PYTHON PROGRAMMING CODES FOR WEB-SCRAPING (DONE ON PYTHON IDLE 3.9 AND PYCHARM, ANACONDA)

Main.py

import csv  
import requests  
from bs4 import BeautifulSoup  
url = "https://www.amazon.co.uk/All-New-Fire-Tablet-Alexa-Display/dp/B07952CV7L/ref=cm\_cr\_arp\_d\_product\_top?ie=UTF8"  
s = requests.get(url)  
  
soup = BeautifulSoup(s.content,"lxml")  
  
links = soup.find\_all("div", {"class" : "info"})  
  
output = open("./products","w")  
writer = csv.writer(output)  
writer.writerow(["PRODUCTS"])  
writer.writerow(["Name","Price US / piece","Links"])  
#list\_of\_records = []  
for link in links:  
 record = []  
 name = link.contents[1].text  
 record.append(name)  
 price = link.contents[3].text.split()[1]  
 record.append(price)  
 alink = "http:" + link.contents[1].find('a').get('href')  
 record.append(alink)  
 writer.writerow(record)  
 print(name + price)  
 print()

Web\_Scraping.py

import csv  
import requests  
session = requests.Session()  
from Tools.scripts.gprof2html import header  
from bs4 import BeautifulSoup  
  
def getAmazonSearch(search\_query):  
 url = "https://www.amazon.co.uk/s?k=Fire+HD+8+Tablet%2C+16+GB%2C+Blue%E2%80%94with+Ads+%28Previous+Generation+-+8th%29&i=amazon-devices&ref=nb\_sb\_noss"  
 print(url)  
 page = requests.get(url, cookies = cookie, headers=header)  
 if page.status\_code==200:  
 return page  
 else:  
 return "Error"  
  
data\_asin=[]  
response=getAmazonSearch('Fire HD + 8 Tablet +16 GB + Blue—with Ads + (Previous Generation - 8th)')  
soup=BeautifulSoup(response.content)  
for i in soup.findAll("div",{'class':"sg-col-4-of-24 sg-col-4-of-12 sg-col-4-of-36 s-result-item sg-col-4-of-28 sg-col-4-of-16 sg-col sg-col-4-of-20 sg-col-4-of-32"}):  
 data\_asin.append(i['data-asin'])  
  
def Searchasin(asin):  
 url="https://www.amazon.in/dp/"+asin  
 print(url)  
 page=requests.get(url,cookies=cookie,headers=header)  
 if page.status\_code==200:  
 return page  
 else:  
 return "Error"  
  
link=[]  
for i in range(len(data\_asin)):  
 response=Searchasin(data\_asin[i])  
 soup=BeautifulSoup(response.content)  
 for i in soup.findAll("a",{'data-hook':"see-all-reviews-link-foot"}):  
 link.append(i['href'])  
  
def Searchreviews(review\_link):  
 url="https://www.amazon.in"+review\_link  
 print(url)  
 page=requests.get(url,cookies=cookie,headers=header)  
 if page.status\_code==200:  
 return page  
 else:  
 return "Error"  
  
reviews=[]  
for j in range(len(link)):  
 for k in range(100):  
 response=Searchreviews(link[j]+'&pageNumber='+str(k))  
 soup=BeautifulSoup(response.content)  
 for i in soup.findAll("span",{'data-hook':"review-body"}):  
 reviews.append(i.text)  
  
rev={'reviews':reviews} #converting the reviews list into a dictionary  
review\_data=pd.DataFrame.from\_dict(rev) #converting this dictionary into a dataframe  
review\_data.to\_csv('Scraping reviews.csv',index=False)

Web\_Scraping\_AmazonReviews.py

# importing libraries  
import requests  
from bs4 import BeautifulSoup  
  
  
def main(URL):  
 # openning our output file in append mode  
 File = open("out.csv", "a")  
  
 # specifying user agent, You can use other user agents  
 # available on the internet  
 HEADERS = ({'User-Agent':  
 'Mozilla/5.0 (X11; Linux x86\_64)  
 AppleWebKit/537.36 (KHTML, like Gecko)  
 Chrome/44.0.2403.157 Safari/537.36',  
 'Accept-Language': 'en-US, en;q=0.5'})  
  
 # Making the HTTP Request  
 webpage = requests.get(URL, headers=HEADERS)  
  
 # Creating the Soup Object containing all data  
 soup = BeautifulSoup(webpage.content, "lxml")  
  
 # retreiving product title  
 try:  
 # Outer Tag Object  
 title = soup.find("span",  
 attrs={"id": 'productTitle'})  
  
 # Inner NavigableString Object  
 title\_value = title.string  
  
 # Title as a string value  
 title\_string = title\_value.strip().replace(',', '')  
  
 except AttributeError:  
 title\_string = "NA"  
 print("product Title = ", title\_string)  
  
 # saving the title in the file  
 File.write(f"{title\_string},")  
  
 # retreiving price  
 try:  
 price = soup.find(  
 "span", attrs={'id': 'priceblock\_ourprice'})  
 .string.strip().replace(',', '')  
 # we are omitting unnecessary spaces  
 # and commas form our string  
 except AttributeError:  
 price = "NA"  
 print("Products price = ", price)  
  
 # saving  
 File.write(f"{price},")  
  
 # retreiving product rating  
 try:  
 rating = soup.find("i", attrs={  
 'class': 'a-icon a-icon-star a-star-4-5'})  
 .string.strip().replace(',', '')  
  
 except AttributeError:  
  
 try:  
 rating = soup.find(  
 "span", attrs={'class': 'a-icon-alt'})  
 .string.strip().replace(',', '')  
 except:  
 rating = "NA"  
 print("Overall rating = ", rating)  
  
 File.write(f"{rating},")  
  
 try:  
 review\_count = soup.find(  
 "span", attrs={'id': 'acrCustomerReviewText'})  
 .string.strip().replace(',', '')  
  
 except AttributeError:  
 review\_count = "NA"  
 print("Total reviews = ", review\_count)  
 File.write(f"{review\_count},")  
  
 # print availiblility status  
 try:  
 available = soup.find("div", attrs={'id': 'availability'})  
 available = available.find("span")  
 .string.strip().replace(',', '')  
  
 except AttributeError:  
 available = "NA"  
 print("Availability = ", available)  
  
 # saving the availibility and closing the line  
 File.write(f"{available},\n")  
  
 # closing the file  
 File.close()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
# openning our url file to access URLs  
 file = open("url.txt", "r")  
  
 # iterating over the urls  
 for links in file.readlines():  
 main(links)

Amazon\_Review.py

import scrapy  
class AmazonReviewSpider(scrapy.Spider):  
 name = 'amazon\_review'  
 allowed\_domains = ['https://www.amazon.co.uk/All-New-Fire-Tablet-Alexa-Display/product-reviews/B07952CV7L/ref=cm\_cr\_dp\_d\_show\_all\_btm?ie=UTF8']  
 start\_urls = ['http://https://www.amazon.co.uk/All-New-Fire-Tablet-Alexa-Display/product-reviews/B07952CV7L/ref=cm\_cr\_dp\_d\_show\_all\_btm?ie=UTF8/']  
  
 def parse(self, response):  
 pass

Example.py

import scrapy  
  
class samplespiderSpider(scrapy.Spider):  
 name = 'sampleSpider'  
  
 allowed\_domains = ['amazon.in']  
 start\_urls = ['http://amazon.in/']  
  
 def parse(self,response):  
 pass  
  
  
class AmazonReviewsSpider(scrapy.Spider):  
 # Spider name  
 name = 'amazon\_reviews'  
  
 # Domain names to scrape  
 allowed\_domains = ['amazon.in']  
  
 # Base URL for the MacBook air reviews  
 myBaseUrl = "https://www.amazon.co.uk/All-New-Fire-Tablet-Alexa-Display/product-reviews/B07952CV7L/ref=cm\_cr\_dp\_d\_show\_all\_btm?ie=UTF8&reviewerType=all\_reviews&amp;amp;pageNumber="  
 start\_urls = []  
  
 # Creating list of urls to be scraped by appending page number a the end of base url  
 for i in range(1, 121):  
 start\_urls.append(myBaseUrl + str(i))  
  
 # Defining a Scrapy parser  
 def parse(self, response):  
 data = response.css('#cm\_cr-review\_list')  
  
 # Collecting product star ratings  
 star\_rating = data.css('.review-rating')  
  
 # Collecting user reviews  
 comments = data.css('.review-text')  
 count = 0  
  
 # Combining the results  
 for review in star\_rating:  
 yield {'stars': ''.join(review.xpath('.//text()').extract()),  
 'comment': ''.join(comments[count].xpath(".//text()").extract())  
 }  
 count = count + 1

CODING FOR COMMAND PROMPT(PIP OR CONDA CAN BE USED FOR INSTALLING SCRAPER AND OTHER LIBRARIES)

1. conda install -c conda-forge scrapy

Or

pip install scrapy

1. scrapy startproject amazon\_reviews\_scraping
2. scrapy genspider amazon\_review <https://www.amazon.co.uk/All-New-Fire-Tablet-Alexa-Display/product-reviews/B07952CV7L/ref=cm_cr_dp_d_show_all_btm?ie=UTF8&reviewerType=all_reviews>
3. scrapy runspider amazon\_reviews\_scraping/amazon\_reviews\_scraping/spiders/amazon\_reviews.py -o reviews.csv
4. scrapy runspider spiders/filename.py -t txt -o -> amazonreviews.txt
5. scrapy runspider spiders/AmazonReview.py -o output.csv
6. import pandas as pd
7. pd.read\_csv(“reviews.csv”)
8. #Python Code Exploratory Data Handling Using Panda
9. import pandas as pd import matplotlib as plt pd.read\_csv("reviews.csv") summarised\_results = dataset["stars"].value\_counts() plt.bar(summarised\_results.keys(), summarised\_results.values) plt.show()
10. #Python Code To Visualize The Data On The CSV File

def visualise\_word\_map():

words=" "

for msg in dataset["comment"]:

msg = str(msg).lower()

words = words+msg+" "

wordcloud = WordCloud(width=3000, height=2500, background\_color='white').generate(words)

fig\_size = plt.rcParams["figure.figsize"]

fig\_size[0] = 14

fig\_size[1] = 7

plt.show(wordcloud) plt.axis("off")