🔧 Ejercicio Práctico: Detección de Formularios HTML y Campos de Entrada

📝 Descripción

El objetivo es construir un pequeño script en Python que, al conectarse a un sitio web, identifique y muestre los formularios presentes en la página junto con sus respectivos campos de entrada. Esto simula una etapa inicial de reconocimiento web automatizado, donde un pentester identifica potenciales puntos vulnerables para análisis posteriores.

🎯 Objetivos del ejercicio

- Realizar una solicitud HTTP a una página web.
- Analizar el contenido HTML utilizando una librería de parsing (BeautifulSoup).
- Extraer información de los formularios (`<form>`) y de sus campos (`<input>`, `<textarea>`, `<select>`).
- Mostrar los atributos relevantes: método, acción, y nombres de campos.
- Fomentar la observación de estructuras web como parte de la superficie de ataque.

👨‍💻 Instrucciones

1. Usa Python y la librería `BeautifulSoup` junto con `requests`.
2. Conéctate a una URL pública que contenga al menos un formulario (por ejemplo, https://httpbin.org/forms/post).
3. Extrae todos los formularios del DOM.
4. Por cada formulario, muestra:

- El valor del atributo `method` (GET/POST)
- El valor del atributo `action` (ruta de envío)
- Los nombres (`name`) y tipos (`type`) de todos los campos `<input>`.

5. Identifica y comenta qué formularios podrían ser críticos si no estuvieran validados.

💡 Ejemplo de salida esperada:

Formulario detectado:
- Método: POST
- Acción: /post
- Campos:
  * name: custname | type: text
  * name: custtel | type: tel
  * name: custemail| type: email
  * name: comments | type: textarea

🔐 Consideraciones Éticas

- Solo utilizar este tipo de reconocimiento en sitios públicos o con fines educativos.
- No enviar datos a formularios reales, solo leer su estructura.
- Este análisis es pasivo: no altera el contenido ni ejecuta acciones.

✅ Solución: `detectar_formularios.py`

```
import requests
from bs4 import BeautifulSoup

# URL de prueba con formularios públicos
url = "https://httpbin.org/forms/post"

# Realizar solicitud HTTP GET
respuesta = requests.get(url)

# Analizar el HTML con BeautifulSoup
soup = BeautifulSoup(respuesta.text, 'html.parser')
```

```
# Buscar todos los formularios
formularios = soup.find_all('form')

print(f"🔍 Formularios encontrados: {len(formularios)}\n")

# Analizar cada formulario
for idx, form in enumerate(formularios, start=1):
    metodo = form.get('method', 'GET').upper()
    accion = form.get('action', 'N/A')

    print(f"📄 Formulario #{idx}:")
    print(f"- Método: {metodo}")
    print(f"- Acción: {accion}")

    # Buscar todos los campos de entrada
    campos = form.find_all(['input', 'textarea', 'select'])
    print("- Campos:")
    for campo in campos:
        nombre = campo.get('name', '[sin nombre]')
        tipo = campo.get('type', 'text') if campo.name == 'input' else campo.name
        print(f"   * name: {nombre} | type: {tipo}")

    print("-" * 40)
```

🧪 Posible salida del script:

```
🔍 Formularios encontrados: 1

📄 Formulario #1:
- Método: POST
- Acción: /post
- Campos:
   * name: custname | type: text
   * name: custtel | type: tel
   * name: custemail | type: email
   * name: size | type: select
   * name: topping | type: checkbox
   * name: comments | type: textarea
   * name: delivery | type: radio
----------------------------------------
```

🔎 Reflexión técnica:

- **¿Qué campos podrían ser manipulados en un ataque?**: todos aquellos sin validación en backend, especialmente `email`, `comments` o campos ocultos.

- **¿Qué revela esta información?**: puntos de entrada al sistema, parámetros que podrían ser vulnerables a ataques como XSS o inyecciones.
