Experiment – 4 : Flask

Name of Student	**Rujuta Medhi**
Class Roll No	**D15A_27**
D.O.P.	
D.O.S.	
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:

> A homepage (`/`) with links to a "Profile" page and a "Submit" page using the `url_for()` function.
>
> The "Profile" page (`/profile/<username>`) dynamically displays a user's name passed in the URL.
>
> 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?

A route in Flask is a URL pattern that is mapped to a specific function, allowing users to access different pages or perform actions in a web application.

- Routes define how different URLs should be handled in an application.
- When a user accesses a specific URL, the associated function (view function) is executed.
- Flask routes are defined using decorators, which associate a URL with a function.

2. How Can You Pass Parameters in a URL Route?

Flask allows passing parameters in the URL using dynamic routes. Parameters are specified inside angle brackets (< >) within the route definition.

- These parameters can be extracted and used within the function.
- Flask automatically converts them to strings unless a specific type is defined (e.g., <int:id> for integers).
- This approach is useful for retrieving user-specific data, handling dynamic pages, and processing queries.

3. What Happens if Two Routes in a Flask Application Have the Same URL Pattern?

If two routes have the same URL pattern, Flask will raise an error because it does not know which function to execute when the URL is accessed.

- The application will not start, and Flask will return a routing conflict error.
- To avoid conflicts, each URL should be unique, or URL parameters should be used to differentiate them.

4. Commonly Used HTTP Methods in Web Applications

Web applications use different HTTP methods to interact with the server. The most commonly used ones are:

- GET – Retrieves data from the server.
- POST – Sends data to the server, often used for form submissions.
- PUT – Updates an existing resource.
- DELETE – Removes a resource from the server.
- PATCH – Partially updates an existing resource.

- HEAD – Similar to GET, but returns only headers, not the body.
- OPTIONS – Returns allowed HTTP methods for a resource.

5. What is a Dynamic Route in Flask?

A dynamic route in Flask is a URL pattern that includes placeholders for variable parts. These variables are used to capture user-specific or dynamic data.

- Dynamic routes allow Flask to handle different inputs without defining multiple routes manually.
- The captured variables can be used in the function to generate personalized responses.

6. Example of a Dynamic Route That Accepts a Username as a Parameter

A dynamic route can be created to accept a username as a parameter in the URL. When a user accesses the route with their username, Flask retrieves it and processes it accordingly.

- This is useful for user profiles, dashboards, and customized experiences.
- The username is passed in the URL and can be displayed or used in backend processing.

7. What is the Purpose of Enabling Debug Mode in Flask?

Enabling debug mode in Flask provides a better development experience by allowing developers to detect and fix errors quickly. The main benefits of debug mode include:

- Automatic Code Reloading – The Flask application restarts automatically whenever a code change is detected, eliminating the need to restart the server manually.

- Interactive Debugger – If an error occurs, Flask displays a detailed traceback in the browser, allowing developers to inspect variables and diagnose issues.

- Error Logging – Provides detailed error messages and logs, making debugging more efficient.

Debug mode should only be enabled during development. In production, it can expose sensitive information and should be turned off for security reasons.

8. How Do You Enable Debug Mode in a Flask Application?

Debug mode in Flask can be enabled through different methods, such as:

- Setting a configuration flag – Flask provides a built-in way to enable debugging by modifying app settings.

- Using environment variables – Debug mode can be enabled through system-level environment settings.

- Running the Flask development server with debugging enabled – This ensures that errors are displayed interactively in the browser.

When debug mode is ON, Flask provides real-time error tracking, automatic reloading, and debugging tools, making development smoother and more efficient. However, it must be disabled in production to prevent security vulnerabilities.

Output:

app.py

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

app = Flask(__name__)

# Homepage Route
@app.route('/')
def home():
    return render_template('index.html')

# Profile Page Route (Dynamic)
@app.route('/profile/<username>')
def profile(username):
```

```
    return render_template('profile.html', username=username)

# Submit Page Route (GET & POST)
@app.route('/submit', methods=['GET', 'POST'])
def submit():
    if request.method == 'POST':
        username = request.form['name']
        age = request.form['age']
        return f"Thank you, {username}! You are {age} years old."
    return render_template('submit.html')

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

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home</title>
</head>
<body style="font-family: Arial, sans-serif; text-align: center; padding: 20px; background-color:
#f4f4f4;">
    <div style="background: white; padding: 20px; border-radius: 10px; box-shadow: 0px 0px
10px rgba(0,0,0,0.1); max-width: 400px; margin: auto;">
        <h1>Welcome to Flask App</h1>

        <p><a href="{{ url_for('submit') }}" style="color: blue;">Submit Your Details</a></p>
    </div>
</body>
</html>
```

Profile.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Profile</title>
</head>
<body style="font-family: Arial, sans-serif; text-align: center; padding: 20px; background-color:
#f4f4f4;">
```

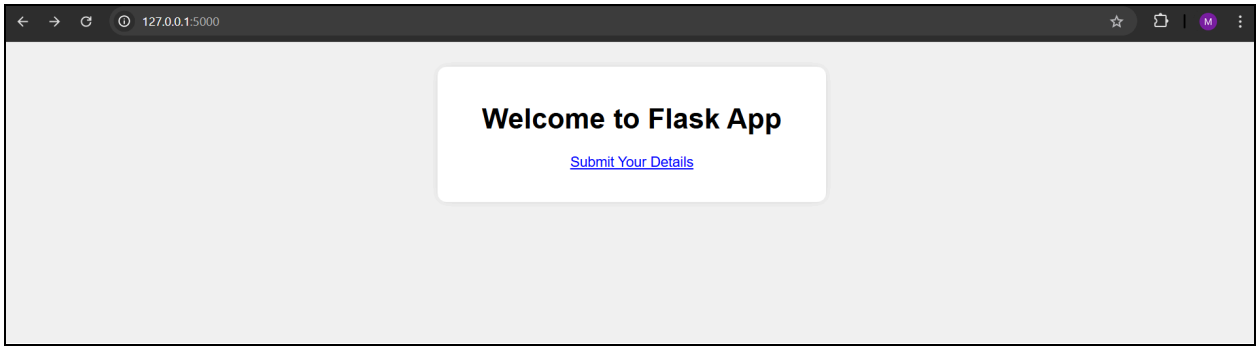
```
    <div style="background: white; padding: 20px; border-radius: 10px; box-shadow: 0px 0px 10px rgba(0,0,0,0.1); max-width: 400px; margin: auto;">
        <h1>Profile Page</h1>
        <p>Welcome, <strong>{{ username }}</strong>!</p>
        <a href="{{ url_for('home') }}" style="color: blue;">Back to Home</a>
    </div>
</body>
</html>
```

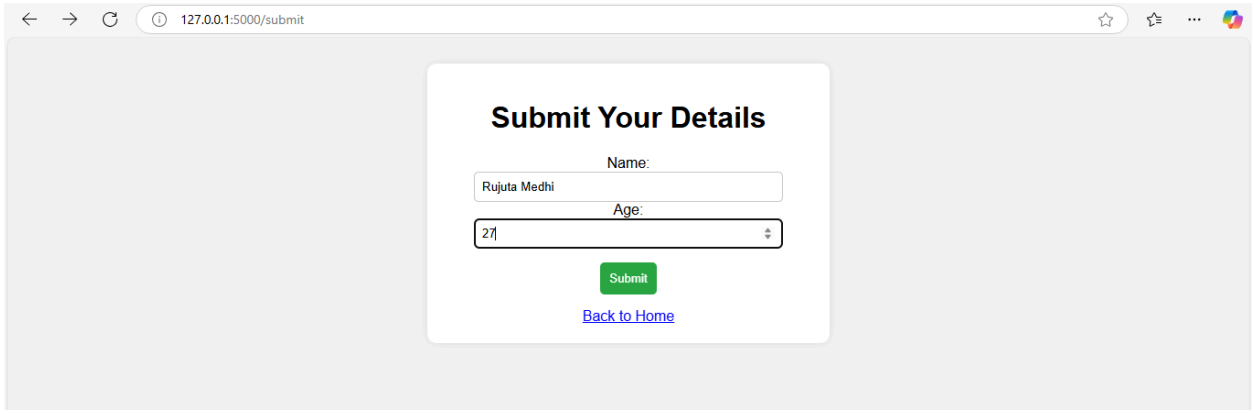
Submit.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Submit</title>
</head>
<body style="font-family: Arial, sans-serif; text-align: center; padding: 20px; background-color: #f4f4f4;">
    <div style="background: white; padding: 20px; border-radius: 10px; box-shadow: 0px 0px 10px rgba(0,0,0,0.1); max-width: 400px; margin: auto;">
        <h1>Submit Your Details</h1>
        <form method="POST" style="display: flex; flex-direction: column; align-items: center;">
            <label for="name">Name:</label>
            <input type="text" id="name" name="name" required style="padding: 8px; width: 80%; border: 1px solid #ccc; border-radius: 5px;">

            <label for="age">Age:</label>
            <input type="number" id="age" name="age" required style="padding: 8px; width: 80%; border: 1px solid #ccc; border-radius: 5px;">

            <button type="submit" style="background-color: #28a745; color: white; padding: 10px; border: none; border-radius: 5px; cursor: pointer; margin-top: 15px;">Submit</button>
        </form>
        <a href="{{ url_for('home') }}" style="color: blue; display: block; margin-top: 15px;">Back to Home</a>
    </div>
</body>
</html>
```


127.0.0.1:5000
Welcome to Flask App
Submit Your Details


127.0.0.1:5000/submit
Submit Your Details
Name:
Rujuta Medhi
Age:
27
Submit
Back to Home


127.0.0.1:5000/submit
Thank you, Rujuta Medhi! You are 27 years old.