Q1#In the context of web development, GET and POST are HTTP methods used to send data from a client (usually a web browser) to a server.

#The GET method is used to retrieve data from the server. When a client sends a GET request,

#the server returns the requested data in the response. The data can be included in the URL as query parameters or in the request headers.

#The POST method, on the other hand, is used to submit data to the server for processing.

#When a client sends a POST request, the data is included in the body of the request rather than in the URL.

#This makes it suitable for sending sensitive information or large amounts of data.

Q2#In Flask, the request object is a global object that represents the current HTTP request.

#It allows you to access the data submitted by a client in a request.

#You can use request to access form data, query parameters, JSON data, and more.

#For example, if you want to retrieve the value of a form field submitted via a POST request, you can use request.form['field_name'].

Q3#The redirect() function in Flask is used to redirect the client to a different URL.

#It is commonly used after processing a form submission or completing a certain action to redirect the user to a different page.

#When you call redirect('url'), Flask sends a response to the client with the appropriate redirect status code (usually 302) and the new URL.

#The client's web browser then makes a new request to the provided URL, and the server responds accordingly.

Q4#Templates in Flask are files containing HTML or other markup languages mixed with placeholders and control structures.

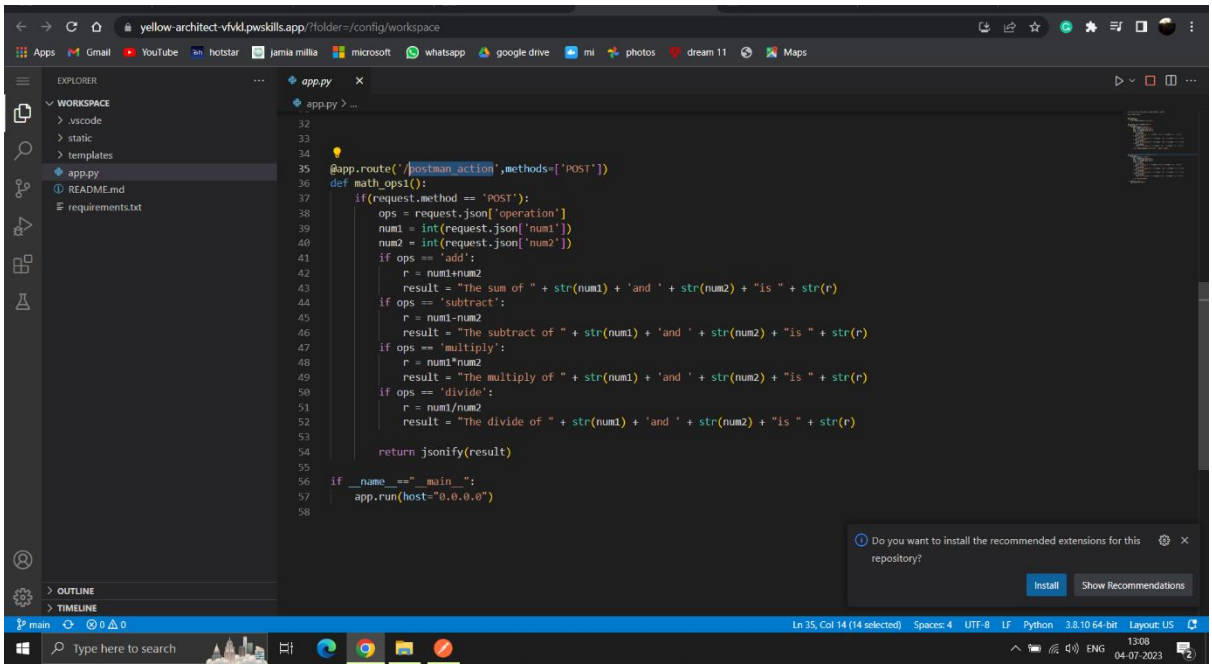
#They allow you to separate the presentation logic from the application logic in a web application.

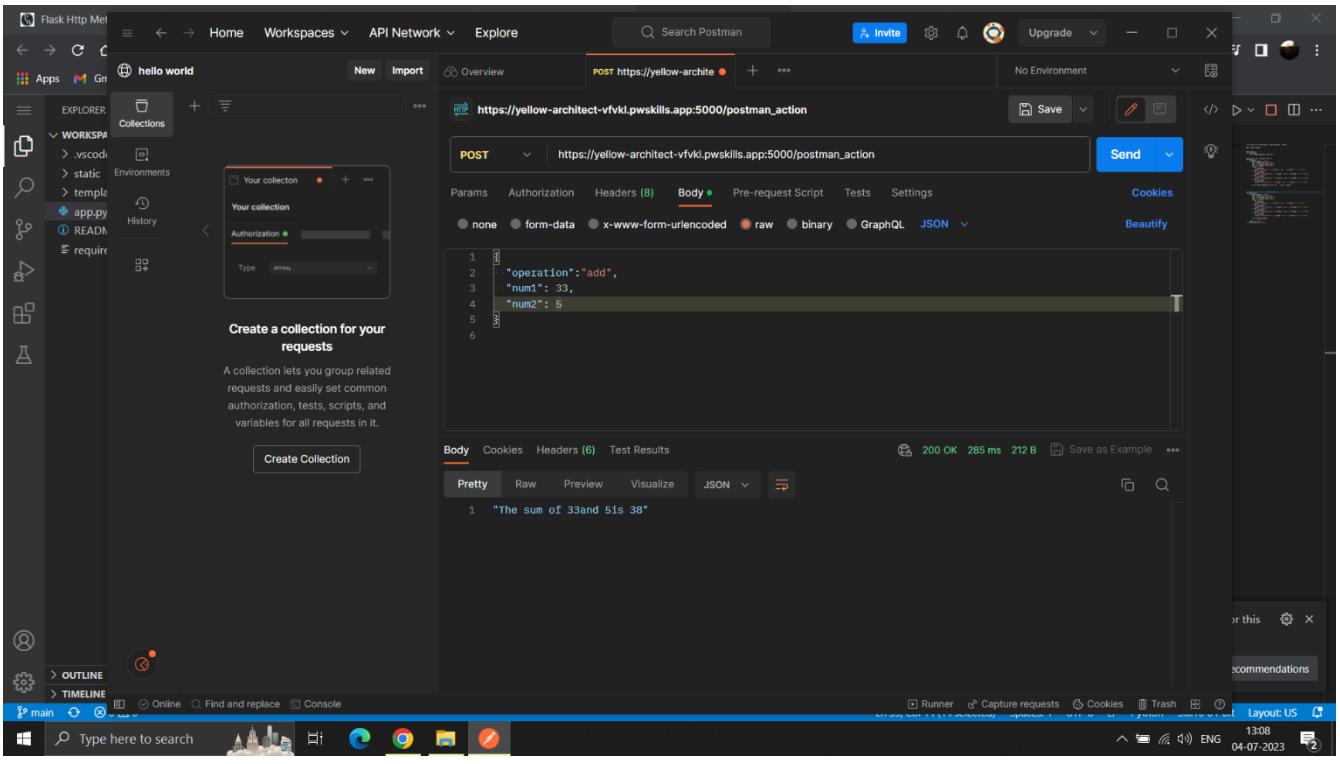
#The render_template() function in Flask is used to render a template and generate the final HTML that will be sent as a response to the client.

#It takes the name of the template file as a parameter and can also accept additional arguments to pass data to the template.

#By using templates, you can create dynamic web pages that can display different content based on variables, conditions, and loops.The render_template() function combines the template file with the provided data to produce the final HTML output.

Q5


yellow-architect-vfvkl.pwskills.app/?folder=/config/workspace
app.py
@app.route('/postman_action',methods=['POST'])
def math_ops1():
if(request.method == 'POST'):
ops = request.json['operation']
num1 = int(request.json['num1'])
num2 = int(request.json['num2'])
if ops == 'add':
r = num1+num2
result = "The sum of " + str(num1) + 'and ' + str(num2) + "is " + str(r)
if ops == 'subtract':
r = num1-num2
result = "The subtract of " + str(num1) + 'and ' + str(num2) + "is " + str(r)
if ops == 'multiply':
r = num1*num2
result = "The multiply of " + str(num1) + 'and ' + str(num2) + "is " + str(r)
if ops == 'divide':
r = num1/num2
result = "The divide of " + str(num1) + 'and ' + str(num2) + "is " + str(r)
return jsonify(result)
if __name__=="__main__":
app.run(host="0.0.0.0")
Do you want to install the recommended extensions for this repository?
Install
Show Recommendations


Home
Workspaces
API Network
Explore
Search Postman
Invite
Upgrade
hello world
New
Import
https://yellow-architect-vfvkl.pwskills.app:5000/postman_action
Save
POST
https://yellow-architect-vfvkl.pwskills.app:5000/postman_action
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"operation":"add",
"num1": 33,
"num2": 5
}
Create a collection for your requests
A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it.
Create Collection
Body
Cookies
Headers (6)
Test Results
200 OK
285 ms
212 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
"The sum of 33and 5is 38"