GSA Capital Partners LLP

Developer Exercise - June 2018

GSA Capital Developer Exercise

This exercise is to test your ability to understand a small requirements specification and implement it. The aim is to create a .NET Core 2.x application (in C#) that loads data from a set of files into a SQL Server Express LocalDB and exposes a RESTful API that will be called by our test suite.

Definitions

In this exercise, the following definitions apply:

(Trading) Strategy – a particular set of rules for managing an investment portfolio, resulting in a profit & loss (P&L) dollar number each day. Each strategy trades in exactly one region.

Region – a major region of the world: Europe (EU), America (US) or Asia Pacific (AP).

Capital – Dollar amount invested in a strategy. Each strategy can have different amounts invested, and that amount can change over time.

Daily Return - Daily P&L divided by capital at the start of the month.

Files Provided

You have been provided with files pnl.csv, capital.csv and properties.csv. These contain source data for the project. The data has been fabricated for this exercise, and is not based on any internal GSA information.

pnl.csv contains P&L data over time for 15 different trading strategies ‘Strategy1’ to ‘Strategy15’. The numbers are US$ P&L on a single day for the strategy.

capital.csv contains the US$ amounts invested in each strategy at the beginning of the month.

properties.csv contains the (single) region of each strategy.

Requirements

You should develop a .NET Core 2.x application (in C#) that loads the csv files specified above into a SQL Server Express LocalDB. For the purpose of this exercise, this may be done during the initialisation of the application.

The import process should store the data in a suitable format (either normalised or denormalised), provided that the underlying meaning of the data is not changed.

The application should also expose a RESTful API on port 8000 that will be called by our test suite to test your code.

The API should consist of the following endpoints, with each response matching the schema illustrated in the corresponding example. Note that the values used in the example responses are fabricated and not derived from the provided data.

Monthly Capitals

Returns a time series of monthly capital values for the strategies.

GET `/api/monthly-capital/`

Optional query string parameters

```
   strategies - (string) Comma delimited list of strategy names (e.g. 'Strategy1,Strategy2')
```

Response (application/json) - (e.g.: /api/monthly-capital/?strategies=Strategy1,Strategy2)

```
[
  { "strategy": "Strategy1", "date": "2017-01-01", "capital": 1000 },
  { "strategy": "Strategy2", "date": "2017-01-01", "capital": 2000 },
  ...
]
```

Cumulative P&Ls

Returns a daily time series of *cumulative* P&Ls *aggregated* by region. Regions can be filtered according to the `region` parameter. The start date of the time series can be specified using the `startDate` parameter.

GET `/api/cumulative-pnl/`

Optional query string parameters

```
   region – (string, either 'AP', 'EU' or 'US')
   startDate – (string, in the format yyyy-MM-dd)
```

Response (application/json)
(e.g.: /api/cumulative-pnl/?startDate=2017-01-01&region=EU)

```
[
  { "region": "EU", "date": "2017-01-01", "cumulativePnl": 1000 },
  { "region": "EU", "date": "2017-01-02", "cumulativePnl": 2000 },
  ...
]
```

Compound daily returns

Returns a time series of *compound* daily returns since inception for a strategy as specified in the route. The compound return for each day should be expressed as a decimal and rounded to five decimal places.

GET `/api/compound-daily-returns/<strategy>/`
Response (application/json) – (e.g.: /api/compound-daily-returns/Strategy1/)

```
[
  {
    "strategy": "Strategy1",
    "date": "2017-01-01",
    "compoundReturn": 0.0012
  },
  ...
]
```

If there are any problems with the data that could cause errors, you must adopt a mechanism of your choice to prevent these from occurring.

Development

The application can be developed with Visual Studio 2017 Community Edition, available at the following link (https://www.visualstudio.com/thank-you-downloading-visual-studio/?sku=Community&rel=15).

You may use third-party libraries to assist in achieving your goal. Any such dependencies must be free for commercial use, and you should document their use.

Delivery

Please zip (or tar) the source code only required to run the project (remove all binary / executables e.g bin & obj folders before sending) plus a short readme.
The readme file should include:

- Your name
- Brief instructions for using your solution
- What mechanism you have adopted (if any) for handling bad data in the files given
- Which libraries are used by your project and the reason for their use
- A brief explanation of the architecture and methodologies used within your solution
- Any assumptions you made when interpreting the specification

Please upload the solution to greenhouse via the link provided by your recruitment agent.

Hints

- There is no strict time limit, but aim to spend a few hours.
- If you run out of time, prefer to cut scope rather than quality.
- If any concepts are unclear, first try searching the internet for explanations or examples. If something is still unclear, use your judgement to make a best effort attempt and document what you have chosen to do in your readme file.