Logic Based Chat-Bot

The chatbot is built using Python and interacts with a pre-loaded Pandas DataFrame containing financial data for Microsoft, Tesla, and Apple from 2021 to 2023. It works in a loop, prompting the user for inputs and displaying relevant data based on those inputs.

Predefined Queries:

The chatbot can respond to queries about specific financial metrics for each company and year. The predefined metrics include:

1. **Total Revenue**: Provides the total revenue for a given company and year, along with growth percentage.
2. **Total Net Income**: Gives net income details and growth for the selected year.
3. **Total Assets**: Displays the company's total assets for the requested year.
4. **Total Liabilities**: Shows the company's liabilities for the chosen year.
5. **Cash Flow from Operating Activities**: Offers information on the cash flow generated by the company's operations.
6. **Revenue Growth (%)**: Calculates and shows the revenue growth between consecutive years.
7. **Net Income Growth (%)**: Provides the net income growth between years.
8. **Assets Growth (%)**: Displays the growth percentage in total assets from one year to the next.
9. **Liabilities Growth (%)**: Shows the growth percentage in total liabilities from one year to the next.
10. **Cash Flow from Operations Growth (%)**: Shows the growth percentage in cash flow from operations between consecutive years.

To get the desired information, the user needs to:

1. **Specify the company:** (Microsoft, Tesla, or Apple)
2. **Specify the year:** (2021, 2022, or 2023)
3. **Choose the desired metric:** (Selecting from the numbered options 1-10)

How It Works:

1. **Data Processing:** The chatbot reads a CSV file containing the financial data for various companies across multiple years.
2. **User Interaction:** Users select the company, year, and metric they are interested in through a web interface.
3. **Response Generation:** Based on the user's input, the chatbot retrieves the corresponding financial data and formulates a response. It can answer specific metrics like total revenue, growth, and financial health.

4. **Output:** The chatbot responds with the relevant financial information, formatted to be easy to understand.

Limitations:

1. **Limited Data:** The chatbot only has access to financial data for the three specified companies and the years 2021-2023. It cannot answer queries about other companies or time periods.
2. **Predefined Queries:** It can only respond to queries about the ten predefined financial metrics. It cannot handle more complex or open-ended financial questions.
3. **Error Handling:** While it has some error handling, unexpected inputs or data issues might still cause errors or unexpected behaviour.
4. **No Natural Language Processing:** The chatbot does not understand natural language. Users need to provide inputs in the specific format it expects.

Overall, the chatbot is useful for quickly retrieving and understanding financial information from structured datasets, but it has limitations in terms of dynamic data sources and custom queries.

AI-Powered Financial Chatbot Web Application

This document provides a step-by-step guide to setting up and running the AI-powered financial chatbot web application.

Prerequisites

1. **Python 3**: Ensure that Python 3 is installed on your machine. You can download it from python.org.
2. **Flask**: Flask is a lightweight web framework for Python. You can install it using `pip` (Python's package installer).
3. **Other Python Libraries**: This app requires some additional libraries like `pandas` to handle CSV files.

Install Flask and Pandas

Run the following commands in your terminal or command prompt to install the required packages:

```
pip install flask pandas
```

Directory Structure

Make sure your project files are organised as follows:

```
financial_chatbot/
│
├── app.py                  # Main Flask application file
├── static/
│   └── styles.css          # CSS file for styling
├── templates/
│   └── index.html          # HTML file for the web interface
└── financial_data.csv      # CSV file containing financial data
```

- **app.py**: The main Python file containing the Flask app.
- **static/styles.css**: The CSS file for the styling of the web interface.
- **templates/index.html**: The HTML file for the web page.
- **financial_data.csv**: The CSV file with financial data used by the chatbot.

Configuration

1. **Place the `financial_data.csv` file** in the root directory of the project. This file should contain the financial data that the chatbot will access to answer user queries.
2. **Check CSV Format**: Ensure that your CSV file (`financial_data.csv`) has the following
   `Year, Company, Total Revenue, Net Income, Total Assets, Total Liabilities, Cash Flow from Operating Activities, Revenue Growth (%), Net Income Growth (%), Assets Growth (%), Liabilities Growth (%), Cash Flow from Operations Growth(%)`

Running the Application

Open the Terminal (or Command Prompt) and navigate to your project directory. For

Example : `cd path/to/financial_chatbot`

Run the Flask App using the following command:

```
python app.py
```

Access the Web Application:

- Once the app is running, Flask will start a development server on `http://127.0.0.1:5000`.
- Open a web browser and go to http://127.0.0.1:5000 to interact with the chatbot.

How to Use the Chatbot

1. **Toggle the Chat Window**: Click the chatbot icon at the bottom right corner of the page to open the chat interface.
2. **Enter the inputs:** Just enter the inputs that is company name, year ,metric you want
3. **Receive a Response**: The chatbot will retrieve and display the requested data from the CSV file.

The chatbot supports questions about the following metrics:

- Total Revenue
- Net Income
- Total Assets
- Total Liabilities
- Cash Flow from Operating Activities
- Revenue Growth (%)
- Net Income Growth (%)
- Assets Growth (%)
- Liabilities Growth (%)
- Cash Flow from Operations Growth (%)

Troubleshooting

- **CSS Not Loading**: Ensure that `styles.css` is in the `static` folder and that the `<link>` tag in `index.html` points to it correctly.
- **Template Not Found Error**: Check that `index.html` is in the `templates` folder.
- **Data Not Loading**: Confirm that `financial_data.csv` is in the correct format and in the root directory.