Take-Home Assignment

Objective:
Construct and track an equal-weighted custom index comprising top 100 US stocks based on their market cap. The index should be updated daily as market cap changes. Develop a dashboard to visualize the index's performance and composition over the past month, highlighting any days with changes in the composition. Additionally, provide functionality to export the performance and composition data to well-formatted Excel and PDF files.

Detailed Assignment Requirements:

1. **Data Acquisition:**
    - Use any publicly available API or dataset to identify and fetch data for actively traded US stocks and their daily market caps.
    - Evaluation Criteria for Data Source:
        - Availability of historical data.
        - Coverage of the US stock market.
        - Reliability and ease of use of the API.
    - Examples of Acceptable Sources: Yahoo Finance, Alpha Vantage, IEX Cloud, Quandl, Polygon or similar.
    - Provide a list of data sources evaluated, with a justification for your chosen source.

2. **Data Storage:**
    - Create an SQL-like setup to store stock, price and market cap data.
    - Key Requirements:
        - Use an in-memory SQL engine (e.g., SQLite or DuckDB).
        - Fetch and persist data into the database using SQL queries for further processing.
        - All subsequent data access and transformations should rely on SQL queries.

3. **Index Construction:**
    - Construct an equal-weighted custom index for each trading day.
    - Steps:
        - Fetch the required market data including stock price and market cap for all stocks for the given day.
        - Identify the top 100 stocks by market cap.
        - Rebalance the index composition if the top 100 stocks change.
        - Maintain historical data for the index composition and performance over the past month.

4. **Python Code Requirements:** Write Python code to:
    - Fetch and persist data:
        - Fetch the latest stock and market data from the data source.
        - Save the data into the database.
    - Analyze data:
        - Query the database to get the top 100 stocks by market cap for any given day.
    - Construct/Rebalance the index:
        - Build the index for the past month on a daily basis.
        - Track changes in composition and index performance.

- Export data:
    - Allow the user to export index performance and composition data to:
    - A well-formatted Excel file.
    - A well-formatted PDF file (bonus).
- Dashboard creation:
    - Use any Python visualization library (e.g., Dash, Streamlit, Plotly, or Matplotlib) to create an interactive dashboard.

5. **Dashboard Requirements:** Visualization Goals
    - Performance:
        - A line chart showing the index's performance over the past month.
    - Composition:
        - A table or bar chart showing the composition of the index for any selected day.
    - Composition Changes:
        - Highlight days when changes in the index composition occurred.
    - Summary Metrics (Bonus):
        - Include metrics such as cumulative returns, daily percentage changes, and the number of composition changes.

6. **Submission Instructions:**
    - Code Submission:
        - Host your Python code in a GitHub repository.
        - Ensure that the code is well-documented with:
            - Clear instructions on setup and execution.
            - A README.md file outlining the project, dependencies, and data source.
        - Follow Python programming best practices.
    - Presentation:
        - Prepare a 2-5 slide PowerPoint or Google Slides presentation covering:
            - Your analysis process.
            - Key findings and insights from the index performance.
            - Design decisions for the code and dashboard.
            - Challenges faced and how you addressed them.

7. **Scalability and Maintenance Discussion:**
    - Include a section in your presentation discussing:
        - How you would maintain and update the dashboard as new data becomes available.
        - Suggestions for scaling up the solution, such as:
            - Adding more data sources.
            - Incorporating advanced analytics.
            - Enhancing the dashboard with real-time updates or additional visualization features.

Guidelines:

- **Creativity and Insight:**
    - While the assignment has specific requirements, feel free to be creative in your analysis and presentation. Highlight your mastery of Python and SQL for reporting.

- **Code Quality:**
    - Write clean, efficient, and well-commented code.
    - Follow best practices for Python programming and data visualization.
- **Dashboard Design:**
    - Ensure the dashboard is user-friendly and visually appealing.
    - The dashboard should effectively communicate your findings to someone without a technical background.
- **Documentation:**
    - ***Include any necessary documentation to help us understand and run your code.***