**API TESTING USING PYTHON**

By mukesh otwani & others

**What is an API**:--

* Application programming interface
* It allows to talk from one software to other software

For example let us take swigy---Zomato---&---ola---uber where we can see maps are being used by these application. So this map is being provided by google maps system i.e Google maps API. But these apps cant access the maps data directly , so before accessing the data from the maps API google add a protection which is nothing but Google Maps API. So apps like zomato will not ask data directly from google but zomato ask data from maps API. So API acts as a bridge || messenger || mediator for data to travel from one application || software to other application || software. When ever zomato wants data of maps then it **REQUESTS** the API for the data & API **REQUESTS** the maps i.e system whether the data can be provided or not , if YES then then system **RESPONSE** to API & API provide the data to zomato. If system says NO the API will return an error to zomato.

Similarly,we see so many apps while login we get options like Sign up with Google & sign up with Facebook. This option is given because apps cant go to each and every customer to verify so verification is already done by Google and facebook databases, so apps take the verification from google and facebook. This is through Google APIs Explorer & FACEBOOK for Developers. Checks for valid user or not.

Other examples are goibibo—makemytrip—other travel sites uses the different flight carriers API for tickets listing—booking—prices etc

In a restaurant the waiter acts as an API between the kitchen and customer having the menu list.

So API provides the protection for the data through API KEYS i.e is a security code which is given to the developer by the system so using this API KEYS the developer can access the data from through the API.

KEYS not only provide SECURITY but also with the help of KEYS system can monitor the information i.e by whom or which developer is accessing the data etc.

UESES OF APIs :---

* In our projects we no need to write the login function from scratch but using APIs we can use signup with google and signup with facebook.
* Weather or temperature related projects , where we can integrate any application that provide weather API to know the temperature of different locations.

REST :--- Representational state transfer

Initial requirements:--

* pip install requests
* pip install jsonpath

<https://reqres.in/> 🡪 This is the website provide you with the fake API for practice Testing API

import requests

p = requests.get("https://reqres.in/api/users?page=2")

print(p)

import requests

p = requests.get("https://reqres.in/api/users?page=2")

print(p)

print(type(p)) #we get the type of the variable

print(dir(p)) #To know the list of properties inside the p variables

import requests

p = requests.get("https://reqres.in/api/users?page=2")

code = p.status\_code

assert code == 200 , "code doesnt match" #this gives us no error as the code is 200

# here status code is nothing but response code

assert code == 201 , "code doesnt match" #this gives us the error

import requests

p = requests.get("https://reqres.in/api/users?page=2")

code = p.status\_code

print(p.text)

# will get response in plain string i.e content of the response in unicode

print("\*"\*50)

print(p.content)

# return in byte format

print("\*"\*50)

print(p.json())

# return the data in serialized format i.e encoded response of data

print(p.headers)

#it will return all the headers

print(p.cookies)

#it will return the cookies

print(p.encoding)

#it will return the encoding

print(p.url)

#it will return the final url which we will be hitting

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**CRUD**

Create – post

Read – get

Update – put/patch

Delete – delete

put – Update/Replace

patch – Update/Modify

In put – all the properites of the object be provided while making request

If record exist – then that record will be updated/replaced

If record doesn’t exist --- then a new record will be created for the user

{

“Name”:”Mukesh”

“Phone”:90090

“Address”:”Bangalore”

}

In patch – specify the property that you want to update

{

“Address”:”vizag”

}

START :--- **get method in api**

import requests

"""

taking a dumy api

"""

resp = requests.get("https://reqres.in/api/users?page=2")

#printing the response code i.e 200

print(resp)

# to know the type of resp variable

print(type(resp))

print(dir(resp))

# directory i.e dir

# it shows the dir available in the api

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Difference between text & content & json :----

Text :-- will return the plain string format i.e content of response in unicode

Content :-- will return the content in bytes format

Json :-- will return json encoded content

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**JSON FILE VIEWER :---**

<http://jsonviewer.stack.hu/> --- this url will is to view the json data

copy & paste the below JSON in the above link

{'page': 2, 'per\_page': 6, 'total': 12, 'total\_pages': 2, 'data': [{'id': 7, 'email': 'michael.lawson@reqres.in', 'first\_name': 'Michael', 'last\_name': 'Lawson', 'avatar': 'https://reqres.in/img/faces/7-image.jpg'}, {'id': 8, 'email': 'lindsay.ferguson@reqres.in', 'first\_name': 'Lindsay', 'last\_name': 'Ferguson', 'avatar': 'https://reqres.in/img/faces/8-image.jpg'}, {'id': 9, 'email': 'tobias.funke@reqres.in', 'first\_name': 'Tobias', 'last\_name': 'Funke', 'avatar': 'https://reqres.in/img/faces/9-image.jpg'}, {'id': 10, 'email': 'byron.fields@reqres.in', 'first\_name': 'Byron', 'last\_name': 'Fields', 'avatar': 'https://reqres.in/img/faces/10-image.jpg'}, {'id': 11, 'email': 'george.edwards@reqres.in', 'first\_name': 'George', 'last\_name': 'Edwards', 'avatar': 'https://reqres.in/img/faces/11-image.jpg'}, {'id': 12, 'email': 'rachel.howell@reqres.in', 'first\_name': 'Rachel', 'last\_name': 'Howell', 'avatar': 'https://reqres.in/img/faces/12-image.jpg'}], 'support': {'url': 'https://reqres.in/#support-heading', 'text': 'To keep ReqRes free, contributions towards server costs are appreciated!'}}

Then you will be above to see the json where we found some data

import requests

"""

taking a dumy api

"""

resp = requests.get("https://reqres.in/api/users?page=2")

print(resp.json())

print(resp.headers)

# this will return all the headers available

print(resp.cookies)

#will return all the cookies

print(resp.encoding)

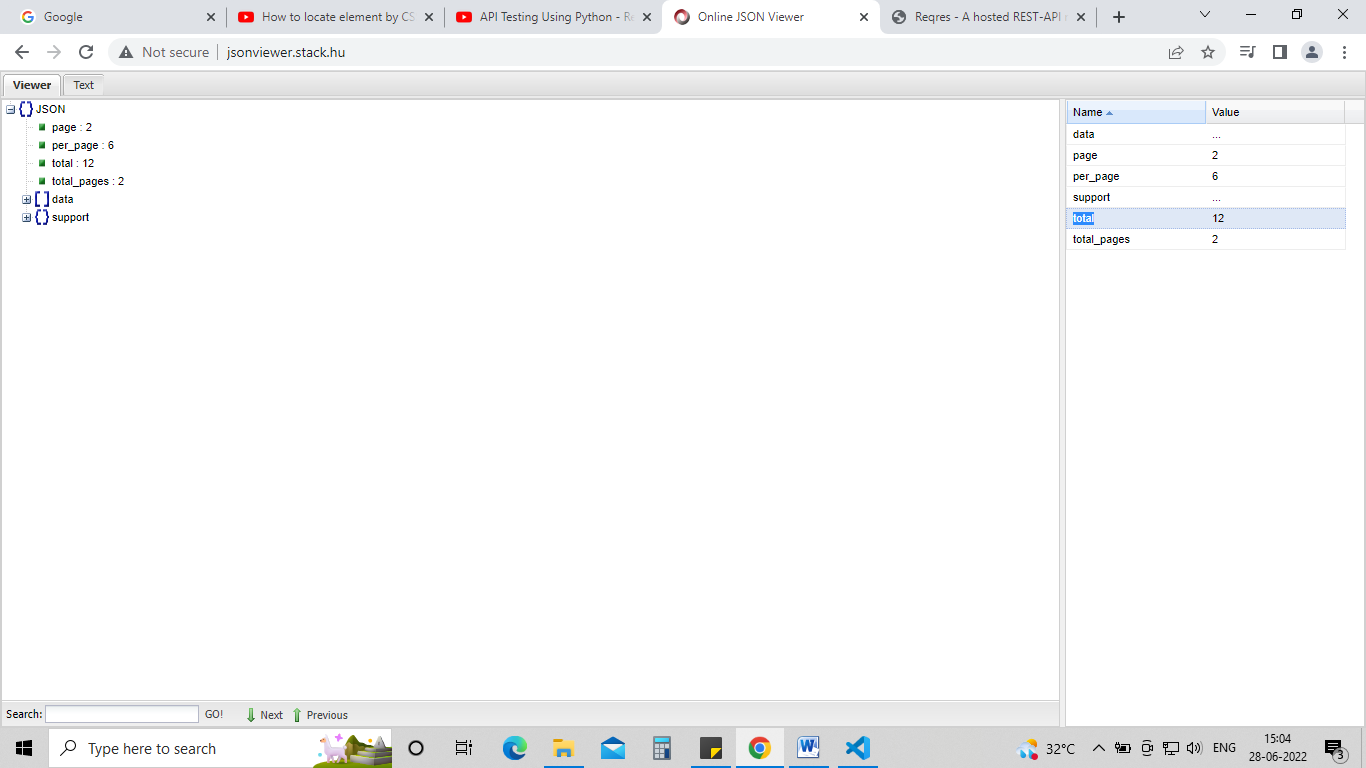
#will return the encoding

print(resp.url)

#will return finall url

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TO VIEW THE JSON FORMAT OF THE LINK :---



import requests

resp = requests.get("https://reqres.in/api/users?page=2")

#we will check how to validate certain response here

json\_response = resp.json()

#calling the json format from that variable

"""This will print total number of pages available in that json

 here 12 pages availabel

 """

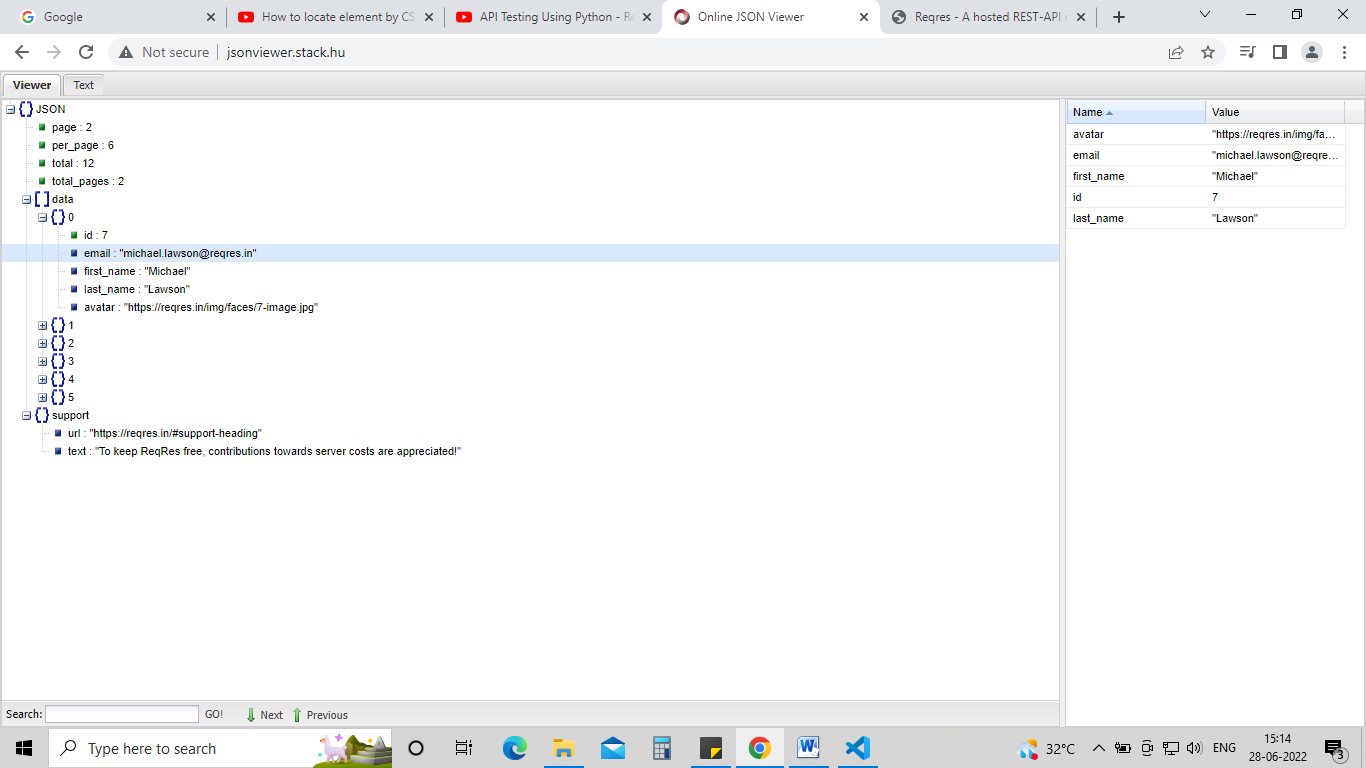
print(json\_response['total'])

assert json\_response['total'] == 12 , "total pages count not matching"

#we asserted whether the pages are 12 or not

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**TO GET DATA FROM JSON FORMAT :---**



import email

import requests

resp = requests.get("https://reqres.in/api/users?page=2")

json\_response = resp.json()

"""

shown below we will be able to GET data from the

json format of the link

here we are getting the email from 0 record

"""

print(json\_response["data"][0]["email"])

#we are asserting the email ends with reqres.in or not

assert (json\_response["data"][0]["email"]).endswith("reqres.in")

#we are doing a failed assert here

assert (json\_response["data"][0]["email"]).endswith("gmail.in"), "email format not matching"

similarly we printed the last name from second record and asserted whether the last name is none or not

print(json\_response["data"][2]["last\_name"])

assert json\_response["data"][2]["last\_name"] != None

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""created a dictionary and assigned it to a variable

and passing it as a params"""

p = {"page":2}

resp = requests.get("https://reqres.in/api/users",params=p)

print(resp.url)

#printing the url to check correct or not

OR

resp = requests.get("https://reqres.in/api/users?page=2")

both are same here

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**post method in api**

post method is used to create a data

def post(url, data=None, json=None, \*\*kwargs):

    r"""Sends a POST request.

    :param url: URL for the new :class:`Request` object.

    :param data: (optional) Dictionary, list of tuples, bytes, or file-like

        object to send in the body of the :class:`Request`.

    :param json: (optional) json data to send in the body of the :class:`Request`.

    :param \\*\\*kwargs: Optional arguments that ``request`` takes.

    :return: :class:`Response <Response>` object

    :rtype: requests.Response

    """

    return request('post', url, data=data, json=json, \*\*kwargs)

here we will create a user data and send or post it using post method

import requests

"""created a dict that contains data """

payload = {

    "name": "morpheus",

    "job": "leader"

}

"""using post method sending the payload to that url

as shown in below """

resp = requests.post("https://reqres.in/api/users",data=payload)

#will print the response code

#here we are expecting 201

# 201 means user data created

print(resp)

"""this will print the user data in json format"""

print(resp.json())

"""asserting job is leader or not in the data of the user"""

assert resp.json()['job'] == 'leader'

"""asserting that id is not none"""

assert resp.json()['id'] != None

"""doing a false assert"""

assert resp.json()['name'] != 'morpheus', "name is not matching"

OUTPUT :--

<Response [201]>

{'name': 'morpheus', 'job': 'leader', 'id': '267', 'createdAt': '2022-06-28T11:02:47.892Z'}

Traceback (most recent call last):

File "g:\WORKSPACE\TESTING\practise\api\createuserapi.py", line 27, in <module>

assert resp.json()['name'] != 'morpheus', "name is not matching"

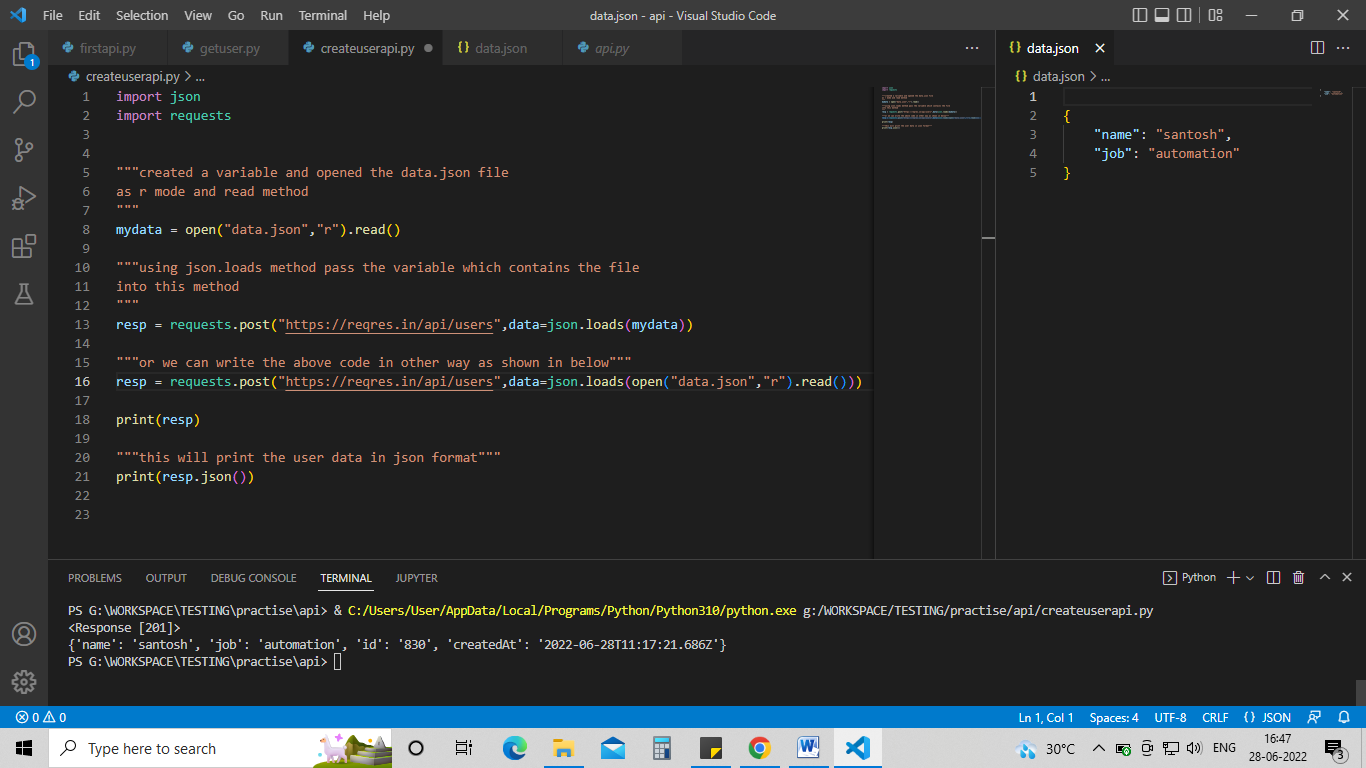
AssertionError: name is not matching

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**HOW TO POST A LARGE DATA EX:- A JSON FILE :--**

First create a .json file and store the data in it

For example I have created a data.json file and stored data in it



Data.json file contains the large or bulk data in json format as shown in the above pic

Code :--

import json

import requests

"""created a variable and opened the data.json file

as r mode and read method

"""

mydata = open("data.json","r").read()

"""using json.loads method pass the variable which contains the file

into this method

"""

resp = requests.post("https://reqres.in/api/users",data=json.loads(mydata))

"""or we can write the above code in other way as shown in below"""

resp = requests.post("https://reqres.in/api/users",data=json.loads(open("data.json","r").read()))

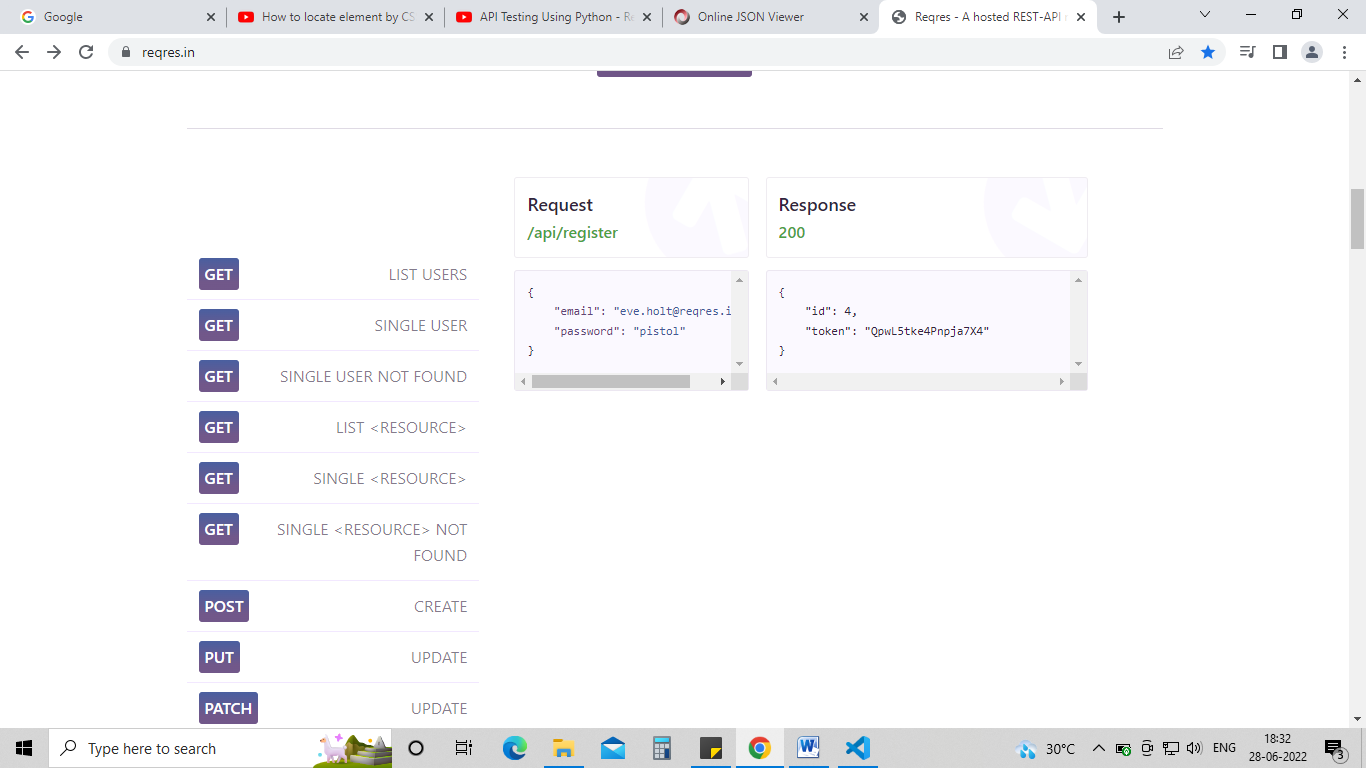
print(resp)

"""this will print the user data in json format"""

print(resp.json())

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Another example for POST :---



import requests

payload = {

    "email": "eve.holt@reqres.in",

    "password": "pistol"

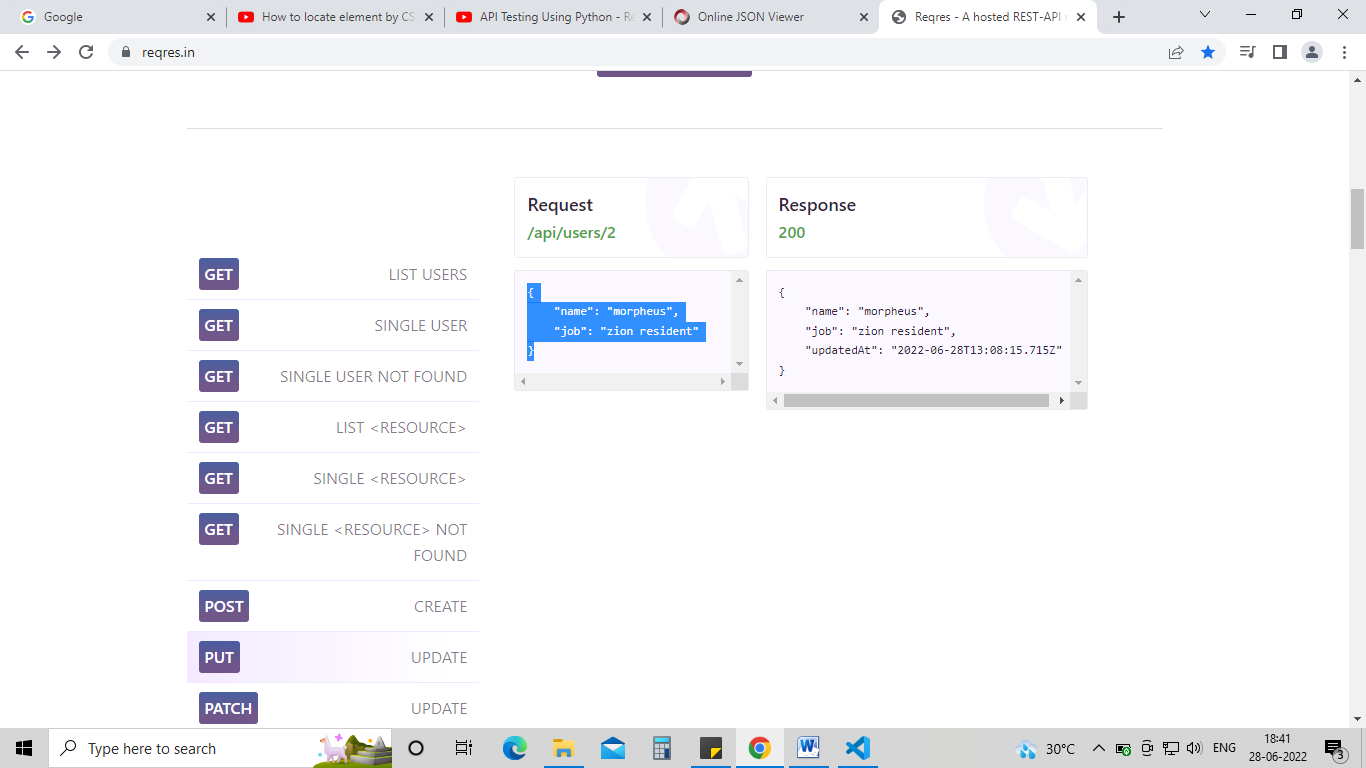
}

resp = requests.get("https://reqres.in/api/register",data=payload)

print(resp)

print(resp.json()['token'])

**PUT METHOD**



from urllib import response

import requests

import json

payload = {

    "name": "morpheus",

    "job": "zion resident"

}

resp = requests.put("https://reqres.in/api/users/2",data=payload)

print(resp)

print(resp.json())

print(resp.headers.get("content-Type"))

assert resp.json()['job'] == 'zion resident'

OUTPUT :--

<Response [200]>

{'name': 'morpheus', 'job': 'zion resident', 'updatedAt': '2022-06-28T13:11:15.915Z'}

application/json; charset=utf-8

**PATCH METHOD**

Similar to PUT method

from urllib import response

import requests

import json

payload = {

    "name": "API"

}

resp = requests.post("https://reqres.in/api/users/2",data=payload)

print(resp)

print(resp.json())

print(resp.headers.get("content-Type"))

assert resp.json()['name'] == 'API'

**DELETION METHOD**

import requests

#here we are deleting user 2 data

resp = requests.delete("https://reqres.in/api/users/2")

#asserting dleted or not

assert resp.status\_code == 204 , "user deletion failed"

**TIMEOUT METHOD**

When you are trying to hit an API but – server is slow – down – and you are getting the response after like 10sec or 15 sec

So in these cases we use timeout method , giving it a time limit

Ex if we are doing a payment and we give the timeout for 30 sec

i.e if the payment is done with in the 30 sec time limit then make it success or if it is done after 30 sec it should not pass

import requests

r = requests.get("https://httpbin.org/delay/3",timeout=5)

# here we gave timeout as 5 sec

# and we want response in 3 sec so it will pass here

r = requests.get("https://httpbin.org/delay/5",timeout=3)

# here we gave timeout for 3 sec

# but we called response for 5 sec

#so it will fail

print(r.status\_code)

<https://httpbin.org/> --- this link as necessary API methods for practice

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**AUTHENTICATION**

<https://requests.readthedocs.io/en/latest/user/authentication/> ---- read this documentation

<https://the-internet.herokuapp.com/> ---- use this url for auth practice

import requests

resp = requests.get("https://the-internet.herokuapp.com/basic\_auth",auth=('admin','admin'))

#here the username & password should be given as a tuple

print(resp.status\_code)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*