

# Aplicación Flask con Plantillas - Documentación

## Descripción del Proyecto

He creado una aplicación web básica utilizando Flask que demuestra el uso de plantillas HTML dinámicas con el motor de plantillas Jinja2. La aplicación incluye:

1. Una plantilla base (base.html) que sirve como estructura común para todas las páginas
2. Tres rutas diferentes que renderizan plantillas específicas
3. Uso de herencia de plantillas para evitar duplicación de código
4. Paso de datos dinámicos desde el backend (Python/Flask) al frontend (HTML/Jinja2)

## Estructura del Proyecto

/flask\_app/

|

|— static/

|   └─ styles.css    # Archivo CSS para estilos

|

|— templates/

|   └─ base.html    # Plantilla base

|   └─ index.html    # Página de inicio

|   └─ pagina1.html   # Página con lista dinámica

|   └─ pagina2.html   # Página con tabla dinámica

|

|— app.py            # Aplicación Flask principal

mi\_aplicacion > static > styles.css > body

```
1  body {
2      background-color: #f8f9fa;
3      font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
4      margin: 0;
5      padding: 0;
6      color: #333;
7  }
8
9  h1 {
10     background-color: #343a40;
11     color: white;
12     margin: 0;
13     padding: 20px;
14     text-align: center;
15 }
16
17 nav {
18     background-color: #495057;
19     padding: 10px;
20     text-align: center;
21 }
22
23 nav a {
24     color: white;
25     text-decoration: none;
26     margin: 0 15px;
27     font-weight: bold;
28 }
29
```

```
mi_aplicacion > static > styles.css > body
30  nav a:hover {
31  |    text-decoration: underline;
32  }
33
34  div {
35  |    padding: 20px;
36  }
37
38  table {
39  |    width: 100%;
40  |    border-collapse: collapse;
41  |    margin-top: 15px;
42  }
43
44  table, th, td {
45  |    border: 1px solid #dee2e6;
46  }
47
48  th, td {
49  |    padding: 10px;
50  |    text-align: left;
51  }
52
53  th {
54  |    background-color: #e9ecef;
55  }
56
```

mi\_aplicacion > templates > base.html > ...


```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4     <meta charset="UTF-8">
5     <title>{% block title %}Mi App Flask{% endblock %}</title>
6     <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
7
8 </head>
9 <body>
10     <h1>Mi Aplicación con Flask</h1>
11     <nav>
12         <a href="/">Inicio</a> |
13         <a href="/pagina1">Página 1</a> |
14         <a href="/pagina2">Página 2</a>
15     </nav>
16     <hr>
17     <div>
18         {% block content %}{% endblock %}
19     </div>
20 </body>
21 </html>
22
```

mi\_aplicacion > templates > index.html > ...


```
1 {% extends "base.html" %}
2 {% block title %}Inicio{% endblock %}
3 {% block content %}
4     <p>Bienvenido a mi aplicación básica con Flask y Jinja2.</p>
5 {% endblock %}
6
```

mi\_aplicacion > templates > pagina1.html > ...

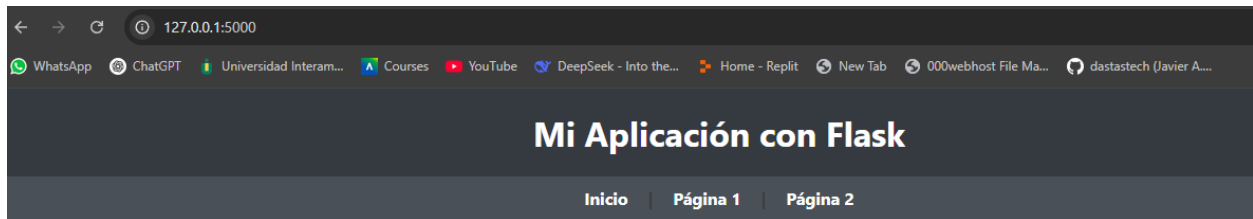
```
1 {% extends "base.html" %}
2 {% block title %}Página 1{% endblock %}
3 {% block content %}
4     <h2>Lenguajes y tecnologías</h2>
5     <ul>
6         {% for item in items %}
7             <li>{{ item }}</li>
8         {% endfor %}
9     </ul>
10 {% endblock %}
11
```

mi\_aplicacion > templates >  pagina2.html > ...

```
1  {% extends "base.html" %}
2  {% block title %}Página 2{% endblock %}
3  {% block content %}
4  <h2>Usuarios Registrados</h2>
5  <table border="1">
6      <tr><th>Nombre</th><th>Edad</th></tr>
7      {% for usuario in usuarios %}
8          <tr>
9              <td>{{ usuario.nombre }}</td>
10             <td>{{ usuario.edad }}</td>
11          </tr>
12      {% endfor %}
13  </table>
14  {% endblock %}
15
```

mi\_aplicacion >  app.py > ...

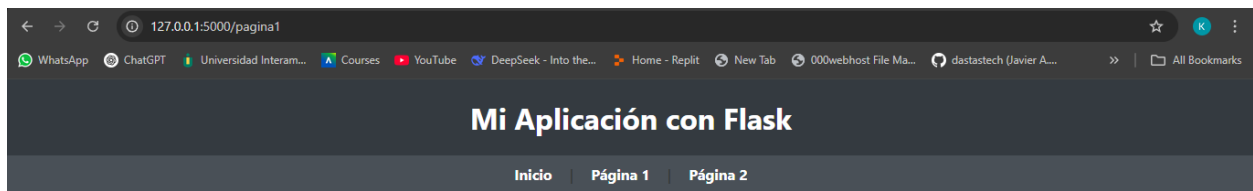
```
1  from flask import Flask, render_template
2
3  app = Flask(__name__)
4
5  @app.route('/')
6  def index():
7      return render_template("index.html")
8
9  @app.route('/pagina1')
10 def pagina1():
11     datos = ["Python", "Flask", "Jinja2"]
12     return render_template("pagina1.html", items=datos)
13
14 @app.route('/pagina2')
15 def pagina2():
16     usuarios = [
17         {"nombre": "Bloom", "edad": 18},
18         {"nombre": "Spencer", "edad": 23}
19     ]
20     return render_template("pagina2.html", usuarios=usuarios)
21
22 if __name__ == '__main__':
23     app.run(debug=True)
24
```



Bienvenido a mi aplicación básica con Flask y Jinja2.

Descripción: La página de inicio (index.html) hereda de la plantilla base y muestra un simple mensaje de bienvenida. Aquí podemos ver:

- El título de la aplicación
- La barra de navegación común
- El contenido específico de la página de inicio

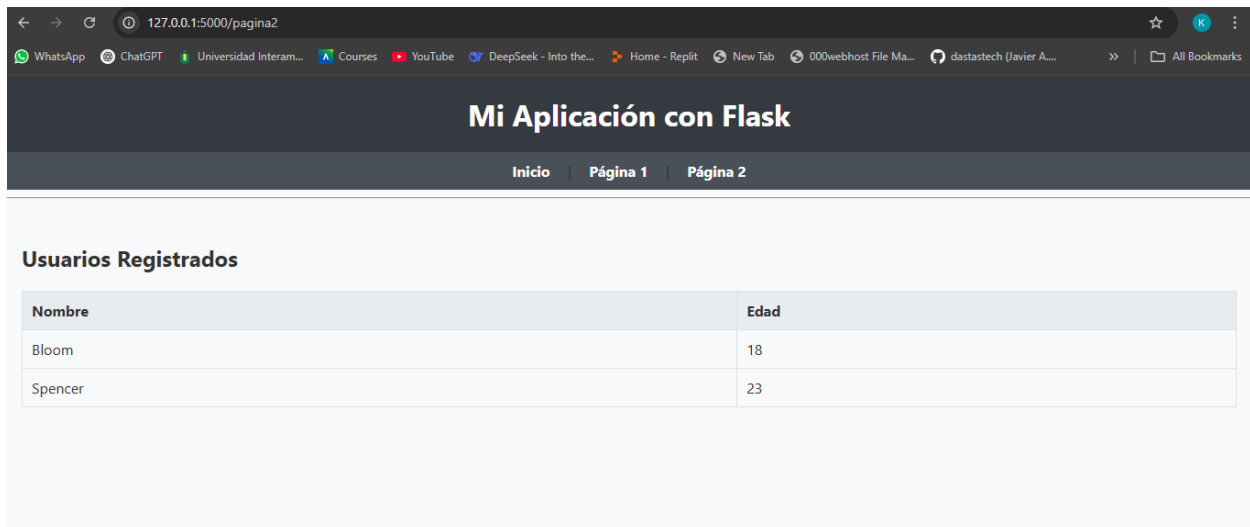


### Lenguajes y tecnologías

- Python
- Flask
- Jinja2

Descripción: Esta página (pagina1.html) muestra una lista de tecnologías que se pasan dinámicamente desde el backend:

- Los datos se definen en la ruta Flask como una lista Python
- Se iteran en la plantilla usando la sintaxis `{% for %}` de Jinja2
- Hereda todos los elementos comunes de la plantilla base



Mi Aplicación con Flask	
Inicio	Página 1
	Página 2
Usuarios Registrados	
Nombre	Edad
Bloom	18
Spencer	23

Descripción: La segunda página (pagina2.html) muestra una tabla de usuarios con sus edades:

- Los datos provienen de una lista de diccionarios en Python
- Se renderiza como una tabla HTML usando bucles Jinja2
- Demuestra cómo estructurar datos más complejos para su presentación

## Reflexión sobre la separación Back-End/Front-End en el proyecto Flask

La implementación de este proyecto con Flask y Jinja2 me permitió experimentar de primera mano los beneficios arquitectónicos de separar claramente el back-end del front-end. Esta división no es solo organizativa, sino que impacta directamente en la calidad del desarrollo:

### 1. Claridad estructural mejorada

Al diferenciar las responsabilidades (lógica en `app.py` vs presentación en `templates/`), cada componente se vuelve más legible. Por ejemplo, cuando reviso las rutas en `app.py`, puedo entender inmediatamente qué datos se preparan para cada vista sin distraerme con detalles HTML.

Simultáneamente, las plantillas se enfocan exclusivamente en cómo se muestran esos datos, usando sintaxis clara de Jinja2 como los bloques {% for %}.

## **2. Escalabilidad demostrada**

Al añadir la tercera página (pagina2.html), pude reutilizar el 90% del código existente. La arquitectura me permitió:

- Extender la plantilla base con {% extends %}
- Concentrarme solo en el nuevo contenido específico
- Inyectar datos complejos (lista de diccionarios) sin modificar la lógica existente

[https://github.com/Seliz05/COMP-2052/tree/main/mi\\_aplicacion](https://github.com/Seliz05/COMP-2052/tree/main/mi_aplicacion)