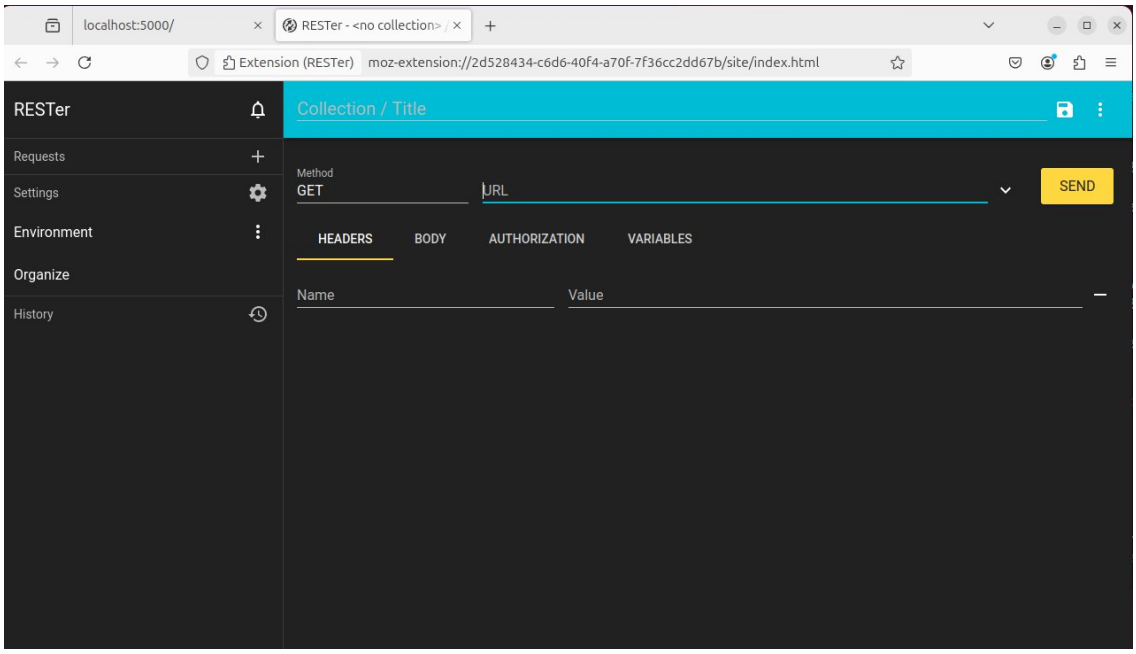
CKCS 145: Lab 3 - Data Submission

---

1. Install the RESTer add-on for Firefox Browser. When you visit the RESTer add-on web site you can click on the “Add to Firefox” button to install it on Firefox.



Open lab3-app.py ~/lab3 Save

```python
from flask import Flask, jsonify, request

app=Flask(__name__)

@app.route('/',methods=['GET','POST'])

def home():
    if(request.method=='GET'):
      data='hello world'
      return jsonify({'data':data})

@app.route('/home/<int:num>',methods=['GET'])

def disp(num):
    return jsonify({'data':num**2})

app.run(debug=True)
```

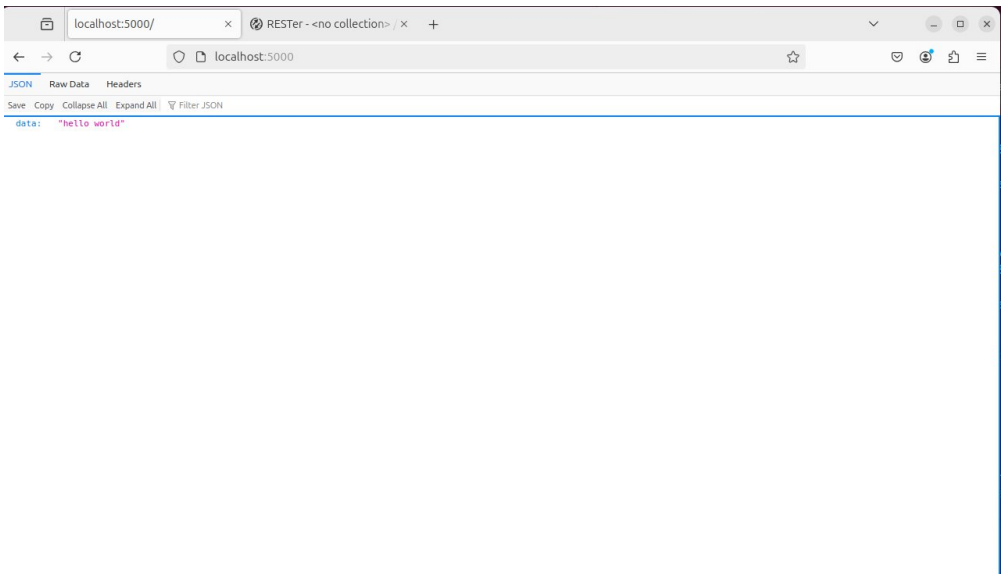
8. The above application is middleware. This means that it is a service that may be accessed on port 5000 using a web browser. The URL that may be used is http://localhost:5000. What do

you see in your web browser when you visit this URL? What kind of request did you send? You can use Web Developer Tools from Firefox to determine the request type.

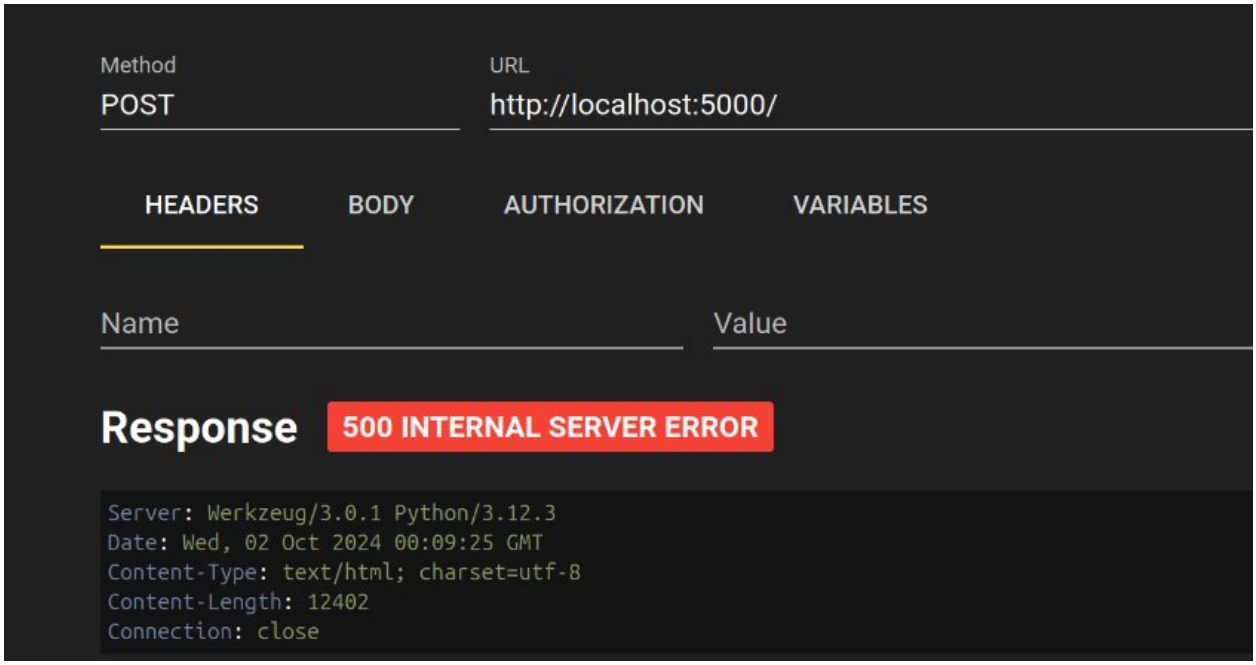
It used ‘GET’ and execute:

if(request.method == 'GET'):

data = 'hello world'



9. Use RESTer to submit a POST request to http://localhost:5000. What do you see in your web browser when you visit this URL? What do you have to modify in the code from step 6 in order to respond to a POST request?



What do you have to modify in the code from step 6 in order to respond to a POST request?

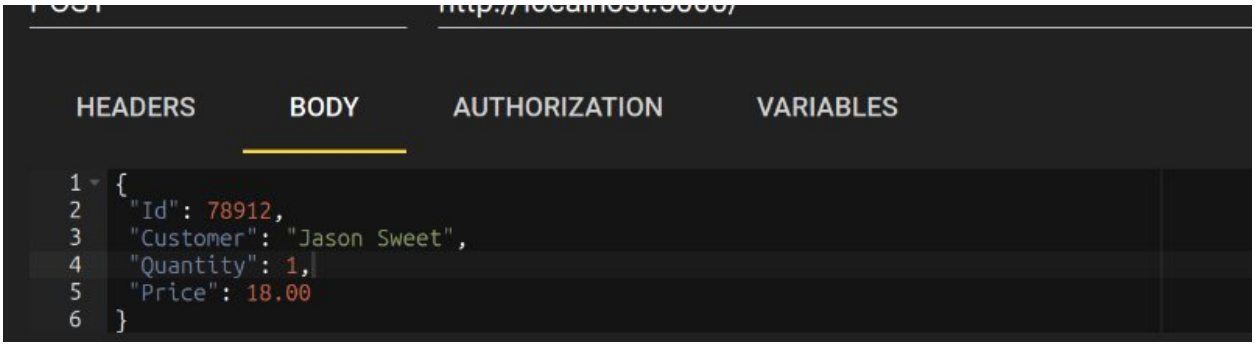
I have to add ‘POST’ request in the “lab3-app.py”

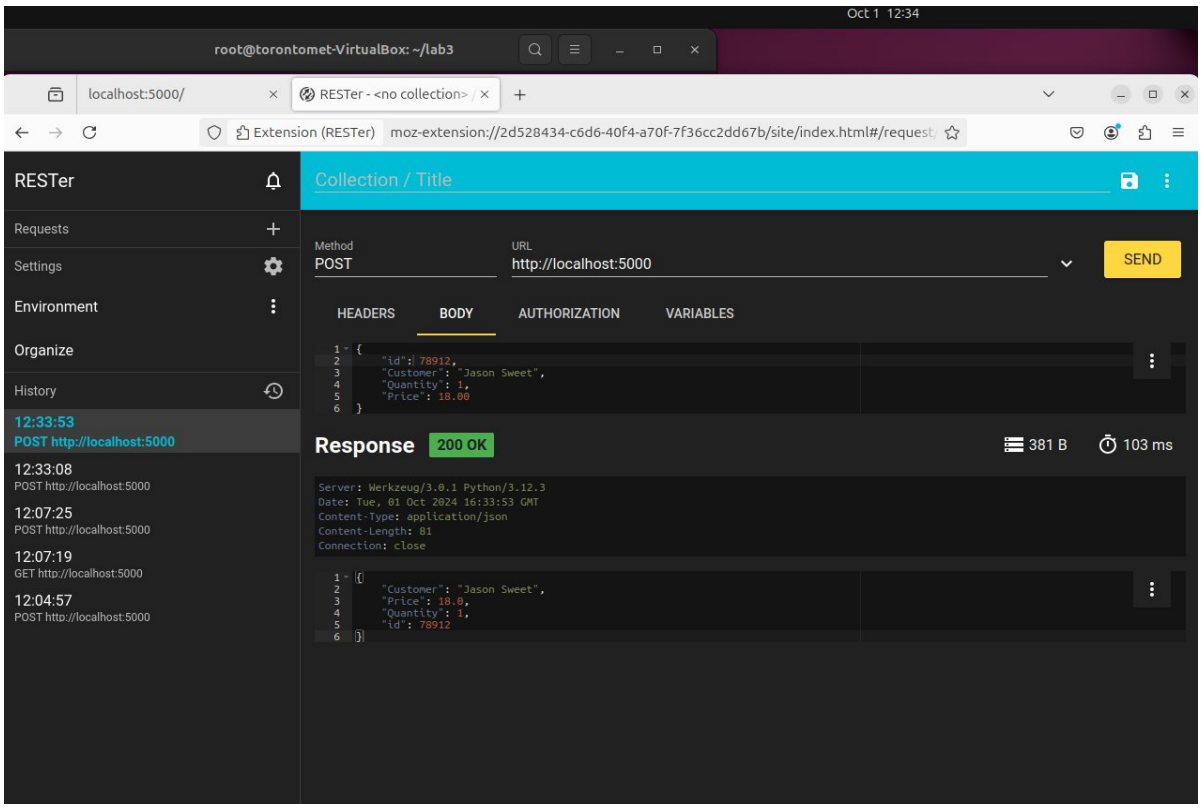
Add the following code to home() method. Now the home() method has the ability to respond to the POST request

```
@app.route('/', methods = ['GET', 'POST'])

@app.route('/home', methods = ['GET'])
def home():
    if(request.method == 'GET'):
        data = 'hello world'
        return jsonify({'data': data})
    if(request.method == 'POST'):
        data = request.get_json()
        return jsonify(data)


@app.route('/home/<int:num>', methods = ['GET'])

def disp(num):
    return jsonify({'data': num**2})
```

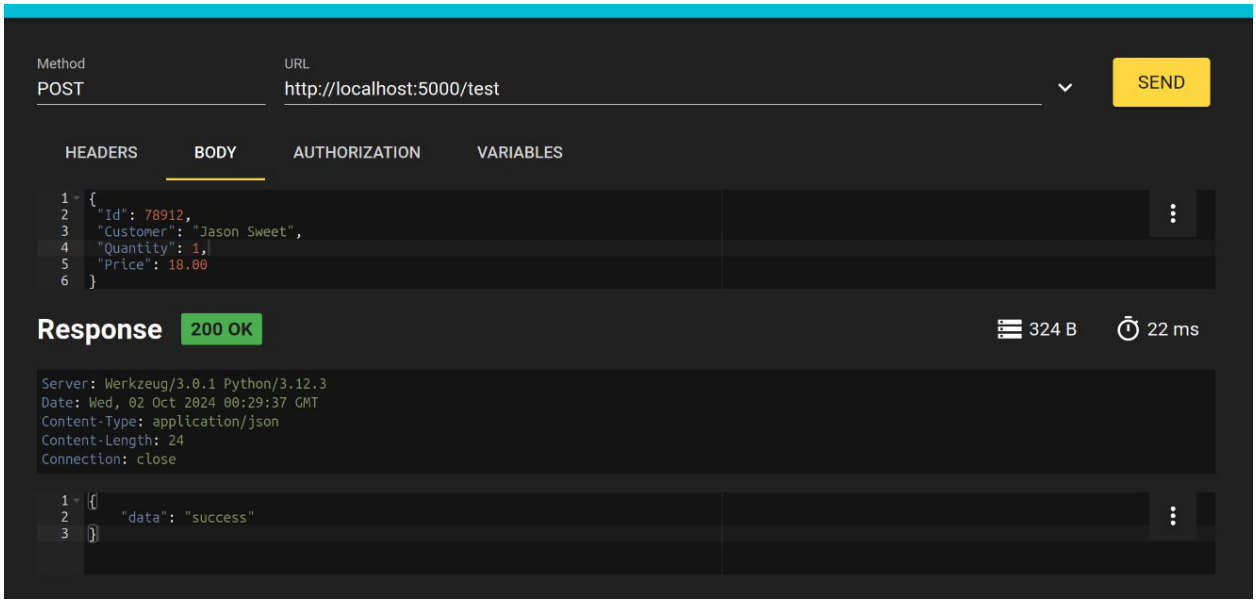
Use RESTer to submit a POST request to http://localhost:5000. Add a HEADER with the name “Content-type” and the value of “application/json”. The BODY should be below.

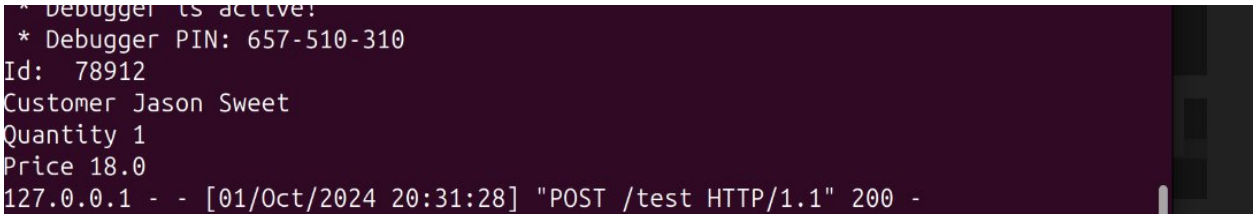
{ "Id": 78912, "Customer": "Jason Sweet", "Quantity": 1, "Price": 18.00 }





Use the RESTer request created in step 14 to submit to http://localhost:5000/test. What do you see on the command prompt of the middleware? What do you see as a response from your Flask application? How are these generated?





On the Command Prompt (where the Flask application is running), you will see the following output:

```
Id: 123

Customer: John Doe

Quantity: 10

Price: 100
```

This output is generated by the `print` statements inside the `post_test_route()` function, which print the values extracted from the JSON request.

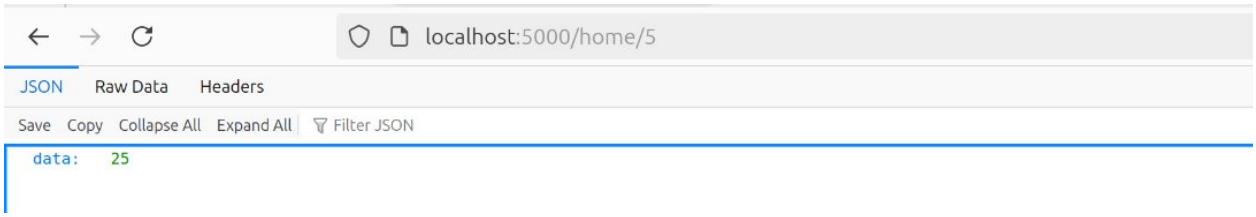
RESTer Response: RESTer will display the following JSON response:

```
{"data": "success"}
```

This confirms that the Flask application received the `POST` request with `application/json` content type, processed it, and returned the correct response.

21. Use a web browser and visit http://localhost:5000/home/5. What do you see? Then visit http://localhost:5090/home/17. What do you see? What method handles this route?

- Visiting http://localhost:5000/home/5 gives the square of 5.

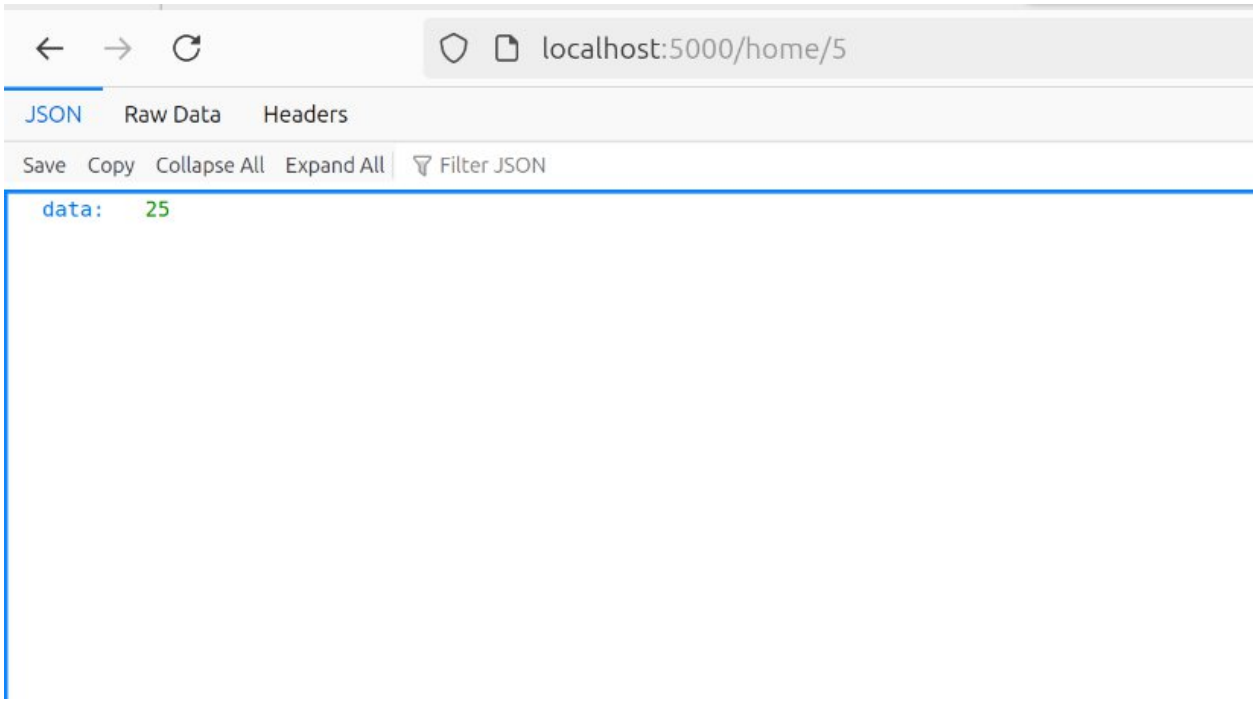


- Visiting http://localhost:5090/home/17 results in a connection error because the app is running on port 5000. However with port 5000 , we are able to get data: 289

22. What happens if you use a web browser and visit http://localhost:5000/home? If this causes an error, how can this be fixed?

- Visiting http://localhost:5000/home without a number causes a 404 error, but you can fix this by adding a default route.

@app.route('/home', methods=['GET'])


localhost:5000/home/5
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
data: 25