Project 1: Web Scraping E-Commerce Flask App

📌 1. Project Title

Web Scraping E-Commerce Data with Flask Display

📝 2. Objective

To build a web application that scrapes data from an e-commerce website (BooksToScrape), stores the data in an Excel file, and presents it in a user-friendly interface using Flask with added search, pagination, and image display features.

🔍 3. Problem Statement

Manually browsing product data from e-commerce websites is time-consuming and repetitive. This project automates the data extraction process and displays it in an organized, searchable, and visual way, ideal for analysis or quick insights.

🧰 4. Tools and Technologies Used

Tool/Library	Purpose
Python	Core programming language
Flask	Web framework for UI
BeautifulSoup (bs4)	Web scraping
Requests	Sending HTTP requests
Pandas	Storing and exporting data
Matplotlib / Seaborn	Data visualization
HTML + CSS	Frontend (Jinja templating)

🧠 5. Project Description

The app scrapes all available books from `https://books.toscrape.com/`, extracting the following fields:

- **Title** 📖
- **Price** 💷
- **Rating** ⭐
- **Product Description** 📝
- **Cover Image** 🖼️

After scraping, it:

- Saves data into `books_data.xlsx`
- Displays top N books with **pagination**
- Enables **searching** by book title
- Shows book cover images (if available)
- Uses emojis to improve user experience

🔄 6. Working Mechanism

1. **Scraper Logic (`scraper.py`)**:
    - Uses BeautifulSoup to parse the website's HTML
    - Navigates multiple pages (pagination)
    - Extracts required fields for each book
    - Saves results in a Pandas DataFrame and Excel file

2. **Flask App (`app.py`)**:
    - Loads data from Excel
    - Serves the homepage with search and pagination
    - Uses `index.html` to display styled results with emojis and images

---

📂 7. Folder Structure

csharp
CopyEdit

```
project_1_WebScraping_Flask/
├── app.py                  # Flask backend
├── scraper.py              # Scraper logic
├── books_data.xlsx         # Output data file
├── templates/
│   └── index.html          # UI template
├── requirements.txt        # Python libraries
└── README.md               # Project description
```

⚙️ 8. How to Run the Project

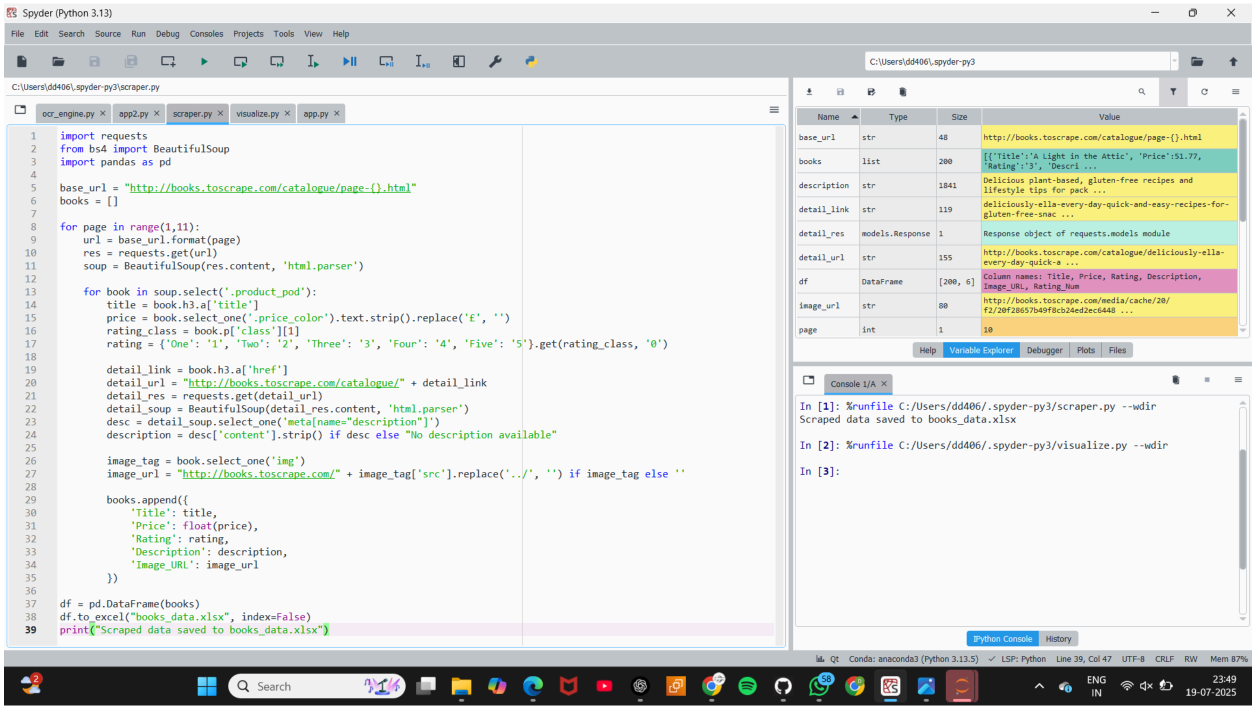
1. Clone the repository

```
https://github.com/dhanalakshmim-eng/cantilever.git

cd cantilever/WebScraping_Ecommerce
```

2. Install required libraries

```
pip install -r requirements.txt
```


Spyder (Python 3.13)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\dd406\.spyder-py3
C:\Users\dd406\.spyder-py3\scraper.py
ocr_engine.py app2.py scraper.py visualize.py app.py

```python
import requests
from bs4 import BeautifulSoup
import pandas as pd

base_url = "http://books.toscrape.com/catalogue/page-{}.html"
books = []

for page in range(1,11):
    url = base_url.format(page)
    res = requests.get(url)
    soup = BeautifulSoup(res.content, 'html.parser')

    for book in soup.select('.product_pod'):
        title = book.h3.a['title']
        price = book.select_one('.price_color').text.strip().replace('£', '')
        rating_class = book.p['class'][1]
        rating = {'One': '1', 'Two': '2', 'Three': '3', 'Four': '4', 'Five': '5'}.get(rating_class, '0')

        detail_link = book.h3.a['href']
        detail_url = "http://books.toscrape.com/catalogue/" + detail_link
        detail_res = requests.get(detail_url)
        detail_soup = BeautifulSoup(detail_res.content, 'html.parser')
        desc = detail_soup.select_one('meta[name="description"]')
        description = desc['content'].strip() if desc else "No description available"

        image_tag = book.select_one('img')
        image_url = "http://books.toscrape.com/" + image_tag['src'].replace('../', '') if image_tag else ''

        books.append({
            'Title': title,
            'Price': float(price),
            'Rating': rating,
            'Description': description,
            'Image_URL': image_url
        })

df = pd.DataFrame(books)
df.to_excel("books_data.xlsx", index=False)
print("Scraped data saved to books_data.xlsx")
```

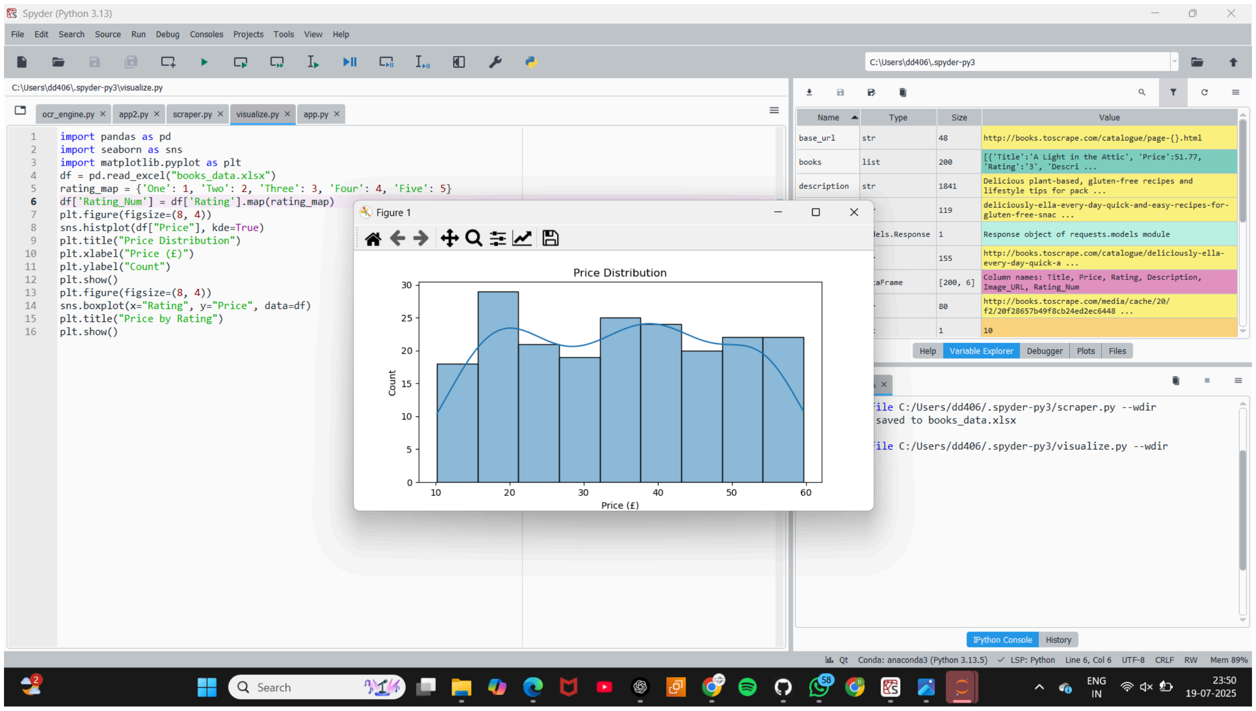
Name	Type	Size	Value
base_url	str	48	http://books.toscrape.com/catalogue/page-{}.html
books	list	200	[{'Title':'A Light in the Attic', 'Price':51.77, 'Rating':'3', 'Descri ...
description	str	1841	Delicious plant-based, gluten-free recipes and lifestyle tips for pack ...
detail_link	str	119	deliciously-ella-every-day-quick-and-easy-recipes-for-gluten-free-snac ...
detail_res	models.Response	1	Response object of requests.models module
detail_url	str	155	http://books.toscrape.com/catalogue/deliciously-ella-every-day-quick-a ...
df	DataFrame	[200, 6]	Column names: Title, Price, Rating, Description, Image_URL, Rating_Num
image_url	str	80	http://books.toscrape.com/media/cache/20/f2/20f28657b49f8cb24ed2ec6448 ...
page	int	1	10

```
In [1]: %runfile C:/Users/dd406/.spyder-py3/scraper.py --wdir
Scraped data saved to books_data.xlsx

In [2]: %runfile C:/Users/dd406/.spyder-py3/visualize.py --wdir

In [3]:
```

IPython Console History
Qt Conda: anaconda3 (Python 3.13.5) LSP: Python Line 39, Col 47 UTF-8 CRLF RW Mem 87%
Search
ENG IN 23:49 19-07-2025

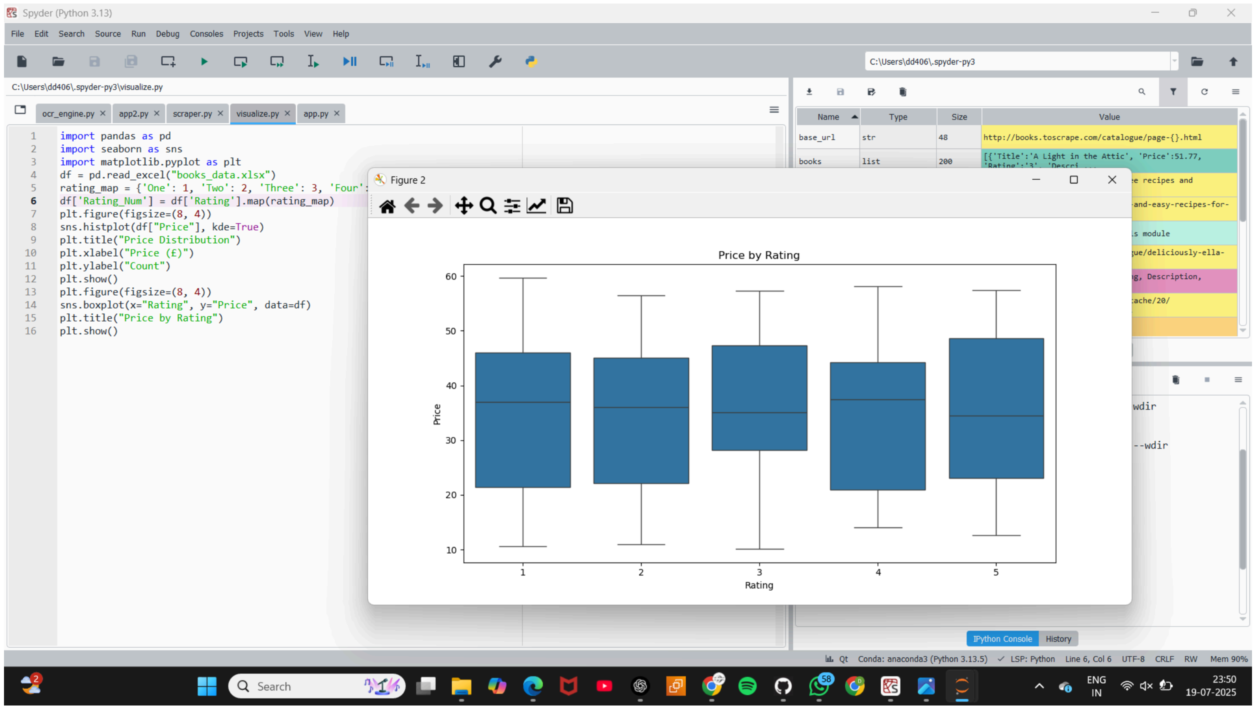

Spyder (Python 3.13)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\dd406\.spyder-py3
C:\Users\dd406\.spyder-py3\visualize.py
ocr_engine.py app2.py scraper.py visualize.py app.py

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_excel("books_data.xlsx")
rating_map = {'One': 1, 'Two': 2, 'Three': 3, 'Four': 4, 'Five': 5}
df['Rating_Num'] = df['Rating'].map(rating_map)
plt.figure(figsize=(8, 4))
sns.histplot(df["Price"], kde=True)
plt.title("Price Distribution")
plt.xlabel("Price (£)")
plt.ylabel("Count")
plt.show()
plt.figure(figsize=(8, 4))
sns.boxplot(x="Rating", y="Price", data=df)
plt.title("Price by Rating")
plt.show()
```

```
ile C:/Users/dd406/.spyder-py3/scraper.py --wdir
 saved to books_data.xlsx

ile C:/Users/dd406/.spyder-py3/visualize.py --wdir
```

IPython Console History
Qt Conda: anaconda3 (Python 3.13.5) LSP: Python Line 6, Col 6 UTF-8 CRLF RW Mem 89%
Search
ENG IN 23:50 19-07-2025

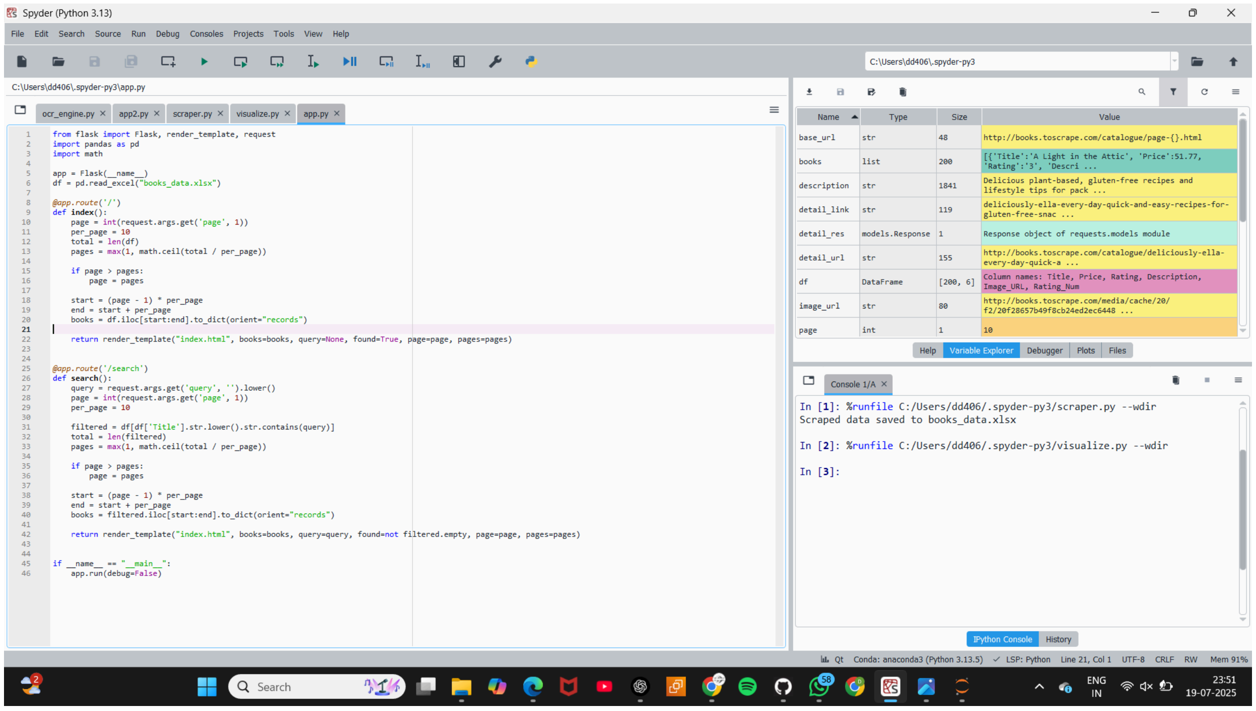


```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_excel("books_data.xlsx")
rating_map = {'One': 1, 'Two': 2, 'Three': 3, 'Four':
df['Rating_Num'] = df['Rating'].map(rating_map)
plt.figure(figsize=(8, 4))
sns.histplot(df["Price"], kde=True)
plt.title("Price Distribution")
plt.xlabel("Price (£)")
plt.ylabel("Count")
plt.show()
plt.figure(figsize=(8, 4))
sns.boxplot(x="Rating", y="Price", data=df)
plt.title("Price by Rating")
plt.show()
```

Spyder (Python 3.13)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\dd406\.spyder-py3\visualize.py
ocr_engine.py app2.py scraper.py visualize.py app.py
Figure 2
Price by Rating
Price
Rating
60 50 40 30 20 10
1 2 3 4 5
IPython Console History
Conda: anaconda3 (Python 3.13.5) LSP: Python Line 6, Col 6 UTF-8 CRLF RW Mem 90%
23:50 19-07-2025

3.Run the app

python app.py



```
from flask import Flask, render_template, request
import pandas as pd
import math

app = Flask(__name__)
df = pd.read_excel("books_data.xlsx")

@app.route('/')
def index():
    page = int(request.args.get('page', 1))
    per_page = 10
    total = len(df)
    pages = max(1, math.ceil(total / per_page))

    if page > pages:
        page = pages

    start = (page - 1) * per_page
    end = start + per_page
    books = df.iloc[start:end].to_dict(orient="records")

    return render_template("index.html", books=books, query=None, found=True, page=page, pages=pages)


@app.route('/search')
def search():
    query = request.args.get('query', '').lower()
    page = int(request.args.get('page', 1))
    per_page = 10

    filtered = df[df['Title'].str.lower().str.contains(query)]
    total = len(filtered)
    pages = max(1, math.ceil(total / per_page))

    if page > pages:
        page = pages

    start = (page - 1) * per_page
    end = start + per_page
    books = filtered.iloc[start:end].to_dict(orient="records")

    return render_template("index.html", books=books, query=query, found=not filtered.empty, page=page, pages=pages)


if __name__ == "__main__":
    app.run(debug=False)
```

Name	Type	Size	Value
base_url	str	48	http://books.toscrape.com/catalogue/page-{}.html
books	list	200	[{'Title':'A Light in the Attic', 'Price':51.77, 'Rating':'3', 'Descri ...
description	str	1841	Delicious plant-based, gluten-free recipes and lifestyle tips for pack ...
detail_link	str	119	deliciously-ella-every-day-quick-and-easy-recipes-for-gluten-free-snac ...
detail_res	models.Response	1	Response object of requests.models module
detail_url	str	155	http://books.toscrape.com/catalogue/deliciously-ella-every-day-quick-a ...
df	DataFrame	[200, 6]	Column names: Title, Price, Rating, Description, Image_URL, Rating_Num
image_url	str	80	http://books.toscrape.com/media/cache/20/f2/20f28657b49f8cb24ed2ec6448 ...
page	int	1	10

```
In [1]: %runfile C:/Users/dd406/.spyder-py3/scraper.py --wdir
Scraped data saved to books_data.xlsx

In [2]: %runfile C:/Users/dd406/.spyder-py3/visualize.py --wdir

In [3]:
```

Spyder (Python 3.13)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\dd406\.spyder-py3\app.py
ocr_engine.py app2.py scraper.py visualize.py app.py
C:\Users\dd406\.spyder-py3
Help Variable Explorer Debugger Plots Files
Console 1/A
IPython Console History
Conda: anaconda3 (Python 3.13.5) LSP: Python Line 21, Col 1 UTF-8 CRLF RW Mem 91%
23:51 19-07-2025

4.Visit in browser:
`http://localhost:5000/`

🎥 `Demo`

`[Click here to watch the demo video]`

`https://drive.google.com/file/d/1kxsY9_9gxRVDpgiRWcEpgcEePxSf_AZO/view?usp=sharing`

📈 9. Optional Visualizations (Extension)

- Use Matplotlib/Seaborn to visualize:
  - Distribution of ratings
  - Price ranges
  - Most frequent words in book titles/descriptions

✅ 10. Key Highlights

- Practical real-world scraping practice
- Excel output for analysis
- Clean and responsive UI
- Basic pagination and search
- Fully local project (no external APIs used)

🛡️ 11. Limitations

- Site structure-dependent (changes may break scraper)
- Scraping limited to open-access pages
- No database backend (can be extended)

💡 12. Future Enhancements

- Integrate SQLite or MongoDB to store data persistently
- Add sorting filters (e.g., price, rating)
- Auto-refresh scraper to get new books weekly
- Add charts and dashboards for visual analytics

👨‍💻 13. Author

Dhana Lakshmi M
B.E. Computer Science
Email: dd406652dhana@gmail.com

GitHub: https://github.com/dhanalakshmim-eng/cantilever.git