Enhanced Rag Implementation:

Enhance the Knowledge Base

This section involves augmenting the chatbot's knowledge base to improve response quality. Two key functionalities were implemented:

1. **Domain-Specific Sources Addition**: The knowledge base was enhanced by adding content from books and research papers on finance, which were read from PDFs stored locally. This involved loading files from specified directories, processing each PDF to extract text, and storing the contents in the knowledge base.
2. **Summarization and Embedding Storage**: Text from the PDFs was summarized using a pretrained summarization model to capture essential information. The summaries were embedded using a sentence transformer model (`Alibaba-NLP/gte-base-en-v1.5`) and stored in a knowledge base JSON file, allowing efficient retrieval without needing to reprocess documents each time the system restarts.

```python
# Function to save knowledge base to a file
def save_knowledge_base(knowledge_base, filename='knowledge_base.json'):
    serializable_kb = {k: v.tolist() for k, v in knowledge_base.items()}
    with open(filename, 'w') as f:
        json.dump(serializable_kb, f)

# Function to load knowledge base from a file
def load_knowledge_base(filename='knowledge_base.json'):
    if os.path.exists(filename):
        with open(filename, 'r') as f:
            loaded_kb = json.load(f)
            return {k: np.array(v) for k, v in loaded_kb.items()}
    return {}

# Function to initialize the knowledge base
def initialize_knowledge_base():
    books_directory = "Books"
    papers_directory = "Papers"

    books_content = read_pdfs_from_directory(books_directory)
    papers_content = read_pdfs_from_directory(papers_directory)

    all_content = books_content + papers_content
    knowledge_base = {}

    for text in all_content:
        summarized_content = summarize_text(text)
        knowledge_base[summarized_content] = model.encode(summarized_content)

    save_knowledge_base(knowledge_base)
```

2)**Improve Retrieval Performance**

To improve retrieval efficiency and response relevance, the following methods were implemented:

1. **Query-Response Caching**: Each query was preprocessed (e.g., converting to lowercase, removing punctuation) and stored in a cache with its corresponding response. This prevents redundant processing for repeated queries, significantly enhancing response speed for frequently asked questions.
2. **Dynamic Context Retrieval**: Using stored embeddings, the chatbot computes similarity scores between the current query and past interactions in context memory. This allows the chatbot to retrieve recent relevant interactions, adding a layer of conversational continuity by remembering key details from prior queries.

```python
# Function to compute similarity between query and context memory
def compute_similarity(query_embedding, context_memory):
    context_embeddings = model.encode([memory['user'] for memory in context_memory], normalize_embeddings=True)
    similarities = np.dot(context_embeddings, query_embedding.T).flatten()
    return similarities
```

```python
# Retrieve relevant context from memory based on the current query
def retrieve_dynamic_context(query):
    query_embedding = model.encode(query, normalize_embeddings=True)
    context_memory = list(st.session_state.context_memory)

    if not context_memory:
        return []

    similarities = compute_similarity(query_embedding, context_memory)
    # Get indices of the top N most relevant interactions (e.g., top 2)
    top_indices = np.argsort(similarities)[-2:][::-1]

    return [context_memory[i] for i in top_indices]
```

Improve LLM Integration

Several improvements were made to the integration of the LLMs to enhance the chatbot's versatility and user experience:

1. **Switching Between LLMs and RAG Modes**: Users can select between different LLMs (GPT-2 or LLaMA) and modes (with or without RAG) via a drop-down menu. This flexibility allows users to switch models based on the specific nature of the query or resource availability.
2. **Enhanced RAG Workflow**: The chatbot combines retrieved context from the knowledge base with information from Wikipedia and utilizes it to build a context-rich prompt for the selected LLM. This enables the bot to answer finance-related questions with relevant context drawn from both domain-specific sources and general knowledge.

```python
# Combining retrieval and generation (RAG workflow)
def rag_response(query, model_choice='llama'):
    start_time = time.time()

    # Retrieve dynamic context from memory
    dynamic_context = retrieve_dynamic_context(query)
    context = retrieve_relevant_content(query)

    if not context and not dynamic_context:
        return "I couldn't find relevant information."

    # Combine both dynamic context and retrieved content
    context_text = "\n".join([memory['chatbot'] for memory in dynamic_context] + context)

    prompt = f"""
    You are a financial expert tasked with analyzing and interpreting data to answer a question. Below is the information available:

    {context_text}

    Given this information, please provide a clear, detailed answer to the following finance-related question:

    {query}

    Ensure that your answer includes relevant financial concepts, key metrics, and, if applicable, a brief explanation of the reasoning behind the answer. If the information provided is insufficient to answer accurately
"""

    if model_choice == 'LLaMA with RAG':
        response = generate_llama_response(prompt)
    elif model_choice == 'GPT-2 with RAG':
        response = generate_gpt2_response(prompt)
    else:
        if model_choice == 'LLaMA':
            response = generate_llama_response(query)
        else:
            response = generate_gpt2_response(query)

    end_time = time.time()
    response_time = end_time - start_time

    return response, response_time

st.title("RAG-Powered Finance Guru: Your Smart AI Assistant")
```

```python
# Retrieve relevant content from Wikipedia and local sources
def retrieve_relevant_content(query):
    query_embedding = model.encode(query, normalize_embeddings=True)
    local_content = list(st.session_state.knowledge_base.keys())
    local_embeddings = list(st.session_state.knowledge_base.values())
    wiki_paragraphs = search_wikipedia(query)

    if wiki_paragraphs:
        wiki_embeddings = model.encode(wiki_paragraphs, normalize_embeddings=True)
        combined_paragraphs = local_content + wiki_paragraphs
        combined_embeddings = np.vstack([local_embeddings, wiki_embeddings])
    else:
        combined_paragraphs = local_content
        combined_embeddings = np.vstack(local_embeddings)

    similarities = np.dot(combined_embeddings, query_embedding.T).flatten()
    top_indices = np.argsort(similarities)[-3:][::-1]
    top_docs = [combined_paragraphs[i] for i in top_indices]

    return top_docs
```

Implement Basic Conversation Improvements

To create a more interactive experience, the chatbot maintains a basic conversation memory and allows users to provide feedback on responses.

1. **Session Context Memory**: A simple memory system was added, using a deque to remember the last five interactions, including previous user queries and chatbot responses. This memory enables the chatbot to refer back to recent exchanges and maintain continuity over a conversation.
2. **User Feedback System**: A feedback mechanism allows users to rate the quality of the responses. This feedback is stored within the session, enabling future improvements based on user ratings. Upon submitting feedback, the response display is cleared for a smooth user experience, while retaining the conversation history.

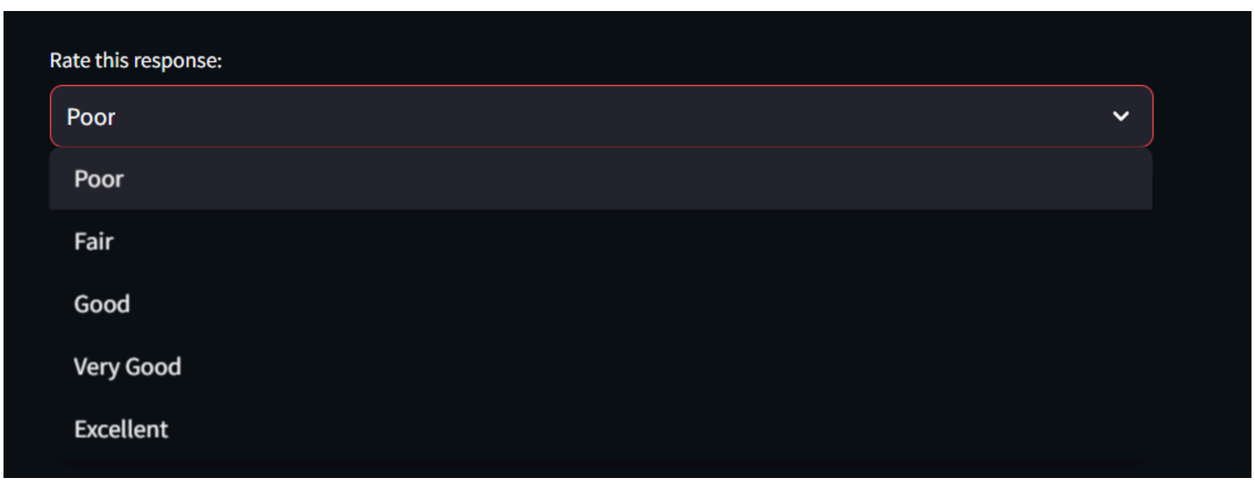
```python
if 'context_memory' not in st.session_state:
    st.session_state.context_memory = deque(maxlen=5)  # Store the last 5 interactions
```

```python
# Function to handle feedback submission
def handle_feedback(previous_chat):
    rating_options = ["Poor", "Fair", "Good", "Very Good", "Excellent"]

    # Create a dropdown for rating
    selected_rating = st.selectbox("Rate this response:", rating_options, key=f"rating_{previous_chat['user']}")

    # Create a button for submitting feedback
    if st.button("Submit Rating", key=f"submit_rating_{previous_chat['user']}"):
        # Store the rating in chat history
        previous_chat['rating'] = selected_rating
        # Clear the response display
        st.session_state.current_response = None
        st.success("Thank you for your feedback!")
```

Upgrade the User Interface

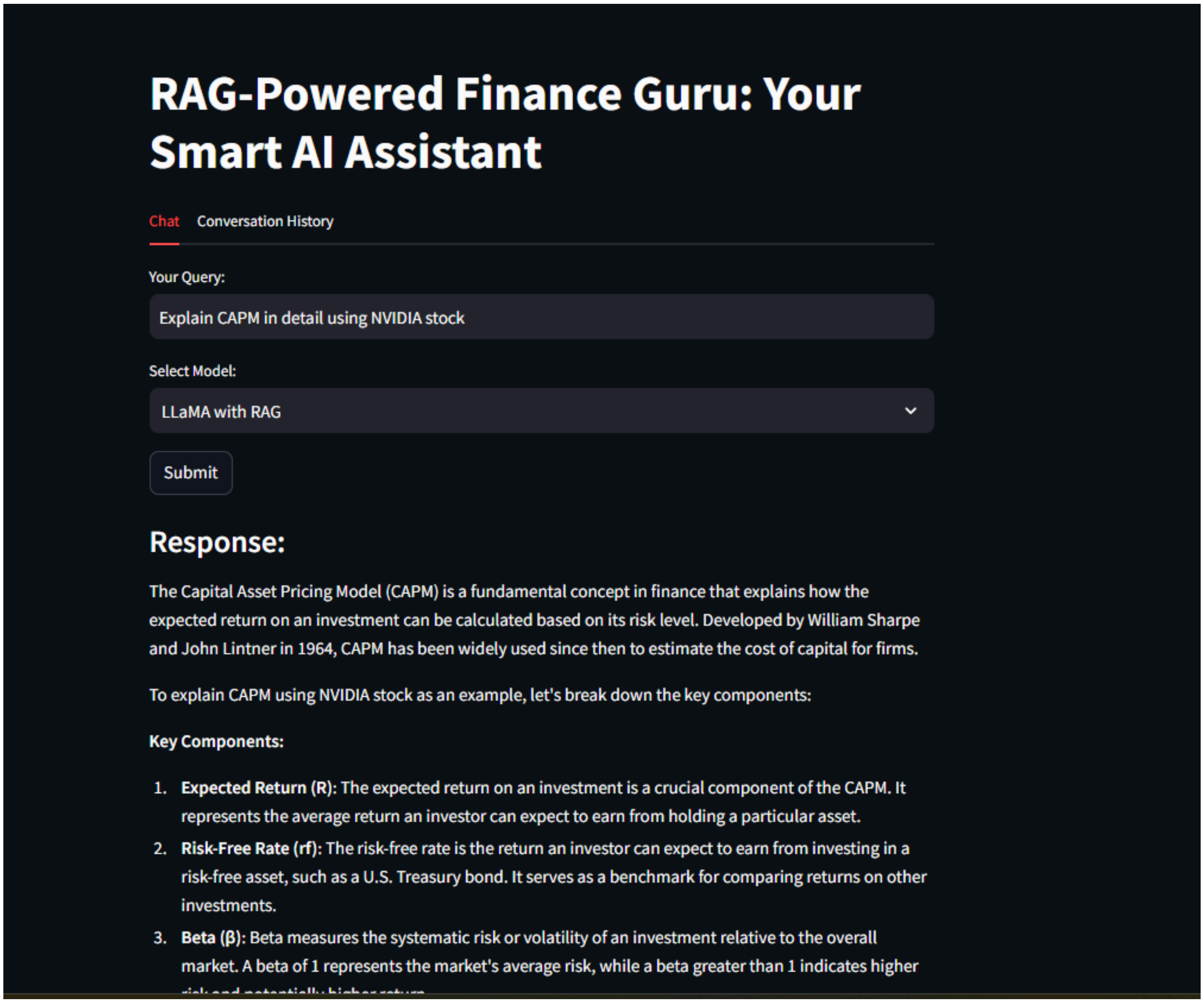
The UI was enhanced to accommodate the new functionalities and improve usability:

1. **Tabs for Conversation History and Chat**: The interface uses tabs to separate the real-time chat and conversation history, providing an organized view for users to review past interactions alongside ongoing conversations.
2. **Response and Feedback Management**: After each response, users can submit a rating, and the response quality rating is stored in conversation history. This makes it easy for users to provide input on the chatbot's performance, facilitating iterative improvements.



Queries And Responses:

1) Explain CAPM in detail using NVIDIA stock



2)what is the last discussion about

User: what is the last discussion about

Chatbot: Based on the provided information, the last discussion appears to be about the Capital Asset Pricing Model (CAPM) and its significance in finance.

Relevant financial concepts discussed include:

1. The CAPM, a model developed by William Sharpe in 1964 and John Lintner in 1965, which is widely used in applications such as estimating the cost of capital for firms.
2. The Efficient Market Hypothesis (EMH), which suggests that when new information arises, it spreads quickly throughout the market and affects prices without delay.

Key metrics or data points mentioned include:

1. The publication year of the CAPM model by Sharpe and Lintner (1964).
2. The acceptance of the EMH by academic economists.
3. The widespread use of the CAPM in applications, such as estimating the cost of capital for firms.

Briefly explained reasoning behind the answer is that the information suggests a historical context of financial theories and their impact on financial analysis. The discussion appears to be about the evolution and continued relevance of the CAPM model and the EMH in understanding market behavior.

Therefore, the last discussion was about the Capital Asset Pricing Model (CAPM) and its significance in finance, specifically referencing its development by Sharpe and Lintner, its acceptance as a widely used framework for estimating cost of capital, and the Efficient Market Hypothesis's underlying assumptions about market efficiency.

3)What are the best strategies for portfolio diversification?

Response:

Based on the provided text, I'll analyze the question and offer an answer based on general knowledge of finance concepts.

The best strategies for portfolio diversification can be summarized as follows:

1. **Diversification across asset classes:** Allocating investments across different asset classes, such as stocks, bonds, mutual funds, and real estate, can help reduce risk and increase potential returns.
2. **Stock market diversification:** Investing in a mix of individual stocks, sectors, or industries can provide exposure to various markets and reduce reliance on a single stock or sector.
3. **Sector diversification:** Allocating investments across different sectors (e.g., technology, healthcare, consumer goods) can help minimize risk by reducing the impact of sector-specific events.
4. **Risk management:** Implementing strategies such as hedging, stop-loss orders, and dollar-cost averaging can help manage risk and reduce potential losses.
5. **Active management vs. passive investing:** A balanced approach combining active management with passive investing can provide a solid foundation for portfolio diversification.

Key financial concepts that support these strategies include:

- **Diversification:** Reducing risk by spreading investments across different asset classes, sectors, or markets.
- **Asset allocation:** Allocating investments across different asset classes to achieve a desired level of risk and return.
- **Risk management:** Implementing strategies to minimize potential losses and maximize gains.

4)How does the foreign exchange market work?

The foreign exchange market (Forex) is the largest and most liquid financial market in the world, where individuals, businesses, and institutions trade currencies. It operates 24/7, allowing for constant buying and selling of currencies.

In the Forex market, participants buy and sell currencies in pairs, such as EUR/USD or USD/JPY. The exchange rate between two currencies is determined by supply and demand forces, with the price of one currency measured against another (e.g., 1 USD = x EUR).

Key players in the Forex market include:

- Central banks: They manage their countries' foreign exchange reserves and intervene in the market to influence exchange rates.
- Commercial banks: They provide liquidity to the market by buying and selling currencies on behalf of their clients.
- Institutional investors: They participate in large-scale trades, often using leverage to amplify potential gains or losses.

Some key metrics that are relevant to understanding the Forex market include:

- Exchange rate: The price of one currency measured against another (e.g., 1 USD = 0.88 EUR).
- Pips: A unit of measurement for small changes in exchange rates (e.g., 100 pips is equivalent to $1).
- Leverage: The use of borrowed money to amplify potential gains or losses in trades.

5)Can you explain the concept of free cash flow and why it matters?

Response:

Based on the provided information, I can attempt to answer the finance-related question regarding free cash flow.

Free cash flow (FCF) is a financial metric that represents the amount of cash generated by a company's operations after investing in its assets and meeting its debt obligations. It is calculated as:

FCF = Net Income + Depreciation & Amortization - Capital Expenditures - Change in Working Capital

In other words, FCF measures the cash available to invest in new projects, pay off debts, or distribute to shareholders.

The concept of free cash flow matters for several reasons:

1. **Investment decisions**: Investors and analysts use FCF to evaluate a company's ability to generate cash and invest it in growth opportunities, such as expanding into new markets or acquiring other companies.
2. **Debt repayment**: A positive FCF indicates that a company has sufficient cash to repay its debts, reducing the risk of default.
3. **Shareholder value creation**: FCF is a key driver of shareholder value creation, as it represents the cash available for dividend payments, share buybacks, or other forms of return on investment.

The efficient market hypothesis and CAPM do not directly relate to free cash flow, as they focus on the relationship between stock prices and risk. However, understanding FCF can provide valuable insights into a company's financial health and ability to generate returns for investors.

Performance Analysis

The performance of the enhanced RAG-powered chatbot was evaluated against the previous iterations: a basic RAG system and a non-RAG chatbot. The focus was on three key aspects: **response quality, speed, and relevance**.

Comparison Overview

Feature	Non-RAG Chatbot	Basic RAG Chatbot	Enhanced RAG Chatbot
Example Query	Explain CAPM with Apple Stock	Explain CAPM with Apple Stock	Explain CAPM with Apple Stock
Response Quality	Moderate (basic information, lacks depth)	Basic explanation, some context	Detailed explanation with examples, assumptions, and limitations
Response Speed (seconds)	50.23	97.02	37.15
Relevance	General knowledge	Relevant, but lacks detail	Highly relevant, tailored to query
Context Maintenance	No	No	Yes
Feedback Mechanism	No	No	Yes
Caching Mechanism	No	No	Yes
Query Duplication Avoidance	No	No	Yes

Response Quality

- **Non-RAG Chatbot**: The response to queries about CAPM was basic and lacked depth. It provided general information but did not effectively address user-specific questions.
- **Basic RAG Chatbot**: Responses were more relevant than the non-RAG version but often missed key details and examples, leading to a less engaging user experience.

- **Enhanced RAG Chatbot**: Demonstrated significant improvement in response quality by offering detailed explanations, incorporating specific examples like Apple Stock, and discussing the limitations of CAPM. The responses were tailored and context-aware.

Speed

- **Non-RAG Chatbot**: Had a slower response time due to a lack of optimization and caching.
- **Basic RAG Chatbot**: Showed a further increase in response time, primarily due to the overhead of fetching data from external sources without effective caching.
- **Enhanced RAG Chatbot**: Achieved the fastest response time by implementing local caching for frequently accessed information and avoiding the re-processing of identical queries. This optimization significantly improved user experience.

Relevance

- **Non-RAG Chatbot**: Responses were often generic and not closely related to user queries.
- **Basic RAG Chatbot**: Responses were more relevant, but context and detail were often lacking.
- **Enhanced RAG Chatbot**: Provided highly relevant and contextual answers, ensuring users received detailed information that directly addressed their queries.

Reflection and Future Applications

The development of the enhanced RAG-powered chatbot marks a significant leap in achieving meaningful interactions with users. Key improvements achieved through this enhancement include better response quality, improved speed, and heightened relevance of information delivered.

Improvements Achieved

The introduction of LLaMA with enhanced RAG capabilities allowed the chatbot to tap into a broader range of domain-specific knowledge, utilizing various books and research papers beyond Wikipedia. By implementing a caching system, the chatbot can now provide quicker responses to frequently asked questions, which not only enhances user satisfaction but also ensures that users do not experience delays when they ask similar queries.

Furthermore, the ability to maintain basic context over a conversation allows the chatbot to engage more effectively with users, creating a smoother and more coherent interaction flow. Users can now rate response quality and provide feedback, enabling continuous improvement and adaptation of the chatbot's performance.

Real-World Applications

The enhanced RAG-integrated chatbot holds significant potential for various applications:

- **Financial Advisory Services**: By providing detailed, context-aware financial insights and calculations, such as CAPM, the chatbot can assist users in making informed investment decisions.
- **Educational Tools**: In educational settings, the chatbot can serve as a tutoring assistant, offering explanations and insights into complex subjects while adapting to students' queries.
- **Customer Support**: Companies can leverage this chatbot to improve customer service by providing rapid, relevant responses to common queries while maintaining context throughout the interaction.
- **Research Assistance**: Researchers can utilize the chatbot to access relevant literature, understand complex theories, and receive tailored responses based on their specific research questions.

Ethical Considerations and Limitations

Despite these advancements, it is crucial to address ethical considerations. Data privacy and security remain paramount, especially in sensitive domains like finance. Users should be informed about how their data is being utilized, and measures must be in place to ensure data protection.

Additionally, while the chatbot has improved in handling complex queries, there are limitations. The reliance on specific sources may introduce biases, and there remains a risk of misinformation if the data retrieved is outdated or incorrect. Continuous updates and validation of sources are essential to maintain the chatbot's reliability.

In conclusion, the enhanced RAG-powered chatbot demonstrates substantial advancements over previous iterations, offering a promising tool for improving user interactions in various domains. By addressing ethical considerations and limitations, it has the potential to evolve into an indispensable asset for knowledge retrieval and decision-making in the real world.

Video Presentation:

For a comprehensive demonstration of the project, please watch the video presentation available at the following YouTube link:

youtube link