***Building Currency Converter in Python***

**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/  (Links to an external site.)](https://www.frankfurter.app/) .

The goal of your program is to display the current conversion rate between 2 currency codes. It will also calculate the inverse conversion rate between the given 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  (Links to an external site.)](https://www.frankfurter.app/docs/#currencies) )
* Extracting the current conversion rate for the specified currency codes (documentation: [https://www.frankfurter.app/docs/#latest  (Links to an external site.)](https://www.frankfurter.app/docs/#latest) )

**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 GBP AUD

Your script will return the following outputs:

|  |  |  |
| --- | --- | --- |
| **Scenario** | **Example** | **Output** |
| Success | python main.py GBP AUD | Today's (2021-07-16) conversion rate from GBP to AUD is 1.8649. The inverse rate is 0.53622 |
| Missing argument | python main.py | [ERROR] You haven't provided 2 currency codes |
| Incorrect argument | python main.py usd AAA | AAA is not a valid option |
| API error |  | There is an error with API call |

Your program will be composed of multiple files:

* **main.py** : main program used for entering the input parameters (currency codes) and display the results
* **api.py** : python script that will contain the code for calling API endpoints
* **currency.py** : python script that will contain the code for checking if currency code is valid, store results and format final output
* **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.

|  |  |  |
| --- | --- | --- |
| **File** | **Defined Function/Class** | **Description** |
| main.py | def get\_rate | Main function that will check input arguments, extract the conversion rate and format the output |
| api.py | def call\_api | Function used to make API calls to the Frankfurter app |
| api.py | def format\_currencies\_url | Function used for formatting the URL for the ‘currency’ endpoint from the Frankfurter app |
| api.py | def get\_currencies | Function that will extract the list of available currencies from the Frankfurter app |
| api.py | def format\_latest\_url | Function used for formatting the URL for the ‘latest’ endpoint from the Frankfurter app |
| currency.py | def check\_valid\_currency | Function that will check if a given currency code belongs to the list of available currencies from the Frankfurter app |
| currency.py | class Currency | Class defining the relevant information to be stored as attributes, methods for calculating inverse rate and formatting output |
| currency.py | def extract\_api\_result | Function that will read API output, instantiate a Currency class and calculate the inverse rate |
| test\_api.py | class TestFormatUrl | Class defining the unit tests for format\_currencies\_url and format\_latest\_url functions |
| test\_api.py | class TestAPI | Class defining the unit tests for get\_currencies and call\_api functions |
| test\_currency.py | class TestValidCurrency | Class defining the unit tests for check\_valid\_currency  function |
| test\_currency.py | class TestExtractApi | Class defining the unit tests for extract\_api\_result function   and Currency class |

Here is the flow chart of this program:



**Submission:**

You will submit a zip file containing your python scripts and documentation. You can find the structure template here: [link  (Links to an external site.)](https://drive.google.com/file/d/1JIJTuqyK-VAp1ydUjLCxJeu0Nb1hk6AP/view?usp=sharing)

The zip file needs to contain the following files:

* main.py
* api.py
* currency.py
* test\_api.py
* test\_currency.py
* README.md
* Pipfile
* Pipfile.lock

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

**Note:**

the pseudo-code is the part where you tried to describe the logics you want to implement programmatically. The description I put in each function is their high level objectives. What you need to put is to explain how you are going to get to it.