CIAL dun & bradstreet

Technical assignment

Hi,

As part of your assessment, we have prepared a small assignment for you. If you have any questions, feel free to contact us via fabiola.rozendo@cialdnb.com
Task is to be completed within seven days from when you start working on it.

Once you are done with the assignment, push your code to github and email us the link to
fabiola.rozendo@cialdnb.com
silvia.tomaz@cialdnb.com
alexsandro.valencio@cialdnb.com

GitHub repo should contain a README.md with clear instructions how to run the codewith a (brief) technical overview.

Good luck!

Goal

Your assignment is to implement a Stocks REST API using Python and a popular web framework of your choice. It could be but not limited to FastAPI/ Flask/ Django.

Brief

In this assignment, you will be building an API web application that retrieves stock data from an external financial API and performs minor data scraping from the Marketwatch financial website.

The application will expose two endpoints that make use the most of data available on these sources. Among other requirements, you will also be expected to write unit tests to ensure the correctness of your implementation.

Requirements

- Write the application in Python (3.10 or above);
- Implement a REST API returning JSON responses;
- Structure the Python project according to the best practices;
- Commit all code into a GitHub repository;
- Feel free to combine and use any other technology, library and external resource.
- Implement a **Stock model**. Each stock should have the following attributes and data types:
    - Status: String
    - purchased_amount: Integer
    - purchased_status: String
    - request_data: Date (YYYY-MM-DD)
    - company_code: String
    - company_name: String
    - Stock_values: Object
        - open: Float
        - high: Float
        - low: Float
        - close: Float
    - performance_data: Object
        - five_days: Float
        - one_month: Float
        - three_months: Float
        - year_to_date: Float
        - one_year: Float
    - Competitors: Array[Object]
        - name: String

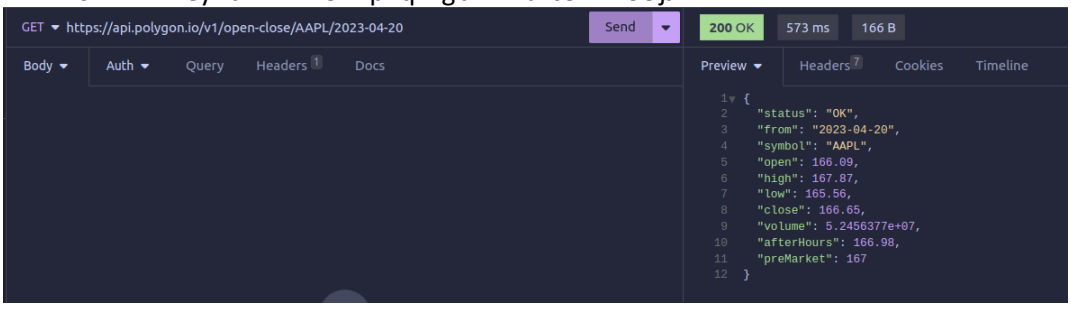
- market_cap: Object
    - Currency: String
    - Value: Float
- The application should expose two endpoints for the Stock resource:
    - [GET] /stock/{stock_symbol}: returns the stock data for the given symbol. The endpoint should return a JSON object that includes all the fields of the Stock model mentioned above. Refer to the sources section for instructions from where to populate the above-mentioned Stock fields.
    - [POST] /stock/{stock_symbol}: update the stock entity with the purchased amount based on received argument: “amount” (of type Integer). Example request body:
      {“amount”: 5.33} should return a response with 201 status code and message: “{amount} units of stock {stock_symbol} were added to your stock record”
- Implement caching per stock mechanism on the GET route so that data will not need to be fetched again in short intervals
- Implement data persistence into a database (PostgresDB) for stocks and their purchased amounts in the above-mentioned POST route.
- Create a Dockerfile that builds an image containing the ready-to-run API application. One should be able to invoke “docker run” and interact with the API locally on port 8000
- Implement logs for the application

Sources

This section describes the sources to be used in order to populate the Stock model fields:

Polygon.io

- Website: Polygon.io
- Service: Daily Open/Close
- Endpoint: [GET] /v1/open-close/{stocksTicker}/{date}
- Documentation: https://polygon.io/docs/stocks/get_v1_open-close__stocksticker___date
- API Key: bmN7i7CrzrpKqFvgbB1fEaztCwZKSUjJ

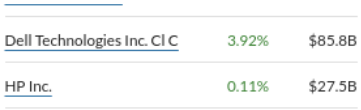


- cURL:

```
curl --request GET \
  --url https://api.polygon.io/v1/open-close/AAPL/2023-04-20 \
  --header 'Authorization: Bearer bmN7i7CrzrpKqFvgbB1fEaztCwZKSUjJ'
```

MarketWatch

- Website: https://www.marketwatch.com
- Data: Performance and Competitors needs to be scraped from the MarketWatch stock page using received stock symbol, available in the following path: https://www.marketwatch.com/investing/stock/<stock_symbol>. Ex: https://www.marketwatch.com/investing/stock/aapl
  -

  - 

Process

Once you receive the task, you can ask us anything about it. Structure and organize your questions wisely because you can send us a list of questions only twice during this process. You have 7 days to finish the task, starting from the moment you receive it. Once you are done with the project, zip up your local clone of the git repository and send it to us with the GitHub url of the repo.

Review

Your project will be reviewed from a logic and structure point of view. Logic includes solution implementation and running your code through different American companies listed on https://www.marketwatch.com/tools/markets/stocks/country/united-states. Structure includes code quality and overall project/code organization.