St. Francis Institute of Technology Borivali (West), Mumbai-400103

(Autonomous Institute)
Department of Information Technology

Academic Year: 2024-25

Class: TE-ITA/B Semester: VI

Subject: Web Lab

Experiment –9: To Design a Weather App using Flask.

1. **Aim:** To design an app using Flask Framework.
2. **Objectives:** Aim of this experiment is that, the students will be able
   - To install Flask Framework
   - To understand Basics of Flask.
   - To understand Flask Application
3. **Outcomes:** After study of this experiment, the students will be able
   - To build applications.
   - To build URL
   - To understand HTTP methods.
4. **Prerequisite:** Basic understanding of HTML and Python etc
5. **Requirements:** Personal Computer, Windows operating system, VSCode, Python 2.6 or higher, browser, Internet Connection, google doc, latest version of Python.
6. **Pre-Experiment Exercise:**
   Brief Theory: Refer shared material
7. **Laboratory Exercise**
   A. **Procedure:**
      Install Python 3 on local machine
      Set up a programming environment via the command line
      Installactivate Python environment
      Install Flask using the pip package installer
      a. **Answer the following:**
         - Flask Variables and rules?
         - Flask session?
      b. **Attach screenshots:**
         - Flask SS

8. **Post-Experiments Exercise**
   A. **Extended Theory:**
      Nil
   B. **Questions:**
      - Flask applications?
   C. **Conclusion:**
      - Write what was performed in the experiment.
      - Write the significance of the topic studied in the experiment.
   D. **References:**

1. Flask Web Development, by Miguel Grinberg

Name: Vishal Rajesh Mahajan Web Lab EXP09

Class: TE IT A Roll No: 56

Laboratory Exercise

A. Answer the following:

Q1] Flask Variables and Rules

Answer: In Flask, URL rules define how URLs are mapped to view functions using the @app.route() decorator.Variable Rules allow parts of the URL to be dynamic.

Syntax:

```
@app.route('/user/<username>')
def profile(username):
    return f'User: {username}'
```

Data Types in Variable Rules:

- string (default): Accepts any text without a slash.
- int: Accepts only integers.
- float: Accepts floating-point numbers.
- path: Accepts text including slashes.
- uuid: Accepts UUID strings.

Q2] Flask Session

Answer: Flask session stores data across requests for a user. It is implemented on the client-side using secure cookies.

To use sessions:

- A secret_key must be set in the Flask app to sign the cookies.

```
from flask import Flask, session

app = Flask(__name__)
app.secret_key = 'secret'

@app.route('/login/<user>')
def login(user):
    session['username'] = user
    return 'Logged in'
```

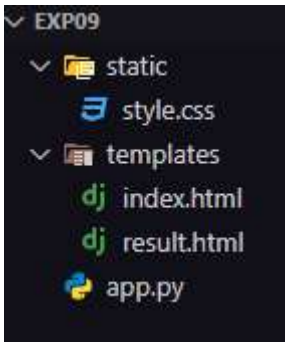
```
@app.route('/logout')
def logout():
    session.pop('username', None)
    return 'Logged out'

@app.route('/profile')
def profile():
    user = session.get('username')
    return f'Logged in as {user}' if user else 'Not logged in'
```

Session Methods:

- session['key'] = value – sets a session variable.
- session.get('key') – retrieves a value.
- session.pop('key') – removes a session variable.

B. Attach screenshots:

1. Folder Structure



2. app.py

```
from flask import Flask, render_template, request
import requests

app = Flask(__name__)

@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == "POST":
        city = request.form['city']
        country = request.form['country']
        api_key = ""

        url = f"http://api.openweathermap.org/data/2.5/weather?q={city},{country}&units=imperial&appid={api_key}"
        print(url)
        response = requests.get(url)
        data = response.json()
        if response.status_code == 200:
            temp = round(data['main']['temp'])
            humidity = data['main']['humidity']
            wind_speed = data['wind']['speed']

            return render_template("result.html", temp=temp, humidity=humidity, wind_speed=wind_speed, city=city)
        else:
            error = data.get("message", "City not found")
            return render_template("index.html", error=error)

    return render_template("index.html")

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

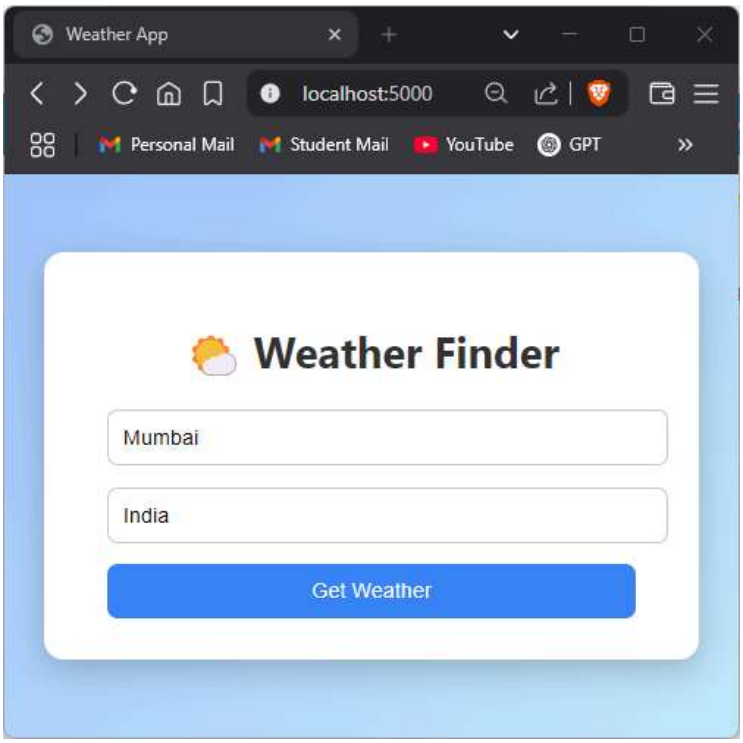
3. Index.html

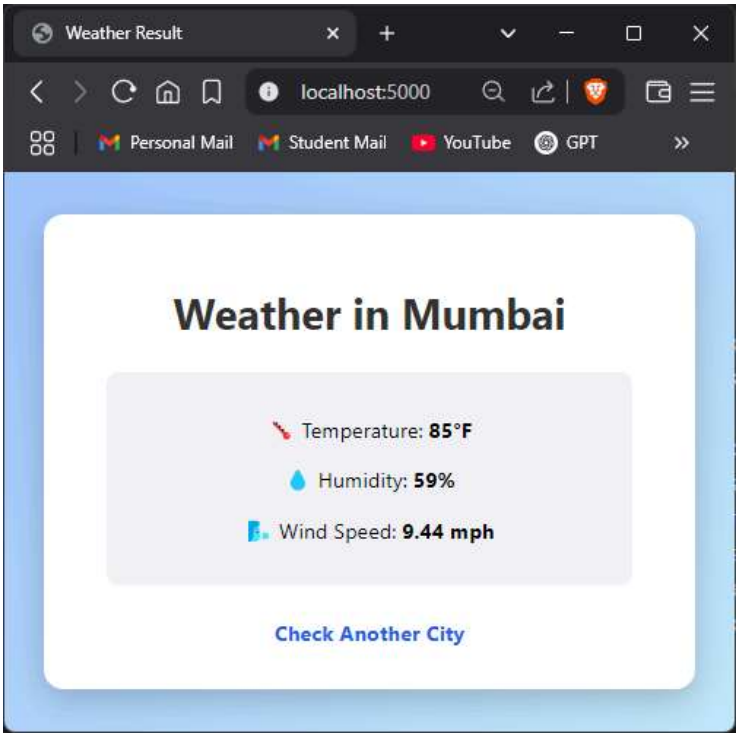
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Weather App</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}" />
  </head>
  <body>
    <div class="container">
      <h1>🌤️ Weather Finder</h1>
      {% if error %}
      <p class="error">{{ error }}</p>
      {% endif %}
      <form method="POST">
        <input type="text" name="city" placeholder="Enter City (e.g. Mumbai)" required/>
        <input type="text name="country" placeholder="Enter Country Code (e.g. IN)"  required />
        <button type="submit">Get Weather</button>
      </form>
    </div>
  </body>
</html>
```

4. result.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <title>Weather Result</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}"/>
  </head>
  <body>
    <div class="container">
      <h1>Weather in {{ city }}</h1>
      <div class="result-box">
        <p>🌡️ Temperature: <strong>{{ temp }}°F</strong></p>
        <p>💧 Humidity: <strong>{{ humidity }}%</strong></p>
        <p>🌬️ Wind Speed: <strong>{{ wind_speed }} mph</strong></p>
      </div>
      <a href="/" class="back-btn">Check Another City</a>
    </div>
  </body>
</html>
```

Output:


Weather App
localhost:5000
Personal Mail
Student Mail
YouTube
GPT
Weather Finder
Mumbai
India
Get Weather


Weather Result
localhost:5000
Personal Mail
Student Mail
YouTube
GPT
Weather in Mumbai
Temperature: 85°F
Humidity: 59%
Wind Speed: 9.44 mph
Check Another City