Flask-2

June 19, 2023

0.1 Q1. Explain GET and POST methods.

0.1.1 GET:

–> Sending the request itself via url.

–> It is used to retrieve data from a server.

–> When a GET request is sent, the data is appended to the URL as query parameters.

–> GET requests should be used for read-only operations where the data is retrieved from the server.

0.1.2 POST:

–> Sending the request itself via body.

–> The POST method is used to send data to a server to create or update a resource.

–> When a POST request is sent, the data is included in the body of the request rather than the URL.

–> POST is not idempotent and can have side effects on the server.

0.2 Q2. Why is request used in Flask?

The request object in Flask is used to access incoming request data in a Flask application. It provides access and handle various components of an HTTP request, allowing us to process incoming data, make decisions, and generate appropriate responses based on the client's request. It provides a convenient way to interact with the incoming data and extract relevant information to build dynamic and responsive web applications. The request object allows us to retrieve data from the request headers, form data, query parameters, cookies, and more.

0.3 Q3. Why is redirect() used in Flask?

Redirect() function in Flask is used to perform URL redirection, allowing us to navigate the client's browser to a different URL or route within our application. It

is useful for route redirection, handling form submissions, preventing duplicate submissions, managing authentication and authorization, and maintaining proper URL structure.

0.4 Q4. What are templates in Flask? Why is the render_template() function used?

In Flask, templates are files containing HTML code mixed with placeholders and control structures. Templates allow us to separate the presentation logic from the application logic, making it easier to maintain and modify the user interface of our web application.

The render_template() function is used to render a template file and generate an HTML response that can be sent back to the client's browser. It takes the name of the template file as an argument and can also accept additional keyword arguments to pass data to the template.

0.5 Q5. Create a simple API. Use Postman to test it. Attach the screenshot of the output in the Jupyter Notebook.

[3]:
```python
from flask import Flask, jsonify, request

app = Flask(__name__)

# Define a simple API route
@app.route('/api/hello', methods=['GET'])
def hello():
    return jsonify({'message': 'Hello, World!'})

# Define an API route that accepts POST requests
@app.route('/api/greet', methods=['POST'])
def greet():
    name = request.json.get('name')
    if name:
        return jsonify({'message': f'Hello, {name}!'})
    else:
        return jsonify({'message': 'Name parameter is missing.'}), 400

if __name__ == '__main__':
    app.run(host="0.0.0.0")
    app.run(debug=True)
```

```
 * Serving Flask app '__main__'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
```

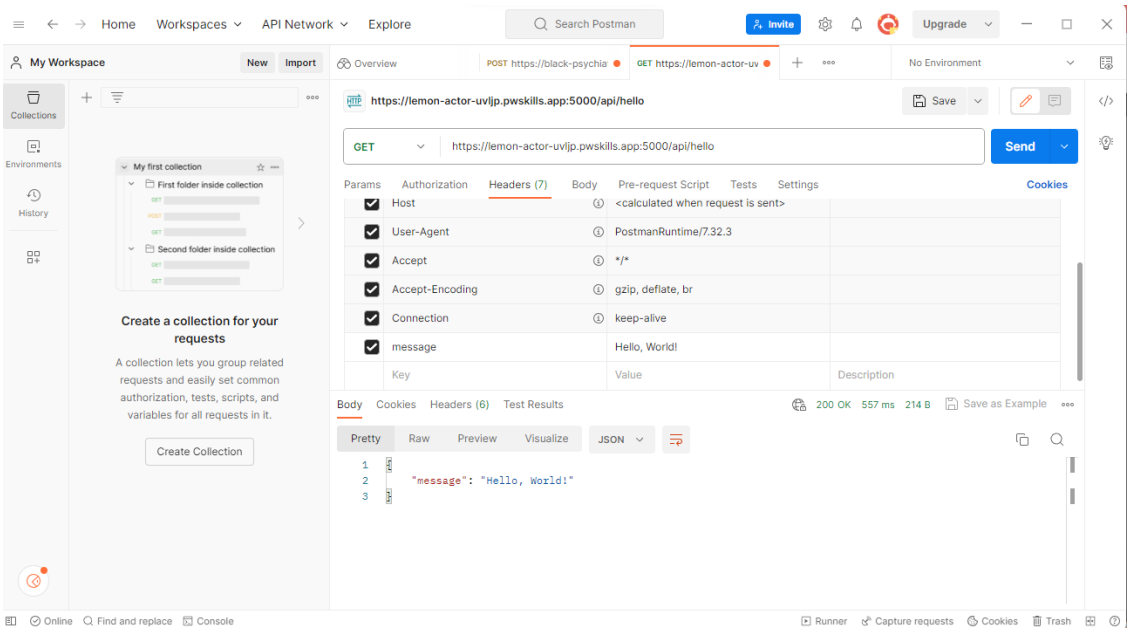
```
 * Running on http://127.0.0.1:5000
 * Running on http://172.18.0.9:5000
Press CTRL+C to quit
172.18.0.2 - - [19/Jun/2023 08:59:44] "GET /api/hello HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:00:16] "GET /api/hello HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:08:35] "GET /api/greet HTTP/1.1" 405 -
172.18.0.2 - - [19/Jun/2023 09:09:37] "GET /api/greet HTTP/1.1" 405 -
172.18.0.2 - - [19/Jun/2023 09:10:52] "POST /api/greet HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:11:21] "POST /api/greet HTTP/1.1" 415 -
172.18.0.2 - - [19/Jun/2023 09:12:51] "POST /api/greet HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:14:28] "GET /api/greet HTTP/1.1" 405 -
172.18.0.2 - - [19/Jun/2023 09:15:01] "POST /api/greet HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:15:04] "GET /api/greet HTTP/1.1" 405 -
172.18.0.2 - - [19/Jun/2023 09:15:06] "GET /api/greet HTTP/1.1" 405 -
172.18.0.2 - - [19/Jun/2023 09:15:20] "GET /api/greet HTTP/1.1" 405 -
172.18.0.2 - - [19/Jun/2023 09:15:23] "GET /api/greet HTTP/1.1" 405 -
172.18.0.2 - - [19/Jun/2023 09:16:13] "GET /api/hello HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:16:24] "GET /api/hello HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:17:20] "POST /api/greet HTTP/1.1" 200 -
172.18.0.2 - - [19/Jun/2023 09:17:28] "POST /api/greet HTTP/1.1" 200 -

 * Serving Flask app '__main__'
 * Debug mode: on

WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
Traceback (most recent call last):
  File "/opt/conda/lib/python3.10/runpy.py", line 196, in _run_module_as_main
    return _run_code(code, main_globals, None,
  File "/opt/conda/lib/python3.10/runpy.py", line 86, in _run_code
    exec(code, run_globals)
  File "/opt/conda/lib/python3.10/site-packages/ipykernel_launcher.py", line 17,
in <module>
    app.launch_new_instance()
  File "/opt/conda/lib/python3.10/site-
packages/traitlets/config/application.py", line 991, in launch_instance
    app.initialize(argv)
  File "/opt/conda/lib/python3.10/site-
packages/traitlets/config/application.py", line 113, in inner
    return method(app, *args, **kwargs)
  File "/opt/conda/lib/python3.10/site-packages/ipykernel/kernelapp.py", line
665, in initialize
    self.init_sockets()
  File "/opt/conda/lib/python3.10/site-packages/ipykernel/kernelapp.py", line
309, in init_sockets
    self.shell_port = self._bind_socket(self.shell_socket, self.shell_port)
```

```
  File "/opt/conda/lib/python3.10/site-packages/ipykernel/kernelapp.py", line
246, in _bind_socket
    return self._try_bind_socket(s, port)
  File "/opt/conda/lib/python3.10/site-packages/ipykernel/kernelapp.py", line
222, in _try_bind_socket
    s.bind("tcp://%s:%i" % (self.ip, port))
  File "/opt/conda/lib/python3.10/site-packages/zmq/sugar/socket.py", line 232,
in bind
    super().bind(addr)
  File "zmq/backend/cython/socket.pyx", line 568, in
zmq.backend.cython.socket.Socket.bind
  File "zmq/backend/cython/checkrc.pxd", line 28, in
zmq.backend.cython.checkrc._check_rc
zmq.error.ZMQError: Address already in use
```

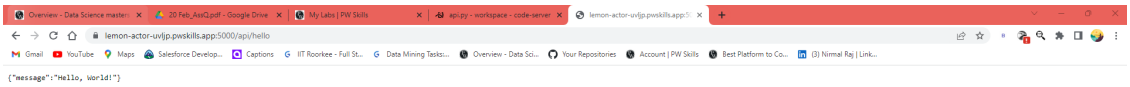
```
An exception has occurred, use %tb to see the full traceback.

SystemExit: 1
```

```
/opt/conda/lib/python3.10/site-packages/IPython/core/interactiveshell.py:3441:
UserWarning: To exit: use 'exit', 'quit', or Ctrl-D.
  warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)
```

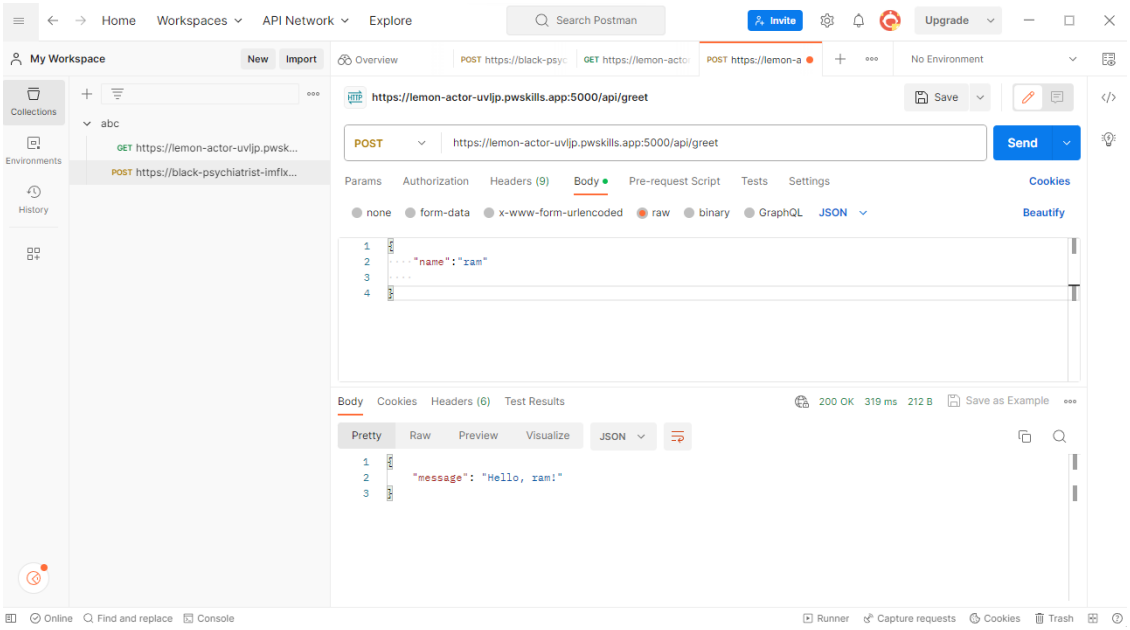
0.5.1 Postman—GET:



0.5.2 Output:



0.5.3 POSTMAN- POST:



[ ]: