LLM CAPSTONE PROJECT

Project Report: Smart Expense Tracking Application

Name: TEJAS C Registration No: 2448553

Abstract

This report details the development of a Smart Expense Tracking Application, a capstone project designed to provide users with intelligent financial insights. The application moves beyond traditional expense logging by incorporating a context-aware chatbot, powered by a Large Language Model (LLM), to analyze spending habits and deliver personalized savings advice. The system architecture is composed of a backend database managed by Supabase, a core logic layer with specialized "agent" modules for financial analysis, and a user-friendly interface built with Streamlit. The primary innovation lies in its ability to use a user's own financial data as a dynamic context vector, enabling the chatbot to provide relevant, actionable recommendations for financial improvement. The project successfully demonstrates a practical application of LLMs in creating interactive and personalized personal finance tools.

Methodology

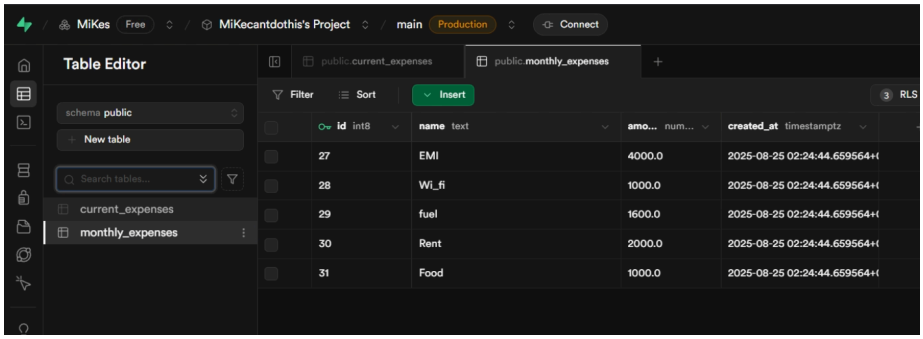
The project was executed using a modular three-tiered architecture, ensuring a clear separation of concerns between data management, core logic, and user interaction.

1. System Architecture The application is structured into three primary modules:

- Database Operations: Manages all data persistence and retrieval.
- Agents: Contains the core analytical and suggestion-generating logic.
- User Interface (UI): Provides the front-end for user interaction and data visualization.

2. Data Management and Backend A SQL database managed via the Supabase platform serves as the application's backend. This choice facilitates robust and

scalable data management, allowing for the secure storage and retrieval of user-specific financial data, including monthly and discretionary expenses. The application interacts with the database to push new expense entries and pull existing data for analysis.



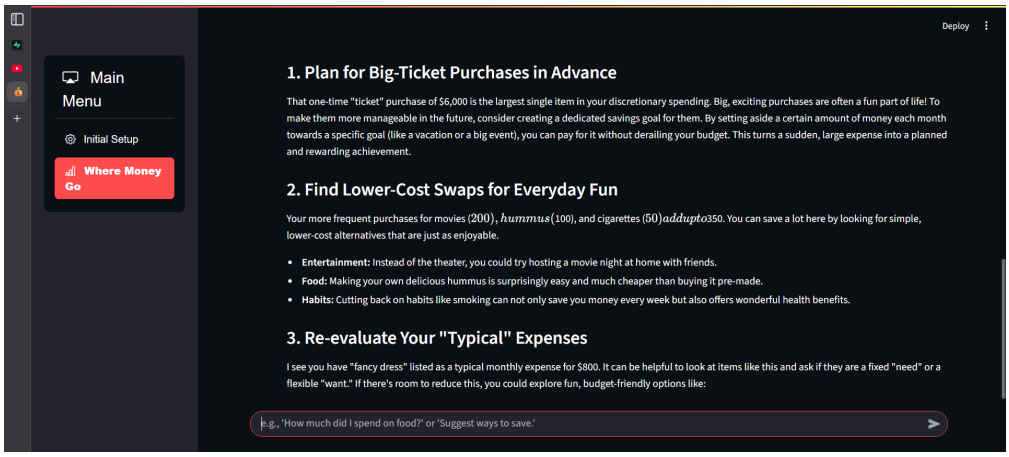
3. Core Logic and AI Agents The intelligence of the application is driven by two specialized Python agents that leverage LLM prompt templates:

- Expense Analysis Agent: This agent is designed to function as an expert financial analyst. It takes a user's expense data (formatted as a pandas DataFrame) and a specific user query as input. It processes this information to provide a clear and concise answer regarding spending patterns.
- Savings Suggestion Agent: This agent acts as a helpful financial advisor. It analyzes a user's complete financial situation, including income, typical monthly expenses, and recent discretionary spending. Based on this holistic view, it generates personalized, actionable tips on how the user can save money, focusing initially on discretionary spending in an encouraging and non-judgmental tone.

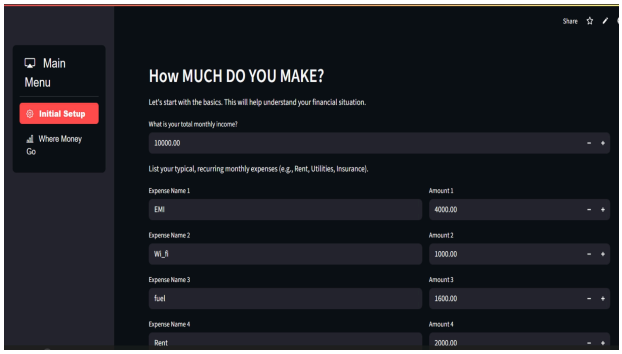
```python
class ExpenseAnalysisAgent:
    """
    An agent that analyzes expense data and answers user questions.
    """
    def __init__(self):
        self.llm = llm
        self.prompt_template = PromptTemplate(
            input_variables=['expenses_data', 'question'],
            template="""
            You are an expert financial analyst. Here is a summary of recent expenses:
            {expenses_data}

            Please answer the following question based on this data:
            Question: {question}

            Provide a clear and concise answer.
            """
        )
        self.chain = self.prompt_template | self.llm | StrOutputParser()

    def analyze(self, expenses_df: pd.DataFrame, question: str) -> str:
        """
        Analyzes the provided expense data to answer a user's question.

        Args:
            expenses_df: A pandas DataFrame of the user's current expenses.
            question: The user's question.

        Returns:
            A string containing the agent's answer.
```

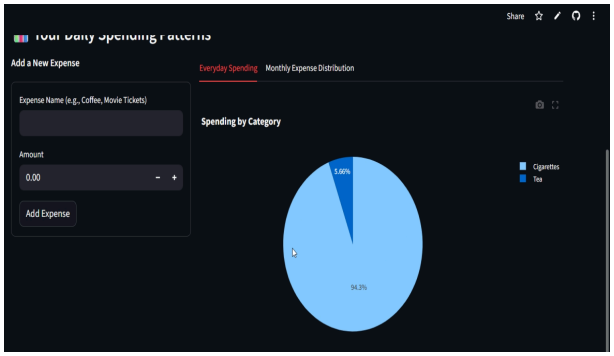
```python
class SavingsSuggestionAgent:
    """
    An agent that suggests ways to save money based on expense data.
    """
    def __init__(self):
        self.llm = llm
        self.prompt_template = PromptTemplate(
            input_variables=['income', 'monthly_expenses', 'current_expenses'],
            template="""
            You are a helpful financial advisor. A user needs help saving money.
            Here is their financial situation:
            - Monthly Income: ${income}
            - Typical Monthly Expenses (e.g., rent, utilities):
            {monthly_expenses}
            - Recent Discretionary Spending:
            {current_expenses}

            Based on this information, please provide 3-5 actionable and personalized tips on how they can save money.
            Focus on the discretionary spending first. Be encouraging and non-judgmental.
            """
        )
        self.chain = self.prompt_template | self.llm | StrOutputParser()

    def suggest(self, income: float, monthly_df: pd.DataFrame, current_df: pd.DataFrame) -> str:
        """
        Generates savings suggestions.

        Args:
            income: The user's monthly income.
            monthly_df: DataFrame of typical monthly expenses.
```

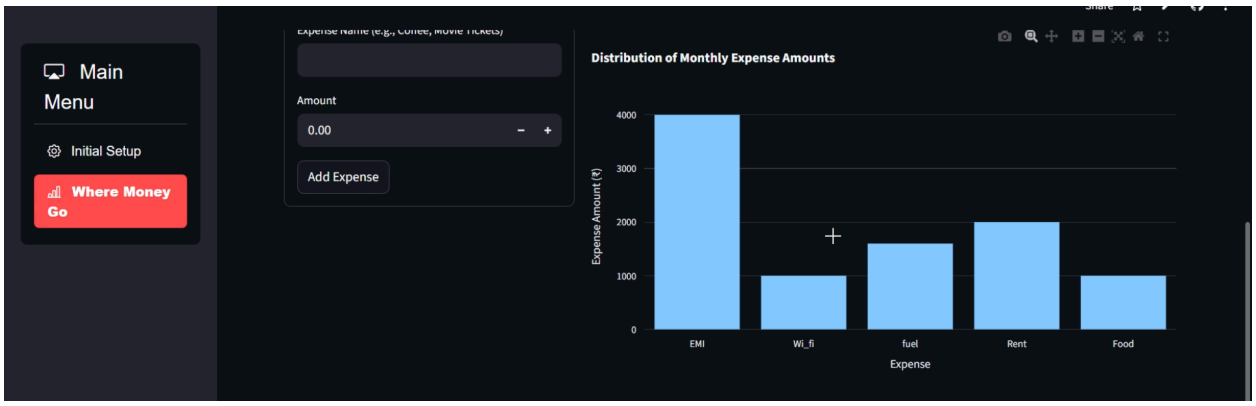
4. Chatbot Implementation and Contextualization The interactive chatbot is the central feature of the application. It is context-driven, meaning its responses are directly informed by the user's financial data. To achieve this, data from the Supabase tables is used to create a context vector. This vector provides the necessary background information for the LLM-powered agents to generate responses and suggestions that are specifically tailored to the user's financial activities, rather than providing generic advice.



5. User Interface The user interface was developed using Streamlit, a Python framework for building simple and interactive web applications. The UI is designed for ease of use, allowing users to perform initial setup, add new expenses, view visualizations of their spending (such as the distribution of monthly expenses by category), and interact directly with the chatbot to ask questions and receive financial guidance.







Conclusion

This project successfully achieved its objective of creating a smart expense tracking application that provides significant value beyond simple data logging. By integrating a context-aware chatbot powered by specialized LLM agents, the application offers users a dynamic and interactive way to understand their financial health. The modular architecture, utilizing Supabase for the backend and Streamlit for the frontend, proved to be an effective and efficient development stack.

The final product is a functional prototype that can accurately track expenses, visualize spending distributions, and, most importantly, provide personalized and actionable savings advice. This demonstrates the powerful potential of applying LLMs to personal finance, transforming raw financial data into meaningful, conversational insights that can empower users to make better financial decisions. Future development could focus on expanding the analytical capabilities, integrating with financial institutions for automatic transaction importing, and adding features for long-term financial goal setting.