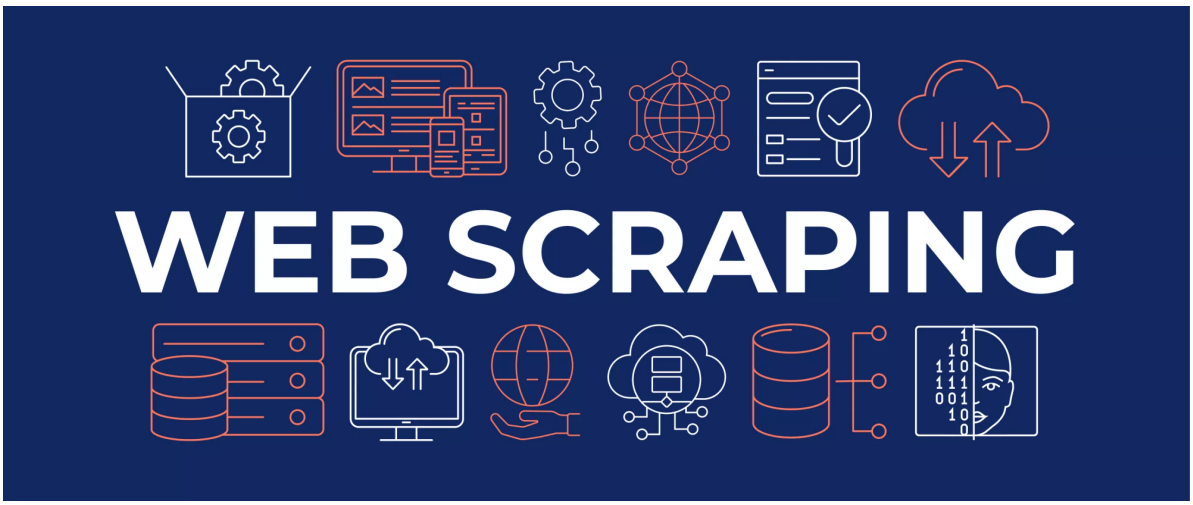
Web Scraping Of Online Bookstore & MongoDB With Python

Context:

The data for this project was scraped from the 'Books to Scrape' website using Python. The scraping was done across the 50 pages of the website.

Aim:

The aim of this project is to scrape data from a website of about 50 pages, transform the data into a CSV file, insert the CSV file into a database using pymongo of MongoDB, and thereafter, retrieve the collection from the databse and read it into a DataFrame which can be analysed, manipulated, and processed for Data Science works.



Importation of the necessary module for the project

In [20]:
```
import pandas as pd
import requests
from bs4 import BeautifulSoup
```

Let's get the url address of the fisrt page of the website we wish to scrape data from:

In [21]:
```
url = "http://books.toscrape.com/catalogue/page-1.html"
```

Let's get a response from the address and then obtain the content of the response. We can thereafter, parse it into html using BeautifulSoup

In [22]:
```
response = requests.get(url)
response = response.content
soup = BeautifulSoup(response, 'html.parser')
```

Let's find the ordinary list(ol) that contains the article in which the contents of the data we wish to scrape are contained in.

In [23]:
```
ol = soup.find('ol')
articles = ol.find_all('article', class_ = 'product_pod')
```

Now, let's performthe above process across the 50 pages of the website starting from the first page.

In [24]:
```
books = []
for i in range(1,51):
    url = f"http://books.toscrape.com/catalogue/page-{i}.html"
    response = requests.get(url)
    response = response.content
    soup = BeautifulSoup(response, 'html.parser')
    ol = soup.find('ol')
    articles = ol.find_all('article', class_ = 'product_pod')


    for article in articles:
        image = article.find('img')
        title = image.attrs['alt']
        star_rating = article.find('p')
        star_rating = star_rating['class'][1]
        price = article.find('p', class_ = 'price_color').text
        price = float(price[1:])
        books.append([title, price, star_rating])
```

Having scrapped our desired date from the website pages, let's now create a pandas DataFrame from it, and thereafter, make a csv file from it.

In [25]:
```
df = pd.DataFrame(books, columns=["Title", "Price", "Star Rating"])
df.to_csv("Books.csv")
```

In [ ]:
```

```

In [26]:
```
#let's read the csv file and get a view of it
books_df = pd.read_csv("Desktop/CSV Files/Books.csv")
books_df.head()
```

Out[26]:

	Unnamed: 0	Title	Price	Star Rating
0	0	A Light in the Attic	51.77	Three
1	1	Tipping the Velvet	53.74	One
2	2	Soumission	50.10	One
3	3	Sharp Objects	47.82	Four
4	4	Sapiens: A Brief History of Humankind	54.23	Five

In [ ]:
```
```

Inserting CSV file into MongoDB

Let's import the libraries we will use to insert the Books.csv file into MongoDB

In [29]:
```
import pandas as pd
import pymongo
import json
```

In [30]:
```
#let's create a client
client = pymongo.MongoClient('mongodb://localhost:27017')
```

In [31]:
```
#let's read our csv file, view the first the fisrt 5 rows and the shape of
books_df = pd.read_csv('Books.csv')
print(f'The shape of books_df is:{books_df.shape}')
books_df.head()
```

```
The shape of books_df is:(1000, 4)
```

Out[31]:

	Unnamed: 0	Title	Price	Star Rating
0	0	A Light in the Attic	51.77	Three
1	1	Tipping the Velvet	53.74	One
2	2	Soumission	50.10	One
3	3	Sharp Objects	47.82	Four
4	4	Sapiens: A Brief History of Humankind	54.23	Five

In [32]:
```
#let's drop the unimportant column
books_df.drop(columns='Unnamed: 0', inplace=True)
books_df.tail()
```

Out[32]:

	Title	Price	Star Rating
995	Alice in Wonderland (Alice's Adventures in Won...	55.53	One
996	Ajin: Demi-Human, Volume 1 (Ajin: Demi-Human #1)	57.06	Four
997	A Spy's Devotion (The Regency Spies of London #1)	16.97	Five
998	1st to Die (Women's Murder Club #1)	53.98	One
999	1,000 Places to See Before You Die	26.08	Five

In [33]:
```
#rename the Star Rating column
books_df.rename(columns = {'Star Rating':'Star_rating'}, inplace=True)
books_df.head()
```

Out[33]:

	Title	Price	Star_rating
0	A Light in the Attic	51.77	Three
1	Tipping the Velvet	53.74	One
2	Soumission	50.10	One
3	Sharp Objects	47.82	Four
4	Sapiens: A Brief History of Humankind	54.23	Five

In [ ]:
```

```

Let's read our dataframe into a json format which is the format of storing data in MongoDB:

In [34]:
```
data = books_df.to_dict(orient = 'records')
print(data)
```

```
[{'Title': 'A Light in the Attic', 'Price': 51.77, 'Star_rating': 'Three
'}, {'Title': 'Tipping the Velvet', 'Price': 53.74, 'Star_rating': 'One
'}, {'Title': 'Soumission', 'Price': 50.1, 'Star_rating': 'One'}, {'Title
': 'Sharp Objects', 'Price': 47.82, 'Star_rating': 'Four'}, {'Title': 'Sa
piens: A Brief History of Humankind', 'Price': 54.23, 'Star_rating': 'Fiv
e'}, {'Title': 'The Requiem Red', 'Price': 22.65, 'Star_rating': 'One'},
{'Title': 'The Dirty Little Secrets of Getting Your Dream Job', 'Price':
33.34, 'Star_rating': 'Four'}, {'Title': 'The Coming Woman: A Novel Based
on the Life of the Infamous Feminist, Victoria Woodhull', 'Price': 17.93,
'Star_rating': 'Three'}, {'Title': 'The Boys in the Boat: Nine Americans
and Their Epic Quest for Gold at the 1936 Berlin Olympics', 'Price': 22.
6, 'Star_rating': 'Four'}, {'Title': 'The Black Maria', 'Price': 52.15, '
Star_rating': 'One'}, {'Title': 'Starving Hearts (Triangular Trade Trilog
y, #1)', 'Price': 13.99, 'Star_rating': 'Two'}, {'Title': "Shakespeare's
Sonnets", 'Price': 20.66, 'Star_rating': 'Four'}, {'Title': 'Set Me Free
', 'Price': 17.46, 'Star_rating': 'Five'}, {'Title': "Scott Pilgrim's Pre
cious Little Life (Scott Pilgrim #1)", 'Price': 52.29, 'Star_rating': 'Fi
ve'}, {'Title': 'Rip it Up and Start Again', 'Price': 35.02, 'Star_rating
': 'Five'}, {'Title': 'Our Band Could Be Your Life: Scenes from the Ameri
```

We can see now that our data is now stored in dictionary(json) format.

Let's create a database into which we will store our data

In [35]:
```
db = client['WebScraping']
print(db)
```

```
Database(MongoClient(host=['localhost:27017'], document_class=dict, tz_aw
are=False, connect=True), 'WebScraping')
```

In [ ]:

Let's insert our collection(table) into our newly created database:

In [36]:
```
db.Books.insert_many(data)
```

Out[36]:
```
<pymongo.results.InsertManyResult at 0x298194aa6a0>
```

Conclusion: We have successfully created our database and collection with python using pymongo

In [ ]:

Load Collection(table) From MongoDB To Python

In [41]:
```
#let's intall and instantiate PrettyPrinter
from pprint import PrettyPrinter
pp = PrettyPrinter(indent=2)
```

In [51]:
```
#let's print our database
pp.pprint(list(client.list_databases()))
```

```
[ {'empty': False, 'name': 'WebScraping', 'sizeOnDisk': 94208},
  {'empty': False, 'name': 'admin', 'sizeOnDisk': 40960},
  {'empty': False, 'name': 'config', 'sizeOnDisk': 110592},
  {'empty': False, 'name': 'local', 'sizeOnDisk': 81920},
  {'empty': False, 'name': 'players', 'sizeOnDisk': 81920}]
```

In [53]:
```
#let's pull our database
db = client['WebScraping']

#print the list of collection available in db
for c in db.list_collections():
    print(c['name'])
```

```
Books
```

In [54]:
```
#let define our collection
mycollection = db['Books']
mycollection
```

Out[54]:
```
Collection(Database(MongoClient(host=['localhost:27017'], document_class=
dict, tz_aware=False, connect=True), 'WebScraping'), 'Books')
```

In [55]:
```
#let's see how many documents are in mycollection
mycollection.count_documents({})
```

Out[55]:
```
1000
```

In [56]:
```
#let's print our first record
first_record = mycollection.find_one({})
pp.pprint(first_record)
```

```
{ 'Price': 51.77,
  'Star_rating': 'Three',
  'Title': 'A Light in the Attic',
  '_id': ObjectId('6495541fa0480a0e669ef5eb')}
```

In [57]:
```
#let's print all records
all_records = mycollection.find({})
pp.pprint(all_records)
```

```
<pymongo.cursor.Cursor object at 0x000002981BEF0430>
```

In [58]:
```
#To be able to access all the records, we will have to run a for loop on t
for row in all_records:
    pp.pprint(row)
```

```
{ 'Price': 51.77,
  'Star_rating': 'Three',
  'Title': 'A Light in the Attic',
  '_id': ObjectId('6495541fa0480a0e669ef5eb')}
{ 'Price': 53.74,
  'Star_rating': 'One',
  'Title': 'Tipping the Velvet',
  '_id': ObjectId('6495541fa0480a0e669ef5ec')}
{ 'Price': 50.1,
  'Star_rating': 'One',
  'Title': 'Soumission',
  '_id': ObjectId('6495541fa0480a0e669ef5ed')}
{ 'Price': 47.82,
  'Star_rating': 'Four',
  'Title': 'Sharp Objects',
  '_id': ObjectId('6495541fa0480a0e669ef5ee')}
{ 'Price': 54.23,
  'Star_rating': 'Five',
  'Title': 'Sapiens: A Brief History of Humankind',
```

COMMENT: Notice how prettier the output showing our records is on printing it out with PrettyPrinter

In [ ]:
```
```

Let's convert our data from dictionary format into a DataFrame:

In [59]:
```
#let's create another cursor object having exhausted the previous one with
all_records = mycollection.find({})
```

In [62]:
```
#let's pass all_records into a list
list_all_records = list(all_records)
```

In [64]:
```
books_df = pd.DataFrame(list_all_records)
books_df.head()
```

Out[64]:

	_id	Title	Price	Star_rating
0	6495541fa0480a0e669ef5eb	A Light in the Attic	51.77	Three
1	6495541fa0480a0e669ef5ec	Tipping the Velvet	53.74	One
2	6495541fa0480a0e669ef5ed	Soumission	50.10	One
3	6495541fa0480a0e669ef5ee	Sharp Objects	47.82	Four
4	6495541fa0480a0e669ef5ef	Sapiens: A Brief History of Humankind	54.23	Five

In [66]:
```
books_df.tail()
```

Out[66]:

	_id	Title	Price	Star_rating
995	6495541fa0480a0e669ef9ce	Alice in Wonderland (Alice's Adventures in Won...	55.53	One
996	6495541fa0480a0e669ef9cf	Ajin: Demi-Human, Volume 1 (Ajin: Demi-Human #1)	57.06	Four
997	6495541fa0480a0e669ef9d0	A Spy's Devotion (The Regency Spies of London #1)	16.97	Five
998	6495541fa0480a0e669ef9d1	1st to Die (Women's Murder Club #1)	53.98	One
999	6495541fa0480a0e669ef9d2	1,000 Places to See Before You Die	26.08	Five

In [67]:
```
books_df.shape
```

Out[67]: (1000, 4)

COMMENT: We now have 1000 row and 4 columns of dataset, however, one can easily drop the _id column.

In [ ]:
```

```

The End. Thank you!!!

In [ ]:
```

```