Experiment 4 : Flask Application using GET and POST

Name of Student	**Soham Satpute**
Class Roll No	**D15A/51**
D.O.P.	**27/02/2025**
D.O.S.	**27/03/2025**
Sign and Grade	

AIM : To design a Flask application that showcases URL building and demonstrates the use of HTTP methods (`GET` and `POST`) for handling user input and processing data.

PROBLEM STATEMENT :

Create a Flask application with the following requirements:

1. **A homepage (`/`) with links to a "Profile" page and a "Submit" page using the `url_for()` function.**
2. **The "Profile" page (`/profile/<username>`) dynamically displays a user's name passed in the URL.**
3. **A "Submit" page (`/submit`) displays a form to collect the user's name and age. The form uses the POST method to send the data, and the server displays a confirmation message with the input.**

Theory:

1. What is a route in Flask, and how is it defined?

In Flask, a **route** is a URL pattern that is associated with a specific function in a web application. When a user visits a particular URL, Flask executes the corresponding function and returns the response. Routes define how the application should respond to different URLs.

A route is defined using the `@app.route()` decorator. For example:

python CopyEdit `from flask`
`import Flask`

```
app = Flask(__name__)

@app.route('/') def home():
return "Welcome to Flask!"
```

In this example, the root URL (`/`) is mapped to the `home` function, which returns a simple text response.

---

2. How can you pass parameters in a URL route?

In Flask, parameters can be passed in a URL route using angle brackets (`< >`). These parameters can be dynamic and help in passing values from the URL to the function.

For example:

python CopyEdit

```
@app.route('/user/<name>')
def greet_user(name):
return f"Hello, {name}!"
```

Here, when a user visits `/user/Sanket`, the function receives `"Sanket"` as the `name` parameter and returns `"Hello, Sanket!"`.

Flask also allows specifying the data type of parameters, such as:

python CopyEdit

```
@app.route('/square/<int:num>') def
square(num):     return f"The square of {num}
is {num**2}"
```

This ensures that `num` is treated as an integer.

3.What happens if two routes in a Flask application have the same URL pattern?

If two routes have the same URL pattern in a Flask application, only the last defined route will take effect, and the previous one will be overridden. Flask does not allow duplicate routes with the same URL pattern, as it would cause ambiguity.

Example of conflicting routes:

python CopyEdit

```
@app.route('/hello') def
hello1():
    return "Hello from function 1"

@app.route('/hello') def hello2():
return "Hello from function 2"
```

In this case, when a user visits `/hello`, Flask will only execute `hello2()`, and `hello1()` will be ignored.

4.What are the commonly used HTTP methods in web applications?

The most commonly used HTTP methods in web applications are:

1. **GET** – Requests data from the server (e.g., retrieving a webpage).
2. **POST** – Sends data to the server (e.g., submitting a form).
3. **PUT** – Updates existing data on the server.
4. **DELETE** – Removes a resource from the server.
5. **PATCH** – Partially updates a resource.

In Flask, these methods can be specified in the `methods` parameter of the `@app.route()` decorator:

python CopyEdit

```
@app.route('/submit', methods=['POST'])
def submit():     return "Form
submitted!"
```

5.What is a dynamic route in Flask?

A **dynamic route** in Flask is a route that contains variables, allowing it to handle multiple different URLs with a single function. Dynamic routes make the web application more flexible by enabling the use of parameters within the URL.

Example:

python CopyEdit

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

If a user visits `/user/Sanket`, the function receives `"Sanket"` as a parameter and responds accordingly.

---

6. Write an example of a dynamic route that accepts a username as a parameter.

python CopyEdit

```
from flask
import Flask

app = Flask(__name__)

@app.route('/user/<username>') def
show_user(username):     return
f"Welcome, {username}!"
 if __name__ ==
'__main__':
app.run(debug=True)
```

In this example, the route `/user/<username>` accepts a `username` parameter from the URL and returns a personalized welcome message.

7.What is the purpose of enabling debug mode in Flask?

Enabling **debug mode** in Flask is useful for development because it provides:

1. **Automatic Code Reloading** – The server automatically restarts when changes are made to the code.
2. **Detailed Error Messages** – Flask displays an interactive debugger when an error occurs, making it easier to identify and fix issues.

However, debug mode should **not** be enabled in a production environment due to security risks.

8.How do you enable debug mode in a Flask application?

Debug mode can be enabled in two ways:

1. **Setting `debug=True` in `app.run()`**

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

2. **Using Environment Variables**
   In the terminal, set the environment variable before running the Flask app:

sh CopyEdit
```
export FLASK_ENV=development flask
run
```

On Windows (Command Prompt):

sh CopyEdit
```
set FLASK_ENV=development flask
run
```

This enables debug mode and allows for easier debugging during development.

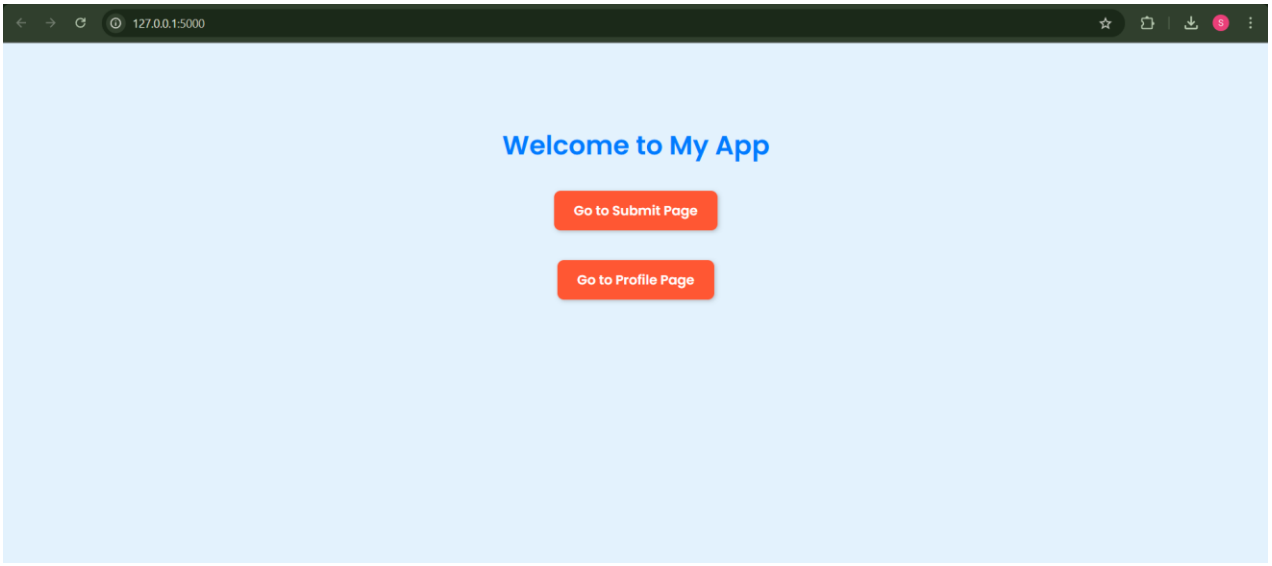
OUTPUT:-

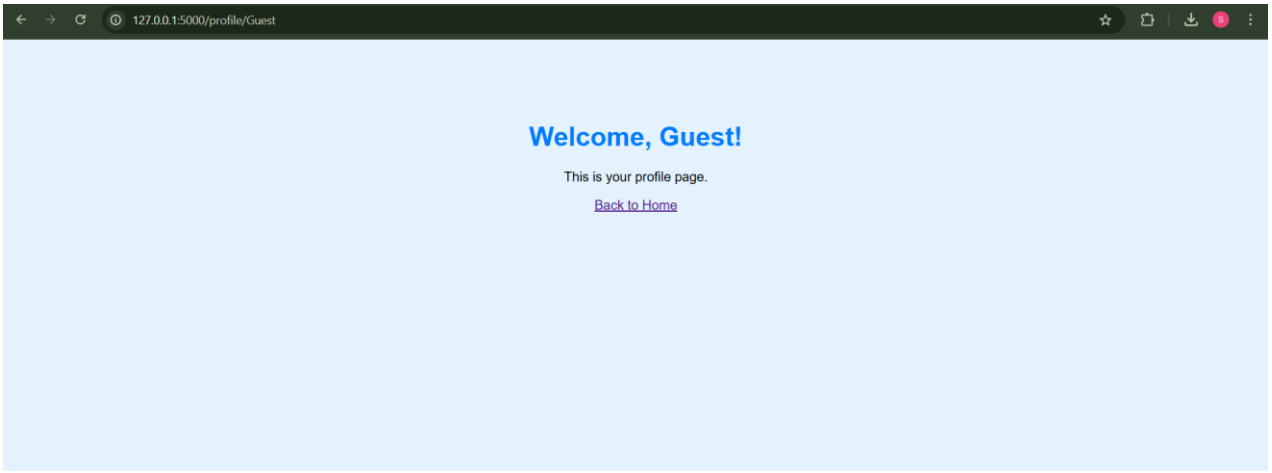
```
from flask import Flask, render_template, request, redirect, url_for

import logging


app = Flask(__name__)



# Configure logging

logging.basicConfig(level=logging.INFO)



@app.before_request

def log_request_info():

    """Logs every request with method and URL"""

                        app.logger.info(f"Request: {request.method} {request.url}")



@app.route('/')

def home():

    return """

    <html>

    <head>

                                                <link href='https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap' rel='stylesheet'>

        <style>

            body { font-family: 'Poppins', sans-serif; background-color: #e3f2fd; text-align: center; }

            .container { margin-top: 100px; }

            .btn { text-decoration: none; color: white; font-weight: bold; padding: 12px 24px; border-radius: 8px; background-color: #ff5733; box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.2); display: inline-block; margin: 10px; transition: 0.3s; }

            .btn:hover { background-color: #c70039; }

            h1 { color: #007BFF; }

        </style>

    </head>

    <body>

        <div class='container'>

            <h1>Welcome to My App</h1>

            <p><a href='/submit' class='btn'>Go to Submit Page</a></p>

                        <p><a href='/profile/Guest' class='btn'>Go to Profile Page</a></p>

        </div>

    </body>

    </html>

    """



@app.route('/profile/<username>')
```

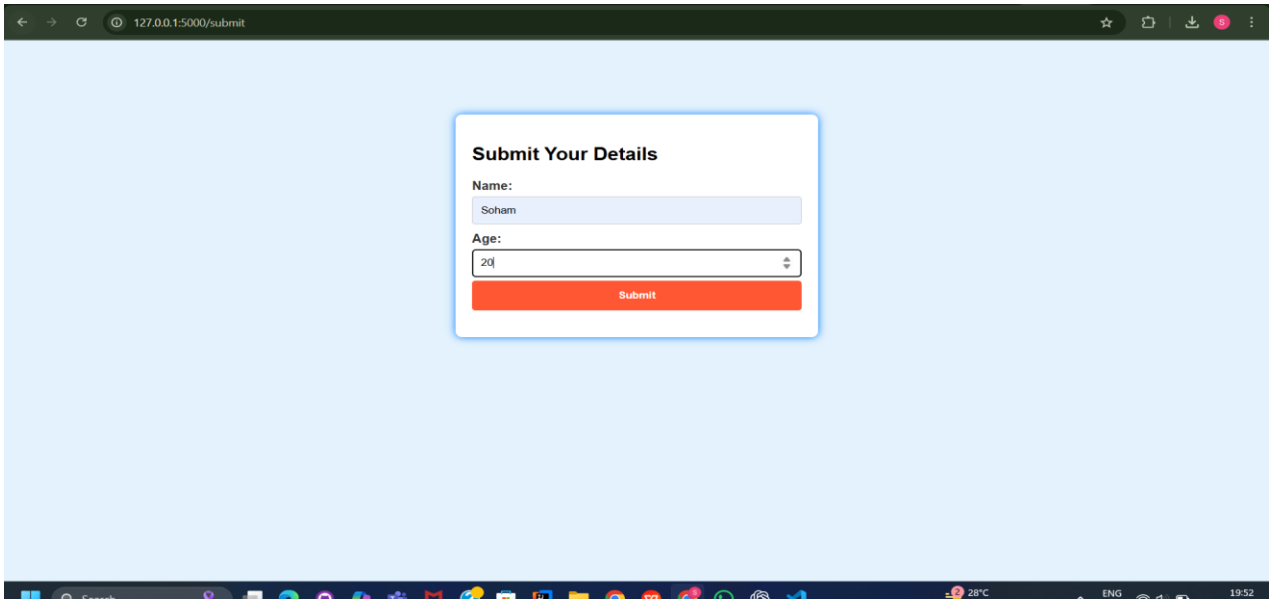
```
def profile(username):
    return f"""
    <html>
    <head>
        <style>
            body {{ font-family: 'Poppins', sans-serif; background-color: #e3f2fd; text-align: center; }}
            .container {{ margin-top: 100px; }}
            h1 {{ color: #007BFF; }}
        </style>
    </head>
    <body>
        <div class='container'>
            <h1>Welcome, {username}!</h1>
            <p>This is your profile page.</p>
                <a href='/' class='btn'>Back to Home</a>
        </div>
    </body>
    </html>
    """

@app.route('/submit', methods=['GET', 'POST'])
def submit():
    if request.method == 'POST':
        name = request.form['name']
        age = request.form['age']
            return redirect(url_for('confirmation', name=name, age=age))
    return """
    <html>
    <head>
        <style>
            body { font-family: 'Poppins', sans-serif; background-color: #e3f2fd; }
            .container { max-width: 400px; margin: 100px auto; background: #fff; padding: 20px; border-radius: 8px; box-shadow: 0px 0px 10px #007BFF; }
            label { font-weight: bold; color: #333; display: block; margin-top: 10px; }
            input { padding: 10px; width: 100%; margin-top: 5px; border: 1px solid #ccc; border-radius: 4px; }
            .btn { background: #ff5733; color: white; border: none; padding: 12px; cursor: pointer; width: 100%; border-radius: 4px; font-weight: bold; transition: 0.3s; }
            .btn:hover { background: #c70039; }
        </style>
    </head>
    <body>
        <div class='container'>
            <h2>Submit Your Details</h2>
```

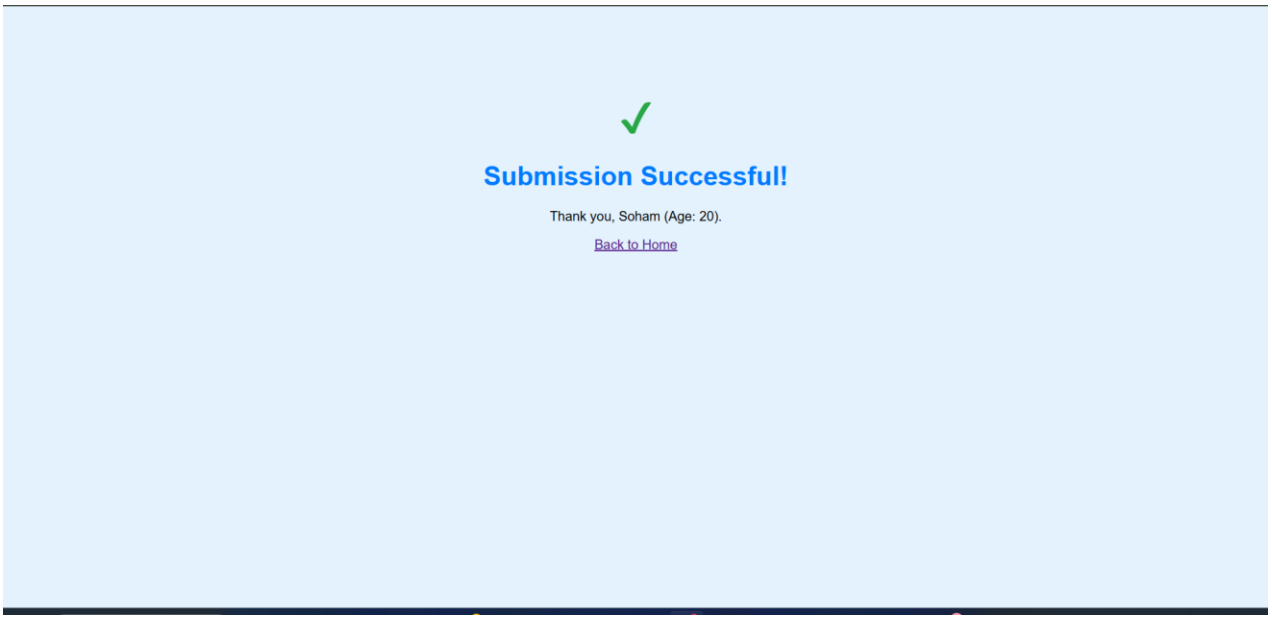
```
                    <form method='POST' action='/submit'>
            <label for='name'>Name:</label>
                <input type='text' name='name' required>
            <label for='age'>Age:</label>
              <input type='number' name='age' required>
            <input type='submit' value='Submit' class='btn'>
          </form>
      </div>
   </body>
   </html>
   """

@app.route('/confirmation')
def confirmation():
        name = request.args.get('name', 'Unknown')
       age = request.args.get('age', 'Not provided')
   return f"""
   <html>
   <head>
      <style>
          body {{ font-family: 'Poppins', sans-serif; background-color: #e3f2fd; text-align: center; }}
          .container {{ margin-top: 100px; }}
          .success-icon {{ font-size: 50px; color: #28a745; }}
          h1 {{ color: #007BFF; }}
      </style>
   </head>
   <body>
      <div class='container'>
                    <div class='success-icon'>&#10004;</div>
          <h1>Submission Successful!</h1>
              <p>Thank you, {name} (Age: {age}).</p>
             <a href='/' class='btn'>Back to Home</a>
      </div>
   </body>
   </html>
   """

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

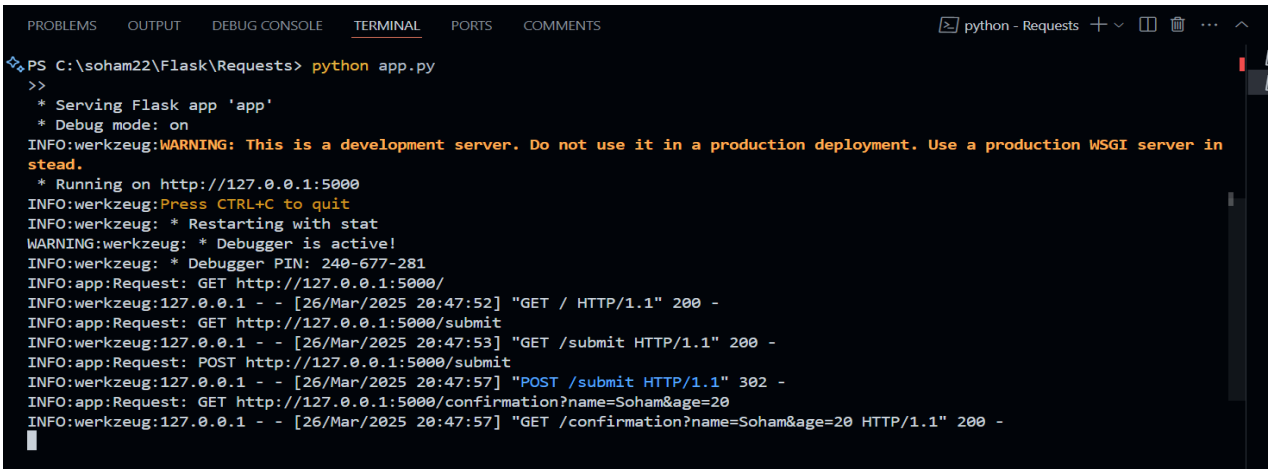
Result:-


127.0.0.1:5000
Welcome to My App
Go to Submit Page
Go to Profile Page


127.0.0.1:5000/profile/Guest
Welcome, Guest!
This is your profile page.
Back to Home


127.0.0.1:5000/submit
Submit Your Details
Name:
Soham
Age:
20
Submit
28°C
ENG
19:52


Submission Successful!
Thank you, Soham (Age: 20).
Back to Home


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
python - Requests
PS C:\soham22\Flask\Requests> python app.py
>>
* Serving Flask app 'app'
* Debug mode: on
INFO:werkzeug: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
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with stat
WARNING:werkzeug: * Debugger is active!
INFO:werkzeug: * Debugger PIN: 240-677-281
INFO:app:Request: GET http://127.0.0.1:5000/
INFO:werkzeug:127.0.0.1 - - [26/Mar/2025 20:47:52] "GET / HTTP/1.1" 200 -
INFO:app:Request: GET http://127.0.0.1:5000/submit
INFO:werkzeug:127.0.0.1 - - [26/Mar/2025 20:47:53] "GET /submit HTTP/1.1" 200 -
INFO:app:Request: POST http://127.0.0.1:5000/submit
INFO:werkzeug:127.0.0.1 - - [26/Mar/2025 20:47:57] "POST /submit HTTP/1.1" 302 -
INFO:app:Request: GET http://127.0.0.1:5000/confirmation?name=Soham&age=20
INFO:werkzeug:127.0.0.1 - - [26/Mar/2025 20:47:57] "GET /confirmation?name=Soham&age=20 HTTP/1.1" 200 -