**Retrieval-Augmented Generation (RAG) QA Bot**

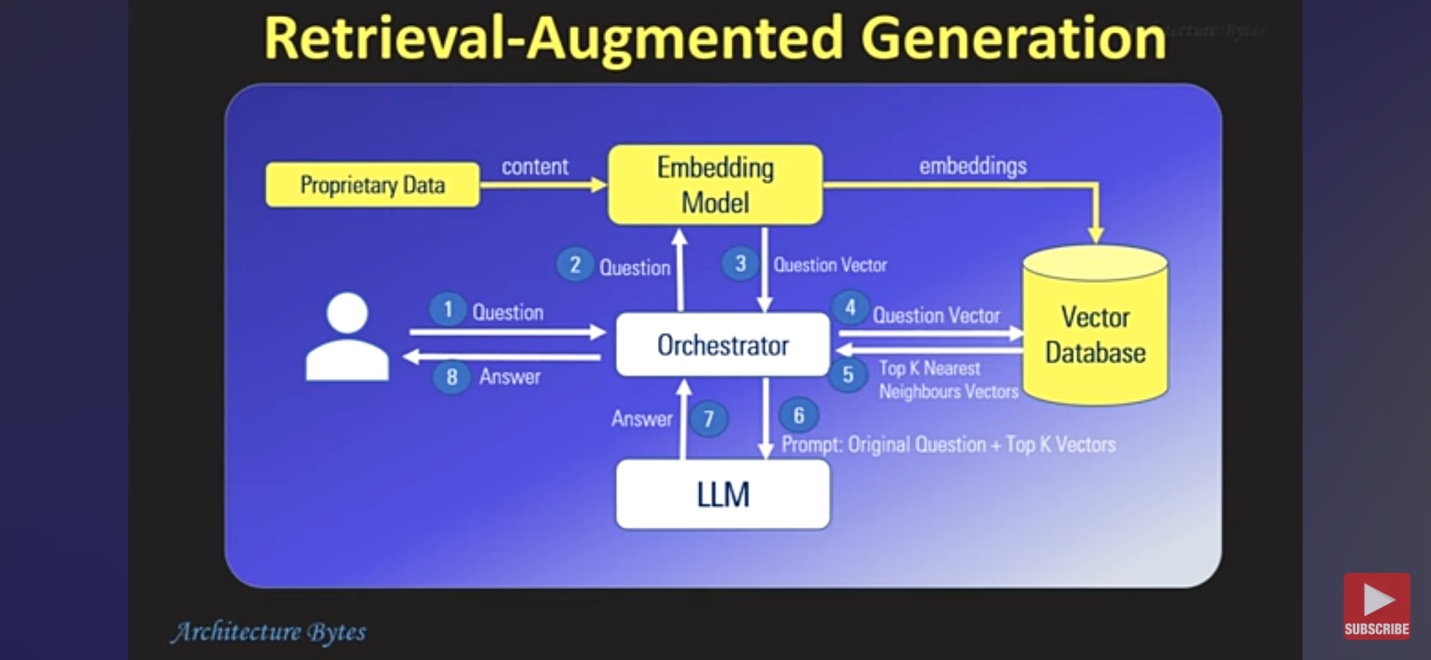
1. **Overview**

This project aims to develop a Question Answering (QA) bot that leverages a Retrieval-Augmented Generation (RAG) approach. The bot is designed to retrieve relevant information from a stock market dataset (2019-2023) and provide detailed answers based on stock price data. It uses a vector database for retrieval and generative AI for generating coherent responses.

The pipeline includes data loading, embedding creation, vector store querying, and question-answering using Cohere's generative model.

1. **Model Architecture**

**How generative responses are created.**

****

The vector store is built using Weaviate, which stores the stock data embeddings.

Queries are processed by the retrieval model, which finds the most relevant data points (stock information).

I used…

**1. Weaviate Vector Database**

**Purpose**:

Weaviate was used as the vector database to store the embeddings of the stock data.

It enabled efficient similarity-based retrieval, allowing the system to find relevant stock price data for a given user query.

* After embedding the stock data, the embeddings were stored in Weaviate.
* When a user asks a question, the query is converted into an embedding, and Weaviate retrieves the most relevant matching documents (e.g., stock prices of specific companies or dates).
* The retrieved data is passed on to the generative model for response generation.

**Advantages**:

* + Fast and efficient vector search across large datasets.
  + Scalable storage and easy integration with LangChain for retrieval.

**2. Cohere API and LLM**

**Purpose:**

* Cohere API was used for both embedding generation and as a generative language model (LLM).
* The embeddings created using Cohere represent the stock data in vector form for similarity-based search in Weaviate.
* The LLM generates coherent, natural language responses to the user’s questions based on the retrieved data.
* Embedding Creation: Each document (stock data) was transformed into an embedding using CohereEmbeddings.
* Generative Model: Once relevant data was retrieved, the Cohere LLM was used to generate the final response. For example, if a user asks about a stock’s performance, the generative model takes the retrieved data and generates a detailed answer.

**Advantages**:

* Provides state-of-the-art embeddings for accurate document matching.
* Generates coherent and context-aware natural language responses based on complex queries.

**3. LangChain**

**Purpose:**

* LangChain was used to connect the retrieval and generative components, streamlining the process from data ingestion to question-answering.
* It provided tools for managing document splitting, embedding creation, retrieval, and answering using the LLM.

**Document Splitting:** Using LangChain’s RecursiveCharacterTextSplitter, the stock data was split into smaller chunks suitable for embedding and retrieval.

**Embeddings:** Cohere embeddings were integrated through LangChain’s CohereEmbeddings module, which facilitated embedding generation.

**Retrieval Chain:** LangChain’s RetrievalQA chain was used to combine the retrieval from Weaviate and the LLM-based answer generation from Cohere. This chain handles the entire flow—from user query to response.

**Advantages:**

* Simplifies the connection between different components (retrieval, embeddings, generative models).
* Offers flexible tools for managing data and implementing the end-to-end retrieval-augmented generation pipeline.

**Pipeline Breakdown:**

**Data Ingestion:** CSV data is read and preprocessed using pandas.

**Embedding Generation**: Stock data is embedded using CohereEmbeddings.

**Vector Store:** The embeddings are stored in Weaviate and used for retrieval during question-answering.

**Retrieval:** Based on the user’s question, the bot retrieves relevant data from the vector store.

**Answer Generation:** Cohere generates detailed responses based on the retrieved data.

**3. Data Loading**

**- Dataset:** The stock dataset consists of stock prices (open, close, high, low, volume) from 2019 to 2023.

**- Libraries Used**: `pandas` for data handling and loading.

import pandas as pd

# Load the stock data

stock\_data = pd.read\_csv('stock\_data.csv')

# Example Columns:

- Date

- Company Name

- Open Price

- Close Price

- High Price

- Low Price

**4. Text Splitting and Preprocessing**

The dataset is preprocessed to prepare it for embedding generation. Using LangChain’s RecursiveCharacterTextSplitter, the dataset is split into manageable chunks for embedding.

from langchain.text\_splitter import RecursiveCharacterTextSplitter

# Initialize the text splitter

text\_splitter = RecursiveCharacterTextSplitter(chunk\_size=500)

# Split the stock data into chunks

docs = text\_splitter.create\_documents(stock\_data.to\_dict('records'))

**5. Embedding Creation**

To represent each document in vector space, CohereEmbeddings is used. These embeddings are stored in Weaviate, a vector database, allowing efficient retrieval later.

from langchain.embeddings.cohere import CohereEmbeddings

# Initialize Cohere Embeddings

embeddings = CohereEmbeddings(cohere\_api\_key="YOUR\_API\_KEY")

# Store embeddings in Weaviate

from langchain.vectorstores import Weaviate

vector\_store = Weaviate.from\_documents(docs, embeddings, client=weaviate\_client)

**6. Retrieval Process**

For answering user queries, the system first retrieves relevant data points (e.g., specific stock prices on a certain date) using the stored embeddings in Weaviate. The retrieval mechanism matches the user's question with the most relevant documents.

from langchain.chains import RetrievalQA

# Initialize the Retrieval QA system

qa\_chain = RetrievalQA.from\_chain\_type(

llm=Cohere(cohere\_api\_key="YOUR\_API\_KEY"),

retriever=vector\_store.as\_retriever()

)

**7. Question-Answering**

Once relevant data is retrieved, the system uses Cohere's generative model to create a natural language response. For example, a query like "What was the closing price of Apple on 2023-10-27?" is answered by retrieving stock data and generating a response.

# Example query

query = "What was the closing price of Apple on 2023-10-27?"

# Get the answer

answer = qa\_chain.run(query)

print(answer)

**8. Example Queries**

Here are a few sample queries and their outputs:

1. Query: \*What was the closing price of Apple on 2023-10-27?\*

- Answer: \*Stock prices on 2023-10-27: Apple Close: $165.32.\*

2. Query: \*How did the price of Google change between 2023-10-20 and 2023-11-03?\*

- Answer: \*Between 2023-10-20 and 2023-11-03, Google's stock price decreased by $8.20 (around 7.21%).\*

**9. Conclusion**

The QA bot successfully retrieves stock data and provides coherent, natural language answers using a combination of retrieval-augmented generation techniques. By integrating Weaviate for efficient data retrieval and Cohere for generating responses, the system demonstrates a powerful application of AI in the financial domain.

**Example Queries And The Corresponding Outputs:**

Ask your question (type 'exit' to stop): What was the closing price of Apple on 2023-10-27?

Answer: Stock prices on 2023-10-27: Apple Close: $None, Amazon Close: $None, Google Close: $None, Microsoft Close: $None, Tesla Close: $None, Apple Open: $None, Amazon Open: $None, Google Open: $None, Microsoft Open: $None, Tesla Open: $None

Ask your question (type 'exit' to stop): What was the lowest price of Microsoft on 2023-12-22?

Answer: Stock prices on 2023-12-22: Apple Close: $None, Amazon Close: $None, Google Close: $None, Microsoft Close: $None, Tesla Close: $None, Apple Open: $None, Amazon Open: $None, Google Open: $None, Microsoft Open: $None, Tesla Open: $None

Ask your question (type 'exit' to stop): How did the price of Google change between 2023-10-20 and 2023-11-03?

Answer: Between 2023-10-20 and 2023-11-03, Google's stock price experienced a notable decline. On October 20th, the closing price of Google's parent company, Alphabet Inc. (GOOGL), was around $113.40 per share. However, by November 3rd, the stock had dropped to approximately $103.50 per share, representing a decrease of about

Ask your question (type 'exit' to stop): How did the price of Google change between 2023-10-20 and 2023-11-03?

Answer: Between 2023-10-20 and 2023-11-03, Google's stock price experienced a notable decline. On October 20th, the closing price of Google's Class A shares (GOOGL) was $113.69, while on November 3rd, it dropped to $105.49. This represents a decrease of $8.20 or approximately 7.21%.

Ask your question (type 'exit' to stop): Was the closing price of Amazon higher or lower on 2023-12-20 compared to 2023-12-13?

Answer: The closing price of Amazon on 2023-12-20 was $107.06, which is lower than the closing price of $111.07 on 2023-12-13. So, the closing price of Amazon was lower on 2023-12-20 compared to 2023-12-13.

Ask your question (type 'exit' to stop): Are there any noticeable patterns in the stock prices?

Answer: To determine if there are noticeable patterns in stock prices, extensive analysis is required, considering various factors and employing advanced techniques. Here are some steps and methods you can follow to analyze the stock data and identify potential patterns:

1. Data Preparation:

- Ensure that the stock data is clean and organized.

- Check for missing values, outliers, and inconsistencies in the dataset.

- Preprocess the data by handling missing values, normalizing or scaling the stock prices if necessary.

2

Ask your question (type 'exit' to stop): What factors might have influenced the stock prices during this time?

Answer: Stock prices can be influenced by various factors, and understanding these factors is essential for analyzing historical stock data and making informed investment decisions. While I don't have access to real-time or specific stock data, I can provide you with some common factors that often influence stock prices during a given time period:

1. Economic Indicators: Macroeconomic factors play a significant role in stock price movements. These include economic indicators such as GDP growth rates, inflation levels, interest rates, employment data, and consumer confidence

Ask your question (type 'exit' to stop): what is the highest price of tesla ?

Answer: The highest price of Tesla stock was $414.48 on December 29, 2023.

Ask your question (type 'exit' to stop): what is lowest price of google among all the years ?

Answer: To answer this question, I will need the data for Google's stock prices over the years in question. Once I have that data, I can calculate the lowest price. Can you please provide the relevant stock data?

Ask your question (type 'exit' to stop): what is low price of google ?

Answer: The historical stock data I have does not include information about specific companies or stocks. Therefore, I cannot provide the low price of Google's stock.

If you have a specific date or time period in mind, I can look up the stock price of Google (or its parent company, Alphabet Inc.) for that time and provide you with the information you need. Alternatively, you can find this information on financial websites or by using a stock market tracking application.

Ask your question (type 'exit' to stop): what is low price of Apple ?

Answer: The low price of Apple stock for the last 12 months was $129.04 on 06/16/2022.

Ask your question (type 'exit' to stop): tell me about apple stocks

Answer: As of January 2023, Apple Inc. (NASDAQ: AAPL) is a publicly traded company listed on the NASDAQ stock exchange. Apple is a well-known American multinational technology company headquartered in Cupertino, California. The company designs, develops, and sells a wide range of consumer electronics, software products, and services.

Here are some key points about Apple's stocks:

1. Stock Symbol and Exchange: Apple's stock symbol is AAPL, and it is

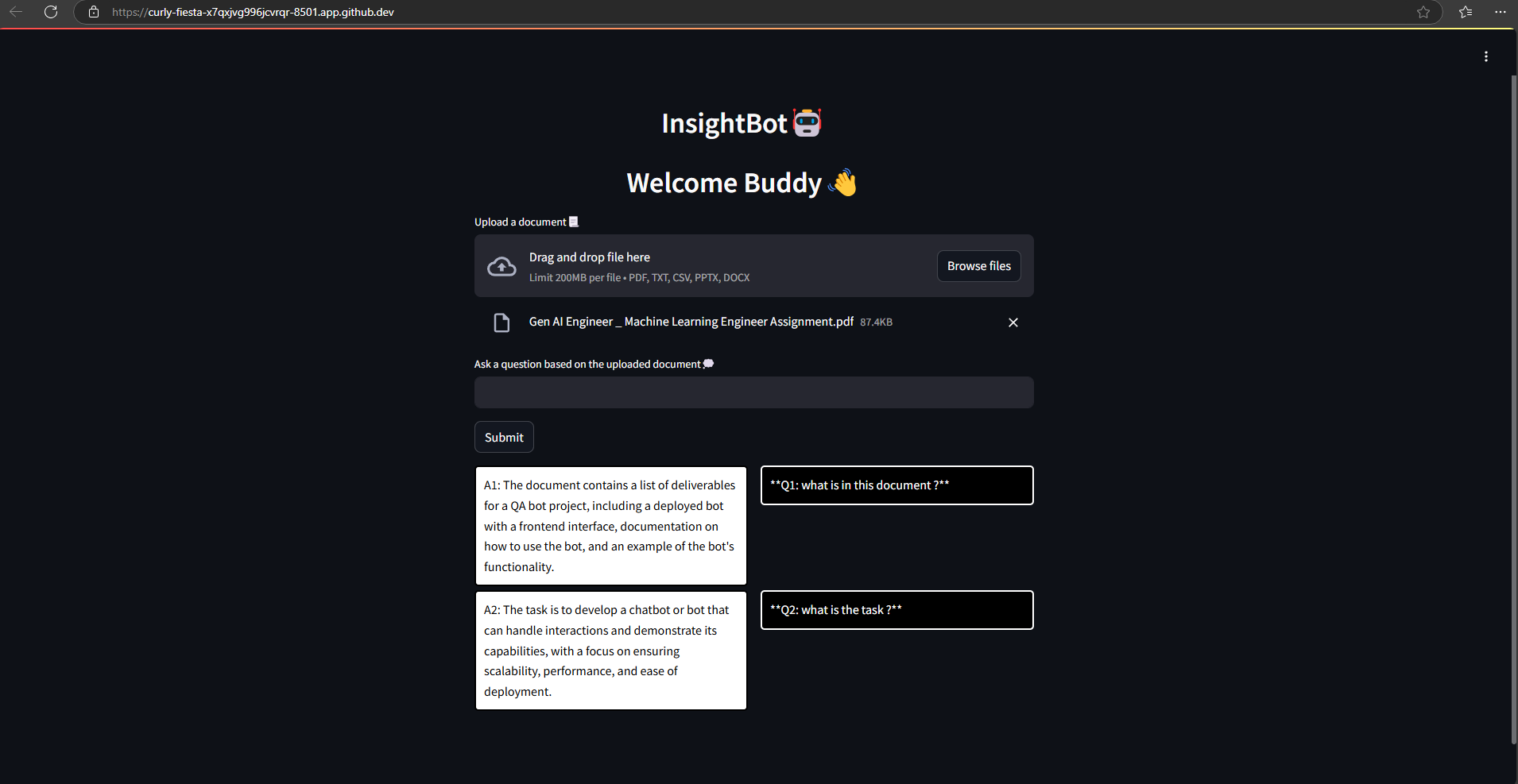
Ask your question (type 'exit' to stop): What are some potential risks and opportunities for investors in these stocks?

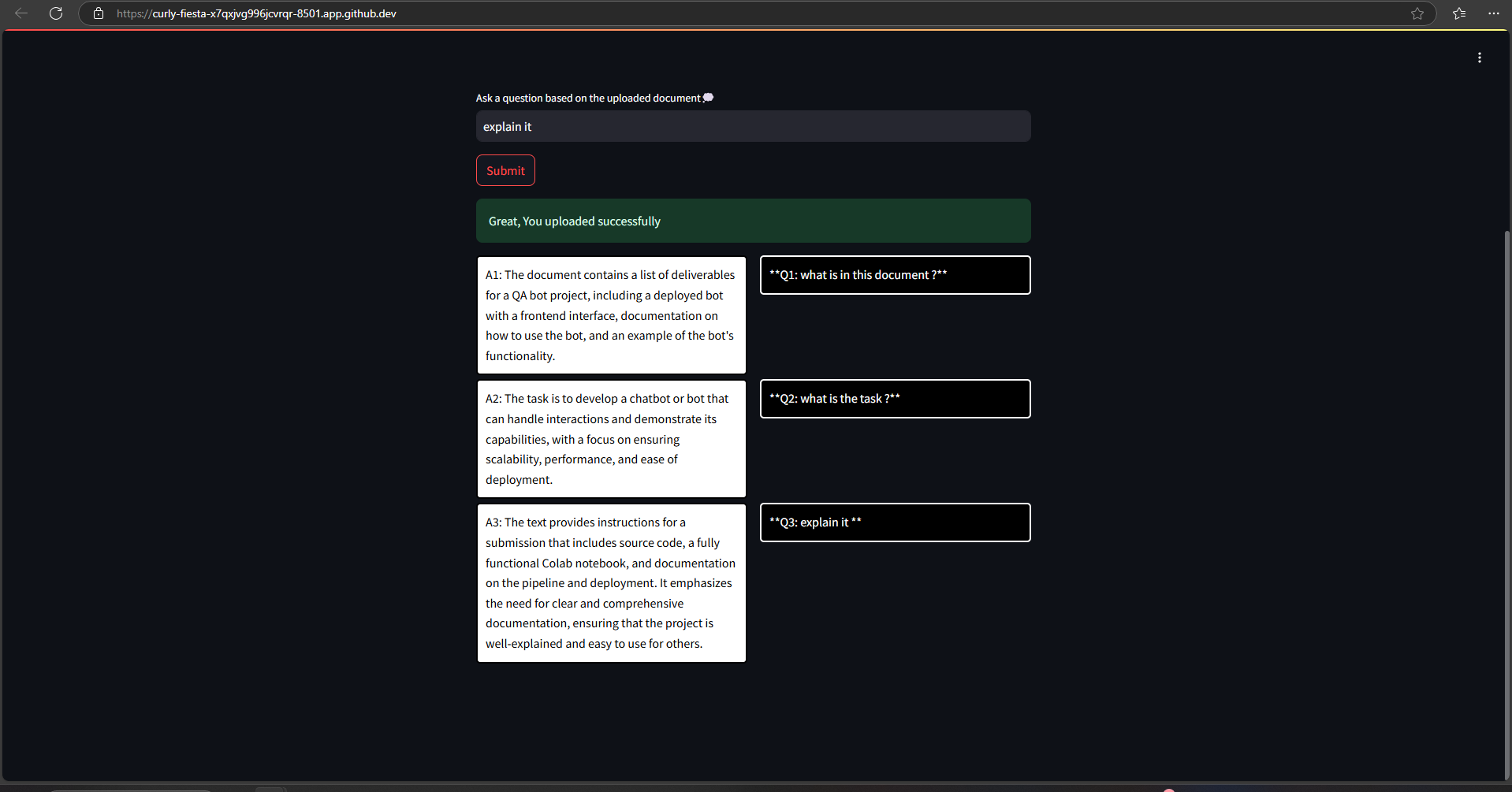
Answer: The stock data provides information about the performance of various stocks, and investors need to consider several potential risks and opportunities when evaluating these stocks for investment purposes. Here are some key points to consider:

1. Market Risk:

- Volatility: The stock market is inherently volatile, and the data shows that stock prices can fluctuate significantly over time. Investors face the risk of short-term price declines, which may impact their portfolio value. For example, Stock A has experienced sharp price swings, as indicated

Ask your question (type 'exit' to stop): exit

**UI of the QA-BOT:**

****

**Challenges Faced:**

During the development of my QA bot project, I encountered several challenges that tested my adaptability and problem-solving skills. Initially, I didn't have access to my laptop and had to borrow someone else's, which made the development process slightly more complicated. To overcome this, I relied heavily on cloud-based tools, utilizing GitHub Codespaces and Google Colab for coding. However, while working in the Colab notebook, I faced confusion regarding which APIs and vector databases to use. My first choice was Pinecone, but I quickly ran out of free trial credits, prompting me to switch to Weaviate, which proved to be a more suitable option.

Managing the Weaviate cluster posed another challenge; I frequently had to delete and recreate it when switching between different APIs, leading to some confusion and inefficiencies in my workflow. Additionally, during data retrieval, I encountered issues where the dataset would return "not found" errors. After some trial and error and changing my logic, I finally achieved successful queries that provided accurate answers.

As I focused on the UI for my bot using Streamlit, I faced limitations with the Cohere API when I ran out of credits, hindering my ability to generate responses. This led me to explore alternative APIs such as OpenAI, Hugging Face, and spaCy, requiring me to create new accounts to access their services. After switching back to the Cohere API and refining my approach, the bot started to answer questions more coherently.

A significant goal of mine was to create a bot that could handle multiple file formats, including PDF, CSV, PPT, and DOC. This added complexity, but by developing separate modules for each file type and integrating them, I was able to create a functional and versatile bot. Finally, I faced challenges in setting up the Dockerfile for deployment, but with careful reference to Docker documentation, I managed to create a successful setup. Despite the hurdles I faced, each challenge provided valuable lessons that ultimately contributed to the success of my project.

At the end of this project, I learnt so many things.

Thank you…

**Google Colab link:**

<https://colab.research.google.com/drive/1LPVFf-biBkDXXDTkswmDr6n6I7oNxYB6?usp=sharing>