

Phase-3 Practice Project-Postman

Repository: https://github.com/Joshnitha/Phase2_Practice-Projects/

Repository link of Assisted Practices :

https://github.com/Joshnitha/Phase1_Practice_Projects/tree/main

Creating Well-Structured .Output for API Clients Using Postman to Get Weather Report.

Description

You are asked to create a well-structured output for their API client using Postman, which will hit that URL and get a detailed report on the weather in a quicker way.

Background of the problem statement:

To get the weather report in a well-structured output, we need to have a set of APIs of the weather application and automatable tool like Postman.

You must use the following:

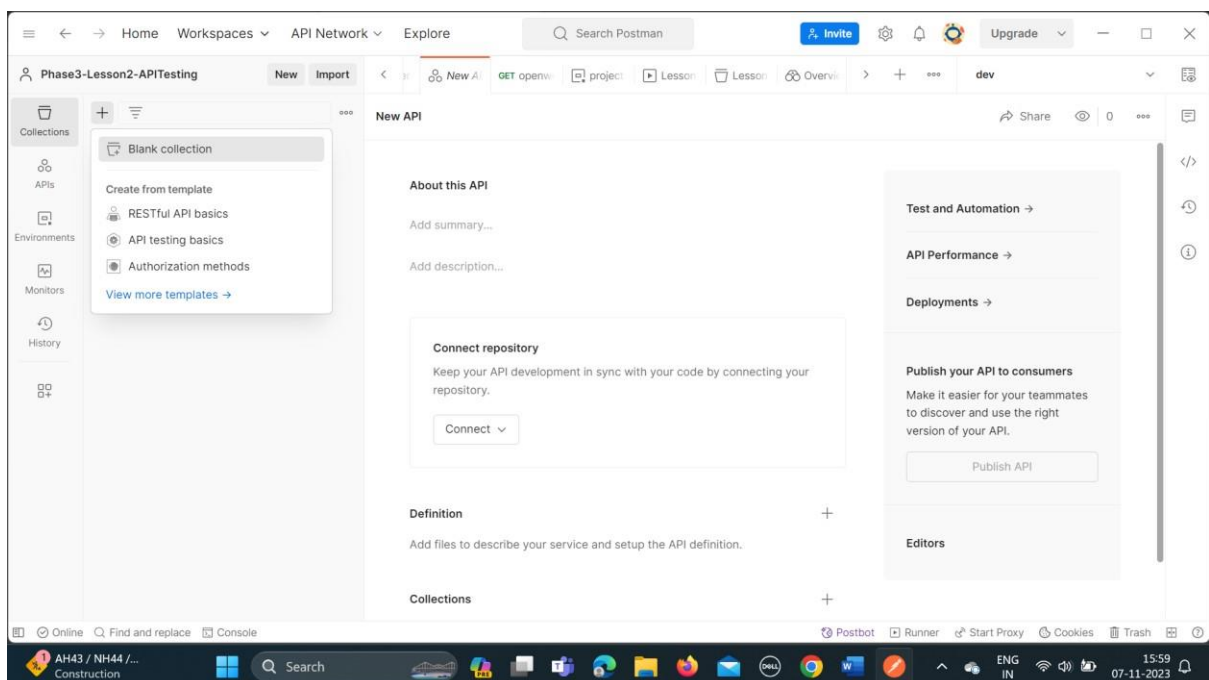
- Postman
- Endpoint

URL(<https://samples.openweathermap.org/data/2.5/weather?q=London,uk&appid=b6907d289e10d714a6e88b30761fae22>)

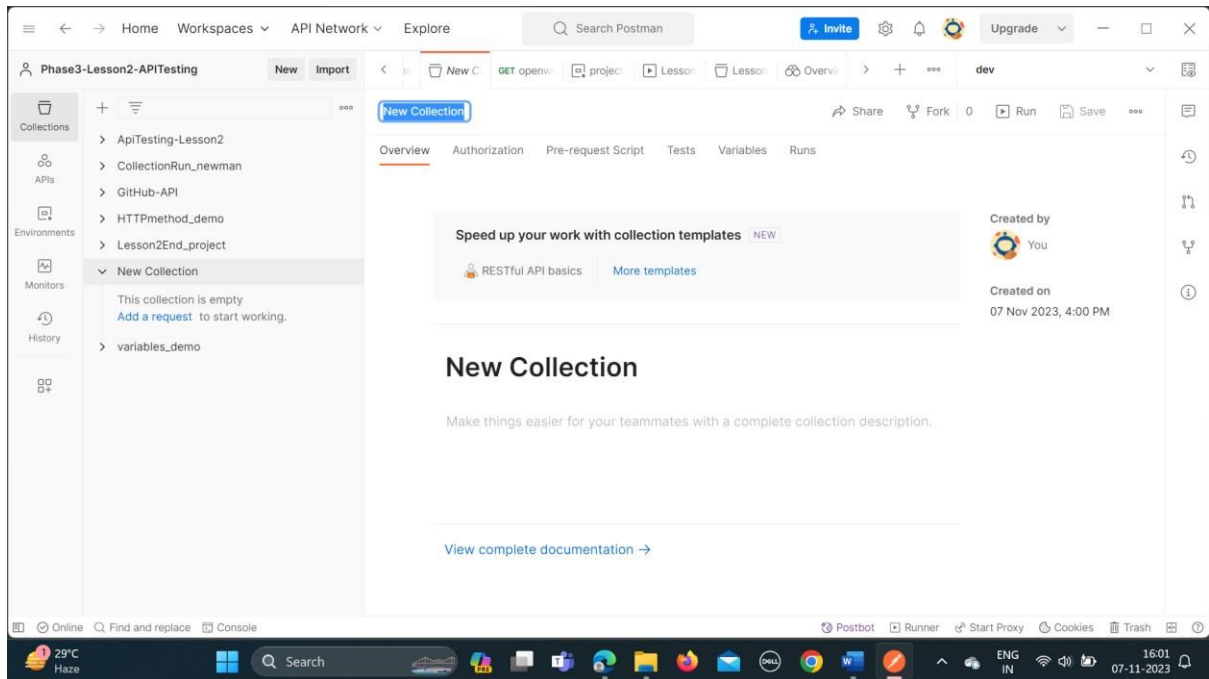
Steps:

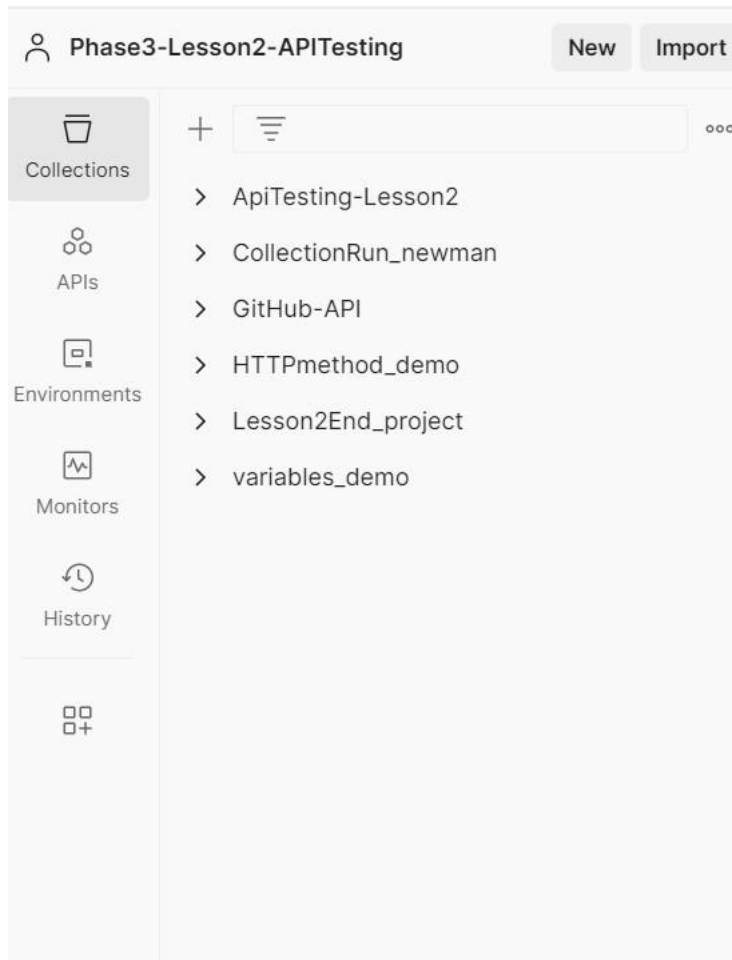
Create a Collection:

Click on the + sign and click on blank collection .



Give a name to collection.

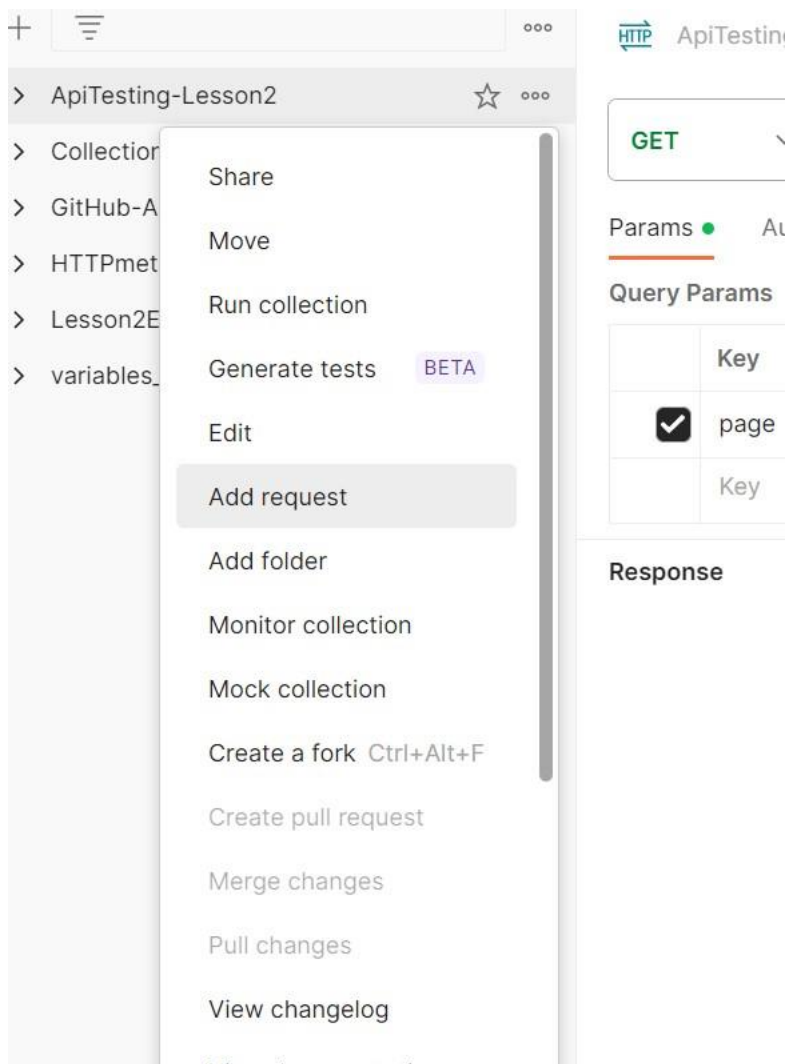




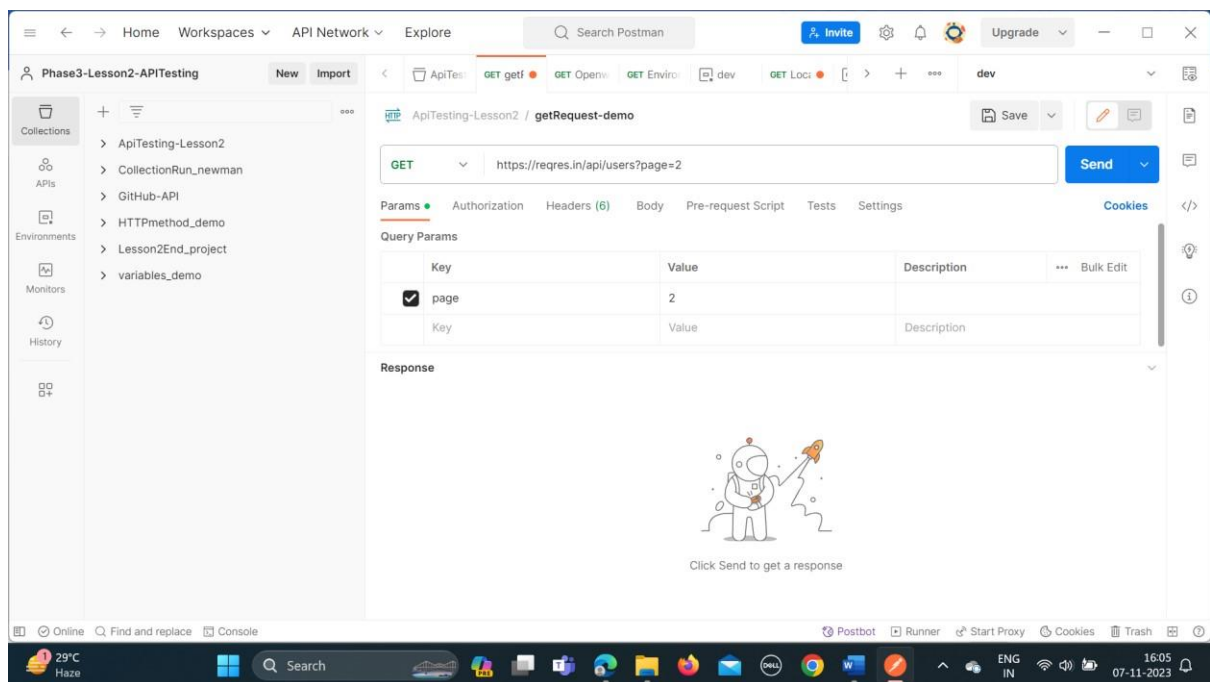
Create a API request:

Click on the 3dots beside the collection which your have created

Click on add request.



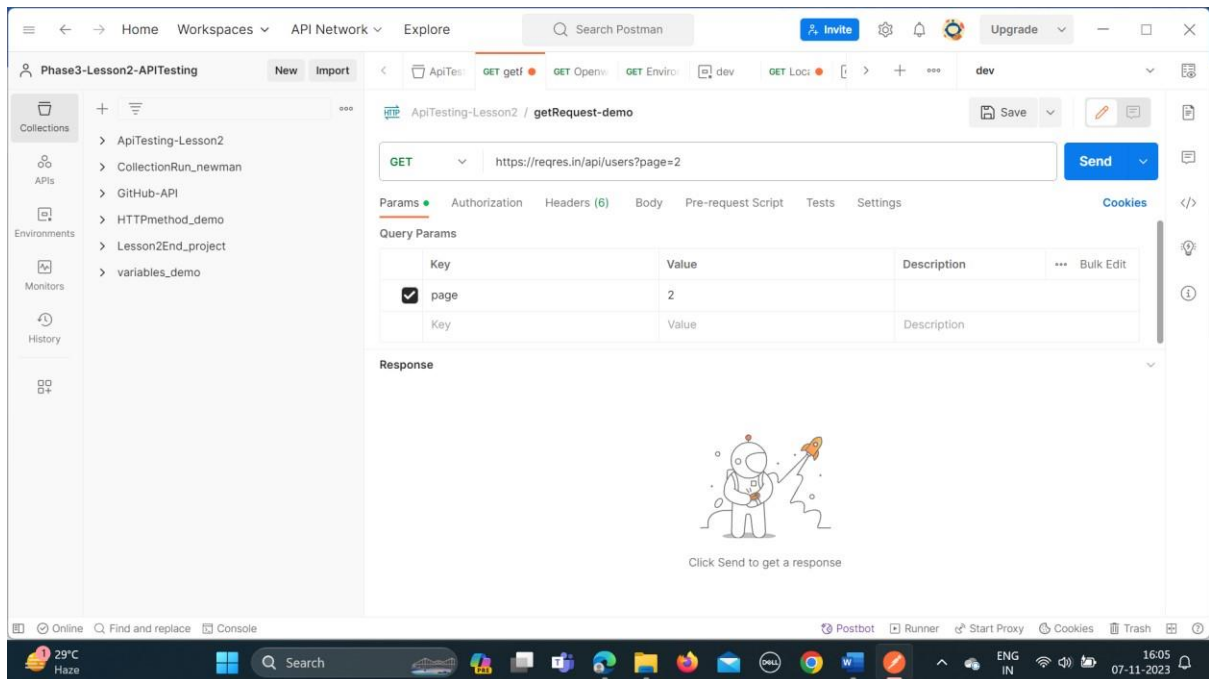
Then give then name to the request



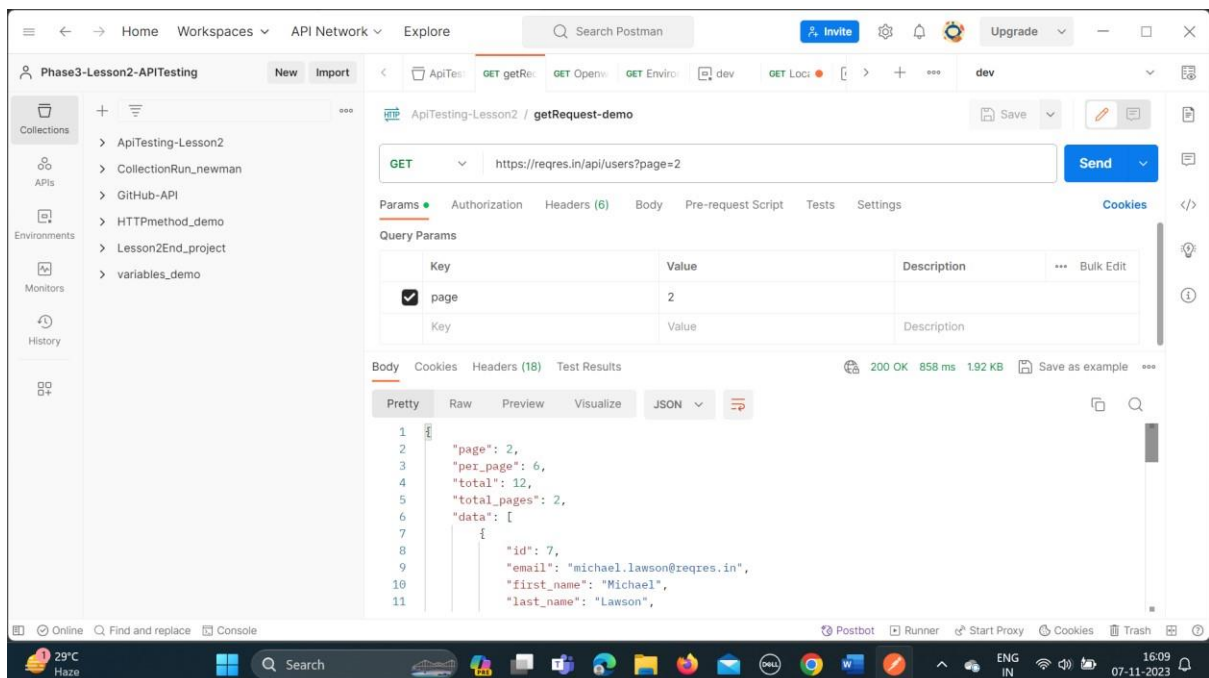
Create a GET request:

Select GET

Give url in the url section.



Click on save and send button,



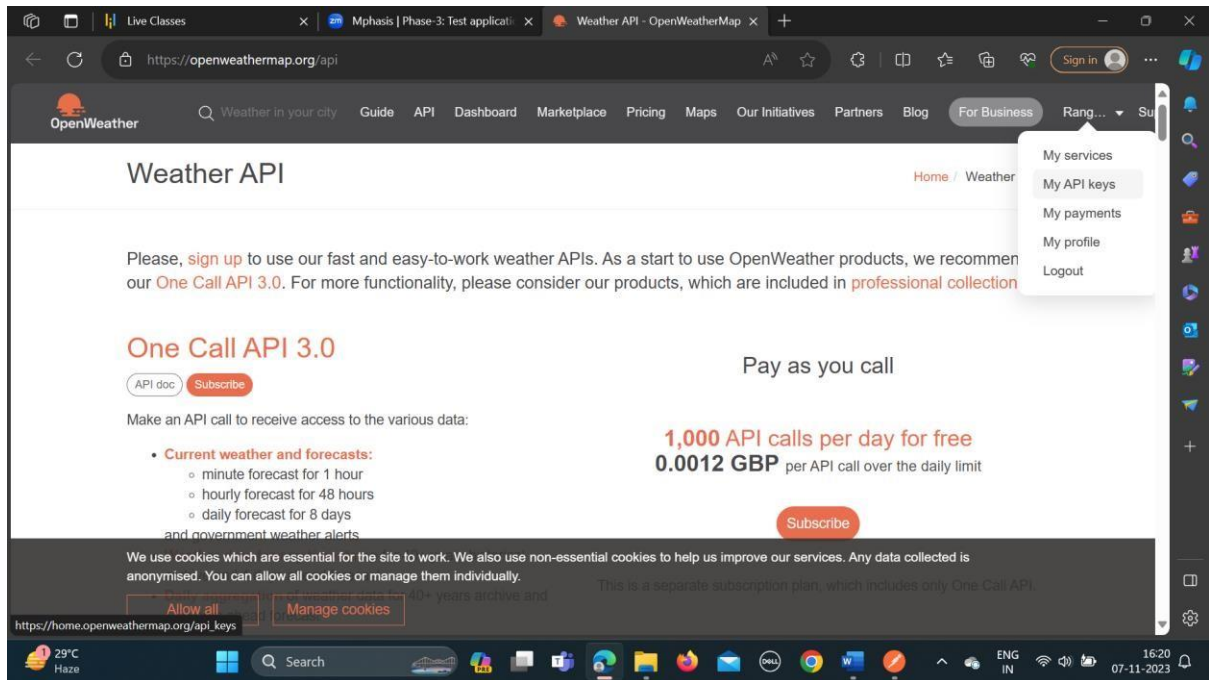
Create a Parameterize request:

Go to this openweathermap page

<https://openweathermap.org/api> create a

account and login to this application.

Now create a api key ⑦ go to u=your account ⑦ click on my api keys



Give a name to api key and click on generate.

Api key will be generated.

The screenshot shows the OpenWeather API keys management interface. At the top, there's a navigation bar with links like 'New Products', 'Services', 'API keys', 'Billing plans', 'Payments', 'Block logs', 'My orders', 'My profile', and 'Ask a question'. Below this, a message states: 'You can generate as many API keys as needed for your subscription. We accumulate the total load from all of them.' The main content area features a table with the following data:

Key	Name	Status	Actions
ccd932683dc3a4207fb62cadf0bc33c3	Default	Active	[Toggle] [Edit] [Delete]
3bdbed5674725f3d1216c2e8edb7ae19	apikeypostman	Active	[Toggle] [Edit] [Delete]

To the right of the table is a 'Create key' section with an input field for 'API key name' and a 'Generate' button. Below the table, there are sections for 'Product Collections' (Current and Forecast APIs, Historical Weather Data), 'Subscription' (How to start, Pricing), and 'Company' (OpenWeather is a team of IT experts and data scientists that has been practising deep weather data science since).

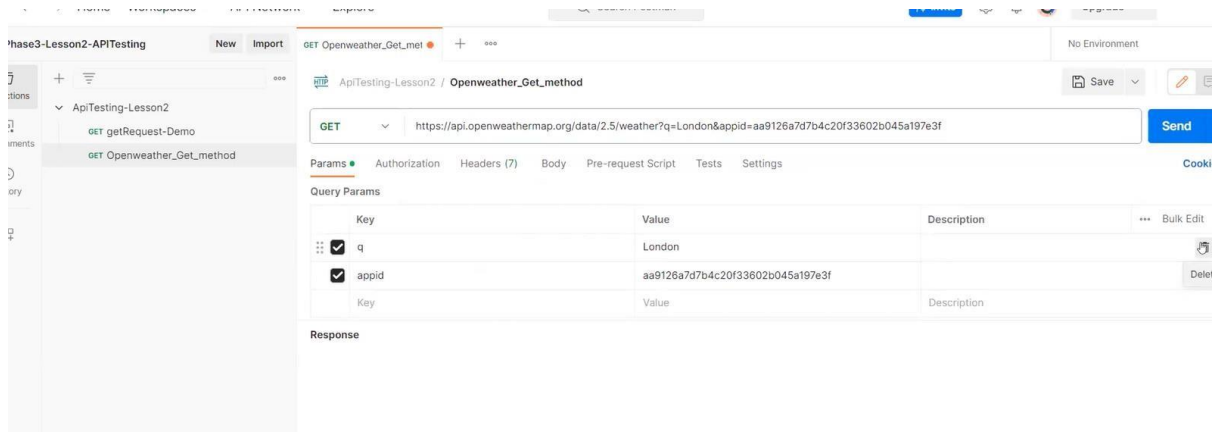
The api request we will use is

```
https://api.openweathermap.org/data/2.5/weather?q={city name}&appid=3bdbed5674725f3d1216c2e8edb7ae19
```

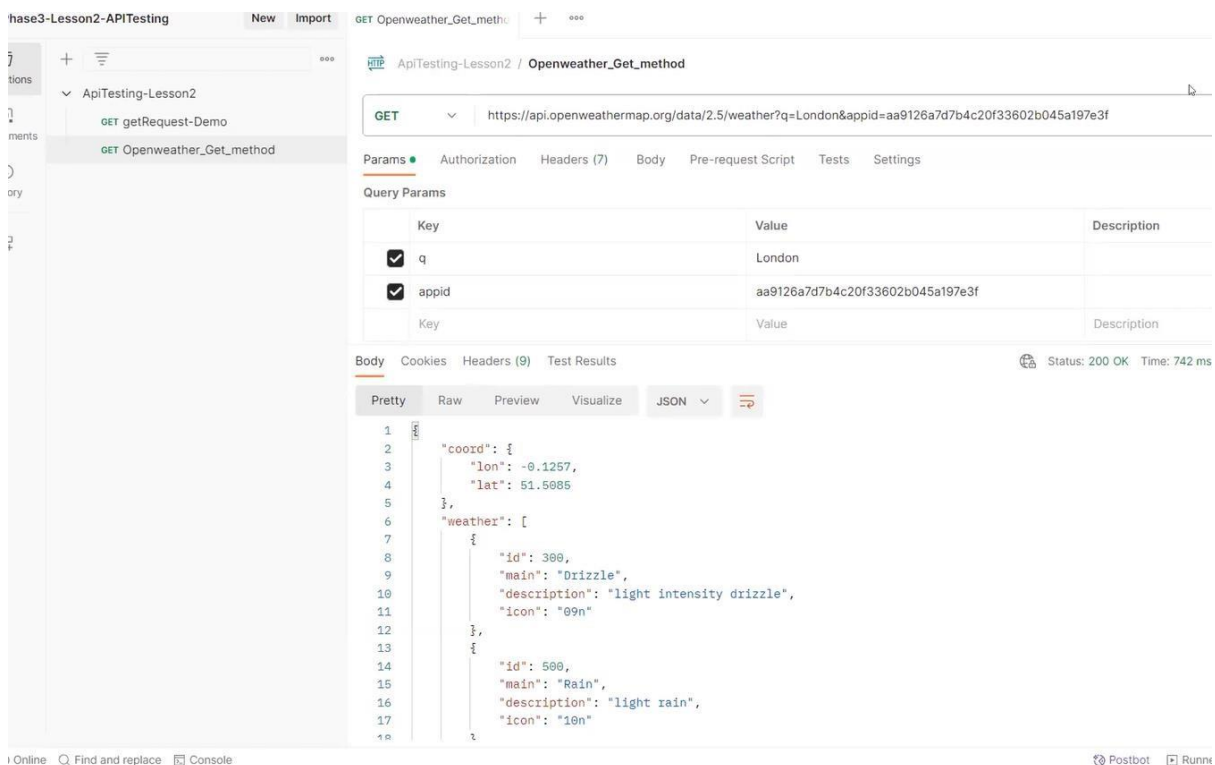
add another get request

The screenshot shows the Postman API client interface. The 'GET' method is selected, and the URL field is populated with 'Openweather_Get-method'. The 'Query Params' section is empty. The 'Response' section shows a placeholder image of a person holding a rocket. The interface includes a sidebar with navigation links, a top bar with 'New', 'Import', and 'GET New Request' buttons, and a bottom bar with 'Send' and 'Cookies' buttons.

Then give the above url and in city name give an city name.



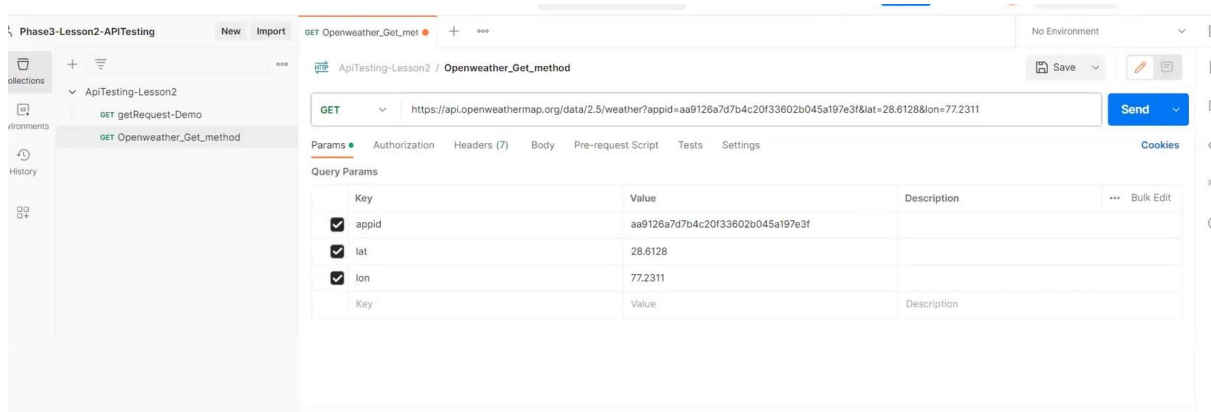
Click on save and send



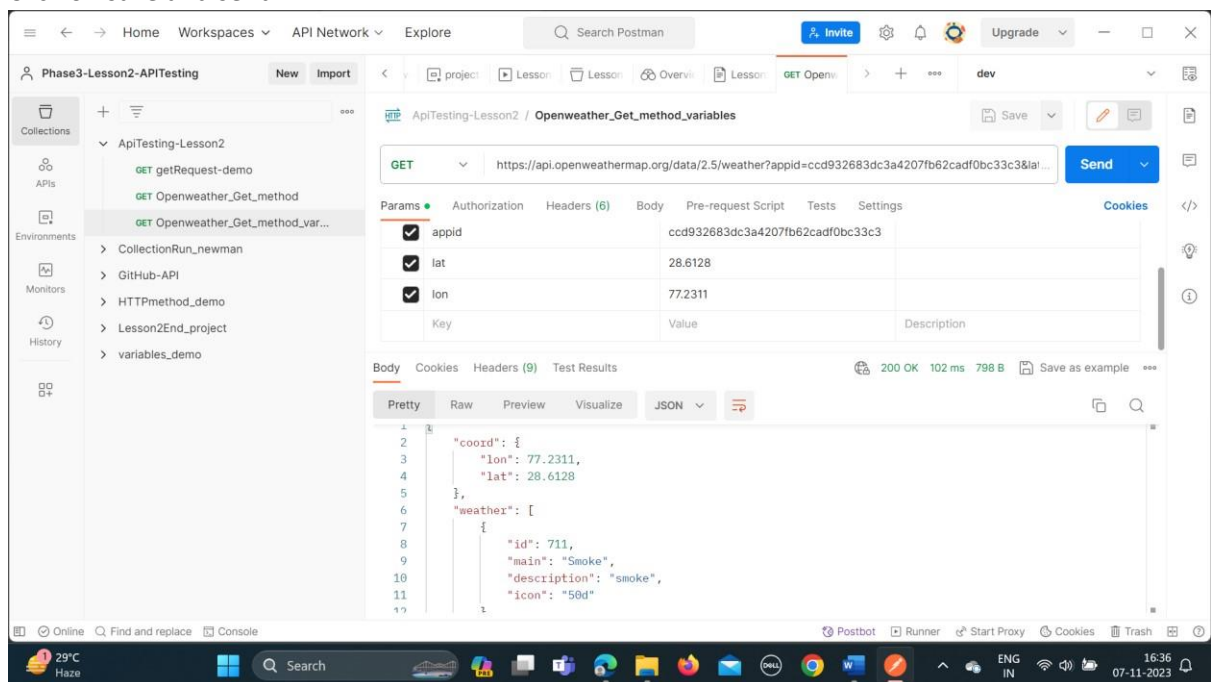
Add and delete the parameters Go

to paramas section of the request

Delete q parameter.



Click on save and send.



Lesson2-EndProject / Openweatherreq Save

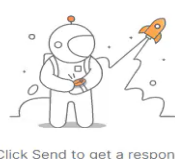
GET ▼ `{{Base_URL}}?appid=aa9126a7d7b4c20f33602b045a197e3f&q={{City}}`

Params • Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

	Key	Value	Description
<input checked="" type="checkbox"/>	appid	aa9126a7d7b4c20f33602b045a197e3f	
<input checked="" type="checkbox"/>	q	{{City}}	
	Key	Value	Description

Response



Click Send to get a response

Create a new environment

se3-Lesson2-APITesting New Import GET Openweatherreq Project-variables dev GET Openweather_Get_method + ... dev ▼

+ ▼

Globals

dev ☒

Project-variables

QA

Project-variables

Filter variables

Variable	Type	Initial value	Current value
<input checked="" type="checkbox"/> Base_URL	default	https://api.openweathermap.org	https://api.openweathermap.org
<input checked="" type="checkbox"/> City	default		
Add new variable			

se3-Lesson2-APITesting New Import GET Openweatherreq Project-variables dev GET Openweather_Get_method + ... dev ▼

+ ▼

ApiTesting-Lesson2

GET getRequest-Demo

GET Openweather_Get_method

GET Openweather_Get_method_var...

CollectionRUN-newman

Github-API

GET GetAllrepos

POST Create a repo

GET Get a specific repo

DEL DeleteRepo

Lesson2-EndProject

GET Openweatherreq

Project-variables

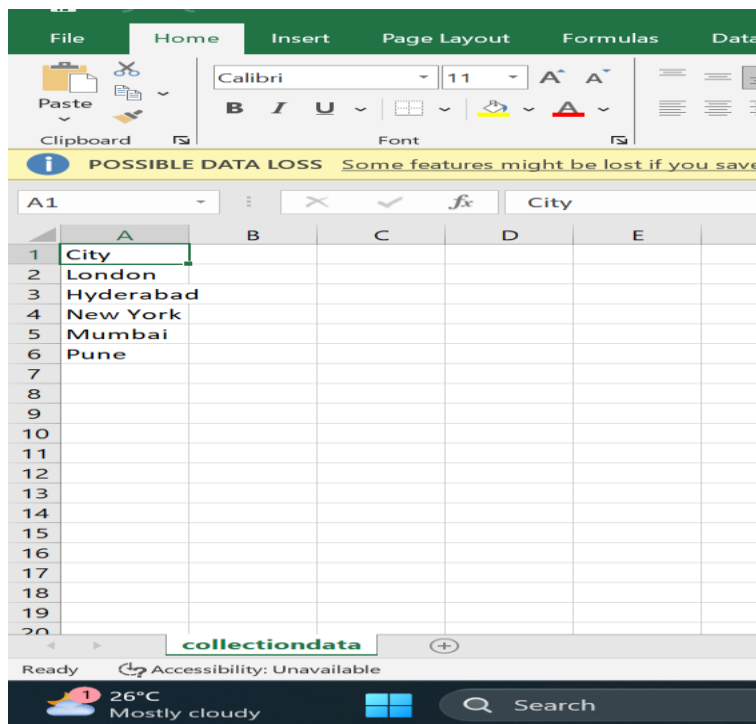
Filter variables

Variable	Type	Initial value	Current value
<input checked="" type="checkbox"/> Base_URL	default	https://api.openweathermap.org	https://api.openweathermap.org
<input checked="" type="checkbox"/> City	default	New Delhi	New Delhi
Add new variable			

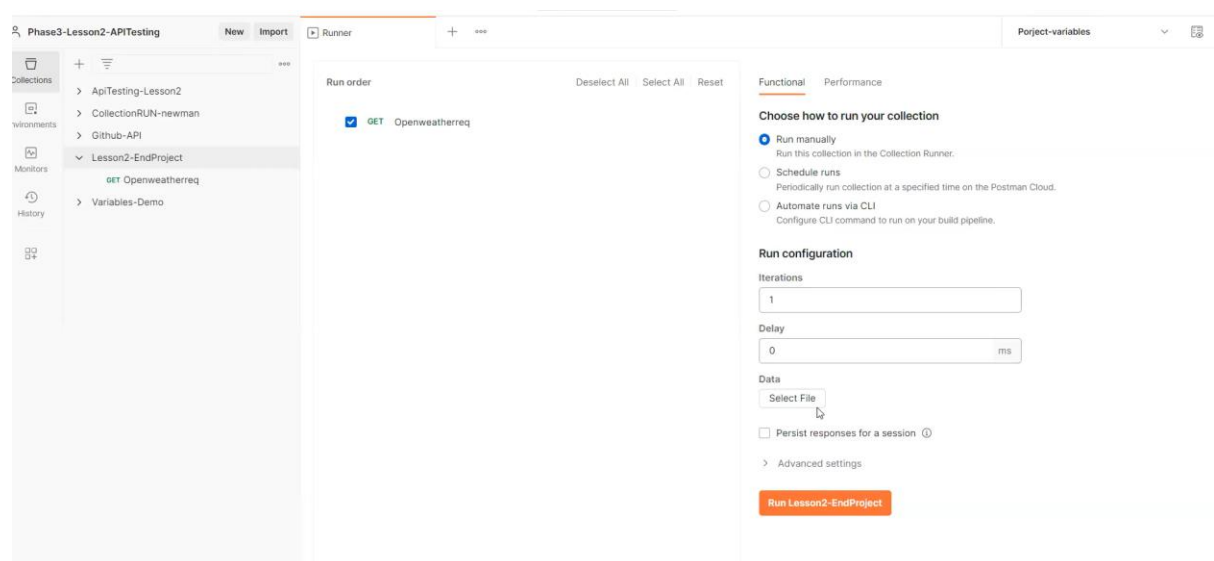
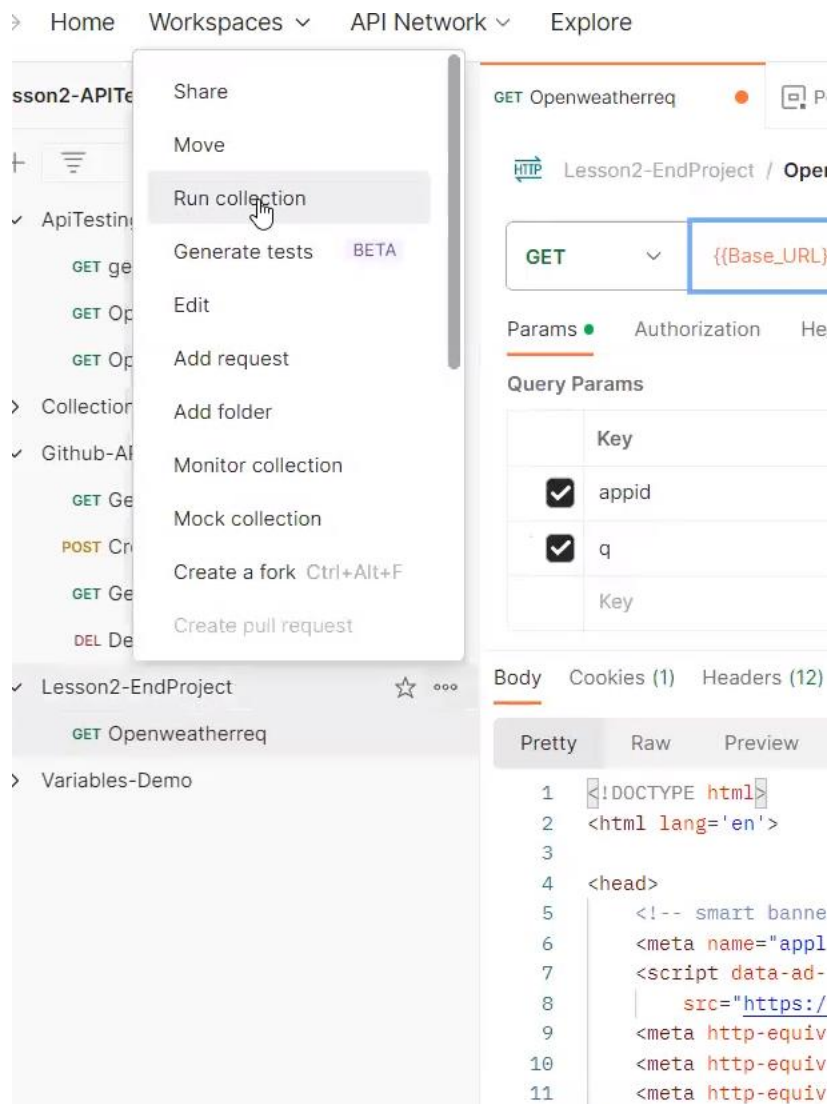
Click on save and send

Then if want to run the collection using CSV

Go desktop and create a excel sheet and save it as CSV(comma delimited)



Click on run collection.



Click on select file and select CSV excel sheet

Deselect All | Select All | Reset

Functional | Performance

Choose how to run your collection

- ☒ Run manually
Run this collection in the Collection Runner.
- ☐ Schedule runs
Periodically run collection at a specified time on the Postman Cloud.
- ☐ Automate runs via CLI
Configure CLI command to run on your build pipeline.

Run configuration

Iterations

Delay

 ms

Data

 collectiondata.csv ×

Data File Type

☐ Persist responses for a session ⓘ

> Advanced settings

Run Lesson2-EndProject

Then click on run

The screenshot shows the Postman interface with the 'Lesson2-EndProject' collection selected. The 'Run' button is highlighted in orange. Below the 'Run' button, the 'Run results' section is visible, showing a table of test results for 5 iterations. The table has columns for Source, Environment, Iterations, Duration, All tests, and Avg. Resp. Time. The results show that all tests passed, with a duration of 1s 862ms and an average response time of 204 ms.

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	Project-variables	5	1s 862ms	0	204 ms

Below the table, the 'All Tests' section shows 'Passed (0)', 'Failed (0)', and 'Skipped (0)'. The 'View Summary' link is also visible.

The 'Iteration 1' section shows a 'GET Openweatherreq' test with a status of '200 OK' and a response time of '574 ms'. The 'No tests found' message is displayed below the test result.

The 'Iteration 2' section shows a 'GET Openweatherreq' test with a status of '200 OK' and a response time of '110 ms'. The 'No tests found' message is displayed below the test result.

The 'Iteration 3' section shows a 'GET Openweatherreq' test with a status of '200 OK' and a response time of '108 ms'. The 'No tests found' message is displayed below the test result.

The 'Iteration 4' section shows a 'GET Openweatherreq' test with a status of '200 OK' and a response time of '114 ms'. The 'No tests found' message is displayed below the test result.

The 'Iteration 5' section shows a 'GET Openweatherreq' test with a status of '200 OK' and a response time of '787 B'. The 'No tests found' message is displayed below the test result.

