Experiment 5 : Flask Application using render_template() function.

Name of Student	**Anuprita Mhapankar**
Class Roll No	**28**
D.O.P.	**06/03/2025**
D.O.S.	**13/03/2025**
Sign and Grade	

AIM : To create a Flask application that demonstrates template rendering by dynamically generating HTML content using the `render_template()` function.

PROBLEM STATEMENT :

Develop a Flask application that includes:

1. A homepage route (`/`) displaying a welcome message with links to additional pages.
2. A dynamic route (`/user/<username>`) that renders an HTML template with a personalized greeting.
3. Use Jinja2 templating features, such as variables and control structures, to enhance the templates.

Theory:

1. What does the `render_template()` function do in a Flask application?

The `render_template()` function is used to render HTML templates stored in the **templates** folder. It dynamically generates web pages by passing variables from the Flask app to the template using **Jinja2**.

2. What is the significance of the templates folder in a Flask project?

- The **templates** folder is the default location where Flask looks for HTML files.
- It maintains a clean separation between business logic (Python code) and presentation logic (HTML).
- Using the **templates** folder allows developers to use **Jinja2** for rendering dynamic content.

- The folder can also store reusable components like base templates, headers, or footers using **template inheritance**.

3. What is Jinja2, and how does it integrate with Flask?

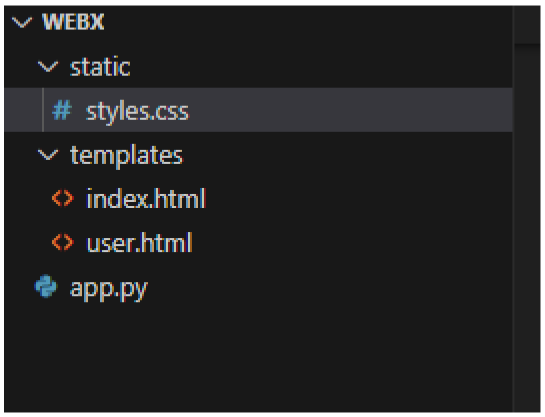
Jinja2 is a templating engine used in Flask to render dynamic HTML content. It allows embedding Python expressions inside HTML files. Using **Jinja2**, you can:

- Display variables
- Apply logic (like loops and conditionals)
- Apply filters for formatting

Flask integrates **Jinja2** by default using the `render_template()` function.

CODE:

Folder Structure:



```
WEBX
  static
    # styles.css
  templates
    index.html
    user.html
  app.py
```

1. app.py (Flask Application)

```
from flask import Flask, render_template

app = Flask(__name__)

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

@app.route('/user/<username>')
def user(username):
    return render_template('user.html', username=username)

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

2. templates/index.html (Homepage Template)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Flask Template Rendering</title>
    <link rel="stylesheet" href="{{ url_for('static',
filename='style.css') }}">
</head>
<body>
    <div class="container">
        <div class="card">
            <h1>Welcome to My Flask App</h1>
            <p>Explore user pages dynamically!</p>
            <div class="links">
                <a href="{{ url_for('user', username='Sneha') }}"
class="btn">Sneha's Page</a>
                <a href="{{ url_for('user', username='Anuprita') }}"
class="btn">Anuprita's Page</a>
            </div>
        </div>
    </div>
</body>
</html>
```

3. templates/user.html (User Template)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>User Page</title>
    <link rel="stylesheet" href="{{ url_for('static',
filename='style.css') }}">
</head>
<body>
    <div class="container">
        <div class="card">
            <h1>Hello, {{ username }}! 👋</h1>
            <p>Welcome to your personalized space.</p>
            <a href="{{ url_for('home') }}" class="btn">Back to
Home</a>
        </div>
    </div>
</body>
</html>
```

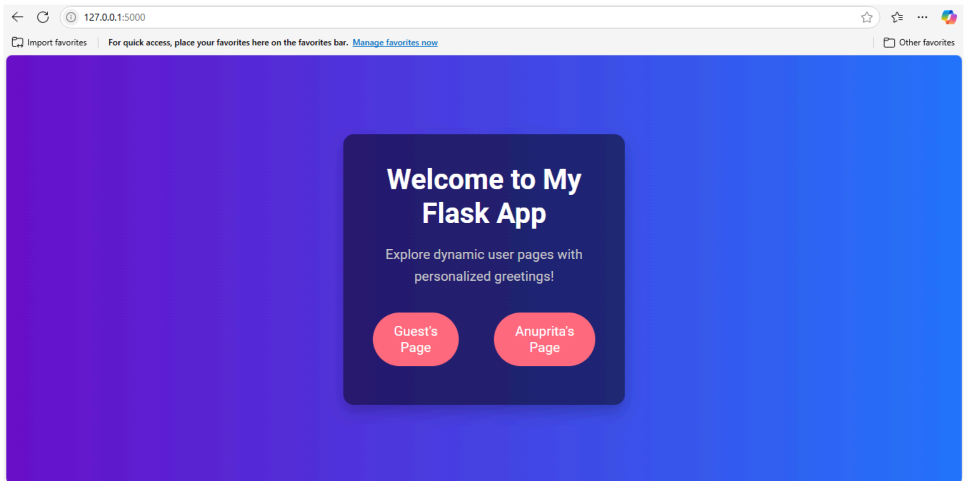
4. static/style.css (Styling the App)

```
body {
    font-family: 'Poppins', sans-serif;
    background-color: #e0f7fa;
    margin: 0;
    padding: 0;
}

.container {
    text-align: center;
    margin-top: 50px;
}

h1 {
    color: #00796b;
}

p {
    font-size: 18px;
    color: #004d40;
}

.btn {
    background-color: #00796b;
    color: white;
    padding: 10px 20px;
    text-decoration: none;
    border-radius: 5px;
}

.btn:hover {
    background-color: #004d40;
}

.card {
    background: white;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
    width: 50%;
    margin: 20px auto;
}

.links {
    display: flex;
    justify-content: center;
    gap: 10px;
    margin-top: 20px;
}
```

Github Link - https://github.com/Anuprita2022-26/WebX_Exp5

OUTPUT:

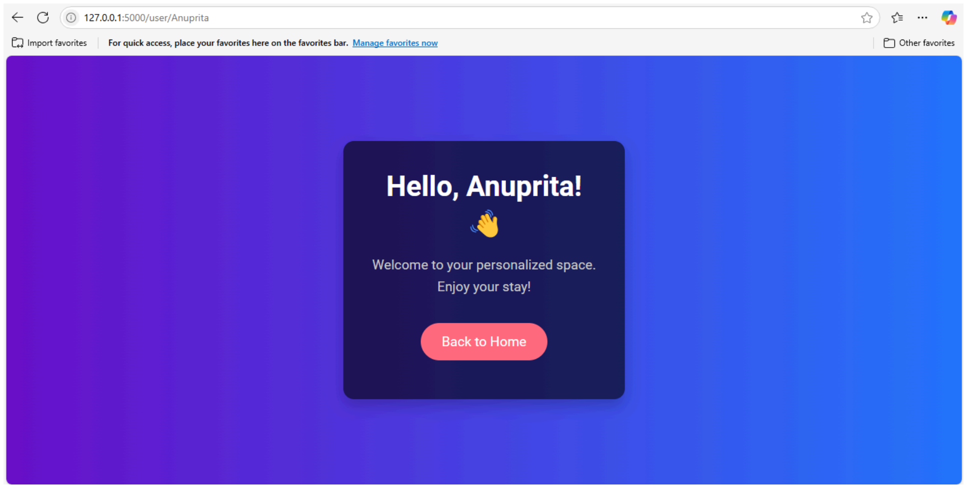
- **Homepage (`/`)**: The homepage displays a welcome message along with two links for user-specific pages (e.g., Sneha's Page and Anuprita's Page).


127.0.0.1:5000
Import favorites
For quick access, place your favorites here on the favorites bar. Manage favorites now
Other favorites
Guest's Page
Anuprita's Page

Welcome to My Flask App

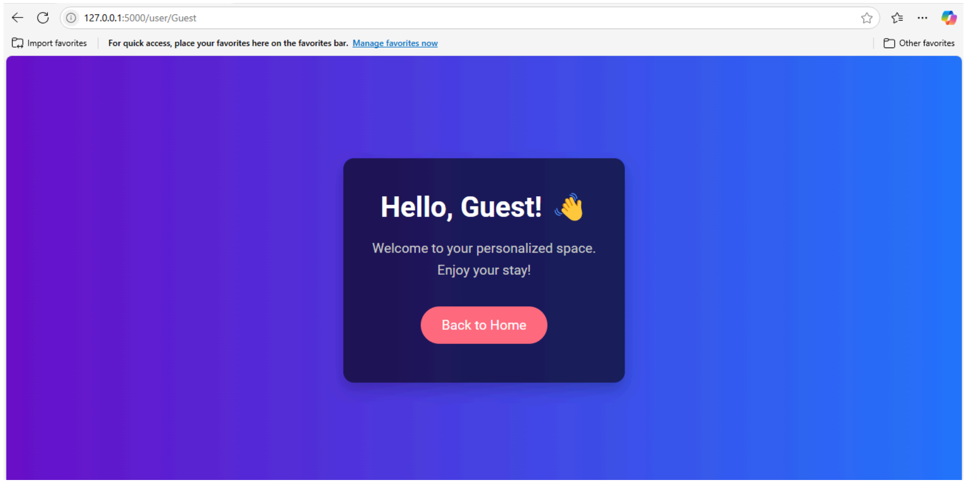
Explore dynamic user pages with personalized greetings!

- **User Page (`/user/<username>`)**: When clicking on any of the user links, the app renders a personalized greeting with the username passed as a URL parameter.


127.0.0.1:5000/user/Anuprita
Import favorites
For quick access, place your favorites here on the favorites bar. Manage favorites now
Other favorites
Back to Home

Hello, Anuprita! 👋

Welcome to your personalized space. Enjoy your stay!


127.0.0.1:5000/user/Guest
Import favorites
For quick access, place your favorites here on the favorites bar. Manage favorites now
Other favorites
Back to Home

Hello, Guest! 👋

Welcome to your personalized space. Enjoy your stay!