

**Assignment -1**  
Flask Programming

Assignment Date	14 September 2022
Student Name	Mr. M.Krishnaraj
Student Roll Number	814619104011
Maximum Marks	2 Marks

**Question-1:**

Write a flask program which should display Name, Email, Phone and it should display the same details once we hit submit.

**Solution:**

```
from flask import Flask, redirect,url_for, request,render_template,json
import os

app = Flask(__name__)

team_member2 = {"1" : "KRISHNARAJ M","2" : "krishnarajgunal@gmail.com", "3" :
"+912344378911"}

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

def api():

if request.method == 'GET':

return team_member2

if request.method == 'POST':

data = request.json

team_member2.update(data)

return "Data is inserted"

@app.route("/data/<id>", methods=["PUT"])

def update(id):

data = request.form['member']

team_member2 [str(id)]=data

return "Data is updated"

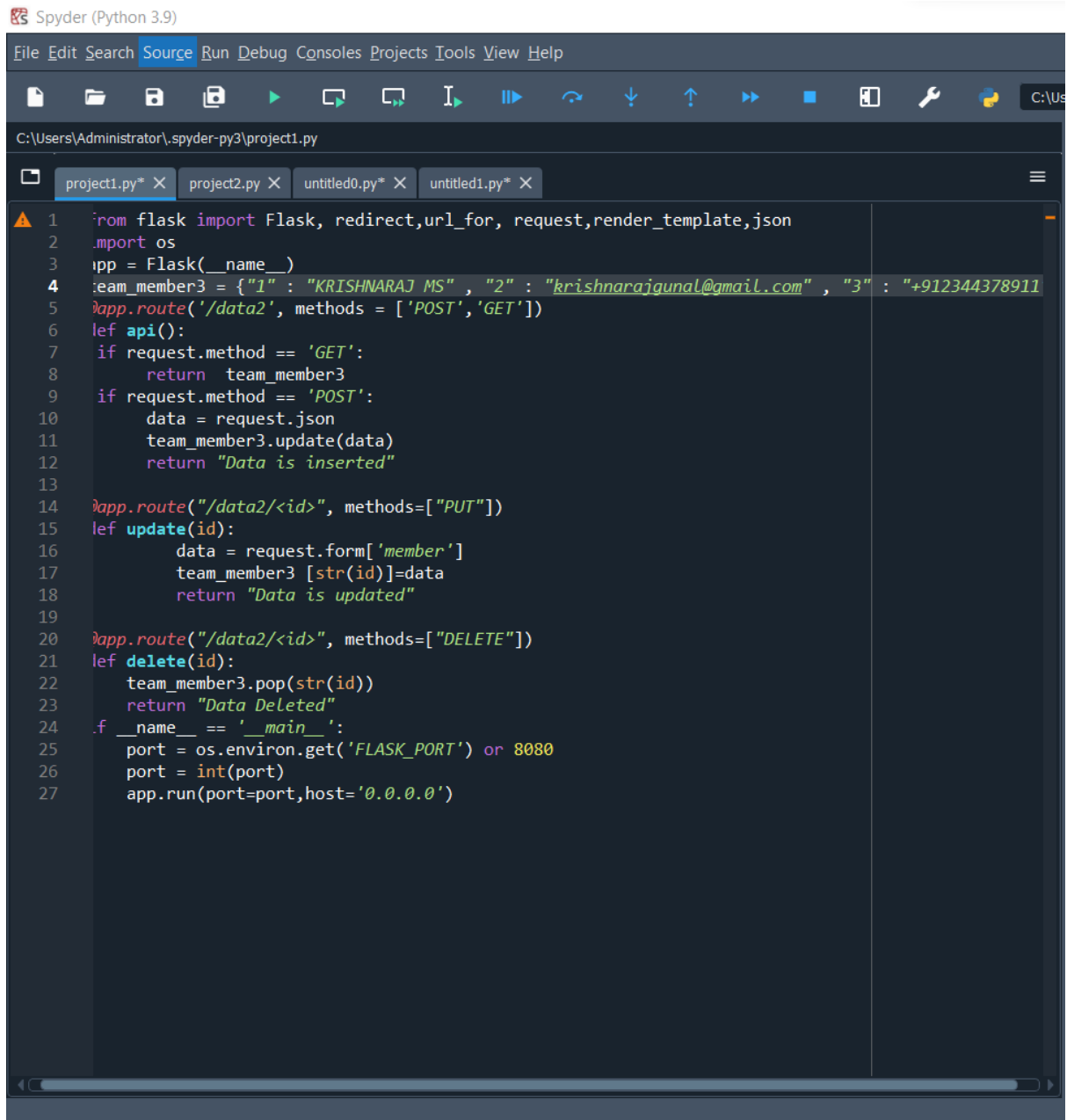
@app.route("/data/<id>", methods=["DELETE"])

def delete(id):

team_member2.pop(str(id))

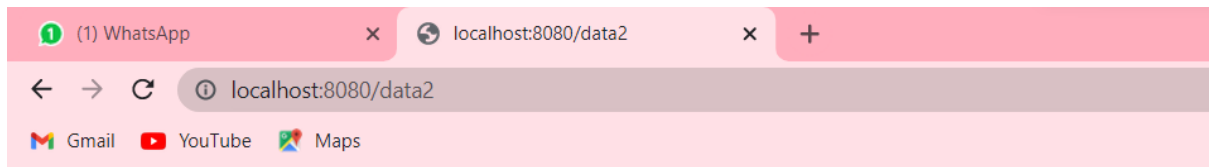
return "Data Deleted"
```

```
if __name__ == '__main__':  
    port = os.environ.get('FLASK_PORT') or 8080  
    port = int(port)  
    app.run(port=port,host='0.0.0.0')
```



The screenshot shows the Spyder Python IDE interface. The top menu bar includes File, Edit, Search, Source, Run, Debug, Consoles, Projects, Tools, View, and Help. Below the menu is a toolbar with icons for file operations and execution. The main editor window displays the code for project1.py, which is a Flask application. The code defines a Flask app, sets up a database (team\_member3), and implements routes for GET, POST, PUT, and DELETE methods. The application is run on port 8080 by default.

```
1 from flask import Flask, redirect, url_for, request, render_template, json  
2 import os  
3 app = Flask(__name__)  
4 team_member3 = {"1": "KRISHNARAJ MS", "2": "krishnarajgunal@gmail.com", "3": "+912344378911"  
5 app.route('/data2', methods = ['POST', 'GET'])  
6 def api():  
7     if request.method == 'GET':  
8         return team_member3  
9     if request.method == 'POST':  
10         data = request.json  
11         team_member3.update(data)  
12         return "Data is inserted"  
13  
14 app.route("/data2/<id>", methods=["PUT"])  
15 def update(id):  
16     data = request.form['member']  
17     team_member3 [str(id)]=data  
18     return "Data is updated"  
19  
20 app.route("/data2/<id>", methods=["DELETE"])  
21 def delete(id):  
22     team_member3.pop(str(id))  
23     return "Data Deleted"  
24 if __name__ == '__main__':  
25     port = os.environ.get('FLASK_PORT') or 8080  
26     port = int(port)  
27     app.run(port=port,host='0.0.0.0')
```



```
{"1": "KRISHNARAJ M", "2": "krishnarajgunal@gmail.com", "3": "+912344378911"}
```

## Question-2:

Write a flask program which should cover cookies and session.

**Solution:**

### Create cookie

```
@app.route('/')
def index():
    return render_template('index.html')
```

This HTML page contains a text input.

```
<html>
<body>

    <form action = "/setcookie" method = "POST">
        <p><h3>Enter userID</h3></p>
        <p><input type = 'text' name = 'nm' /></p>
        <p><input type = 'submit' value = 'Login' /></p>
    </form>

</body>
</html>
```

### Set cookie

```
@app.route('/setcookie', methods = ['POST', 'GET'])
def setcookie():
    if request.method == 'POST':
```

```
User = request.form['nm']
```

```
Resp = make_response(render_template('readcookie.html'))
```

```
Resp.set_cookie('userID', user)
```

```
Return resp
```

Get cookie

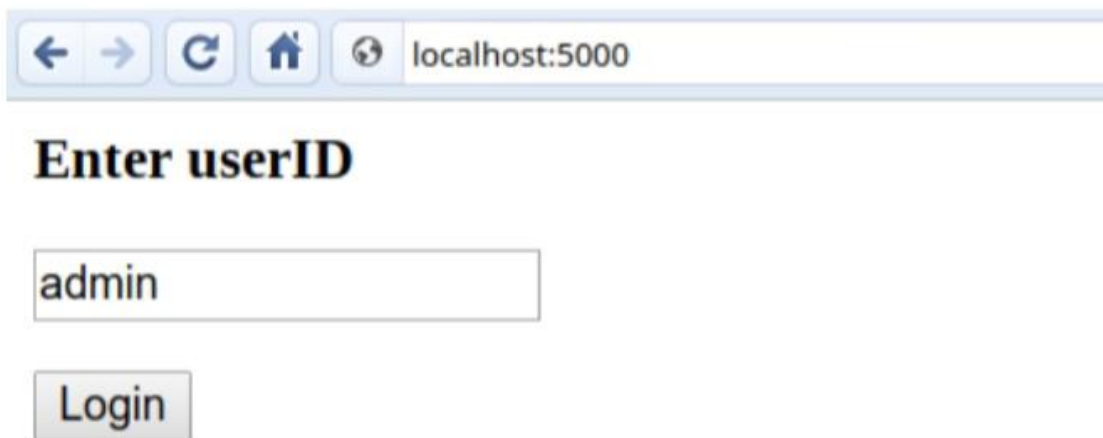
```
@app.route('/getcookie')
```

```
Def getcookie():
```

```
    Name = request.cookies.get('userID')
```

```
    Return '<h1>welcome ' + name + '</h1>'
```

### Output



### SESSION

```
From flask import Flask, render_template_string, request, session, redirect, url_for
```

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

```
Def set_email():
```

```
    If request # Create the Flask application
```

```
App = Flask(__name__)
```

```
App.secret_key = 'BAD_SECRET_KEY'
```

```
..method == 'POST':
```

```
    # Save the form data to the session object
```

```
    Session['email'] = request.form['email_address']
```

```
    Return redirect(url_for('get_email'))
```

```
Return ""
```

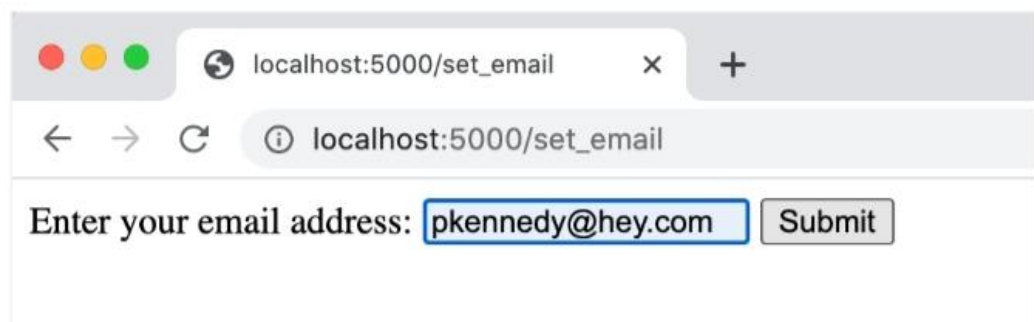
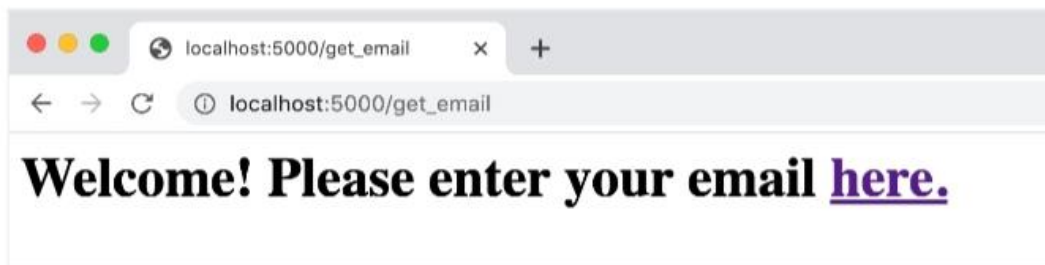
```

<form method="post">
  <label for="email">Enter your email address:</label>
  <input type="email" id="email" name="email_address" required />
  <button type="submit">Submit</button>
</form>
"""

@app.route('/get_email')
def get_email():
    Return render_template_string("""
        {% if session['email'] %}
            <h1>Welcome {{ session['email'] }}!</h1>
        {% else %}
            <h1>Welcome! Please enter your email <a href="{{ url_for('set_email') }}">here.</a></h1>
        {% endif %}
    """)

@app.route('/delete_email')
def delete_email():
    # Clear the email stored in the session objects
    Session.pop('email', default=None)
    Return '<h1>Session deleted!</h1>'
if __name__ == '__main__':
    App.run()

```



### Question-3:

Write a flask program which should display resume details and also have upload resume option by using file uploading.

### Solution:

Upload file html program:

```
<html>
<body>
  <form action = "http://localhost:5000/uploader" method = "POST"
    enctype = "multipart/form-data">
    <input type = "file" name = "file" />
    <input type = "submit"/>
  </form>
</body>
</html>
```

### Flask program:

```
from flask import Flask, render_template, request
from werkzeug import secure_filename
app = Flask(__name__)

@app.route('/upload')
def upload_file():
    return render_template('upload.html')

@app.route('/uploader', methods = ['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        f = request.files['file']
        f.save(secure_filename(f.filename))
        return 'file uploaded successfully'

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

Spyder (Python 3.9)

File Edit Search Source Run Debug Consoles Projects Tools View Help

C:\Users\Administrator\spyder-py3\project2.py

project1.py X project2.py X

```
1 from flask import Flask, render_template, request
2 from werkzeug import secure_filename
3 app = Flask(__name__)
4
5 @app.route('/upload')
6 def upload_file():
7     return render_template('upload.html')
8
9 @app.route('/uploader', methods = ['GET', 'POST'])
10 def upload_file():
11     if request.method == 'POST':
12         f = request.files['file']
13         f.save(secure_filename(f.filename))
14         return 'file uploaded successfully'
15
16 if __name__ == '__main__':
17     app.run(debug = True)
```

