```python
from bs4 import BeautifulSoup
import urllib.request, urllib.error, urllib.parse
import re
import pandas as pd
from urllib.parse import urljoin




global all_books
all_books = []

def parse_book(url):
    #url = 'https://books.toscrape.com/catalogue/tipping-the-velvet_999/index.html'

    str = urllib.request.urlopen(url).read().decode()
    str = str.strip()
    soup = BeautifulSoup(str, 'html.parser')

    #name = soup.find('body').find('div',class_ = 'container-fluid page').find('div',class_ = 'page_inner').find('div',class_ = 'content').find('h1').text.strip()
    name = soup.find('div',class_ = 'content').find('h1').text.strip()

    stock = soup.find('p',class_ = 'instock availability').text.strip()
    stock = re.search(r'[0-9]+',stock).group()

    rating = soup.find('p',class_='star-rating')['class'][1]
    rnum = {
        "One": 1, "Two": 2, "Three": 3, "Four": 4, "Five": 5
    }
    rating = rnum.get(rating,0)

    genre = soup.find('ul',class_='breadcrumb').find('a',href = re.compile(r'category/books/')).text

    p_items = dict()
    p_items['Name']=name
    p_items['Stock']=stock
    p_items['Rating']=rating
    p_items['Genre']=genre
    t_items = ['UPC','Product Type','Price (incl. tax)','Price (excl. tax)','Tax','Number of reviews']
    def tags(tag):
        th_tag =soup.find('th',string=tag)

        if (th_tag):
            td =th_tag.find_next_sibling('td').text.strip()
        else :
```

```python
def parse_book(url):
    p_items['Genre']=genre
    t_items = ['UPC','Product Type','Price (incl. tax)','Price (excl. tax)','Tax','Number of reviews']
    def tags(tag):
        th_tag =soup.find('th',string=tag)

        if (th_tag):
            td =th_tag.find_next_sibling('td').text.strip()
        else :
            td = None
        return td

    for item in t_items:
        p_items[item] = tags(item)
    print(p_items)
    all_books.append(p_items)
    print(len(all_books))

def home_page(url):

    #url = 'https://books.toscrape.com/catalogue/category/books/travel_2/index.html'

    str = urllib.request.urlopen(url).read().decode()
    str = str.strip()
    soup = BeautifulSoup(str, 'html.parser')

    genres = soup.find('ul',class_="nav nav-list").find_all('a')

    links = list()
    for genre in genres:
        link = 'https://books.toscrape.com/' + genre['href']
        links.append(link)

    links = links[1:] # Removing the home link

    for link in links:
        book_link(link)



def book_link(url):
    while True:
        str_ = urllib.request.urlopen(url).read().decode().strip()
        soup = BeautifulSoup(str_, 'html.parser')

        # Find all book links on the current page
        book_tags = soup.select('h3 a')
        for tag in book_tags:
            book_url = urljoin(url, tag['href'])
            try:
                parse_book(book_url)
            except:
                print(book_url, 'Not found')

        # Check for next page
        next_btn = soup.find('li', class_='next')
        if next_btn:
            next_link = next_btn.find('a')['href']
            url = urljoin(url, next_link)
        else:
            break
```

Name	Stock	Rating	Genre	Price
The Perfect Play	4	3	Romance	£59.99
Last One Home	5	3	Fiction	£59.98
Civilization and Its Discontents	3	2	Psychology	£59.95
The Barefoot Contessa Cookbook	6	5	Food and Drink	£59.92
The Diary of a Young Girl	12	3	Nonfiction	£59.90
The Bone Hunters	4	3	Thriller	£59.71
Thomas Jefferson and the Tripoli Pirates	15	1	History	£59.64
Boar Island	12	3	Mystery	£59.48
The Improbability of Love	11	1	Fiction	£59.45
The Man Who Mistook His Wife for a Hat and Other Clinical Tales	6	4	Nonfiction	£59.45
The Gray Rhino	12	4	Add a comment	£59.15
Life Without a Recipe	3	5	Autobiography	£59.04
Listen to Me	1	3	Romance	£58.99
Unlimited Intuition Now	9	4	Default	£58.87
Approval Junkie	5	5	Autobiography	£58.81
Hamilton	14	3	Nonfiction	£58.79
Myriad	1	4	Fantasy	£58.75
The Rose & the Dagger	3	4	Fantasy	£58.64
Candide	4	3	Classics	£58.63
Alight	5	4	Default	£58.59
Catherine the Great	6	4	History	£58.55
Miller's Valley	8	2	Fiction	£58.54
Shameless	1	3	New Adult	£58.35
Silence in the Dark	8	3	Suspense	£58.33
How to Speak Golf	12	5	Default	£58.32
Aristotle and Dante Discover the Secrets of the Universe	14	4	Young Adult	£58.14
The Death of Humanity	16	4	Philosophy	£58.11
Unstuffed	3	1	Nonfiction	£58.09

Row Labels	Sum of Stock	Average of Rating
Academic	5	2.000
Add a comment	516	2.761
Adult Fiction	3	5.000
Art	72	3.625
Autobiography	57	3.000
Biography	43	2.200
Business	133	2.917
Childrens	229	2.621
Christian	37	2.667
Christian Fiction	55	4.167
Classics	55	2.474
Contemporary	33	2.333
Crime	15	1.000
Cultural	15	1.000
Default	1345	2.836
Erotica	15	5.000
Fantasy	372	3.083
Fiction	588	3.185
Food and Drink	319	2.900
Health	49	3.750
Historical	20	3.000
Historical Fiction	194	3.231
History	181	2.944
Horror	136	2.706
Humor	84	3.400
Music	111	3.154

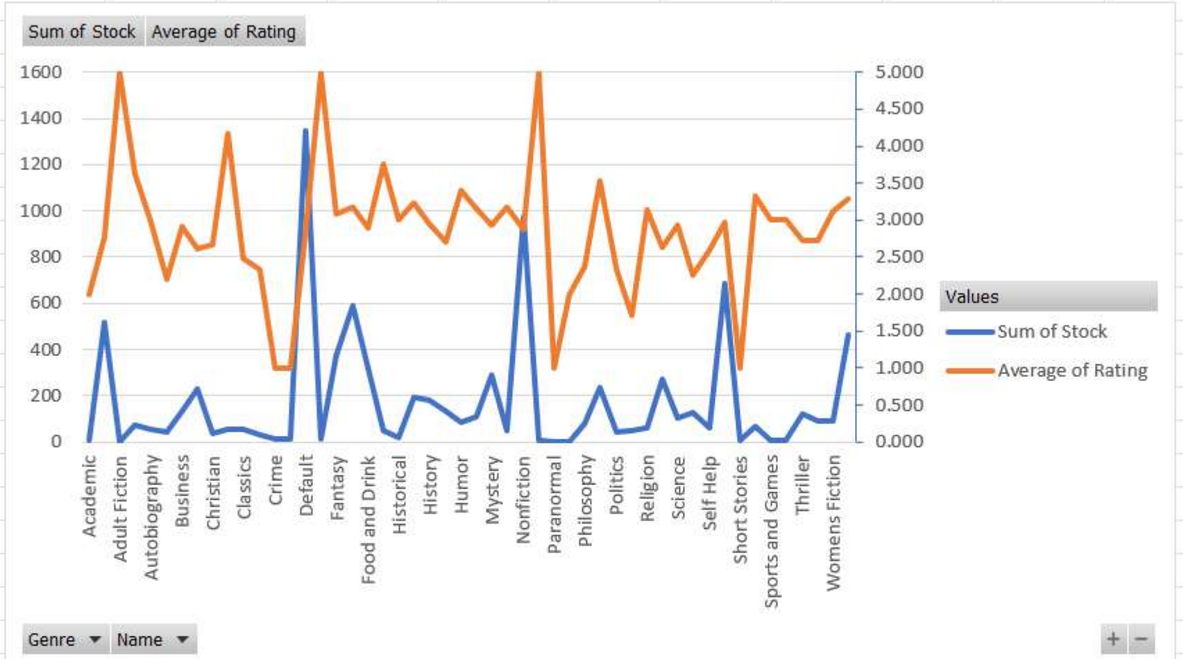
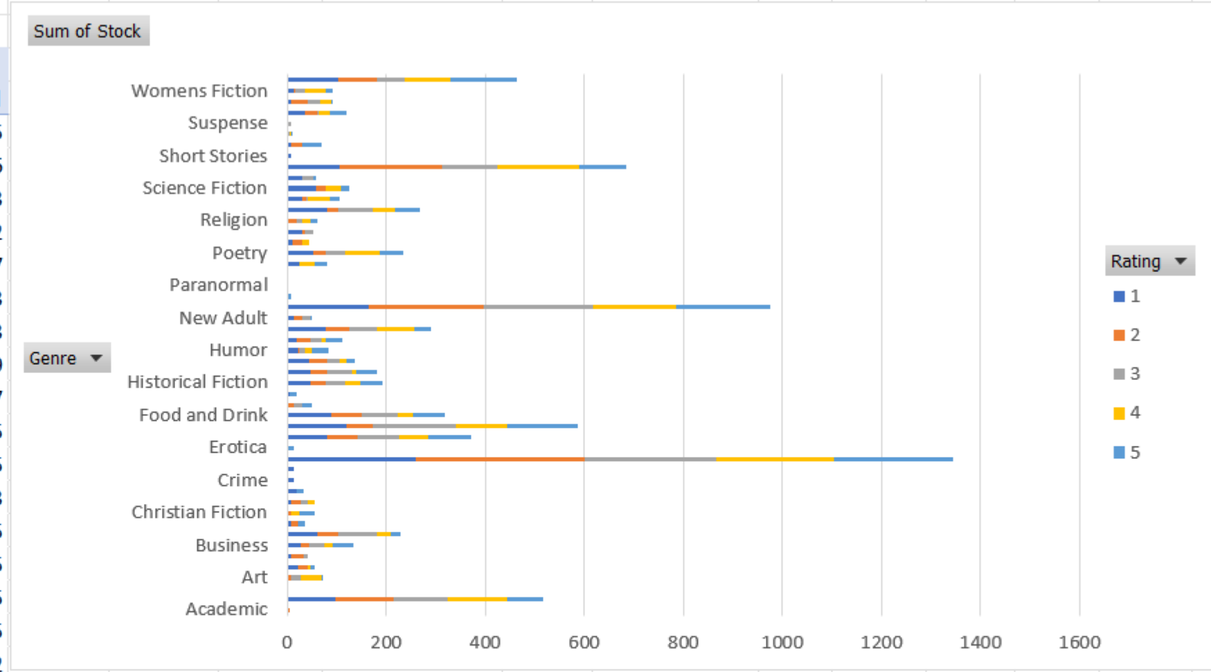


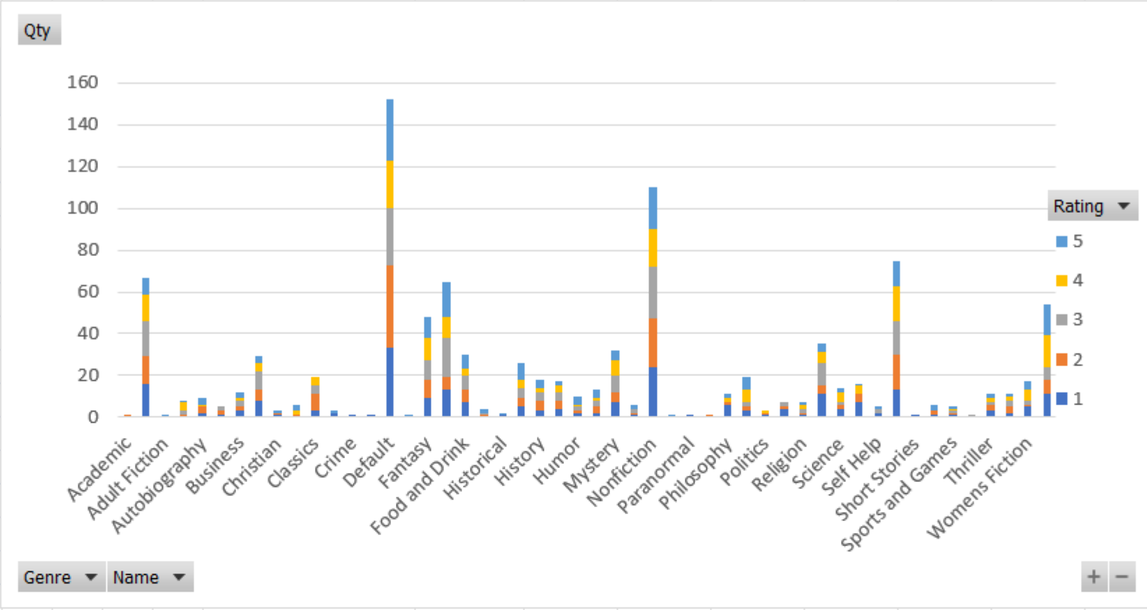
Chart Area

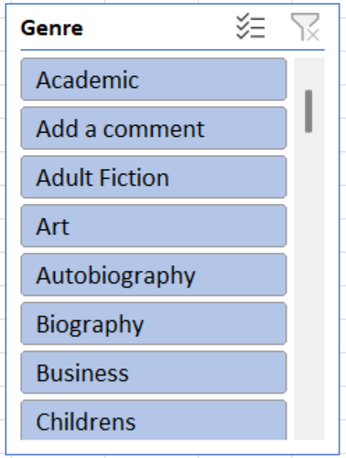
cleaned_books_chart | Stock_by_rating | Rating_by_Stock | Rating_by_Qty | Dashboard

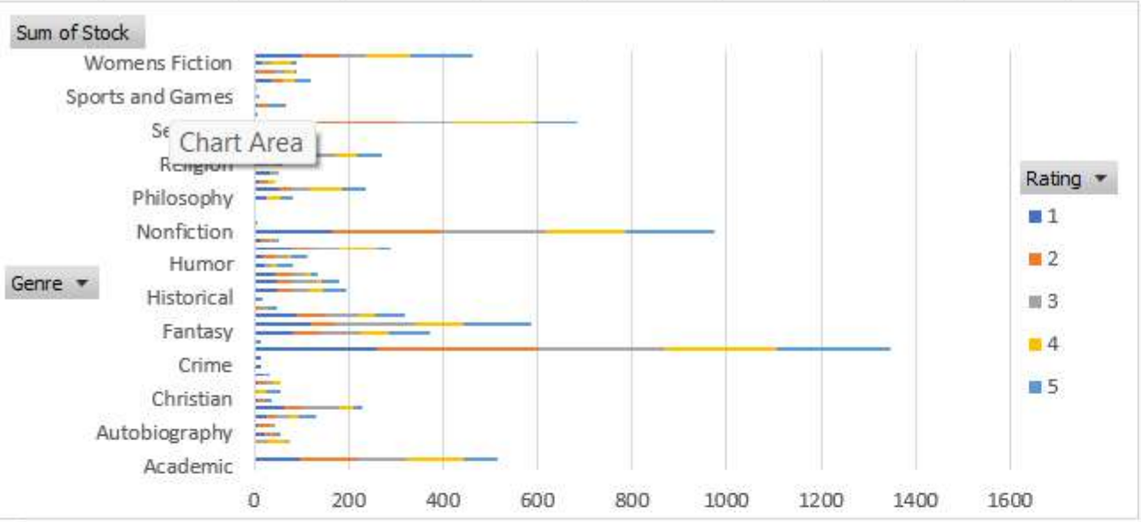
Sum of Stock	Column Labels					
Row Labels	1	2	3	4	5	Grand Total
Academic		5				5
Add a comment	99	117	107	121	72	516
Adult Fiction					3	3
Art		7	22	42	1	72
Autobiography	22	21		3	11	57
Biography	7	26	10			43
Business	27	18	29	19	40	133
Childrens	62	41	79	27	20	229
Christian	7	14			16	37
Christian Fiction		7		18	30	55
Classics	7	20	15	13		55
Contemporary	19				14	33
Crime	15					15
Cultural	15					15
Default	260	341	266	238	240	1345
Erotica					15	15
Fantasy	82	61	82	60	87	372
Fiction	119	54	168	102	145	588
Food and Drink	90	62	71	32	64	319
Health		14	16		19	49
Historical	6				14	20
Historical Fiction	48	29	39	32	46	194
History	47	33	50	11	40	181
Horror	44	37	26	12	17	136
Humor	21	3	12	14	34	84

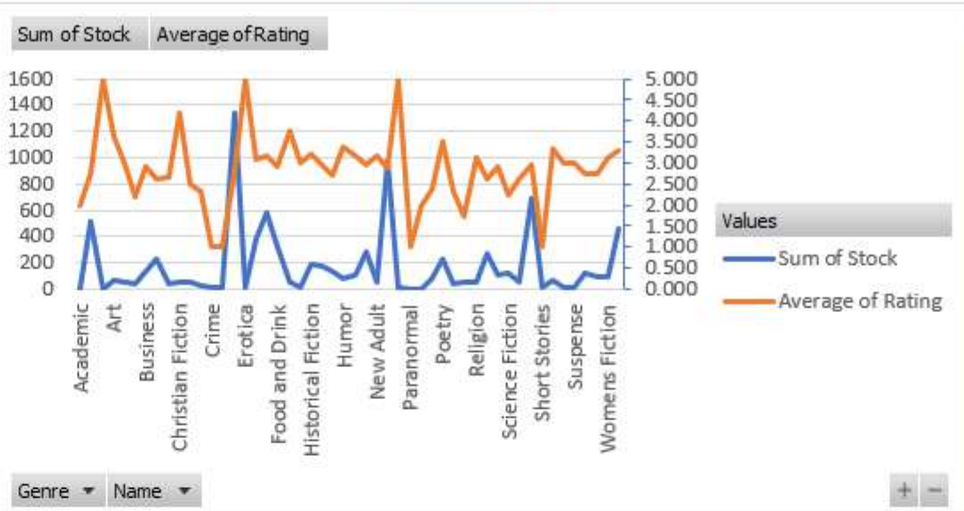


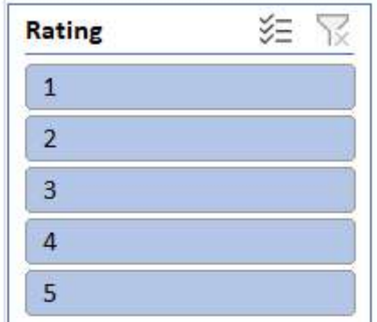
Qty	Column Labels					
Row Labels	1	2	3	4	5	Grand Total
Academic		1				1
Add a comment	16	13	17	13	8	67
Adult Fiction					1	1
Art		1	2	4	1	8
Autobiography	2	3		1	3	9
Biography	1	2	2			5
Business	3	2	3	1	3	12
Childrens	8	5	9	4	3	29
Christian	1	1			1	3
Christian Fiction		1		2	3	6
Classics	3	8	4	4		19
Contemporary	2				1	3
Crime	1					1
Cultural	1					1
Default	33	40	27	23	29	152
Erotica					1	1
Fantasy	9	9	9	11	10	48
Fiction	13	6	19	10	17	65
Food and Drink	7	6	7	3	7	30
Health		1	1		2	4
Historical	1				1	2
Historical Fiction	5	4	5	4	8	26
History	3	5	4	2	4	18
Horror	4	4	4	3	2	17
Humor	2	1	2	1	4	10



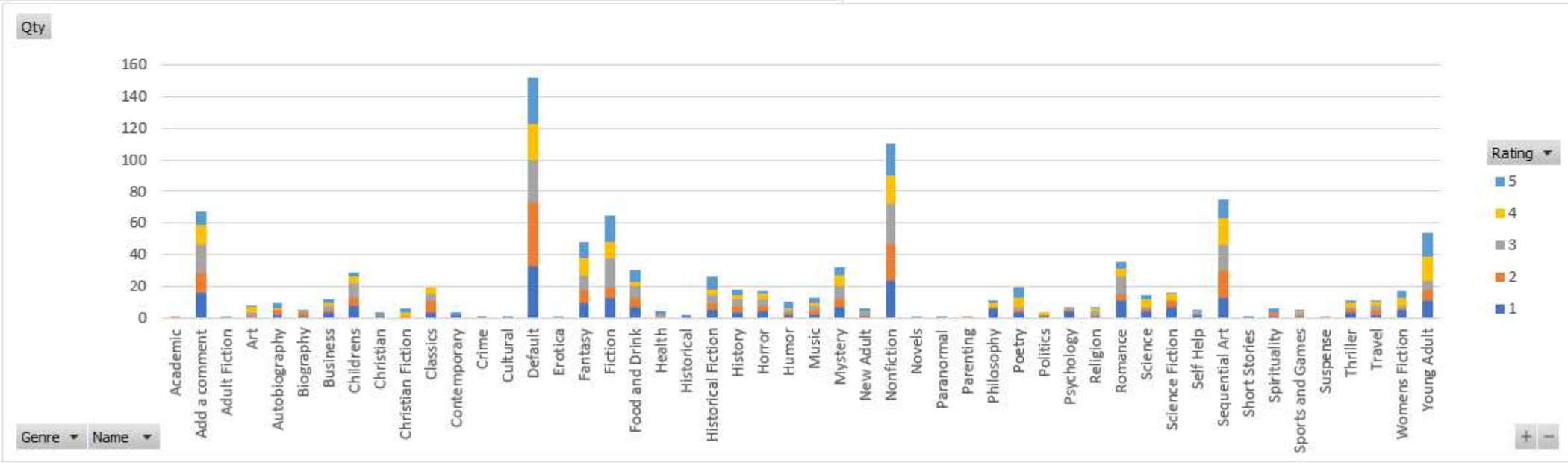



Sum of Stock
Womens Fiction
Sports and Games
Chart Area
Philosophy
Nonfiction
Humor
Historical
Fantasy
Crime
Christian
Autobiography
Academic
Genre
Rating
1
2
3
4
5
0
200
400
600
800
1000
1200
1400
1600


Sum of Stock
Average of Rating
1600
1400
1200
1000
800
600
400
200
0
5.000
4.500
4.000
3.500
3.000
2.500
2.000
1.500
1.000
0.500
0.000
Academic
Art
Business
Christian Fiction
Crime
Erotica
Food and Drink
Historical Fiction
Humor
New Adult
Paranormal
Poetry
Religion
Science Fiction
Short Stories
Suspense
Womens Fiction
Values
Sum of Stock
Average of Rating
Genre
Name


Rating
1
2
3
4
5


Genre
Academic
Add a comment
Adult Fiction
Art
Autobiography
Biography
Business
Childrens
Christian
Christian Fiction
Classics
Contemporary
Crime
Cultural


Qty
160
140
120
100
80
60
40
20
0
Academic
Add a comment
Adult Fiction
Art
Autobiography
Biography
Business
Childrens
Christian
Christian Fiction
Classics
Contemporary
Crime
Cultural
Default
Erotica
Fantasy
Fiction
Food and Drink
Health
Historical
Historical Fiction
History
Horror
Humor
Music
Mystery
New Adult
Nonfiction
Novels
Paranormal
Parenting
Philosophy
Poetry
Politics
Psychology
Religion
Romance
Science
Science Fiction
Self Help
Sequential Art
Short Stories
Spirituality
Sports and Games
Suspense
Thriller
Travel
Womens Fiction
Young Adult
Rating
5
4
3
2
1
Genre
Name

Page No. _____
Date

Project - 01 (DA-01) :-

→ To scrape "books.toscrape.com", prepare an excel sheet & derive conclusions :-

i) Price v/s rating
ii) Distribution of Star Ratings
iii) Stock Availability

- **features** :-

  - Extracts all books across all categories.
  - Collects detailed meta data :-
    - Name
    - Stock
    - Rating
    - UPC
    - Product type
    - Price (exc. tax)
    - Price (inc. tax)
    - Tax
    - No. of reviews
    - Genre
  - Handles pagination & Save it in CSV.

Spiral

Page No.
Date

How it works ??

1. **Home Page Parsing** :- Takes homepage url & give links for all genere's page.

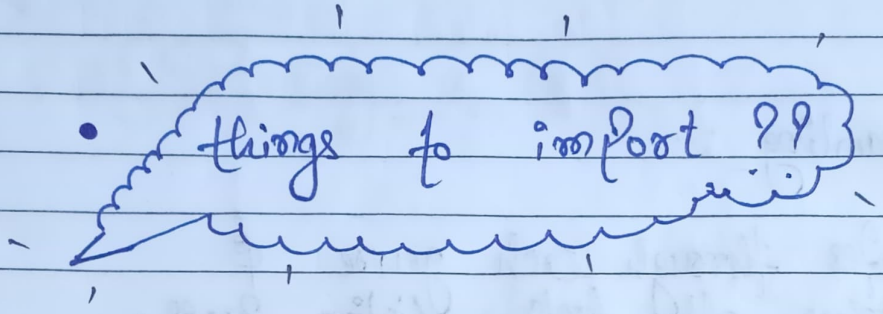
```
def home_page(url):
    Str = urllib.request.urlopen(url).read().decode()
    str = Str.strip()
    Soup = BeautifulSoup(Str, 'html.parser')
```

Created Soup of given (url).

```
genres = Soup.find('ul', class_="nav nav-list")
                .find_all('a')

# Searched targeted class & tag.

Links = list()
for genre in genres:
    link = 'https://books.toscrape.com/' + genre['href']
    Links.append(link)

# In genre['href'] we only get the relative
  adress, so we add it to rest.

Links = Links[1:] # Removed homepage link)

for link in Links:
    book_link(link)
```

Spiral

Page No. _____
Date




```
from bs4 import BeautifulSoup
import urllib.request, urllib.error, urllib.parse
import re
import pandas as pd
from urllib.parse import urljoin
```

```
# created a all-book dict.
global all_books
    all_books = {}
```

Page No. _____
Date

2. Genre Crawling :-

→ Iterates through each genre & fetches all book listing pages.

→ to Pagination (if next page appears moves to next).

```
def book_line (url):
    while True:
        ⋮
        # created soup from url.
        ⋮
    # find all books. ↓
    book_tags = soup.select ('h3 a')
    for tag in book_tags:
        book_url = urljoin (url, tag['href'])
    # used a func?? rather manual join & merge.
        try:
            Parse_book (book_url)
        except:
            print ( book_url, 'Not found' )
    # for next page
    next_btn = soup.find ('li', class_ = "next")
    if next_btn:
        next_link = next_btn.find ('a') ['href']
        url = urljoin (url, next_link)
    else:
        break
```

Spiral

Page No.
Date

3. Parsing a book :-

→ Visit each book page & extract the details.

```
def parse_book(url):
    ⋮
    # Soup created from url
    ⋮
# Extracted all parameters :-
name = soup.find('div', class_='content').find('h1')
        .text.strip()
stock = soup.find('p', class_='instock availability')
        .text.strip()
stock = re.search(r'[0-9]+', stock).group
    # to extract qty. in numbers.
rating = soup.find('p', class_='star rating')['class']
        [1].
rnum = {"One":1, "Two":2, "Three":3,
        "Four":4, "Five":5
    } # text → Num
rating = rnum.get(rating, 0)
```

Spiral

Created a dic. to take output extracted values (P_items)
Created dic. to iterate in single table. (t_items)

```
p_items = dict()
p_items['Name'] = name
p_items['Stock'] = stock
p_items['Rating'] = rating
p_items['Genre'] = genre

t_items = ['UPC', 'Product Type', 'Price (inc. tax)',
           'Price (exc. tax)', 'Tax', 'Number
                of Reviews']
```

Created func? (tags) to iterate each item.

```
def tags(tag):
    th_tag = soup.find('th', string = tag)
```

find inside th table. ↑ with respected names. ↑

```
    if (th_tag):
        td = th_tag.find_next_sibling('td')
             .text.strip()
    else:
        td = None
    return td
```

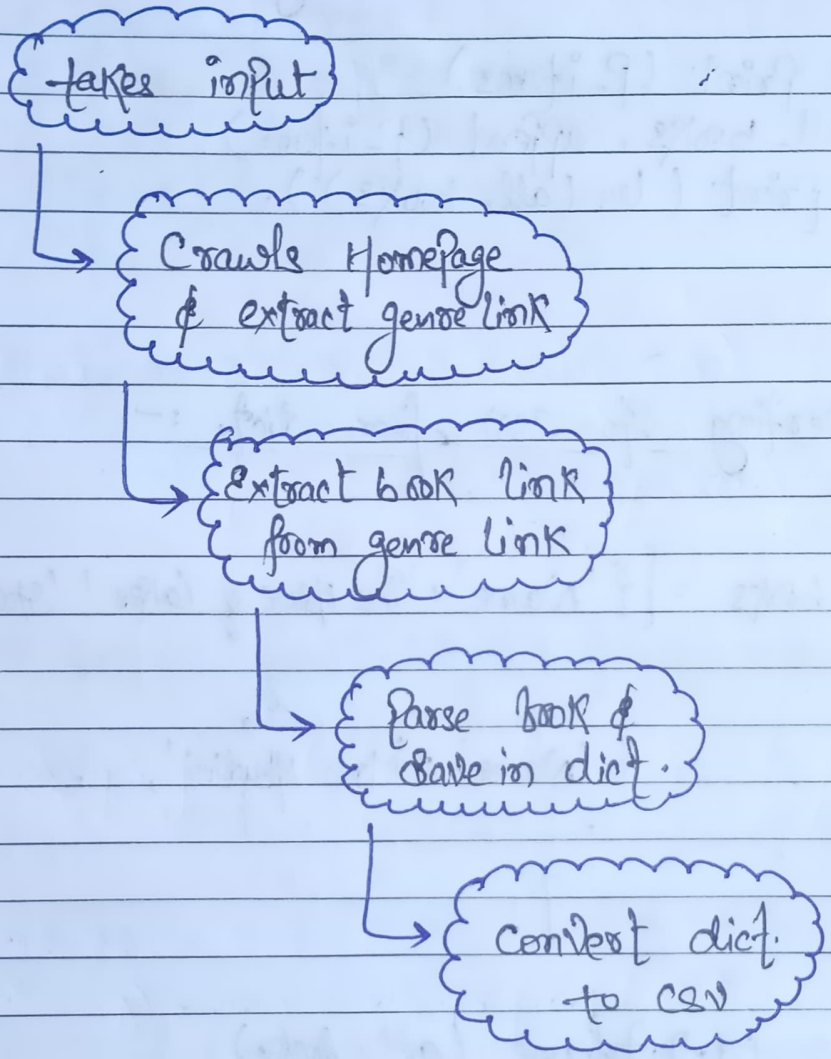
```
for item in t_items:
    p_items[item] = tags(item)

# Print(p_items)
all_books.append(p_items)
# print(len(all_books)).
```

4. Converting to csv from dict. :-

```
# all_books = [{'Name': 'The House of Cards', 'Stock': 15-
                                                    },
              {'Name': 'Mrs. Moudini', ...
                                                    }
              :].

df = pd.Dataframe(all_books)
df.to_csv("books_data.csv", index = False)
```

Page No.
Date

- **Program Work flow :-**



Spiral

Page No.
Date

- Methods() used :-

- Soup.text.strip()

  :- remove tag & prints the inside text.

- tittle = Soup.find('head').find('title')

  :- iterate over tags to find req. element.

  :- We do it many times (nesting)

Eg:-

```
name = Soup.find('body').find('div', class =
'container fluid page').find('div',
class_ = 'page-inner').find( ...
. . . . ).find('h1').text.strip()
```

- re.search()

  :- returns index & other details of matched content.

Spiral

- re.search().group()
  :- returns only matched string.

- rating = rnum.get(rating, 0)
  :- to extract values from key inside dict.

- Soup.find(tag_name, attribute filter)

Eg:-
N_review = Soup.find('a', href = re.compile(r'reviews/'))

↳ find 'a'

↳ inside match attribute

- .prettify()
  :- to beautify HTML codes.
  :- but we don't use methods in it.

- to parse .html comment

<!-- text to comment -->

```
from bs4 import BeautifulSoup, Comment
```

→ Search all comments :-

using

```
Soup.find_all(String = lambda text:
    isinstance(text, Comment))
```

:- returns normal HTML text.

:- Parse it as usual.

Spiral

Page No.
Date

- upc = soup.find('th', string = "UPC")
  - 'th' ↑ tag name
  - string = "UPC" ↑ attribute filter

In UPC,
↳ it doesn't have any class or tag.

So,
we navigate by value.

- link = genre['href']

genre = Books

↳ To iterate links inside tag.

print(link)

↳ ....html

↳ Clean website.

Spiral

Page No.
Date

Cleaning imported csv with pandas :-

Issues :-

(i) Big Names
(ii) Missing Values
(iii) Extra Spaces
(iv) Duplicates
(v) type cast
(vi) clean price, remove symbol
(vii) Remove unnecessary data
  - UPC :- no use
  - No. of reviews :- 0
  - Tax :- 0
  - Product type :- Books.

Price {inc. tax} & Price {exc. tax} :- contains same value {i.e. tax=0}

merged to "Price".

Spiral

Page No. _____
Date

codes :-

```
import pandas as pd
import re

df = pd.read_csv('.....csv') # import
df.head() # Analyze
df.info()

df['Name'] = df['Name'].str.split(':').str[0].str.strip()
df['Name'] = df['Name'].str.replace(r'\s*\(.*?\)', '', regex=True).str.strip()

# trimmed name after : & (...) inside this.

df[df.duplicated()]
    # no duplicates found

# remove irrelevant columns.
df = df.drop('Tax', axis=1)
df = df.drop('Number of Reviews', axis=1)
df = df.drop('Price (excl. Tax)', axis=1)
df = df.drop('Product Type', axis=1)

# rename 'Price (incl. tax)' ⟶ 'Price'.
df = df.rename(columns={'Price (incl. tax)': 'Price'})
df['Price'] = df['Price'].str[1:] # remove symbol.

df.to_csv('new_name.csv', index=False)
    # save
```

Spiral

Page No. _____
Date

Excel Analyzing & Dashboarding :-

→ Created 4 Sheet & a dashboard :-

(I) Cleaned books chart
(II) Stock by rating
(III) Rating by Stock
(IV) Rating by Qty.

Dash board

→ added Slicers & graphs.