🕸️ Web Scraping for Beginners

📌 1. What is Web Scraping?

Web scraping is the process of extracting data from websites automatically, instead of copying it manually.

👉 Example use cases:

- Collecting product prices from e-commerce sites
- Gathering job postings from LinkedIn/Indeed
- Extracting news articles, reviews, or sports data
- Building datasets for machine learning

⚡ 2. Ethical Considerations

- Always check the website's robots.txt file.
- Scrape only public data.
- Respect rate limits → don't overload servers.
- Use data responsibly.

🛠️ 3. Tools & Libraries for Web Scraping

- **Python Requests** → Send HTTP requests, get web pages
- **BeautifulSoup (bs4)** → Parse HTML, extract data
- **lxml** → Fast HTML/XML parser
- **Selenium** → Automate browsers (for dynamic content)
- **Scrapy** → Full web scraping framework

👨‍💻 4. Basic Workflow of Web Scraping

1. **Send Request** → Get the webpage HTML
2. **Parse HTML** → Use a parser to read elements
3. **Extract Data** → Target specific tags, attributes
4. **Store Data** → Save in CSV, Excel, or database

🔎 5. Getting Started with Requests + BeautifulSoup

Install libraries:

```
# Install required libraries
pip install requests beautifulsoup4
```

Example: Scraping Quotes

```
import requests
from bs4 import BeautifulSoup

url = "http://quotes.toscrape.com/"
response = requests.get(url)

soup = BeautifulSoup(response.text, "html.parser")

quotes = soup.find_all("span", class_="text")
authors = soup.find_all("small", class_="author")

for q, a in zip(quotes, authors):
    print(f"{q.text} - {a.text}")
```

> ✅ Output: Extracts all quotes and their authors.

🎯 6. Extracting Specific Elements

Find one element:

```
title = soup.find("title").text
```

Find multiple elements:

```
links = soup.find_all("a")
for link in links:
    print(link.get("href"))
```

🌐 7. Scraping Dynamic Content (with Selenium)

Some websites load content using JavaScript. For those cases:

```
pip install selenium
```

```
from selenium import webdriver

driver = webdriver.Chrome()
driver.get("https://example.com")

html = driver.page_source
soup = BeautifulSoup(html, "html.parser")
```

📊 8. Storing Scraped Data

```
import csv

with open("quotes.csv", "w", newline="", encoding="utf-8") as f:
    writer = csv.writer(f)
    writer.writerow(["Quote", "Author"])
    for q, a in zip(quotes, authors):
        writer.writerow([q.text, a.text])
```

```
import pandas as pd

data = {"Quote": [q.text for q in quotes], "Author": [a.text for a in authors]}
df = pd.DataFrame(data)
df.to_csv("quotes.csv", index=False)
```

🚀 9. Beginner Project Ideas

- 📚 Book Data: Scrape top 100 books from Goodreads
- 💰 Price Tracking: Collect product prices (Amazon, Flipkart)
- 📰 News Headlines: Extract recent news headlines
- 💼 Job Listings: Create a dataset with job titles, companies, locations

📚 10. Next Steps

- Learn regular expressions for text extraction.
- Use APIs whenever possible.
- Explore Scrapy for large-scale scraping.
- Practice handling pagination & login-protected sites.

✨ **Web scraping is the foundation of the Data Science lifecycle. It helps collect raw data — the first step toward analysis and insights.**