Web Scraping in Python - Requests & BeautifulSoup

1. response.text vs response.content

response.text:
- Type: str (decoded string)
- Use when scraping HTML, JSON, or XML
- Internally uses: response.content.decode(encoding)
- Ideal for parsing readable text (for BeautifulSoup, etc.)

response.content:
- Type: bytes (raw binary)
- Use when downloading non-text content (images, PDFs)
- Just returns raw bytes

2. Example: Using response.text

```
import requests
res = requests.get("https://example.com")
print(res.text[:500])  # Outputs HTML as a string
```

3. Example: Using response.content

```
import requests
res = requests.get("https://example.com/logo.png")
with open("logo.png", "wb") as f:
    f.write(res.content)
```

4. JSON APIs with .json()

```
import requests
url = "https://jsonplaceholder.typicode.com/users"
response = requests.get(url)
data = response.json()
for user in data:
    print(user['name'], user['email'])
```

5. Download PDF File

```
url = "https://www.w3.org/WAI/ER/tests/xhtml/testfiles/resources/pdf/dummy.pdf"
res = requests.get(url)
```

```
with open("sample.pdf", "wb") as f:
    f.write(res.content)
```

6. Download Image File

```
url = "https://via.placeholder.com/150"
res = requests.get(url)
with open("image.png", "wb") as f:
    f.write(res.content)
```

7. Handle Encoding Issues

```
res = requests.get("https://example.com")
res.encoding = "utf-8"  # or ISO-8859-1
html = res.text
```

8. Detect Encoding Automatically (chardet)

```
import chardet
raw_data = res.content
encoding = chardet.detect(raw_data)['encoding']
html = raw_data.decode(encoding)
```

9. Summary Table

Goal	Method	Output Format
Scrape HTML	response.text	str
Download file (PDF/img)	response.content	bytes
Work with JSON APIs	response.json()	Python dict/list
Handle encoding	res.encoding = ...	Affects .text