REVIEW Assignment 1 - Building Currency Converter in Python

Start Assignment

Due Sep 9 by 23:59 **Points** 100 **Submitting** a file upload

Building Currency Converter in Python

Frankfurter is an open-source API for current and historical foreign exchange rates published by the European Central Bank.

Get started Fork me

The Brief:

An API (Application Programming Interface) is a software program that provides communication channels following the HTTP protocol between 2 applications. It is usually used for allowing a client to request or update information from a server.

You are tasked to develop a Python program that will perform currency conversion using data fetched from an open-source API: https://www.frankfurter.app/ (https://www.frankfurter.app/) .

The goal of your program is to display the current conversion rate between 2 currency codes at a specific date. It will also calculate the inverse conversion rate between these 2 currencies.

To do so, you will need to call 2 different API endpoints from the Frankfurter app:

- Extracting the list of available currency codes (documentation: https://www.frankfurter.app/docs/#currencies (https://www.frankfurter.app/docs/#currencies))
- Extracting the historical conversion rate for the specified currency codes and a given(documentation: https://www.frankfurter.app/docs/#historical (https://www.frankfurter.app/docs/#historical))

Description:

In this individual assignment, you will develop a python program that will take 2 currency codes as input arguments. Here is the command for running your script:

```
python main.py <date> <currency1> <currency2>
```

Your script will return the following outputs:

Scenario	Example	Output
Success	`python main.py 2022-01-01 GBP AUD`	The conversion rate on 2021-07-16 from GBP to AUD was 1.8649. The inverse rate was 0.5362
Missing argument	`python main.py`	[ERROR] You need to provide 3 arguments in the following order: <date> <currency1> <currency2>
Missing argument	`python main.py 2022-01-01`	[ERROR] You need to provide 3 arguments in the following order: <date> <currency1> <currency2>
Missing argument	`python main.py 2022-01-01 AUD`	[ERROR] You need to provide 3 arguments in the following order: <date> <currency1> <currency2>
Too many argument	`python main.py 2022-01-01 AUD EUR GBP`	[ERROR] You need to provide 3 arguments in the following order: <date> <currency1> <currency2>
Incorrect currency	`python main.py 2022-01-01 usd AAA`	AAA is not a valid currency code
Incorrect currency	`python main.py 2022-01-01 AAA USD`	AAA is not a valid currency code
Incorrect currencies	`python main.py 2022-01-01 AAA bbb`	AAA and bbb are not valid currency codes
Incorrect date	`python main.py 2022/01/01 AAA USD`	Provided date is invalid
Incorrect date	`python main.py 2022-01-41 EUR bbb`	Provided date is invalid
API error		There is an error with Frankfurter API

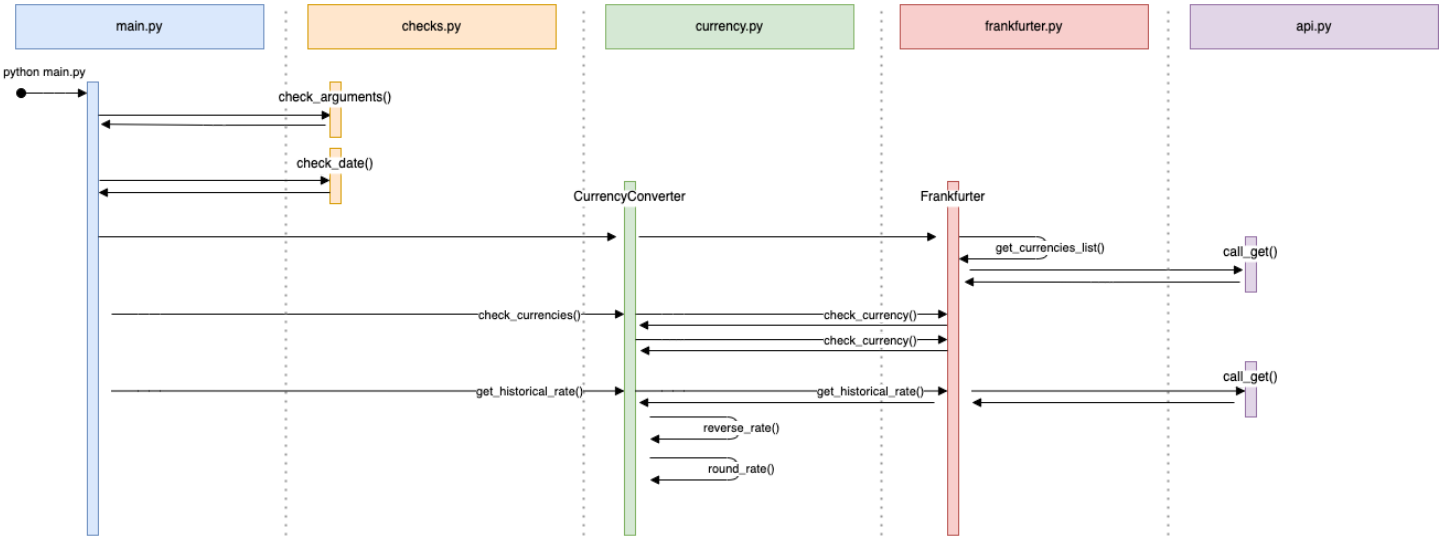
Your program will be composed of multiple files:

- **main.py**: main program used for running your business logics

- **checks.py**: python script that will contain the code for checking inputted arguments and date validity
- **api.py**: python script that will contain the code for making API calls
- **frankfurter.py**: python script that will contain the class used for calling relevant Frankfurter endpoints
- **currency.py**: python script that will contain the class used for extracting currency conversion rate and calculating the inverse rate
- **test_checks.py**: python script for testing code from checks.py
- **test_frankfurter.py**: python script for testing code from frankfurter.py
- **test_api.py**: python script for testing code from api.py
- **test_currency.py**: python script for testing code from currency.py
- **README.md**: a markdown file containing your details (full name, student id), a description of this project, listing of all Python functions and classes and instructions for running your code

Each of these files have been pre-populated. You will need to fill the defined functions and classes with your code. You are allowed to add more custom Python elements if you wish but they need to be compatible with the original defined functions and classes.

Here is the flow chart of this program:



Submission:

You will submit a zip file containing your python scripts and documentation. The name of the zip file **SHOULD** follow this convention: dsp_at1_<*student_id*>.zip

You can find the structure template here: **link (https://drive.google.com/file/d/19l2anxv9RYOl7BB_7CJhZQQl9ioF8WEh/view?usp=sharing)**

The zip file needs to contain the following files:

- main.py
- checks.py
- api.py
- frankfurter.py
- currency.py
- test_checks.py
- test_frankfurter.py
- test_api.py
- test_currency.py
- README.md

All assignments need to be submitted before the due date on Canvas. Penalties will be applied for late submission.

Assessment Criteria:

- Quality and reliability of Python code
- Readability and consistency of coding style
- Level of clarity for documentation of pseudo code and code
- Comprehensibility and relevance of unit tests