PHP Challenge Instructions

Description

This Challenge is designed to test your abilities in building backend web projects with PHP. Additionally, you are encouraged to show us your best skills and knowledge, demonstrate creativity, follow best development practices, and showcase knowledge in other related programming topics.

You are free to use any PHP Framework of your choice; however, it would be a plus if you could use Symfony (so we could evaluate your SF skills) or Slim Framework (so we could evaluate your PHP and OO skills more precisely).

Assignment

The objective of this challenge is to create a REST API application that the user can use to track the value of stocks in the stock market.

You may use the AlphaVantage API service or Stooq API service to get the latest stock market values. Check the format used below. You can replace the `{symbol}` placeholder with the corresponding stock code:

- `https://www.alphavantage.co/query?function=GLOBAL_QUOTE&symbol={symbol}&apikey={api_key}`
- `https://stooq.com/q/l/?s={symbol}&f=sd2t2ohlcvn&h&e=csv`

Mandatory Features

- The application must use an SQL database to store users and record logs of past requests.
- The application must be able to authenticate registered users.

The application must have these three endpoints:

- An endpoint to create a new User, storing the email and information to log in later.
- An endpoint to request a stock quote, like this:

`GET /stock?q=IBM`

```
{
    "name": "IBM",
    "symbol": "IBM",
    "open": 123.66,
    "high": 123.66,
    "low": 122.49,
    "close": 123
}
```

`GET /stock?q=appl.us`

```
  {
      "name": "APPLE",
      "symbol": "APPL.US",
      "open": 123.66,
      "high": 123.66,
      "low": 122.49,
      "close": 123
  }
```

The same endpoint must additionally send an email with the same information to the user who requested the quote.

- An endpoint to retrieve the history of queries made to the API service by that user. The endpoint should return the list of entries saved in the database, showing the latest entries first:

`GET /history`

```
[
    {"date": "2021-04-01T19:20:30Z", "name": "IBM", "symbol": "IBM",
"open": "123.66", "high": 123.66, "low": 122.49, "close": "123"},
    {"date": "2021-03-25T11:10:55Z", "name": "IBM", "symbol": "IBM",
"open": "121.10", "high": 123.66, "low": 122, "close": "122"},
    ...
]
```

Bonus features

The following features are optional to implement, but if you do, you'll be ranked higher in our evaluation process.

- Add unit tests, and integration tests for the endpoints.
- Use RabbitMQ to send the email asynchronously.
- Use JWT instead of basic authentication for endpoints.
- Containerize the app.
- Postman Collection.
- Creation of a simple frontend using Vue to interact with the API.
- Use of Symfony or Slim Framework.

Considerations

- Provide any and all information our evaluators may need.
- Use seeders and migrations instead of SQL dumps if we need any predetermined database information.
- We want to see your skills, so make sure to use everything that the frameworks offers (if applicable).
- Best coding practices adherence, such as coding standards, design patterns, and design principles.
- Provide a detailed Readme with instructions on how to install, use, etc.

Submission

Candidates must provide a public Git repository containing the source code of the application, along with any relevant documentation. They should also include clear instructions on how to set up and run the application locally.