API REST exercises

Tiziano Fagni
BDSCC course, 2024/2025

Use “requests” library to query REST Web services

```python
import requests

# URL of REST Web service
url = "https://api.example.com/data"

# Execution of GET request
response = requests.get(url)

# Verify if the call has been successful
if response.status_code == 200:
    # Extract JSON data from response
    data = response.json()
    print("Received data from Web service:")
    print(data)
else:
    print(f"Error in the request: {response.status_code}")
```

Use “requests” library to query REST Web services (2)

- Use the proper “requests” method for the wanted HTTP request type:
  - requests.get() for GET type
  - requests.post() for POST type
  - requests.put() for PUT type
  - requests.delete() for DELETE type

```
headers = {'Authorization':'YOUR_BEARER_TOKEN'}
data = {'name':'Pippo',...}

response = requests.get(url, headers=headers)  # GET request

response = requests.post(url, json=data, headers=headers)  # POST request

response = requests.put(url, json=data, headers=headers)  #PUT request

response = requests.delete(url, headers=headers)  # DELETE request
```

Every Web service has its own way to specify bearer token

Specify the data to pass to Web service

Call the right method!

Ex1

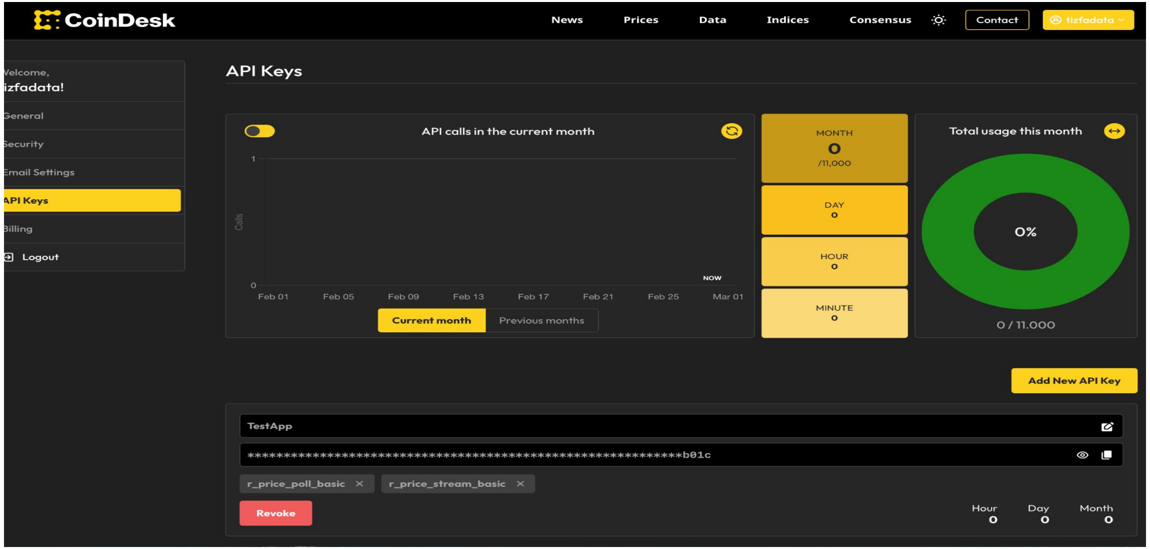
Write a Python program to fetch and display detailed market data for specific cryptocurrencies. Your program should be able to access real-time data for Bitcoin (BTC), Ethereum (ETH), and Ripple (XRP) against USD and EUR.

For each cryptocurrency extract and print on screen:

- Cryptocurrency Name (e.g., BTC, ETH, XRP)
- Market Capitalization in USD and EUR
- Current Price in USD and EUR
- Daily Low Price in USD and EUR
- Daily High Price in USD and EUR
- Daily Trading Volume in USD and EUR

The data is available through REST endpoint:
https://min-api.cryptocompare.com/documentation?key=Price&cat=multipleSymbolsFullPriceEndpoint

Requirements for Ex2: add new API key



TestApp

**b01c

r_price_poll_basic ✕ r_price_stream_basic ✕

1. Go to https://developers.coindesk.com and create new account
2. Go to https://developers.coindesk.com/settings/api-keys and add new key

Use the API key in your Python call or use `&api_key={API_KEY}` in request URL

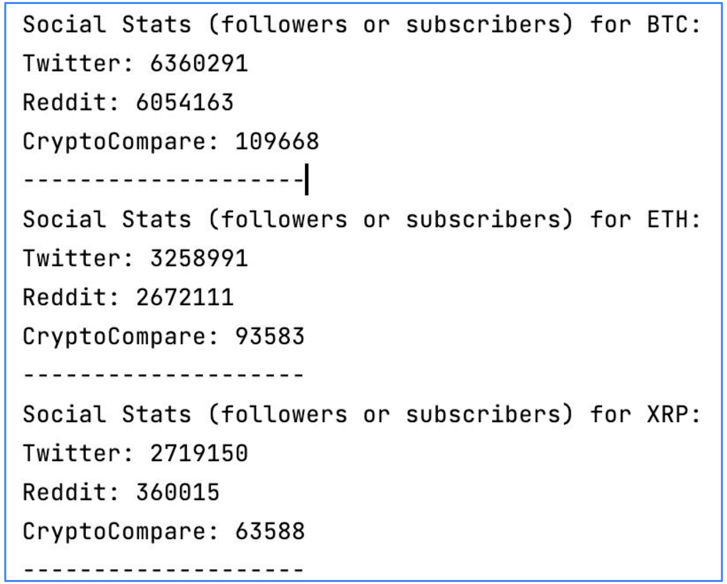
```
# Replace 'API_KEY' with your actual API key
headers = {'Apikey': API_KEY}
response = requests.get(url, headers=headers)
```

Ex2

The goal of this exercise is to develop a Python program that leverages the CryptoCompare API to fetch and analyze the social media presence and popularity of major cryptocurrencies. The students will focus on Bitcoin (BTC), Ethereum (ETH), and Ripple (XRP), exploring their social media statistics including Twitter followers, Reddit subscribers, and CryptoCompare followers.

To do it:

- use coins endpoint to get coins IDs
- use social endpoint to get social data



```
Social Stats (followers or subscribers) for BTC:
Twitter: 6360291
Reddit: 6054163
CryptoCompare: 109668
--------------------
Social Stats (followers or subscribers) for ETH:
Twitter: 3258991
Reddit: 2672111
CryptoCompare: 93583
--------------------
Social Stats (followers or subscribers) for XRP:
Twitter: 2719150
Reddit: 360015
CryptoCompare: 63588
--------------------
```

Ex3

Compare the current weather conditions in a given city with the same day over the past 30 years using the Open-Meteo API. No API key is required!

1. Retrieves today’s weather for a given city.
2. Retrieves historical weather data for the same date over the last 30 years.
3. Calculates trends for:
    a. Temperature (max and min)
    b. Wind speed
    c. Precipitation
4. Generates a CSV file with all data (today and historical data).
5. Prints a summary comparing today’s data with historical averages.

Ex4

Using the TMDB API:

- Search all movies having the word “titans” in their titles (tip: use search/movie endpoint and handle data pagination using *total_pages* attribute in JSON answer).
- For each movie found, obtain all the relevant info for the film such as year, genre, title, overview, language, etc. (tip: use movie/{movie_id} endpoint)
- Write in a JSONL file all found movies with related gathered info

REQUIREMENTS

- You must have a valid account in TMDB web site
- You need to create an API developer key

Ex4: how to authenticate the user

A Include the API key in the URL endpoint →

```
https://api.themoviedb.org/3/search/movie? api_key={API_KEY}
```

B Use the bearer token in headers of request →

```
headers = {
    "accept": "application/json",
    "Authorization": "Bearer
eyJhbGciOiJIUzI1NiJ9.eyJhdWQiOiJhYzMxZTZiMmI3YmU2NmZjN2
YzZTJkMzMzMTc1ZjFiMiIsIm5iZiI6MTczOTg5MTczMC4wMTIsInN1Y
iI6IjY3YjRhNDEyYjMwODM3ZDhmOGUwZDk1NiIsInNjb3BlcyI6WyJh
cGlfcmVhZCJdLCJ2ZXJzaW9uIjoxfQ.ki-SqsfdMR6lwVdCPCsGltLT
NOuSIjWMaeKnx-8Q3dg"
}

response = requests.get(url, headers=headers)
```