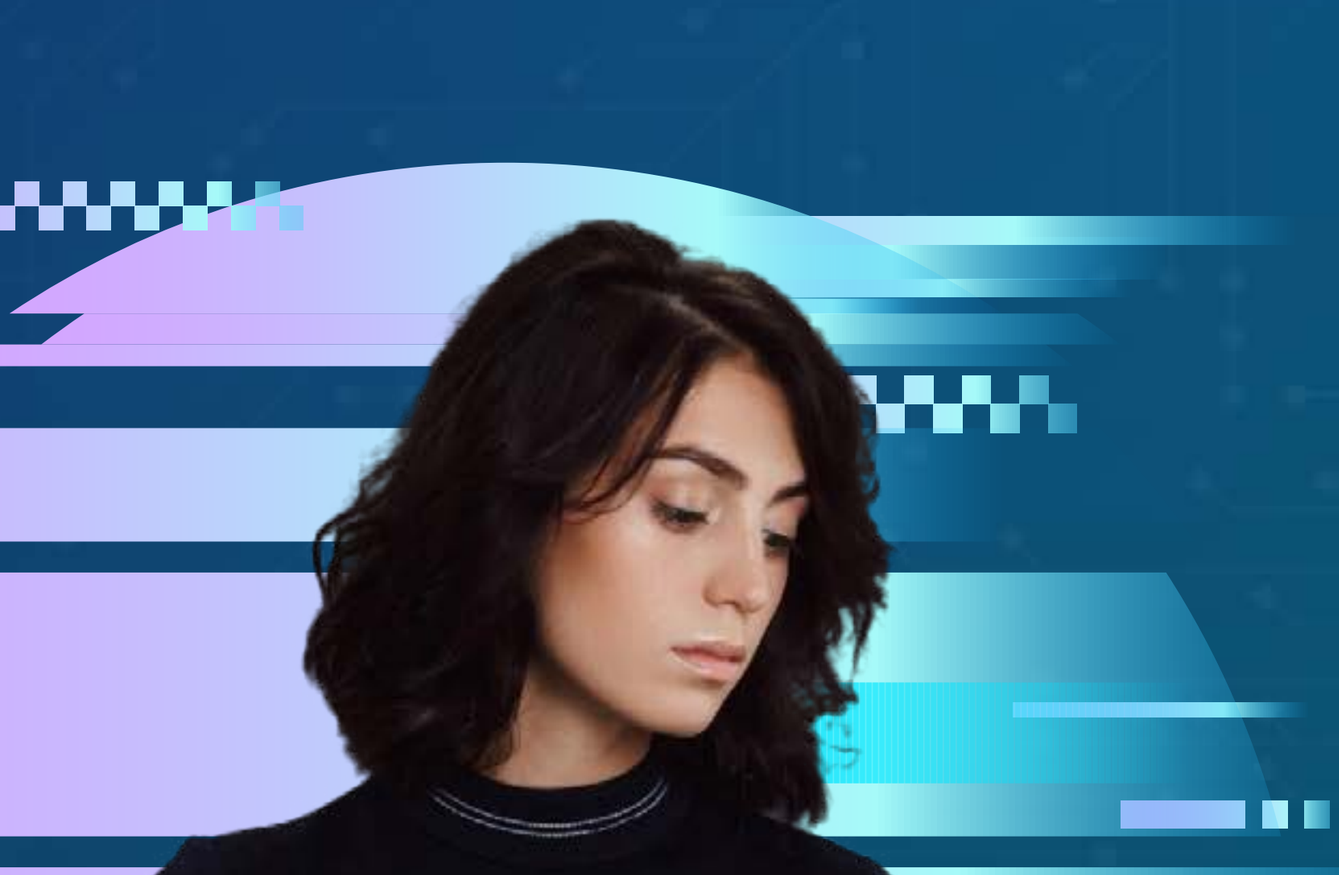
TIC

TALENTO TECH

Contenidos teóricos y conceptos clave de programación

Nivel básico (explorador) Módulo 3



cim

Módulo 3: integración de python con desarrollo web

Desarrollo Web con Flask

Introducción a Flask

Flask es un microframework de Python para el desarrollo web. Es ligero, flexible y escalable, ideal para proyectos de cualquier tamaño. Para comenzar, instala Flask con pip install Flask y crea un archivo principal como app.py.

Ejemplo básico de Flask:

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def home():
    return '¡Hola, Mundo!'

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

Estructura: Crea un archivo app.py y directorios templates/ y static/ para HTML y recursos estáticos.

Ejecuta la App: Usa *python app.py* y visita http://127.0.0.1:5000 en el navegador.

2. Creación de una Aplicación Web Simple con Flask

Rutas y Vistas

Las rutas definen las URL accesibles y las vistas manejan solicitudes a esas URLs. Usa decoradores @app.route() para definir rutas y funciones para manejar la lógica.

Manejo de Rutas y Métodos HTTP:

```
from flask import Flask, request
app = Flask(__name__)
@app.route('/')
def home():
    return 'Página de inicio'

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        usuario = request.form['username']
        return f'Bienvenido, {usuario}'
    return '''
        <form method="post">
            Usuario: <input type="text" name="username"><br>
            <input type="submit" value="Enviar">
        </form>
    '''

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

GET y POST: GET recupera datos, POST envía datos al servidor.

Formulario HTML: Usar formularios para capturar y procesar datos del usuario.

Aplicación Simple:

Crea una aplicación que toma el nombre del usuario y devuelve un mensaje de bienvenida.

```
from flask import Flask, request
app = Flask(__name__)
@app.route('/', methods=['GET', 'POST'])
def welcome():
    if request.method == 'POST':
        nombre = request.form['nombre']
        return f'¡Bienvenido, {nombre}!'
    return '''
        <form method="post">
            Nombre: <input type="text" name="nombre"><br>
            <input type="submit" value="Enviar">
        </form>
    '''

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

3. Renderización de Plantillas HTML con Flask

Flask utiliza una herramienta llamada Jinja2 para renderizar plantillas dinámicas, permitiendo incorporar lógica y datos en HTML.

Uso de Plantillas:

Estructura típica del proyecto con plantillas:

```
my_flask_app/
├── app.py
└── templates/
    ├── index.html
    └── about.html
```

Ejemplo de Renderización:

```
from flask import Flask, render_template
app = Flask(__name__)
@app.route('/')
def home():
    usuario = "Ana"
    return render_template('index.html', usuario=usuario)

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

Plantilla (index.html):

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Página de Inicio</title>
</head>
<body>
    <h1>¡Bienvenido, {{ usuario }}!</h1>
    <p>Este es un ejemplo de una página renderizada con Flask.</p>
</body>
</html>
```

Contexto y Variables: Pasa datos a las plantillas y utiliza {{ variable }} para insertar valores.

Condicionales y Bucles: Usa {% if ... %} y {% for ... %} para lógica en plantillas.

Contexto de Datos:

Pasa listas, diccionarios y más a las plantillas:

```
@app.route('/')
def home():
    frutas = ['Manzana', 'Banana', 'Naranja']
    return render_template('frutas.html', frutas=frutas)
```

Plantilla (frutas.html):

```
<ul>
    {% for fruta in frutas %}
        <li>{{ fruta }}</li>
    {% endfor %}
</ul>
```

4. Envío de Datos de Python a HTML

Flask facilita el envío de datos complejos como listas y diccionarios a las plantillas, permitiendo crear interfaces dinámicas.

Pasar Datos Complejos:

```
@app.route('/')
def home():
    perfil_usuario = {
        'nombre': 'Carlos',
        'email': 'carlos@example.com',
        'edad': 30
    }
    return render_template('perfil.html', perfil=perfil_
usuario)
```

Plantilla (perfil.html):

```
<h1>Perfil de {{ perfil.nombre }}</h1>
<p>Email: {{ perfil.email }}</p>
<p>Edad: {{ perfil.edad }}</p>
```

5. Visualización de Datos en la Web

Integrar gráficos en aplicaciones Flask permite visualizar datos de manera clara y efectiva. Usa bibliotecas como Matplotlib y Seaborn para gráficos en Python, y Chart.js o D3.js para gráficos interactivos en JavaScript.

Incorporación de Gráficos con Python

Uso de Matplotlib:

Matplotlib es una biblioteca potente para crear gráficos en Python. Puedes integrar estos gráficos en aplicaciones Flask utilizando buffers y codificación base64.

```
import matplotlib.pyplot as plt
import io
import base64
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def home():
    # Datos para el gráfico
    x = [1, 2, 3, 4, 5]
    y = [10, 14, 16, 20, 25]

    # Creación del gráfico
    plt.figure(figsize=(6, 4))
    plt.plot(x, y, marker='o', linestyle='-', color='b')
    plt.title('Crecimiento de Ventas')
    plt.xlabel('Mes')
    plt.ylabel('Ventas')

    # Guardar gráfico en un buffer
    buf = io.BytesIO()
    plt.savefig(buf, format='png')
    buf.seek(0)

    # Codificar imagen en base64
    imagen_base64 = base64.b64encode(buf.getvalue()).
decode('utf8')

    return render_template('grafico.html', imagen=ima-
gen_base64)

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

Plantilla (grafico.html):

<h1>Gráfico de Crecimiento de Ventas</h1>

<img src="{{ imagen }}" alt="Gráfico de Ventas">

BIBLIOGRAFÍA

- Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python (2nd ed.). O'Reilly Media.
- Dwyer, G. (2021). Flask by Example: Unleash the full potential of Flask to create Web Applications. Packt Publishing.
- Murray, S. (2017). Interactive Data Visualization for the Web: An Introduction to Designing with D3 (2nd ed.). O'Reilly Media.
- Nelli, F. (2018). Python Data Analytics: With Pandas, NumPy, and Matplotlib (2nd ed.). Apress.
- Meeks, E. (2017). D3.js in Action: Data Visualization with JavaScript (2nd ed.). Manning Publications.
- Zhu, N. Q. (2018). Data Visualization with D3 4.x Cookbook (2nd ed.). Packt Publishing.
- Swaroop, C. H. (2013). A Byte of Python. Self-published. https://python.swaroopch.com/

TIC

TALENTO TECH