🧠 WEB SCRAPING – STAGE 1

COMPLETE CODING + THINKING CHEATSHEET

0️⃣ MENTAL MODEL (MOST IMPORTANT)

Web scraping ka matlab:

> **Browser jo data dekh raha hai, wo data Python se direct server se lena.**

Golden flow (NEVER CHANGE THIS):

```
Website → DevTools → Parent → Children → Data → Structure → Save
```

1️⃣ WEBSITE ANALYSIS (BEFORE CODING)

1. Static vs Dynamic

- **Static** → HTML me data milta hai
- **Dynamic** → JS fetch / XHR / API se aata hai

quotes.toscrape.com = Static

👉 Isliye:

- `requests`
- `BeautifulSoup` kaam karega

2️⃣ DEVTOOLS THINKING (THE X-RAY)

Tum browser me kya karte ho:

1. Repeat hone wali cheez identify
2. Right-click → Inspect
3. Hover karke **poora ek record** pakadna
4. Parent confirm karna

Golden rule:

> **1 hover = 1 record = 1 parent**

Example:

```
<div class="quote">  ← parent
```

---

3 PROJECT SKELETON (MINIMUM)

```
quotes_scraper/
│
├── scraper.py
├── requirements.txt
├── quotes.json
```

---

4 REQUIRED LIBRARIES

```
requests
beautifulsoup4
lxml
```

---

5 BASIC REQUEST TEMPLATE (ALWAYS SAME)

```
import requests

URL = "https://quotes.toscrape.com/"

HEADERS = {
    "User-Agent": "Mozilla/5.0",
    "Accept-Language": "en-US,en;q=0.9",
}

response = requests.get(URL, headers=HEADERS)
```

Why headers?

- Identity dete hain
- 403 se bachate hain

---

6 HTML → DOM TREE

```
from bs4 import BeautifulSoup

soup = BeautifulSoup(response.text, "lxml")
```

Meaning:

- Raw HTML → searchable tree

---

7 RECORD EXTRACTION (CORE LOGIC)

Parent pakadna:

```
quotes = soup.find_all("div", class_="quote")
```

Rule:

- `find_all` → multiple records
- `find` → single child

---

8 CHILD EXTRACTION (INSIDE PARENT)

```
for quote in quotes:
    text = quote.find("span", class_="text").get_text()
    author = quote.find("small", class_="author").get_text()
    tags = quote.find_all("a", class_="tag")
    tag_texts = [tag.get_text() for tag in tags]
```

Important rules:

- **Parent ke andar hi search karo**
- Global search = fragile scraper ❌

---

9 DATA STRUCTURING (REAL SCRAPING)

Ek record = dictionary

```
quote_data = {
    "Text": text,
    "Author": author,
    "Tags": tag_texts
}
```

Sab records = list

```
all_quotes.append(quote_data)
```

Rule:

> **Print is temporary. Structure is permanent.**

🔟 PAGINATION (REAL-WORLD SKILL)

Browser discovery:

```
<li class="next">
    <a href="/page/2/">Next</a>
</li>
```

Pagination pattern:

```
current_url = URL

while True:
    response = requests.get(current_url, headers=HEADERS)
    soup = BeautifulSoup(response.text, "lxml")

    # scrape current page

    next_button = soup.find("li", class_="next")
    if next_button is None:
        break
```

```
    next_link = next_button.find("a")["href"]
    current_url = URL + next_link
```

Golden rule:

> **Request me hamesha `current_url` jaayega**

1️⃣1️⃣ FILE SAVING (JSON)

```
import json
from pathlib import Path

BASE_DIR = Path(__file__).parent

with open(BASE_DIR / "quotes.json", "w", encoding="utf-8") as f:
    json.dump(all_quotes, f, ensure_ascii=False, indent=4)
```

Why JSON first?

- API ready
- DB ready
- Debug friendly

1️⃣2️⃣ COMMON BUGS (YOU FACED THEM)

❌ Wrong variable

```
len(quote_data)  # ❌
len(all_quotes)  # ☑
```

❌ Printing HTML instead of text

```
print(quote)     # ❌
print(text)      # ☑
```

❌ Pagination bug

```
requests.get(URL)         # ❌
requests.get(current_url) # ☑
```

1️⃣3️⃣ REAL-WORLD ISSUES (NOT YOUR FAULT)

Network timeout

- Internet slow
- Server slow
- Happens daily

Solution mindset:

- `timeout=10`
- `time.sleep(1)`
- `try/except`

(Not fully implemented yet — Stage 2)

🧠 FINAL STAGE-1 THINKING SUMMARY

Tum ab ye soch paate ho:

- Data page nahi, **records hota hai**
- Browser = discovery tool
- Python = execution tool
- Parent decide karta hai scraper ki strength
- Structure > print
- Pagination is mandatory
- Failures are normal

👉 **This is scraper engineer thinking.**

🔓 WHAT WE DID *NOT* DO (INTENTIONALLY)

- ❌ Scraper class
- ❌ Pydantic model
- ❌ Async scraping

Because:

> **Architecture tab aata hai jab logic crystal clear ho**

Aur ab **logic crystal clear hai**.
