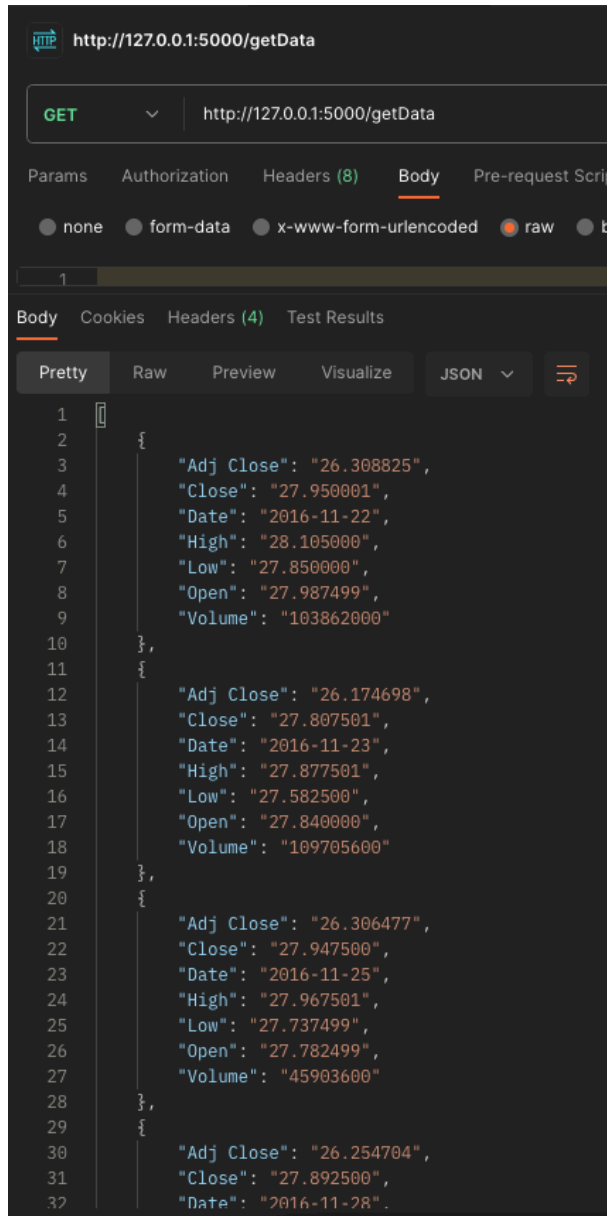
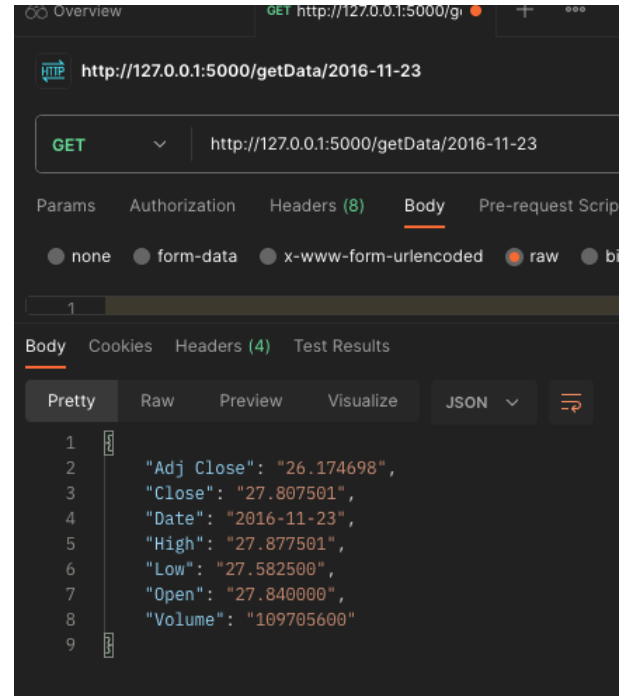


Mohammad Ali
Lab 2 – Flask Tutorial

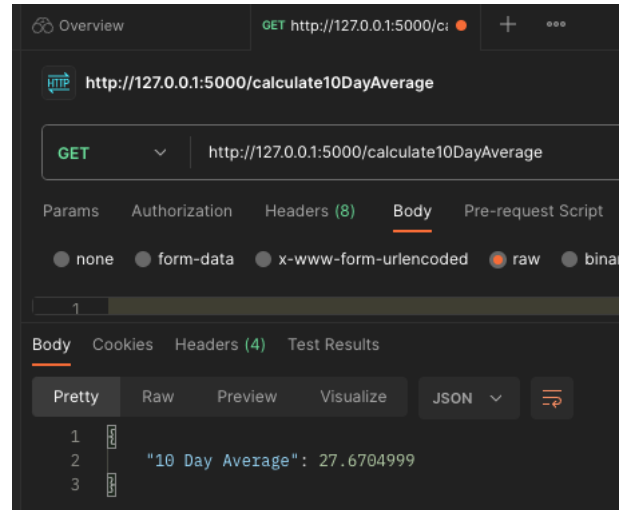
GET:



```
1 {
2   {
3     "Adj Close": "26.308825",
4     "Close": "27.950001",
5     "Date": "2016-11-22",
6     "High": "28.105000",
7     "Low": "27.850000",
8     "Open": "27.987499",
9     "Volume": "103862000"
10  },
11  {
12    "Adj Close": "26.174698",
13    "Close": "27.807501",
14    "Date": "2016-11-23",
15    "High": "27.877501",
16    "Low": "27.582500",
17    "Open": "27.840000",
18    "Volume": "109705600"
19  },
20  {
21    "Adj Close": "26.306477",
22    "Close": "27.947500",
23    "Date": "2016-11-25",
24    "High": "27.967501",
25    "Low": "27.737499",
26    "Open": "27.782499",
27    "Volume": "45903600"
28  },
29  {
30    "Adj Close": "26.254704",
31    "Close": "27.892500",
32    "Date": "2016-11-28"
```



```
1 {
2   "Adj Close": "26.174698",
3   "Close": "27.807501",
4   "Date": "2016-11-23",
5   "High": "27.877501",
6   "Low": "27.582500",
7   "Open": "27.840000",
8   "Volume": "109705600"
9 }
```



```
1 {
2   "10 Day Average": 27.6704999
3 }
```

POST:

POST `http://127.0.0.1:5000/addData`

Params Authorization Headers (10) **Body** Pre-request Script

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw

```
1 {
2   "Date": "2023-11-22",
3   "Open": 150.0,
4   "High": 200.0,
5   "Low": 120.0,
6   "Close": 124.5,
7   "Adj Close": 150.0,
8   "Volume": 1250000
9 }
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Added successfully!"
3 }
```

POST `http://127.0.0.1:5000/getData`

Params Authorization Headers (10) **Body** Pre-request Script

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary

```
1 {
2   "start": "2016-05-12",
3   "end": "2017-05-01"
4 }
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   {
3     "Adj Close": "26.308825",
4     "Close": "27.950001",
5     "Date": "2016-11-22",
6     "High": "28.105000",
7     "Low": "27.850000",
8     "Open": "27.987499",
9     "Volume": "103862000"
10  },
11  {
12    "Adj Close": "26.174698",
13    "Close": "27.807501",
14    "Date": "2016-11-23",
15    "High": "27.877501",
16    "Low": "27.582500",
17    "Open": "27.840000",
18    "Volume": "109705600"
19  },
20  {
21    "Adj Close": "26.306477",
22    "Close": "27.947500",
23    "Date": "2016-11-25",
24    "High": "27.967501",
25    "Low": "27.737499",
26    "Open": "27.782499",
27    "Volume": "45903600"
28  }
29 }
```

PUT:

PUT `http://127.0.0.1:5000/updateData`

Params Authorization Headers (10) **Body** Pre-request Script Tests

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL

```
1 {
2   "Date": "2023-11-22",
3   "Open": 200.0,
4   "High": 300.0,
5   "Low": 150.0,
6   "Close": 160.5,
7   "Adj Close": 160.0,
8   "Volume": 1230000
9 }
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Data for 2023-11-22 updated successfully."
3 }
```

DELETE:

DELETE ▼ http://127.0.0.1:5000/deleteData

Params Authorization Headers (10) **Body** Pre-request Script Tests

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary

```
1 {
2   "Date": "2023-11-22"
3 }
4
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON ▼ ⇌

```
1 {
2   "message": "Data for 2023-11-22 deleted successfully."
3 }
```

DELETE ▼ http://127.0.0.1:5000/deleteAll


Params Authorization **Headers (8)** Body Pre-request Script Tests Settings

	Key	Value
<input checked="" type="checkbox"/>	Content-Type	application/json
<input checked="" type="checkbox"/>	User-Key	NOTADMIN
	Key	Value

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON ▼ ⇌

```
1 {
2   "error": "Access denied. Invalid key."
3 }
```

 http://127.0.0.1:5000/deleteAll

DELETE ▼ http://127.0.0.1:5000/deleteAll

Params Authorization **Headers (8)** Body Pre-request Script Tests Settings

Headers 👁 6 hidden

	Key	Value
<input checked="" type="checkbox"/>	Content-Type	application/json
<input checked="" type="checkbox"/>	User-Key	IAMADMIN123
	Key	Value

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON ▼ ⇌

```
1 {
2   "message": "Access granted. Successful deletion."
3 }
```

How I went about adding each API: (**NOTE:** All APIs have response messages to confirm or deny requests with an associated error code to display to the user)

⇒ **The GET APIs:**

- /getData
 - I created a function called “read_stock_csv” that using the “import csv” was able to append every row in the file into an array called data. The function returned data.
- /getData/<date>
 - I created a function called “get_data_using_date” that took 2 parameters, data & date, and simply looked for the ‘Date’ key in each row, comparing it to the date passed by the user, and when found, return the full row/information associated with that date.
- /calculate10DayAverage (**Note:** Assumes data is sorted by date)
 - Created by using the “calculate_10_day_average” function, that took 1 parameter, data. Set a variable to zero and looked in the last 10 entries of the data and kept adding the values of the ‘Close’ key to the set variable and returned that total divided by 10.

⇒ **The POST APIs:**

- /addData
 - Using the “import request” from flask, I was able to take the data passed by the user, validate it by making sure all required fields were passed, and then simply appending the new data by calling a function “adding_data” that would use the open file with write permissions and using the fieldnames given, it would write the new data into the row.
- /getData
 - Using the request data passed by the user, and making sure it is a valid request, meaning a start and end date is given, the “import datetime” from datetime, is able to parse the start and end dates and using a “sort_data_by_range” function, which just does a date comparison from the ‘Date’ key in each row and appends it into a new list that is returned once the for loop is finished.

⇒ **The PUT APIs:**

- /updateData
 - The route logic is checking if the user provided a valid date and then sets the target_update to the given ‘Date’ key of the input. Update_this stores the row in which the ‘Date’ key of the main data is equal to the requested date and if so, using the “adding_data” function, the requested_data overwrites the previous data.

⇒ **The DELETE APIs:**

- /deleteData
 - This route logic is the same as the /updateData but once the delete_this variable finds a match to the target_delete, it removes the requested data

from the fetched data and gives that “new” data back to be overwritten using the “adding_data” function.

- /deleteAll
 - This route required ADMIN rights so by setting an admin_key in the beginning of the project and allowing the user to pass a key through the header, the admin_key would be compared to the user_key and if they match, using the “adding_data” function, the current data would be overwritten with an empty data set.