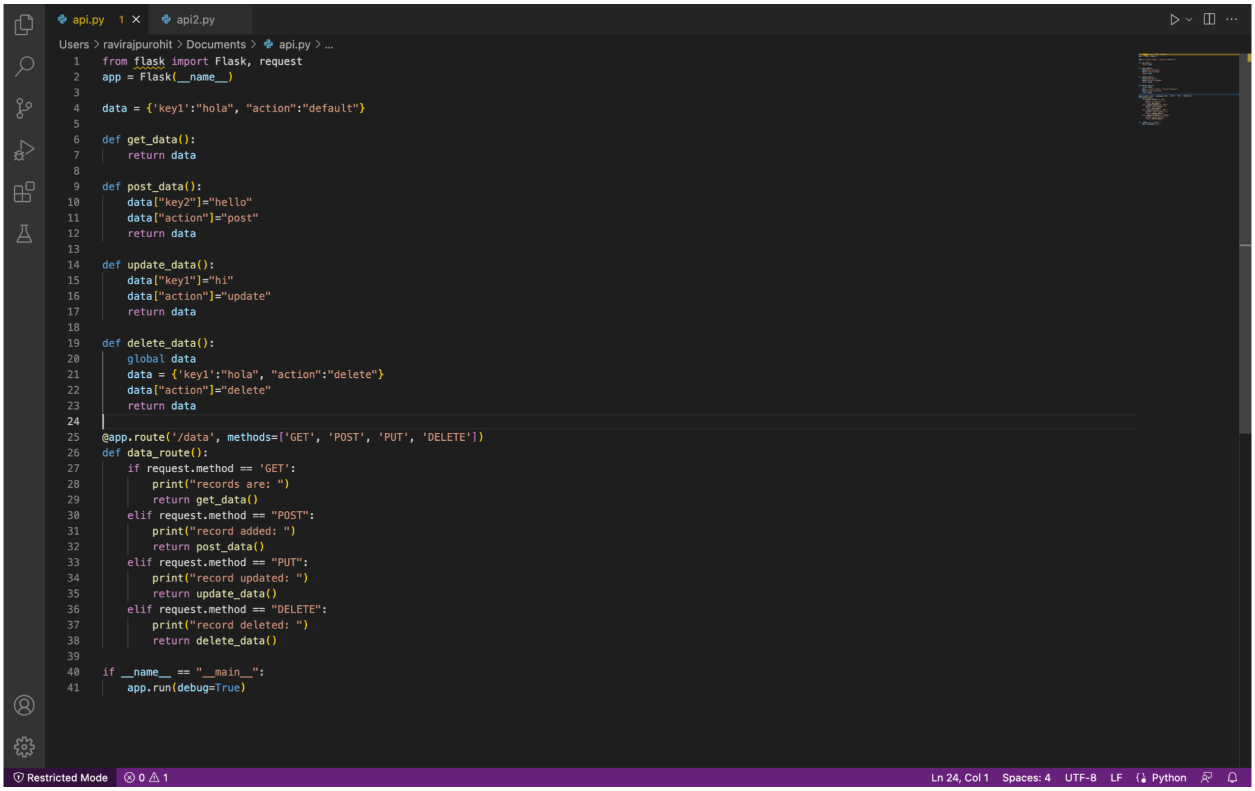
Cloud Computing In Class Lab 2

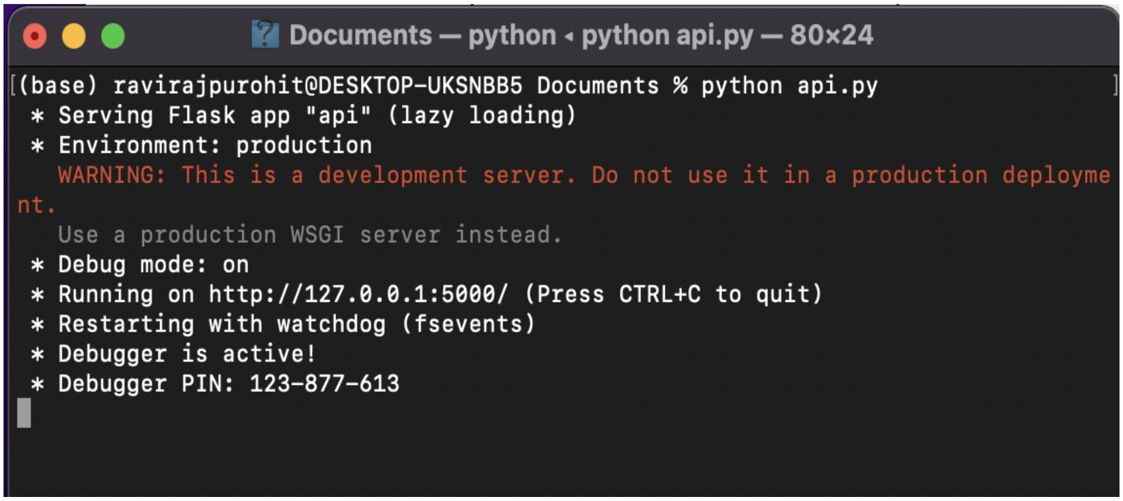
Name - Ravi Rajpurohit UTA ID - 1002079916

Step 1 - Implement a Flask API with GET, POST, PUT (UPDATE) and DELETE methods
Using Python, defined a simple flask app to store some data in a local dictionary


api.py 1
api2.py
Users > ravirajpurohit > Documents > api.py > ...
Restricted Mode
0 1
Ln 24, Col 1 Spaces: 4 UTF-8 LF Python

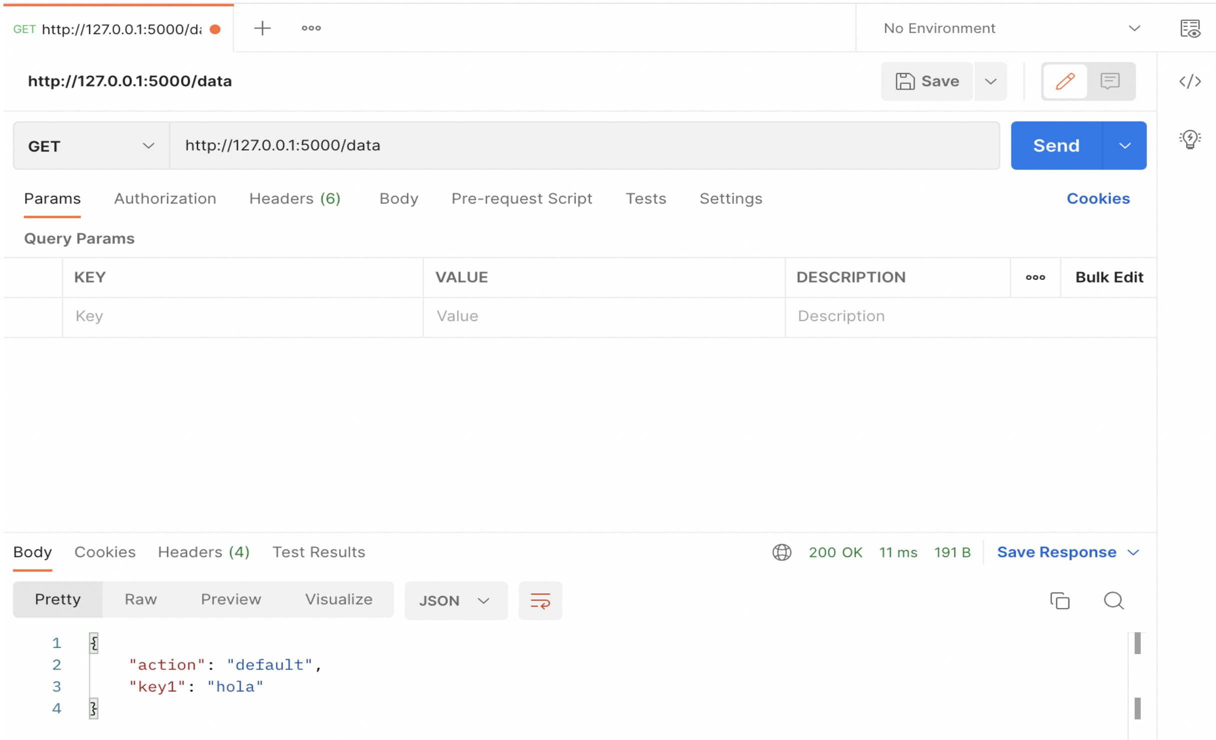
```python
from flask import Flask, request
app = Flask(__name__)

data = {'key1':"hola", "action":"default"}

def get_data():
    return data

def post_data():
    data["key2"]="hello"
    data["action"]="post"
    return data

def update_data():
    data["key1"]="hi"
    data["action"]="update"
    return data

def delete_data():
    global data
    data = {'key1':"hola", "action":"delete"}
    data["action"]="delete"
    return data

@app.route('/data', methods=['GET', 'POST', 'PUT', 'DELETE'])
def data_route():
    if request.method == 'GET':
        print("records are: ")
        return get_data()
    elif request.method == "POST":
        print("record added: ")
        return post_data()
    elif request.method == "PUT":
        print("record updated: ")
        return update_data()
    elif request.method == "DELETE":
        print("record deleted: ")
        return delete_data()

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

Step 2 - Run the flask application



```
[(base) ravirajpurohit@DESKTOP-UKSNBB5 Documents % python api.py                ]
 * Serving Flask app "api" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with watchdog (fsevents)
 * Debugger is active!
 * Debugger PIN: 123-877-613
```

Step 3 - Test the GET method



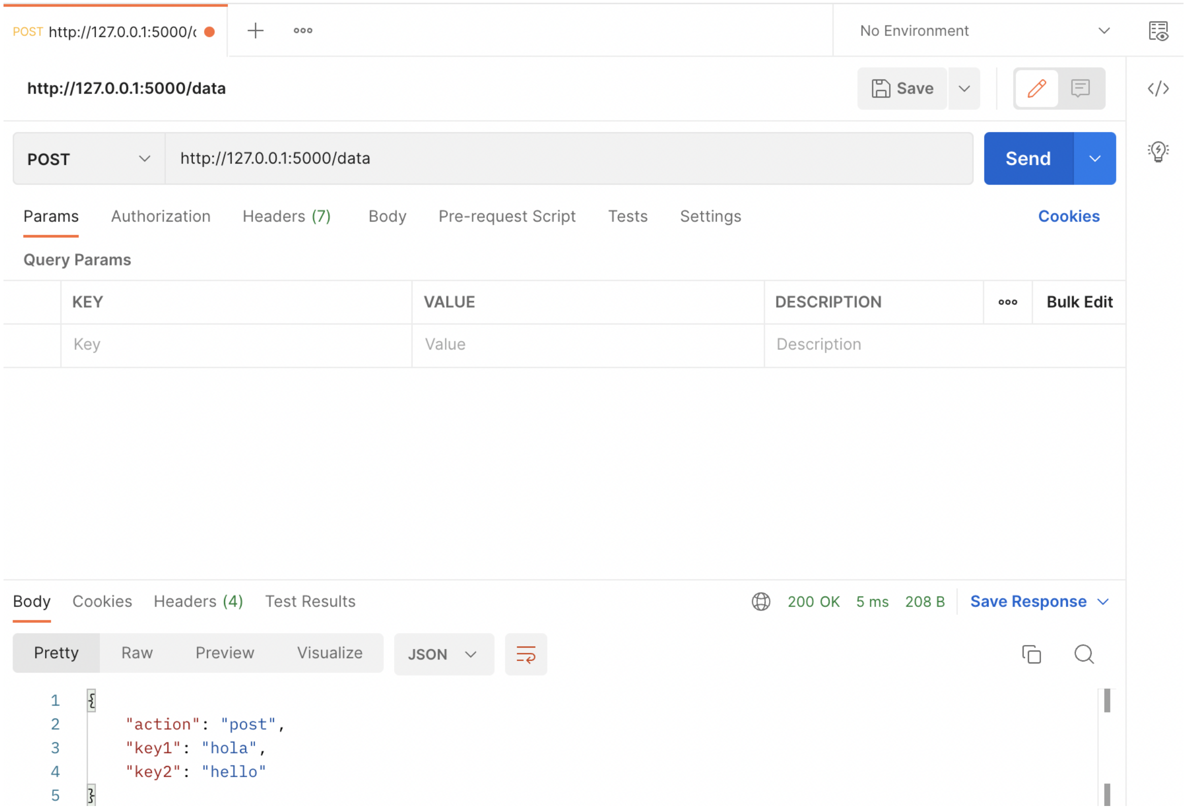
http://127.0.0.1:5000/data

GET http://127.0.0.1:5000/data

200 OK 11 ms 191 B

```
{
    "action": "default",
    "key1": "hola"
}
```

Step 4 - Test the POST method



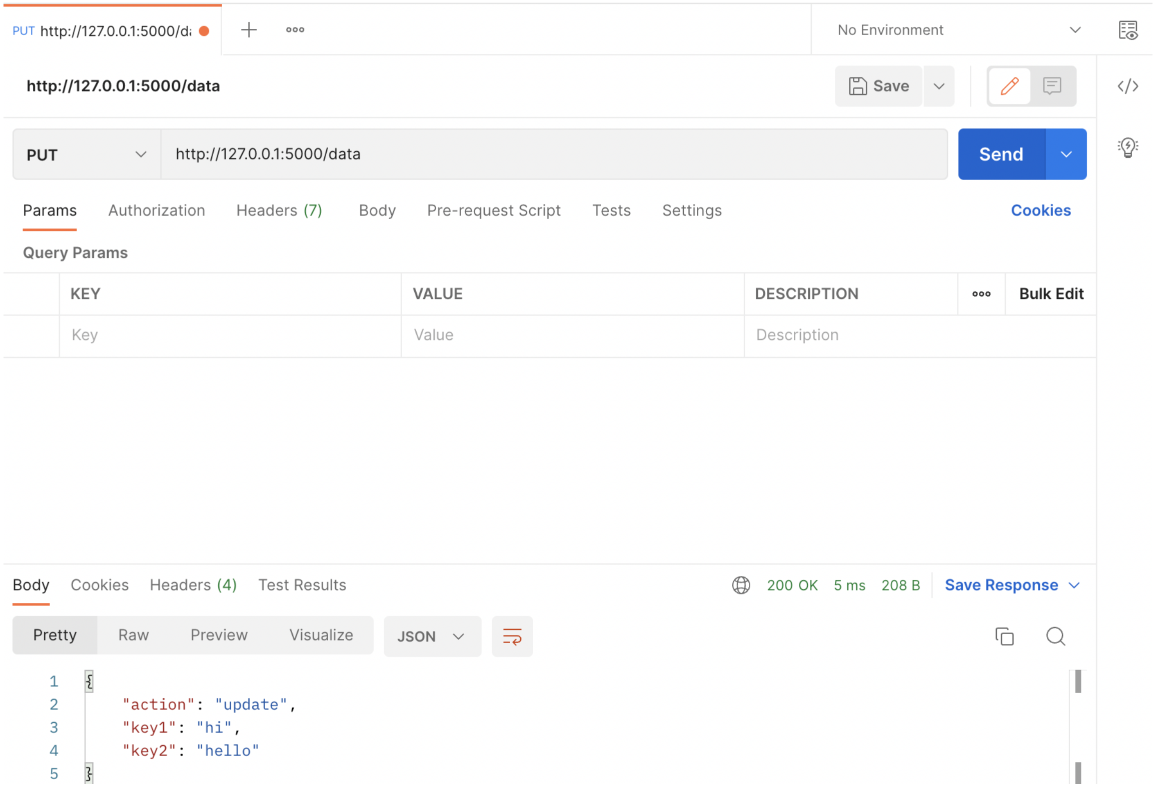
http://127.0.0.1:5000/data

POST http://127.0.0.1:5000/data

200 OK 5 ms 208 B

```
{
    "action": "post",
    "key1": "hola",
    "key2": "hello"
}
```

Step 5 - Test the PUT/UPDATE method



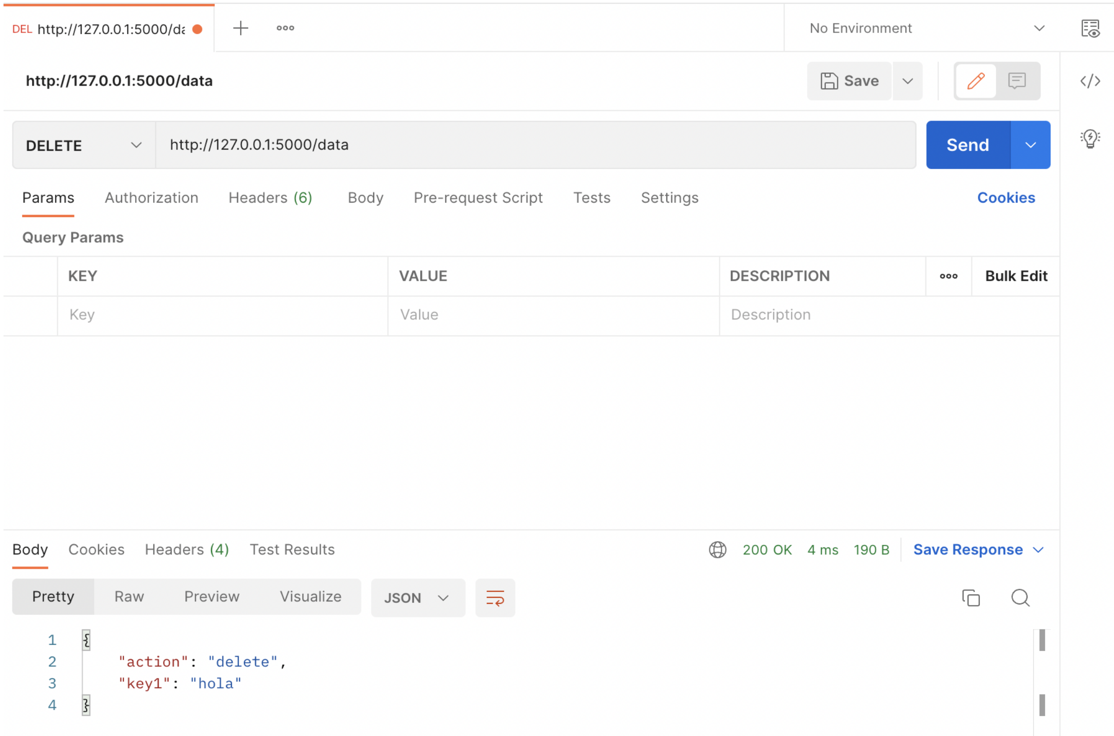
http://127.0.0.1:5000/data

PUT http://127.0.0.1:5000/data

200 OK 5 ms 208 B

```
{
    "action": "update",
    "key1": "hi",
    "key2": "hello"
}
```

Step 6 - Test the DELETE method



http://127.0.0.1:5000/data

DELETE http://127.0.0.1:5000/data

200 OK 4 ms 190 B

```
{
    "action": "delete",
    "key1": "hola"
}
```