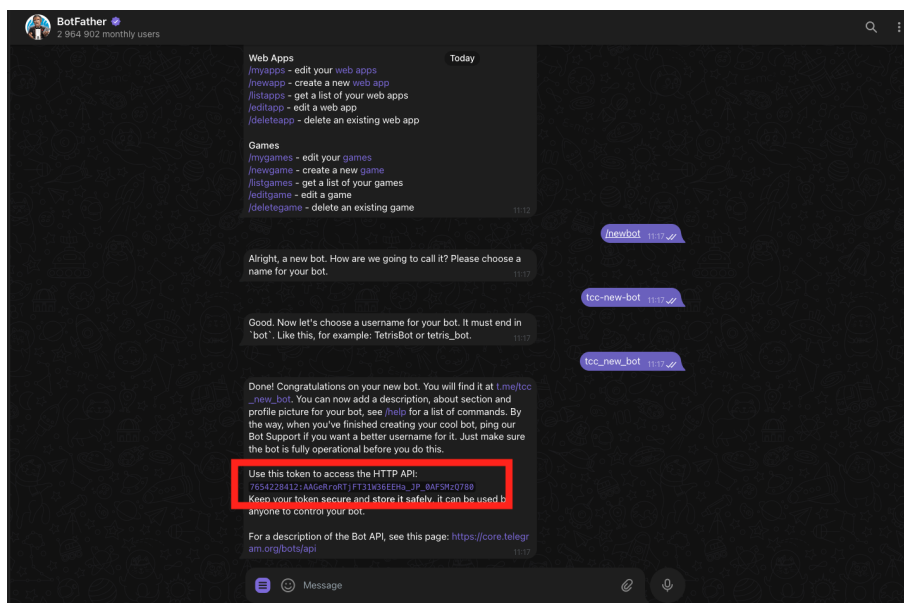


Figure 6. Bot token created

3.3. **Create a Telegram Username:** Ensure that you have a Telegram username set in your account settings. This is necessary to link your Telegram account with the chatbot (Figure 7).

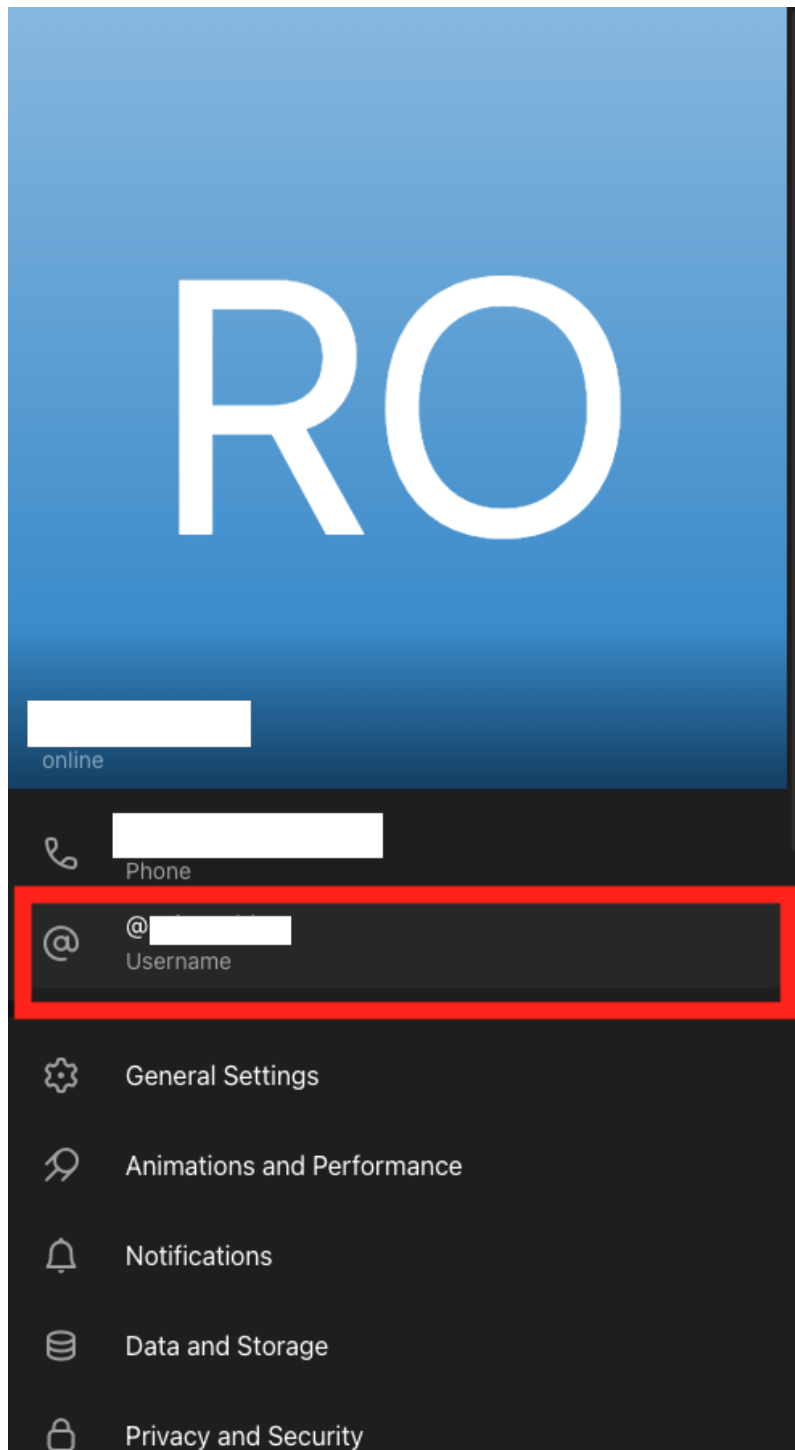


Figure 7. Telegram Username example

- 3.4. **Integrate the Token and Username in MoneyMate:** In your local instance of MoneyMate, log in to your account (you need a checking account to use this feature). In the sidebar, go to the **Chatbot** section. Paste your **access token** and **Telegram username** into the provided fields, as shown in Figure 8.

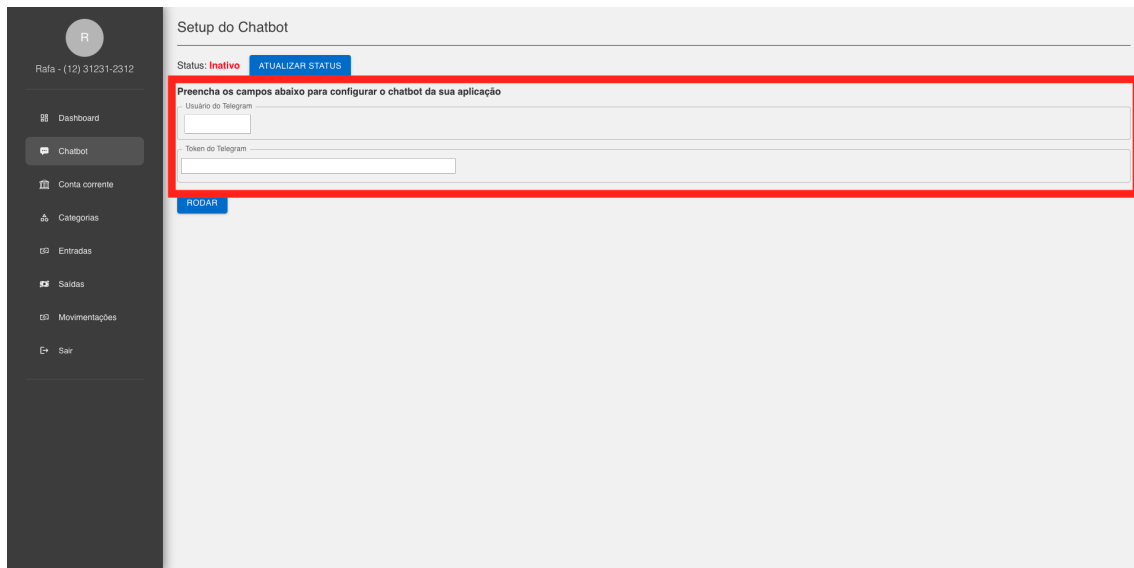


Figure 8. Setup Chatbot on MoneyMate

4. **Start Managing Your Finances:** Once the bot is running, you can register transactions and ask for financial reports. The bot will use your data to generate insights and assist you with managing your budget. An example of creating a transaction via Telegram is shown in Figure 9.

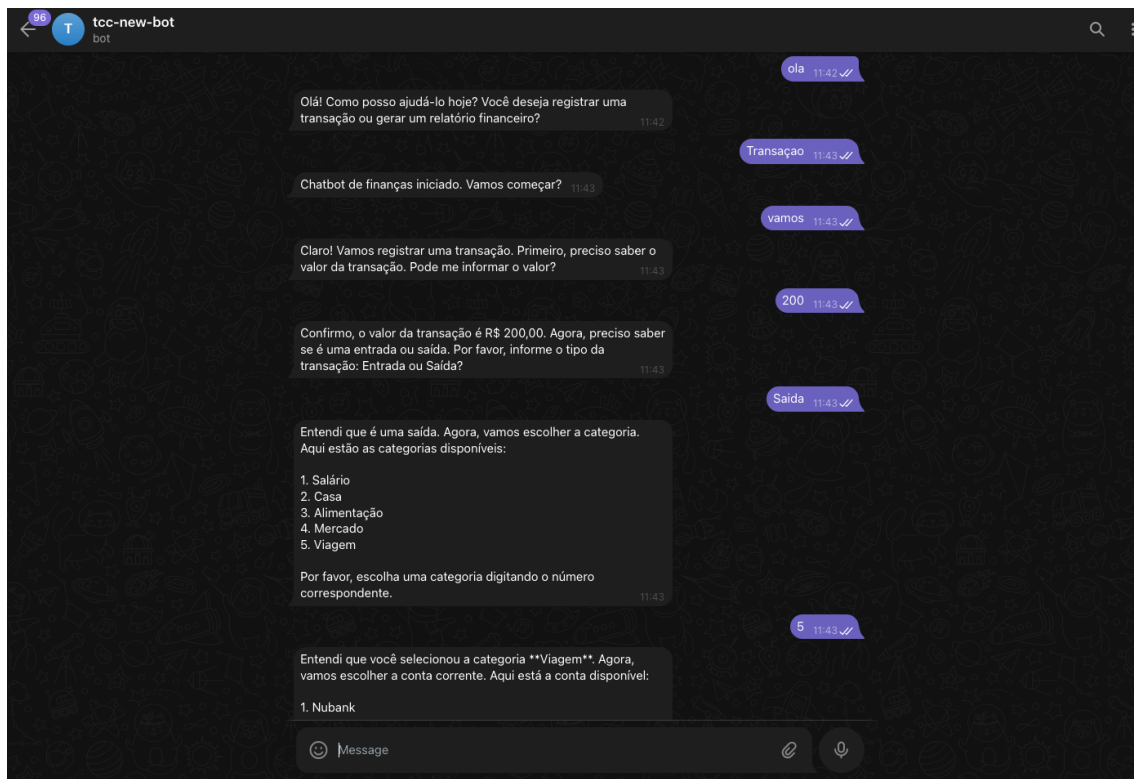


Figure 9. Creating a transaction through Telegram

5. **Chat with Groq:** Ask Groq for intelligent reports and suggestions, such as:
- “How much did I spend on food last month?”

- “Show me my monthly balance.”

Groq uses advanced AI to interpret your questions and deliver meaningful, personalized responses (see Figure 10).

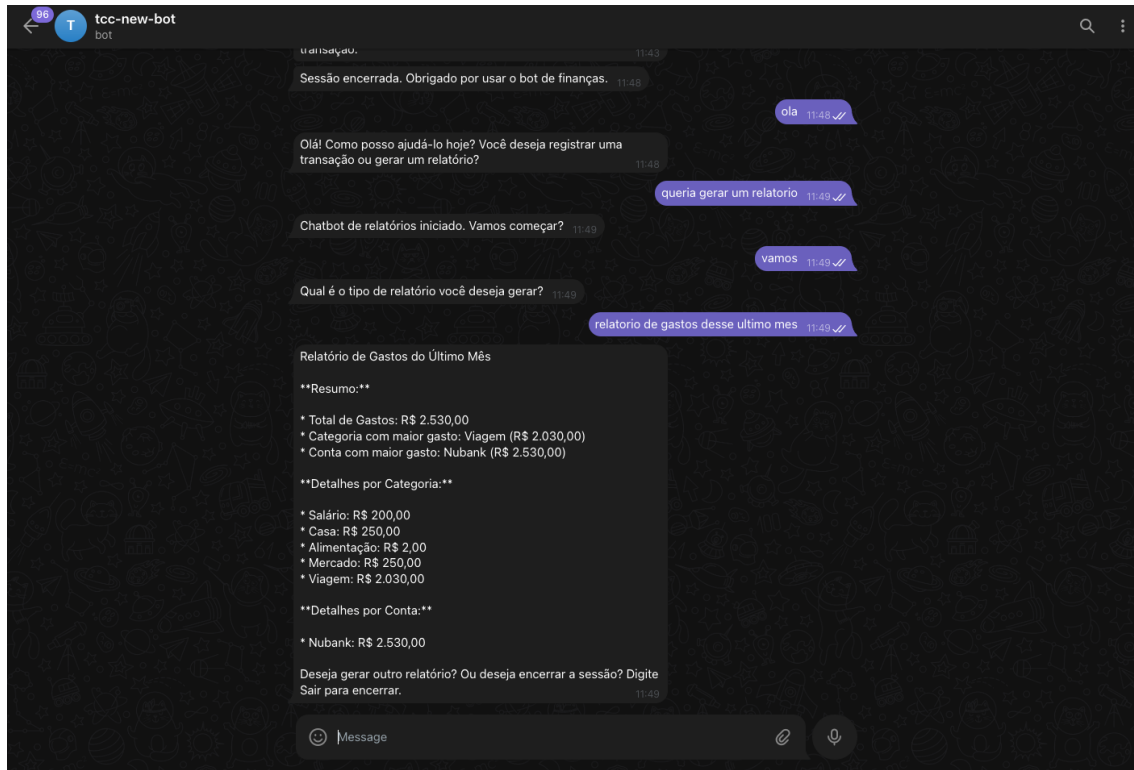


Figure 10. Generating a report through Telegram

6. Future Work

Although MoneyMate significantly improves the user experience compared to existing market solutions by integrating conversational AI and a user-friendly dashboard, there are still areas for enhancement. One notable limitation is the current onboarding process for the chatbot feature. Users are required to have a Telegram account, understand how to interact with BotFather, and manually retrieve and configure tokens. This process assumes a certain level of technical familiarity, which may pose a barrier to adoption for less tech-savvy users.

A promising direction for future development is to simplify and streamline this user experience without compromising data security or the personalized nature of the solution. One potential approach could involve embedding a guided setup directly into the platform, offering step-by-step instructions, visual feedback, or even partial automation of the bot creation and configuration process via Telegram’s API, if permitted by their terms of service.

In addition, supporting other popular messaging platforms, such as WhatsApp, would significantly increase the accessibility of the system. This would allow users to interact with the chatbot through their preferred communication channels and eliminate the dependency on a single platform, making the experience more inclusive and adaptable.

Another major improvement would be the addition of credit card support. In the

current system, there is no functionality to register credit cards or manage credit-based transactions. Incorporating this feature would provide a more complete view of a user's financial situation, especially since many users rely on credit cards for daily purchases. However, implementing such functionality presents considerable challenges, including the need to handle complex variables such as billing cycles, due dates, interest rates, installment payments, and available credit. Developing an intuitive interface for managing this data, while maintaining accuracy and clarity, would be essential for its success.

By addressing these areas, MoneyMate can evolve into a more comprehensive, secure, and accessible financial management solution for a broader range of users.

7. Conclusion

This project demonstrates the potential of integrating AI-powered chatbots with web systems to enhance personal finance management. By leveraging cutting-edge technologies such as LLMs, React, and Express, MoneyMate offers a scalable, intuitive, and intelligent solution that goes beyond traditional financial dashboards.

The use of a conversational interface powered by large language models allows users to interact with their financial data in a more natural and humanized way, reducing barriers often associated with financial tools. Coupled with a robust web dashboard, users benefit from both flexibility and control, choosing the interaction mode that best suits their needs.

Despite its achievements, the system also opens doors for further development, particularly in simplifying the bot setup process, supporting other messaging platforms beyond Telegram, and implementing advanced features like credit card tracking. These additions could significantly broaden the system's appeal and usability, reaching a wider and more diverse audience.

In conclusion, MoneyMate serves as a promising example of how AI, automation, and thoughtful UX design can be combined to create more inclusive and effective tools for personal finance management. As an open-source project, it invites collaboration, customization, and community-driven innovation—paving the way for a future where financial decision-making is smarter, easier, and more accessible to everyone.

References

- [1] Allianz. Allianz global financial literacy survey 2023, 2023. Accessed: 2025-06-06. URL: https://www.allianz.com/en/mediacenter/news/studies/230727_Allianz-Not-knowing-the-financial-basics-costs-people-not-only-self-confidence-but-also-a-lot-of-money.html?utm_source=chatgpt.com.
- [2] Back-End, 2025. Accessed: 2025-06-11. URL: <https://github.com/mtguerson/tcc-api>.
- [3] Bank of America. Erica: Virtual financial assistant by bank of america, 2025. Accessed: 2025-04-04. URL: <https://promotions.bankofamerica.com/digitalbanking/mobilebanking/erica>.

- [4] ChatBot, 2025. Accessed: 2025-06-11. URL: <https://github.com/Reshzera/tcc-telegram-bot>.
- [5] Cleo. Cleo: A conversational chatbot for personal financial management, 2025. Accessed: 2025-04-04. URL: <https://web.meetcleo.com/>.
- [6] Express. Fast, unopinionated, minimalist web framework for node.js, 2025. Accessed: 2025-05-26. URL: <https://expressjs.com/>.
- [7] FinZap. Finzap: Controle financeiro via whatsapp, 2025. Accessed: 2025-04-04. URL: <https://usefinzap.com/pv>.
- [8] Front-End, 2025. Accessed: 2025-06-11. URL: <https://github.com/mtguerson/tcc-front>.
- [9] Groq. Fast ai inference., 2025. Accessed: 2025-05-26. URL: <https://groq.com/>.
- [10] Intuit. Mint: Personal finance & budgeting tool, 2025. Accessed: 2025-04-04. URL: <https://mint.intuit.com/>.
- [11] Kasisto. Kai: Conversational ai platform for financial services, 2025. Accessed: 2025-04-04. URL: <https://kasisto.com/products/kai-platform/>.
- [12] Llama3. open-source ai models you can fine-tune, distill and deploy anywhere., 2025. Accessed: 2025-05-28. URL: <https://www.llama.com/models/llama-3/>.
- [13] Meu assistente. Meu assistente – inteligência artificial, 2025. Accessed: 2025-05-22. URL: <https://ai.meuassistente.com/>.
- [14] Meu Dinheiro. Meu dinheiro: Controle financeiro pessoal online, 2025. Accessed: 2025-04-04. URL: <https://www.meudinheiroweb.com.br/>.
- [15] Mobills. Mobills: Aplicativo de controle financeiro pessoal, 2025. Accessed: 2025-04-04. URL: <https://www.mobills.com.br/>.
- [16] Neon. Neon: Ship faster with postgres, 2025. Accessed: 2025-06-09. URL: <https://neon.com/>.
- [17] Node. Run javascript everywhere, 2025. Accessed: 2025-05-26. URL: <https://nodejs.org/>.
- [18] Organizze. Organizze: Gerenciador financeiro online, 2025. Accessed: 2025-04-04. URL: <https://www.organizze.com.br/>.
- [19] PostgreSQL. The world's most advanced open source relational database, 2025. Accessed: 2025-05-26. URL: <https://www.postgresql.org/>.
- [20] React. The library for web and native user interfaces, 2025. Accessed: 2025-05-22. URL: <https://react.dev/>.
- [21] Telegram. A cloud-based mobile and desktop messaging app with a focus on security and speed., 2025. Accessed: 2025-05-26. URL: <https://web.telegram.org/>.
- [22] Telegram BotFather. Botfather: Official telegram tool for creating and managing bots, 2025. Accessed: 2025-06-06. URL: <https://telegram.me/BotFather>.
- [23] TelegramAPI. Allow you to build your own customized telegram clients., 2025. Accessed: 2025-05-26. URL: <https://core.telegram.org/>.

- [24] Vercel. Vercel: Your complete platform for the web., 2025. Accessed: 2025-06-09. URL: <https://vercel.com/>.
- [25] You Need A Budget. Ynab: Budgeting software for personal finance, 2025. Accessed: 2025-04-04. URL: <https://www.ynab.com>.