EXPERIMENT NO. 4 - Flask Application using GET and POST

Name of the student	**Ria Chaudhari**
Class and Roll no	**D15A07**
DOP	**27/02/2025**
DOS	**6/3/2025**
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?

A **route** in Flask is a URL pattern that determines which function should be executed when a user accesses a specific URL in the web application.

Definition: Routes are defined using the `@app.route()` decorator in Flask.

Example:

@app.route('/')

def homepage():

  return "Welcome to the Homepage"

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

You can pass parameters in a URL route by defining placeholders in the route using angle brackets (`<>`). Flask will extract the parameter value and pass it to the view function.

Example:

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

def greet_user(username):

    return f"Hello, {username}!"
```

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

If two routes in a Flask application have the **same URL pattern**, Flask will raise an error because each route must be unique. The application will not start.

Incorrect Example:

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

def profile1():

    return "Profile 1"

@app.route('/profile')  # Duplicate route

def profile2():

    return "Profile 2"
```

This will cause an error: AssertionError: View function mapping is overwriting an existing endpoint function.

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

The most commonly used HTTP methods in web applications are:

GET → Retrieve data (default method).
POST → Submit data to the server.
PUT → Update existing data.
DELETE → Remove data.
PATCH → Partially update data.

5.What is a dynamic route in Flask?

A **dynamic route** is a route that includes a variable (placeholder) in the URL. It allows passing dynamic values to the view function.

Example of a dynamic route:

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

def show_profile(username):
    return f"User Profile: {username}"
```

Accessing /user/Alice will display: User Profile: Alice.

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

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

def profile(username):

    return f"Welcome, {username}!"
```

Accessing /profile/JohnDoe will return: "Welcome, JohnDoe!".

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

Enabling **debug mode** in Flask allows:

1.**Automatic Code Reloading** → Changes to the code take effect without restarting the server.
2.**Detailed Error Messages** → If an error occurs, Flask provides an interactive debugger with a traceback.

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

You can enable debug mode by setting `debug=True` when running the application.

Example:

```
if __name__ == '__main__':

    app.run(debug=True)
```

Alternatively, you can enable debug mode using environment variables:

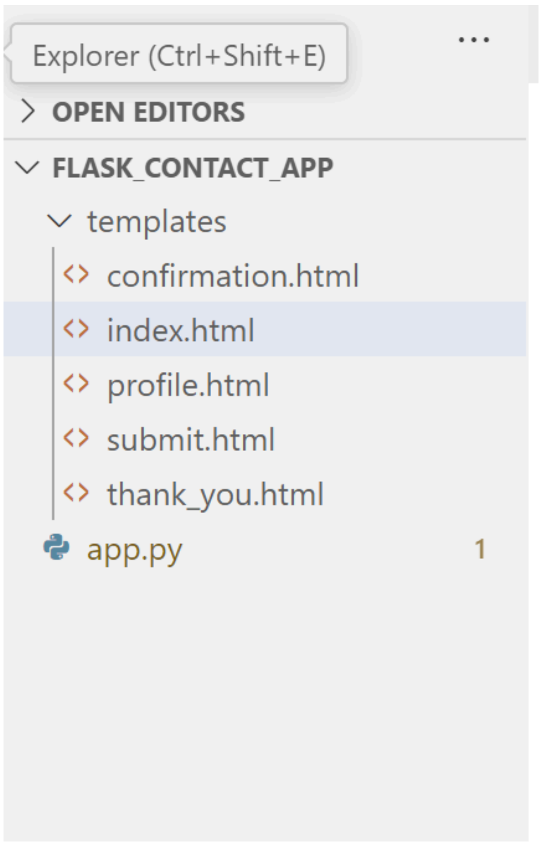
```
export FLASK_ENV=development

flask run
```

This will enable debugging and automatic reloading.

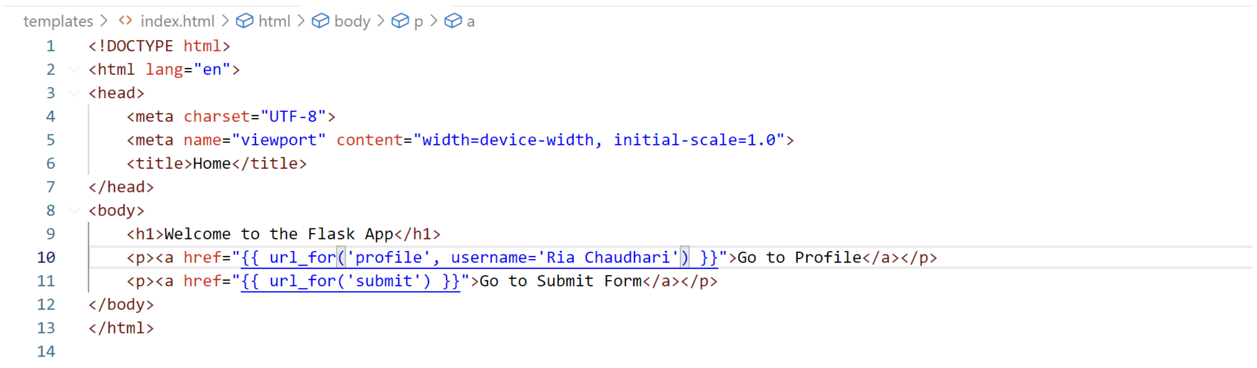
OUTPUT:

Folder Structure:



```
Explorer (Ctrl+Shift+E)
> OPEN EDITORS
v FLASK_CONTACT_APP
  v templates
    confirmation.html
    index.html
    profile.html
    submit.html
    thank_you.html
  app.py                1
```

index.html

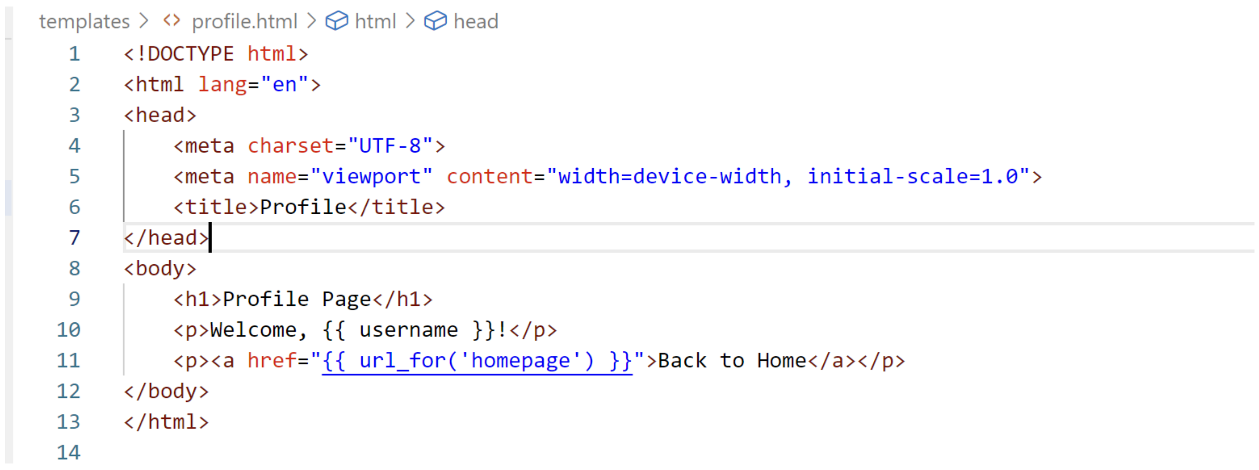


templates > index.html > html > body > p > a

```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>
    <h1>Welcome to the Flask App</h1>
    <p><a href="{{ url_for('profile', username='Ria Chaudhari') }}">Go to Profile</a></p>
    <p><a href="{{ url_for('submit') }}">Go to Submit Form</a></p>
</body>
</html>

```

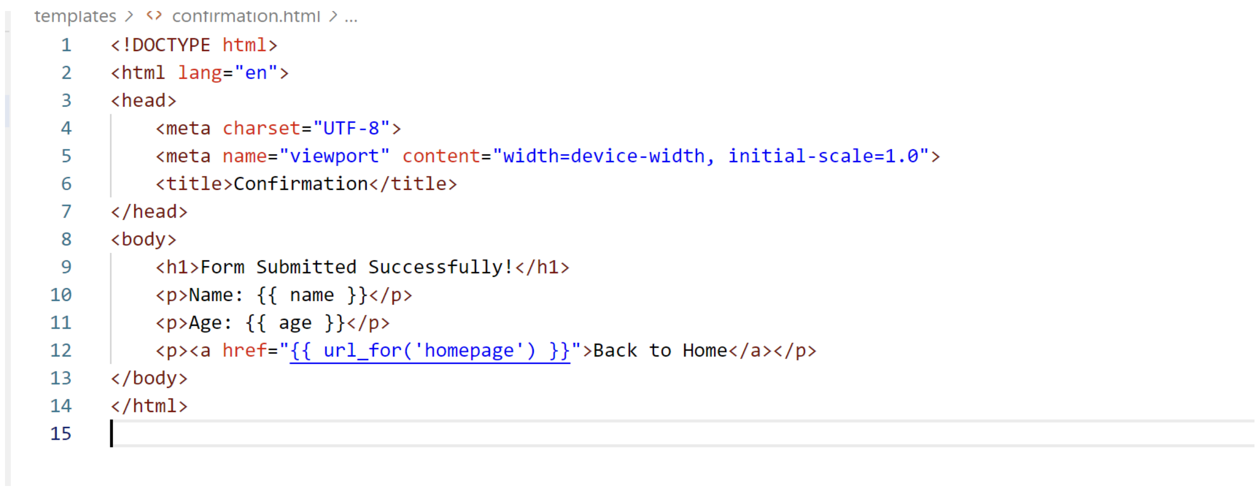
profile.html



```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>
    <h1>Profile Page</h1>
    <p>Welcome, {{ username }}!</p>
    <p><a href="{{ url_for('homepage') }}">Back to Home</a></p>
</body>
</html>

```

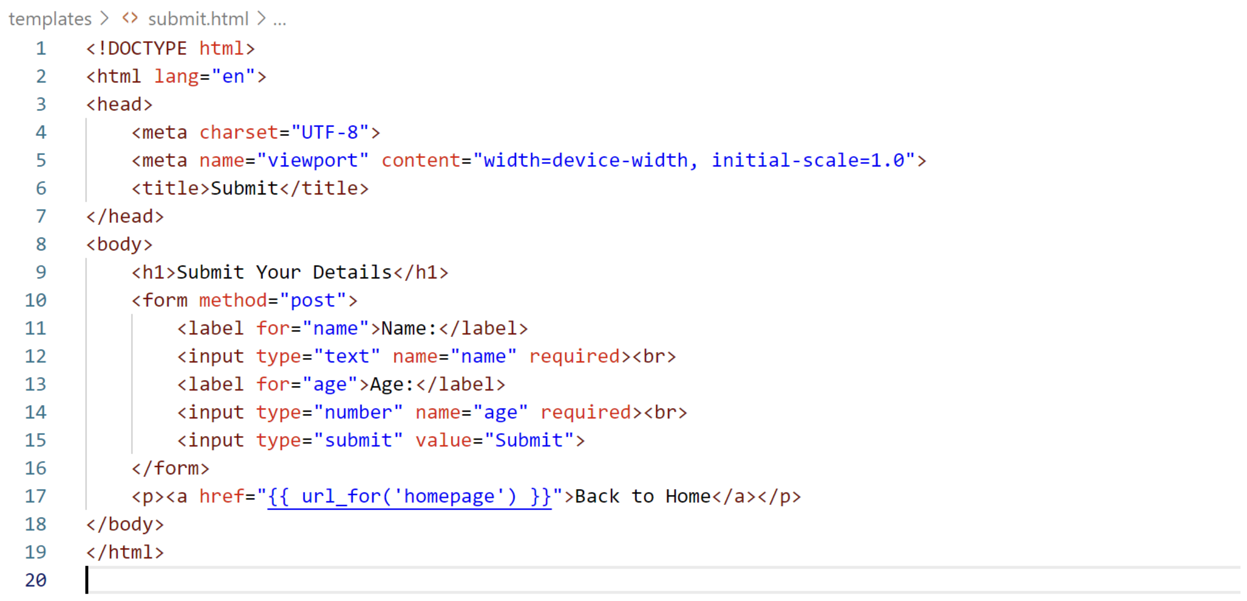
confirmation.html



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Confirmation</title>
</head>
<body>
    <h1>Form Submitted Successfully!</h1>
    <p>Name: {{ name }}</p>
    <p>Age: {{ age }}</p>
    <p><a href="{{ url_for('homepage') }}">Back to Home</a></p>
</body>
</html>

```

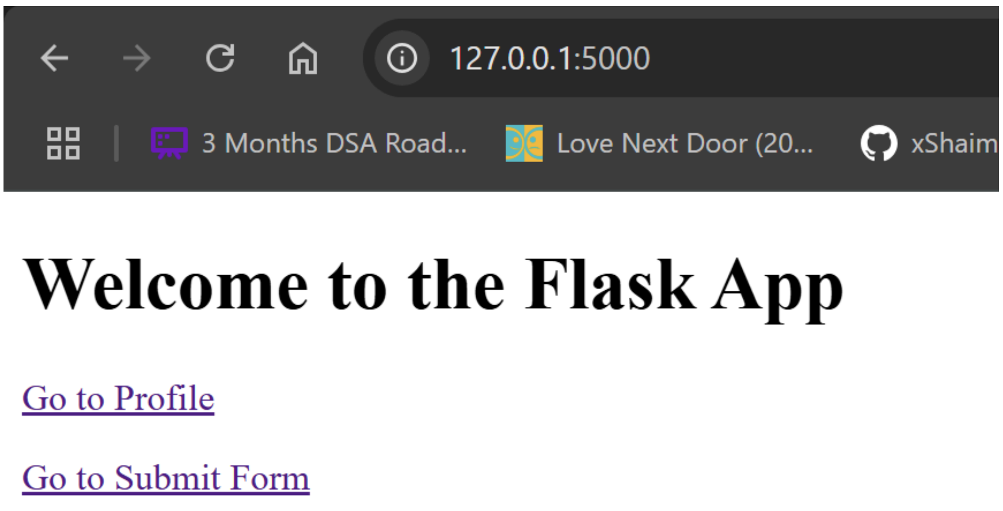
submit.html



```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>
    <h1>Submit Your Details</h1>
    <form method="post">
        <label for="name">Name:</label>
        <input type="text" name="name" required><br>
        <label for="age">Age:</label>
        <input type="number" name="age" required><br>
        <input type="submit" value="Submit">
    </form>
    <p><a href="{{ url_for('homepage') }}">Back to Home</a></p>
</body>
</html>

```

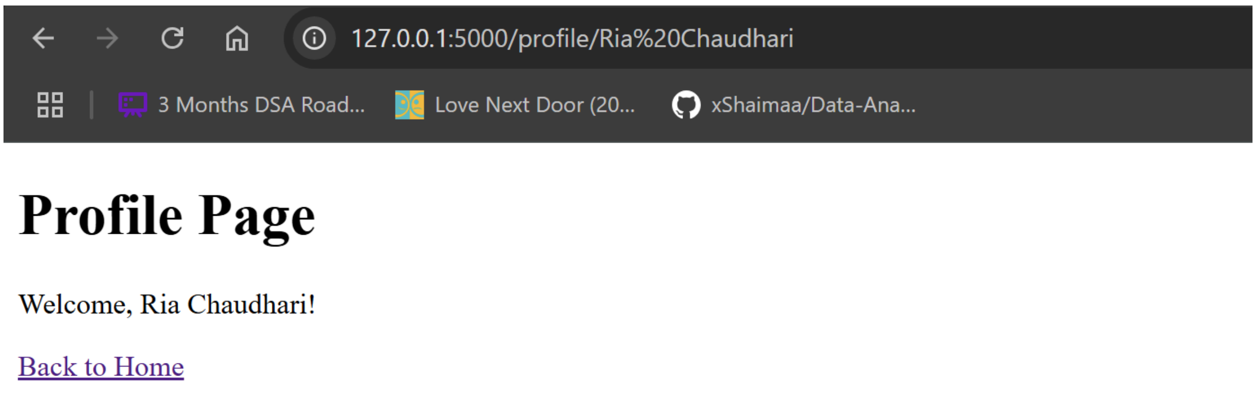
Output:



Welcome to the Flask App

Go to Profile

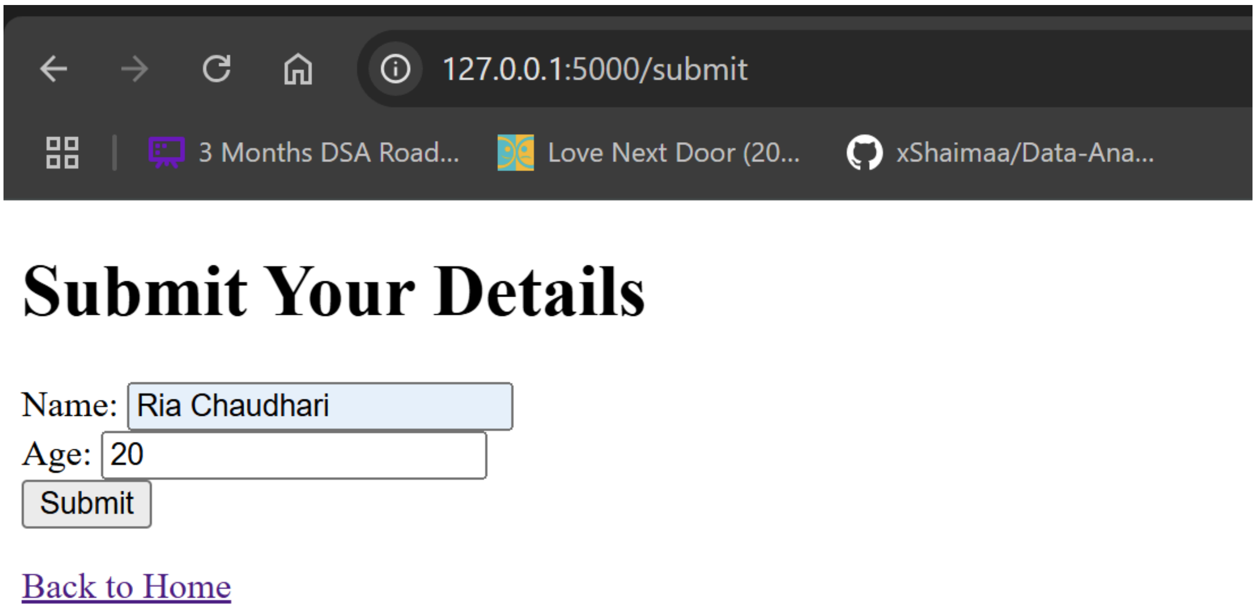
Go to Submit Form



Profile Page

Welcome, Ria Chaudhari!

Back to Home

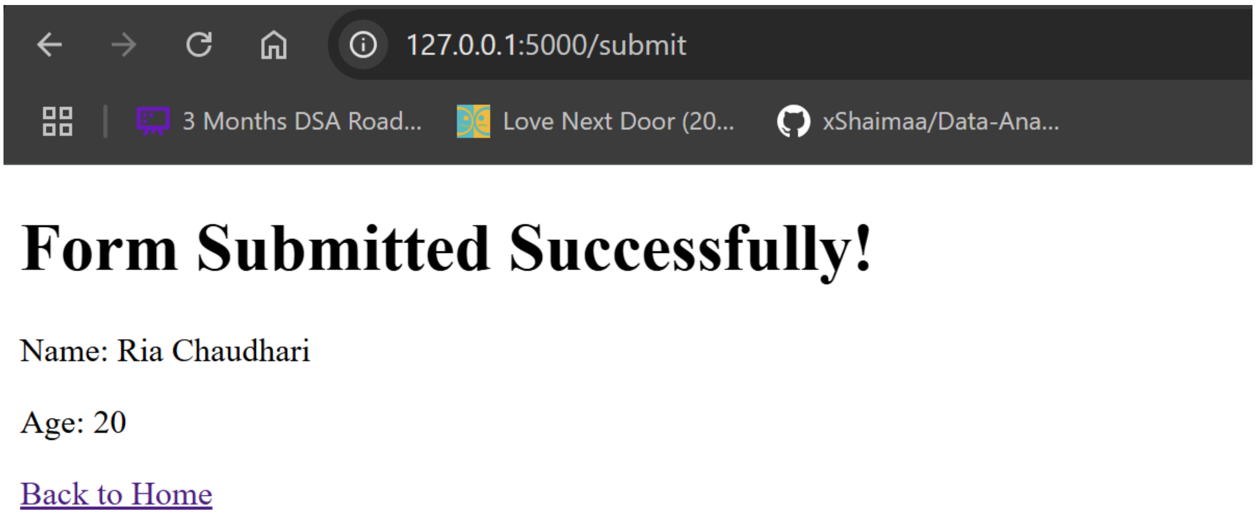

127.0.0.1:5000/submit

3 Months DSA Road... Love Next Door (20... xShaimaa/Data-Ana...

Submit Your Details

Name: Ria Chaudhari
Age: 20
Submit

Back to Home


127.0.0.1:5000/submit

3 Months DSA Road... Love Next Door (20... xShaimaa/Data-Ana...

Form Submitted Successfully!

Name: Ria Chaudhari

Age: 20

Back to Home

Conclusion:
The experiment demonstrated the creation of a simple Flask application using GET and POST methods. The application effectively handled dynamic routing using URL parameters and processed user input via a form using the POST method. Additionally, it showcased URL building using the url_for() function and maintained user data using sessions.