**You said:**

to continue working one WebScraperProject now we can start coding for next stage

**ChatGPT said:**

Great 👍 Let’s move WebScraperProject into the **next coding stage**.

Based on where we left off (project structure + repo setup), the next logical step is to **start implementing the scraper logic inside src/**.

Here’s a suggested progression for this stage:

**🔹 Stage 2: Core Scraping Engine**

We’ll focus on building a minimal but working scraper.

**1. Define entry point**

Inside src/, create a main.py that will act as the runner.

# src/main.py

from scraper.scraper import WebScraper

def run():

url = "https://example.com"

scraper = WebScraper(url)

data = scraper.scrape()

print(data)

if \_\_name\_\_ == "\_\_main\_\_":

run()

**2. Build the scraper class**

Inside src/scraper/scraper.py:

import requests

from bs4 import BeautifulSoup

class WebScraper:

def \_\_init\_\_(self, url: str):

self.url = url

def fetch\_html(self):

response = requests.get(self.url, headers={"User-Agent": "Mozilla/5.0"})

response.raise\_for\_status()

return response.text

def parse\_html(self, html: str):

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

# Example: extract all links

links = [a["href"] for a in soup.find\_all("a", href=True)]

return {"url": self.url, "links": links}

def scrape(self):

html = self.fetch\_html()

return self.parse\_html(html)

**3. Organize into modules**

Inside src/scraper/, you can later expand with:

* parser.py → Different parsing strategies
* utils.py → Helper functions
* storage.py → Saving scraped data

**4. Testing**

Inside tests/test\_scraper.py:

import pytest

from scraper.scraper import WebScraper

def test\_scraper\_links():

url = "https://example.com"

scraper = WebScraper(url)

data = scraper.scrape()

assert "links" in data

assert data["url"] == url

Run with:

pytest

👉 Next stages after this could be:

* Handling multiple pages / crawling.
* Exporting results (CSV, JSON, DB).
* Adding configs (YAML/JSON).
* CLI interface.

Do you want me to **start coding only the basic scraper (fetch + parse)** now, or also include **data storage (like saving scraped links to JSON/CSV)** in this stage?