Direct Research

Data Scraping for Competitive Market Analysis & Behavior

CSE (498R)

Prepared By

Md.Muttashim Mishel Mawn

ID: 1520800042

Supervised By

Muhammad Shafayat Oshman

Lecturer

Electrical and Computer Engineering

Department of Electrical & Computer Engineering

North South University

Dhaka 1229

Summer 2022

DECLARATION

I, hereby, declare that the work presented in this study or thesis report is the outcome of the design and development work performed by me under the supervision of **Muhammad Shafayat Oshman**, Lecturer, Electrical & Computer Engineering, North South University as a course work of CSE/EEE 498R (Direct Research). We also declare that no part of this report has been taken from other works without reference.

Signature of Students

_____

Md.Muttashim Mishel Mawn

APPROVAL

This Thesis report titled 'Data Scraping for Competitive Market Analysis and Behavior' submitted by Md.Muttashim Mishel Mawn ID: 1520800042 to the Department of Electrical and Computer Engineering, North South University, has been accepted as Direct Research Term Final Report.

Signatures

_____

Muhammad Shafayat Oshman

Lecturer, ECE

_____

Dr. Rajesh Palit

Professor and Chairman, ECE

Abstract

I have proposed a data analysis based study for a competitive market analysis from our country's online based market. Because of now a days a lot of online sale service provides various product at our door but it's so difficult for a newcomer to establish a sustaining business. Competitive analysis is one of the many aspects of digital marketing. It also requires data from various dynamic website. The main objective of our study is to develop a crawler to gather data to compare them and bring up a result that shows us the market status, which type of product they stock most and the demand of a specific product. And for this project I am going to use python library bs4 and scrappy framework for data scraping after that for data filtering I am going to use python pandas. The outcome of this project will help the new comers in the market to understand market status and consumer also can find out the availability of a certain product and what are their services according to specific brands and products.

Table of Contents

List of Figures

Introduction

Background

In 2022 business is more competitive than 10 years before. Advanced technology gives us the path and the solvent. In Bangladesh business for a newcomer is way harder than the traditional family business man who took the business experience from his father and his father took the business experience from his grandfather. And this is one of the reasons that our new generation is scared to start a new business.

Right now the market is competitive. No one gives you the secret sauce to get success. And I can say competitive market analysis is one of the most important aspects of digital marketing. Maybe everyone can get the initial success by following basic steps of starting a business but what about sustaining business development. If we talk about sustaining business development we should focus on product analysis and need to read consumer philosophy and their capability.

Problem Statement

Nowadays the competitive market is not newcomer friendly. A lot of newcomers just started their first business but that cannot even sustain. The reason behind this is not enough knowledge about market status, product analysis and consumer psychology to buy a product from you. My approach is to develop a crawler that will scrape data from various ecommerce websites and compare products by price range, demand and supply ratio, services that sellers provide us and how often consumer's write a review. If we are able to compare this we can get the data of the number of stock products in a specific price range. So we can determine the demand of a specific product. After all, it will help newcomers and even the experienced business man too.

Contributions

The proposed methodology may be among some first proposal. There are number of research done in competitive market analysis but not in this particular way. Our objectives are to develop and state the statistics from data that shown in web based ecommerce site. When the newcomers come to the market how they will read the current market status.

Feasibility Study

We discussed about possible solutions in this part.

Possible Solution

There may be many procedures for categorization and scraping data. But first I am trying to use beautiful soup library for data scraping and pandas for data filtering. If I will face any kind of critical issue I will try to use python framework called Scrapy.

Dataset Representation

Data representation is the most unique and important part of this project. Firstly, data can have gathered in txt format in each separate format. Reading directly from .txt and store them for separate class may be difficult. So, we can transfer those txt files into CSV format where title, price and category will be remained in the excel file.

Description

A competitive analysis is a strategy that involves researching major competitors to gain insight into their products, sales, and marketing tactics. Implementing stronger business strategies, warding off competitors, and capturing market share are just a few benefits of conducting a competitive market analysis.

Data scraping involves pulling information out of a website and into a spreadsheet. To a dedicated data scraper, the method is an efficient way to grab a great deal of information for analysis, processing, or presentation.

People can analyze data by themselves but it's not efficient to do manually in 2022. So what if a bot does the whole work and gives you the outcome and does not consume more time? So I want to create a crawler that crawl data from various websites and gathers data in a single csv file. Here you can gather data as much as you can. More data will give you a more appropriate outcome in any situation. The main difference between an experienced businessman and newcomer is the experienced businessman has better knowledge about product and consumer. Even newcomers know those facts but data changes over time so everyone should change their strategies over time too. And this is the main weak point for newcomers. If you fail to adopt you can't survive long. For example: when the rainy season has come the number of umbrella sales increases drastically everyone knows it but it does not depend on the rainy season it depends on how often rain comes. So if you stock a pile of umbrella at the beginning of the rainy season, then it can be said that you have adopted such a strategy due to lack of data analysis of your previous year.

Data and statistics will predict the future. So in this competitive market if you want to survive there is no other way then market and product analysis.

Proposed Solution

Platform

Software

*python 3.10.6

*Microsoft Excel

Packages

*Beautiful Soup 4

*Pandas

*Numpy

*Matplotlib

Proposed Solution Flowchart

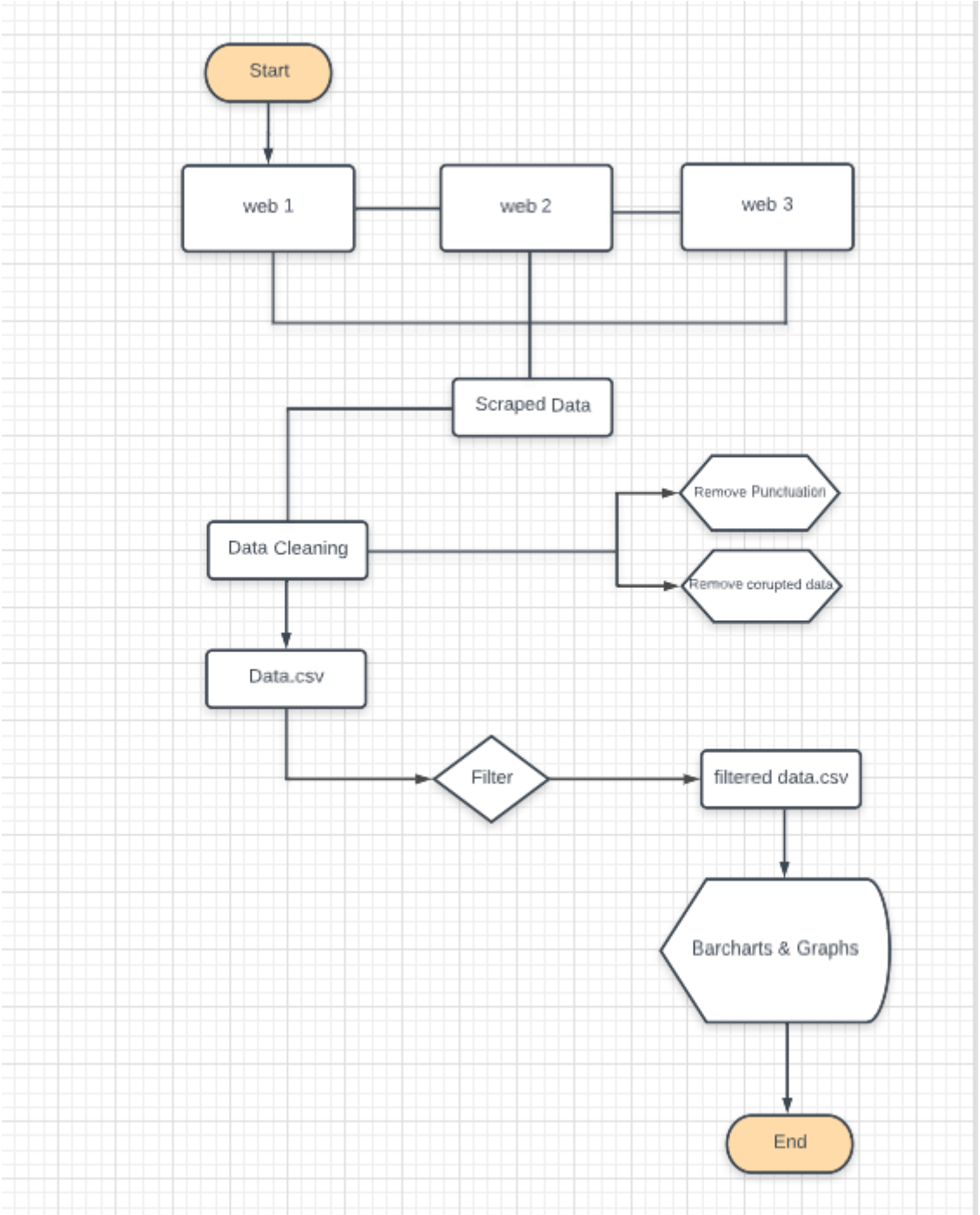


Fig 1: Proposed Solution Block Diagram

Data Set Selection

Dataset selection plays a very important role in supervised machine learning approaches to categorizing. The quality and the quantity of data shape the goal to predict a text to the right output. Web based shops is our main goal to scrap data but not all of the shops we choose computer hardware and peripherals like motherboard, monitor, mouse, keyboard and many more.

Pre-Processing

In this step, our goal is to remove noises from the data. As texts are a much unstructured way to represent information and consist noise that can make the classifier very difficult to do its work. There are some basic ways to preprocess text to extract only relevant pieces of information. We divided the preprocessing step into three steps. But in this project we are working in English language only so that for this application we need one pre-processing step that is

✓Punctuation removal

Punctuation Removal

Since the punctuations play a very little role in order to contribute being a feature and also can be considered as noise we have removed all the punctuations used in the title of a product. We have also removed numerals and special characters for the same reason. Some of the examples of the punctuations, numerals, and special characters that we removed are given below

—, \, \, , ╬, ∏, , s, S, , `, ?, , %, ;, , :, =, ¡, ¤, ¦, §, ª, ®, ¯, ¬, ´, ¾, å, è, ò, ö, ÷, ø, ÿ, ý

Impacts

Impact analysis of this project consists of technical, environmental, financial and societal analysis.

Technical Analysis

Data scrap is one of the important parts of internet or web. Now a day everything is analyzed on web using text mining techniques. It's quite difficult for a person to check every product title and matched with other shops title and then store one of them. That's why data scraping and organize this data is so important for us.

Financial Analysis

I can use this data analysis to categorize products by product title and find out which seller provide us the minimum price among them and many more valuable information's. It's also beneficial for business farms or big companies to predict the market demand stock and consumer state.

Environmental Analysis

Data scraping and competitive market analysis is just a software side project. So I don't think there is any kind of environmental impact of this application.

Societal Analysis

There are impacts on societal analysis. By scraping data and data analysis from online based market system will reduce fake seller, fake product and reduce price hike syndicates. Or maybe reduce illegal stock holder who creates fake demand in the market.

Result Analysis

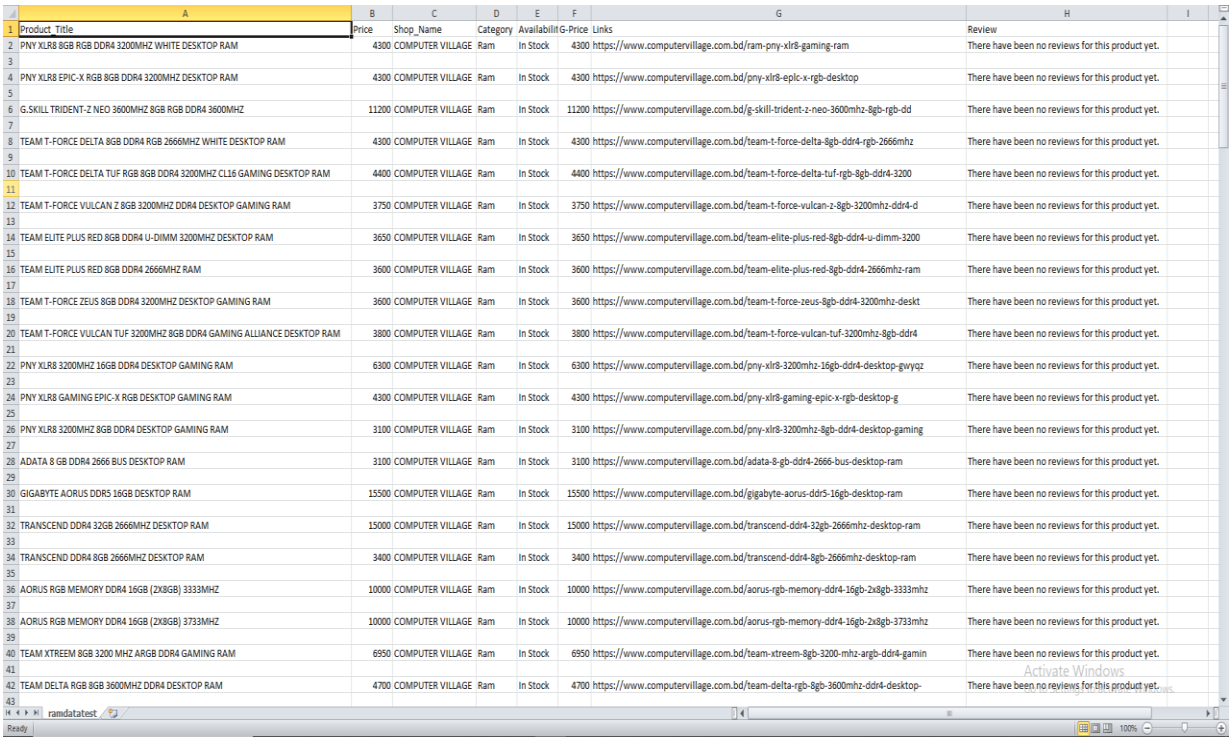
Methodology

My approach to solving this problem is to break the problem down into smaller parts and solve them one by one. The first thing that I need is information. So I decided to scrape data from the top five computer hardware and peripheral shops in Bangladesh and select five types of products from their website. Collecting data from various websites is not that much hard but when I crawled data from websites it will take so much unnecessary stuff and it will prevent our next step.

1709	RAPOO E6700 BLUETOOTH ULTRA-SLIM KEYBOARD WITH TOUCHPAD	3850	COMPUTER VILLAGE	Keyboard	In Stock	3850
1710	RAPOO X9310 WIRELESS ULTRA SLIM ALUMINUM ALLOY KEYBOARD MOUSE COMBO	3475	COMPUTER VILLAGE	Keyboard	In Stock	3475
1711	RAPOO V500 PRO BACKLIT USB MECHANICAL GAMING KEYBOARD YELLOW AND BLUE	3300	COMPUTER VILLAGE	Keyboard	In Stock	3300
1712	RAPOO X8210 WIRELESS MOUSE & KEYBOARD COMBO	2175	COMPUTER VILLAGE	Keyboard	In Stock	2175
1713	TEAM VULCAN Z 8GB DDR4 2666MHZ DESKTOP RAM	4400	Skyland	Ram	In Stock	4400
1714	TEAM ELITE PLUS 16GB DDR4 2400MHZ DESKTOP RAM	5400	Skyland	Ram	In Stock	5400
1715	RAMSTA 4GB DDR4 2400MHZ DESKTOP RAM	2300	Skyland	Ram	In Stock	2300
1716	CORSAIR VENGEANCE LPX 16GB DDR4 3200MHZ DESKTOP RAM	8999	Skyland	Ram	In Stock	8999
1717	TEAM ELITE U-DIMM 4GB 1600MHZ DDR3 RAM	2100	Skyland	Ram	In Stock	2100
1718	G.SKILL TRIDENT Z ROYAL 8GB DDR4 3200MHZ SILVER HEATSINK DESKTOP RAM	8500	Skyland	Ram	In Stock	8500
1719	G.SKILL TRIDENT Z ROYAL SERIES 16GB 3600MHZ RGB SILVER DDR4 RAM	6200	Skyland	Ram	In Stock	6200
1720	CORSAIR VENGEANCE RGB PRO SL 16GB DDR4 3200MHZ RAM WHITE	8000	Skyland	Ram	In Stock	8000
1721	G.SKILL TRIDENT Z NEO RGB 16GB DDR4 3600MHZ GAMING DESKTOP RAM	8300	Skyland	Ram	In Stock	8300
1722	GIGABYTE AORUS RGB 8GB DDR4 3333MHZ DESKTOP GAMING RAM	5200	Skyland	Ram	In Stock	5200
1723	AITC KINGSMAN DDR3 8GB 1600MHZ HEATSINK DESKTOP RAM	2950	Skyland	Ram	In Stock	2950
1724	TEAM ELITE U-DIMM 8GB 2400MHZ DDR4 RAM	3299	Skyland	Ram	In Stock	3299
1725	CORSAIR DOMINATOR PLATINUM RGB 8GB 3200MHZ DDR4 RAM (WHITE)	6300	Skyland	Ram	In Stock	6300
1726	THERMALTAKE TOUGHRAM RGB 8GB 4600MHZ DDR4 DESKTOP RAM	10200	Skyland	Ram	In Stock	10200
1727	AITC 8GB DDR4 UDIMM 2666MHZ DESKTOP RAM	3672	Skyland	Ram	In Stock	3672
1728	AITC KINGSMAN 4GB DDR4 3000MHZ DESKTOP RAM	2299	Skyland	Ram	In Stock	2299
1729	TRANSCEND JETRAM 32GB DDR4 3200MHZ U-DIMM DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1730	TRANSCEND JETRAM 16GB DDR4 2666MHZ U-DIMM DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1731	TRANSCEND JETRAM 8GB DDR4 3200MHZ U-DIMM DESKTOP RAM	3500	COMPUTER VILLAGE	Ram	In Stock	3500
1732	TRANSCEND JETRAM 4GB DDR4 2666MHZ UDIMM DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1733	TRANSCEND 4GB DDR4 2400MHZ UDIMM DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1734	ADATA XPG HUNTER 32GB DDR4 3200MHZ DIMM DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1735	ADATA 16GB DDR4 2666MHZ DESKTOP RAM	8500	COMPUTER VILLAGE	Ram	In Stock	8500
1736	ADATA XPG HUNTER 16GB DDR4 3200MHZ DIMM DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1737	ADATA XPG SPECTRIX D60G RGB 8GB DDR4 3600MHZ GAMING DESKTOP RAM	5500	COMPUTER VILLAGE	Ram	In Stock	5500
1738	ADATA XPG SPECTRIX D60G RGB 8GB 3200MHZ DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1739	ADATA GAMMIX D30 8GB DDR4 3000MHZ GAMING DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1740	ADATA XPG GAMMIX D30 8GB DDR4 2666MHZ BLACK HEATSINK GAMING DESKTOP RAM	Call For Price	COMPUTER VILLAGE	Ram	In Stock	null
1741	PATRIOT SIGNATURE LINE PREMIUM 8GB DDR4 2666MHZ DESKTOP RAM	3000	TECHLAND	Ram	In Stock	3000
1742	G.SKILL TRIDENT Z NEO 32GB (2 X 16GB) 3600MHZ RGB RAM	15500	TECHLAND	Ram	In Stock	15500
1743	CORSAIR VENGEANCE LPX 4GB DDR4 2400MHZ DESKTOP RAM	2500	TECHLAND	Ram	In Stock	2500
1744	CORSAIR VENGEANCE 32GB (2X16GB) DDR5 5600MHZ DESKTOP RAM (BLACK)	38800	TECHLAND	Ram	In Stock	38800
1745	CORSAIR VENGEANCE 32GB (2X16GB) DDR5 4800MHZ DESKTOP RAM (BLACK)	29000	TECHLAND	Ram	In Stock	29000
1746	CORSAIR VENGEANCE 16GB DDR5 4800MHZ DESKTOP RAM (BLACK)	14500	TECHLAND	Ram	In Stock	14500
1747	THERMALTAKE WATERRAM RGB 32GB (4 X 8GB) DDR4 3600MHZ LIQUID COOLING DESKTOP RAM	43500	TECHLAND	Ram	In Stock	43500
1748	THERMALTAKE WATER RAM RGB LIQUID COOLING 32GB(4 X 8GB) DDR4 3200MHZ DESKTOP RAM	39000	TECHLAND	Ram	In Stock	39000

Fig.2 Corrupted Data

So I need to organize all of the data in a uniform format. The more we clean the data the more we will get the accurate outcome. So that data cleaning is the most important section for any kind of data analysis project. After that we need to remove all of the corrupted data from the data list. For this project we are storing product name, product price, product category, product link and product review. If any one of the variables is missing we can call it corrupted data. Because when we will start to filter and compare one with another our project outcome accuracy will decrease. But i didn't edit our main dataset; we just created a fresh new dataset in another file. Finally we have a dataset to work with.



	A	B	C	D	E	F	G	H	I
1	Product_Title	Price	Shop_Name	Category	Availabilit	G-Price	Links	Review	
2	PNY XLR8 8GB RGB DDR4 3200MHZ WHITE DESKTOP RAM	4300	COMPUTER VILLAGE	Ram	In Stock	4300	https://www.computervillage.com.bd/ram-pny-xlr8-gaming-ram	There have been no reviews for this product yet.	
3									
4	PNY XLR8 EPIC-X RGB 8GB DDR4 3200MHZ DESKTOP RAM	4300	COMPUTER VILLAGE	Ram	In Stock	4300	https://www.computervillage.com.bd/pny-xlr8-epic-x-rgb-desktop	There have been no reviews for this product yet.	
5									
6	G.SKILL TRIDENT-Z NEO 3600MHZ 8GB RGB DDR4 3600MHZ	11200	COMPUTER VILLAGE	Ram	In Stock	11200	https://www.computervillage.com.bd/g-skill-trident-z-neo-3600mhz-8gb-rgb-dd	There have been no reviews for this product yet.	
7									
8	TEAM T-FORCE DELTA 8GB DDR4 RGB 2666MHZ WHITE DESKTOP RAM	4300	COMPUTER VILLAGE	Ram	In Stock	4300	https://www.computervillage.com.bd/team-t-force-delta-8gb-ddr4-rgb-2666mhz	There have been no reviews for this product yet.	
9									
10	TEAM T-FORCE DELTA TUF RGB 8GB DDR4 3200MHZ CL16 GAMING DESKTOP RAM	4400	COMPUTER VILLAGE	Ram	In Stock	4400	https://www.computervillage.com.bd/team-t-force-delta-tuf-rgb-8gb-ddr4-3200	There have been no reviews for this product yet.	
11									
12	TEAM T-FORCE VULCAN Z 8GB 3200MHZ DDR4 DESKTOP GAMING RAM	3750	COMPUTER VILLAGE	Ram	In Stock	3750	https://www.computervillage.com.bd/team-t-force-vulcan-z-8gb-3200mhz-ddr4-d	There have been no reviews for this product yet.	
13									
14	TEAM ELITE PLUS RED 8GB DDR4 U-DIMM 3200MHZ DESKTOP RAM	3650	COMPUTER VILLAGE	Ram	In Stock	3650	https://www.computervillage.com.bd/team-elite-plus-red-8gb-ddr4-u-dimm-3200	There have been no reviews for this product yet.	
15									
16	TEAM ELITE PLUS RED 8GB DDR4 2666MHZ RAM	3600	COMPUTER VILLAGE	Ram	In Stock	3600	https://www.computervillage.com.bd/team-elite-plus-red-8gb-ddr4-2666mhz-ram	There have been no reviews for this product yet.	
17									
18	TEAM T-FORCE ZEUS 8GB DDR4 3200MHZ DESKTOP GAMING RAM	3600	COMPUTER VILLAGE	Ram	In Stock	3600	https://www.computervillage.com.bd/team-t-force-zeus-8gb-ddr4-3200mhz-deskt	There have been no reviews for this product yet.	
19									
20	TEAM T-FORCE VULCAN TUF 3200MHZ 8GB DDR4 GAMING ALLIANCE DESKTOP RAM	3800	COMPUTER VILLAGE	Ram	In Stock	3800	https://www.computervillage.com.bd/team-t-force-vulcan-tuf-3200mhz-8gb-ddr4	There have been no reviews for this product yet.	
21									
22	PNY XLR8 3200MHZ 16GB DDR4 DESKTOP GAMING RAM	6300	COMPUTER VILLAGE	Ram	In Stock	6300	https://www.computervillage.com.bd/pny-xlr8-3200mhz-16gb-ddr4-desktop-gwyqz	There have been no reviews for this product yet.	
23									
24	PNY XLR8 GAMING EPIC-X RGB DESKTOP GAMING RAM	4300	COMPUTER VILLAGE	Ram	In Stock	4300	https://www.computervillage.com.bd/pny-xlr8-gaming-epic-x-rgb-desktop-g	There have been no reviews for this product yet.	
25									
26	PNY XLR8 3200MHZ 8GB DDR4 DESKTOP GAMING RAM	3100	COMPUTER VILLAGE	Ram	In Stock	3100	https://www.computervillage.com.bd/pny-xlr8-3200mhz-8gb-ddr4-desktop-gaming	There have been no reviews for this product yet.	
27									
28	ADATA 8 GB DDR4 2666 BUS DESKTOP RAM	3100	COMPUTER VILLAGE	Ram	In Stock	3100	https://www.computervillage.com.bd/adata-8-gb-ddr4-2666-bus-desktop-ram	There have been no reviews for this product yet.	
29									
30	GIGABYTE AORUS DDR5 16GB DESKTOP RAM	15500	COMPUTER VILLAGE	Ram	In Stock	15500	https://www.computervillage.com.bd/gigabyte-aorus-ddr5-16gb-desktop-ram	There have been no reviews for this product yet.	
31									
32	TRANSCEND DDR4 32GB 2666MHZ DESKTOP RAM	15000	COMPUTER VILLAGE	Ram	In Stock	15000	https://www.computervillage.com.bd/transcend-ddr4-32gb-2666mhz-desktop-ram	There have been no reviews for this product yet.	
33									
34	TRANSCEND DDR4 8GB 2666MHZ DESKTOP RAM	3400	COMPUTER VILLAGE	Ram	In Stock	3400	https://www.computervillage.com.bd/transcend-ddr4-8gb-2666mhz-desktop-ram	There have been no reviews for this product yet.	
35									
36	AORUS RGB MEMORY DDR4 16GB (2X8GB) 3333MHZ	10000	COMPUTER VILLAGE	Ram	In Stock	10000	https://www.computervillage.com.bd/aorus-rgb-memory-ddr4-16gb-2x8gb-3333mhz	There have been no reviews for this product yet.	
37									
38	AORUS RGB MEMORY DDR4 16GB (2X8GB) 3733MHZ	10000	COMPUTER VILLAGE	Ram	In Stock	10000	https://www.computervillage.com.bd/aorus-rgb-memory-ddr4-16gb-2x8gb-3733mhz	There have been no reviews for this product yet.	
39									
40	TEAM XTREEM 8GB 3200 MHZ ARGB DDR4 GAMING RAM	6950	COMPUTER VILLAGE	Ram	In Stock	6950	https://www.computervillage.com.bd/team-xtreem-8gb-3200-mhz-argb-ddr4-gamin	There have been no reviews for this product yet.	
41									
42	TEAM DELTA RGB 8GB 3600MHZ DDR4 DESKTOP RAM	4700	COMPUTER VILLAGE	Ram	In Stock	4700	https://www.computervillage.com.bd/team-delta-rgb-8gb-3600mhz-ddr4-desktop-	There have been no reviews for this product yet.	
43									

Fig.3 Dataset

Now we are in data analysis. First we will see how the product stocks within a certain price range. And to determine the range of this product, I extracted the minimum price and maximum price of the product from my data set and divided the price between them into two ways, one lower price range and the other upper price range.

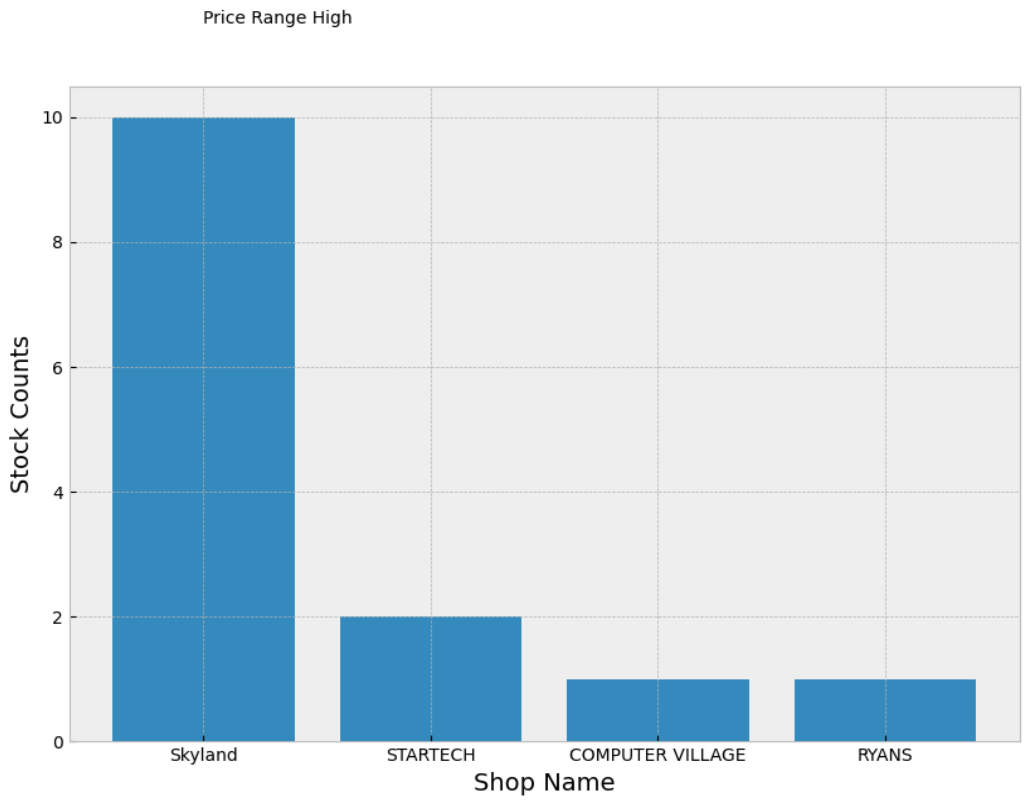


Fig.4 Stock counts in high price range

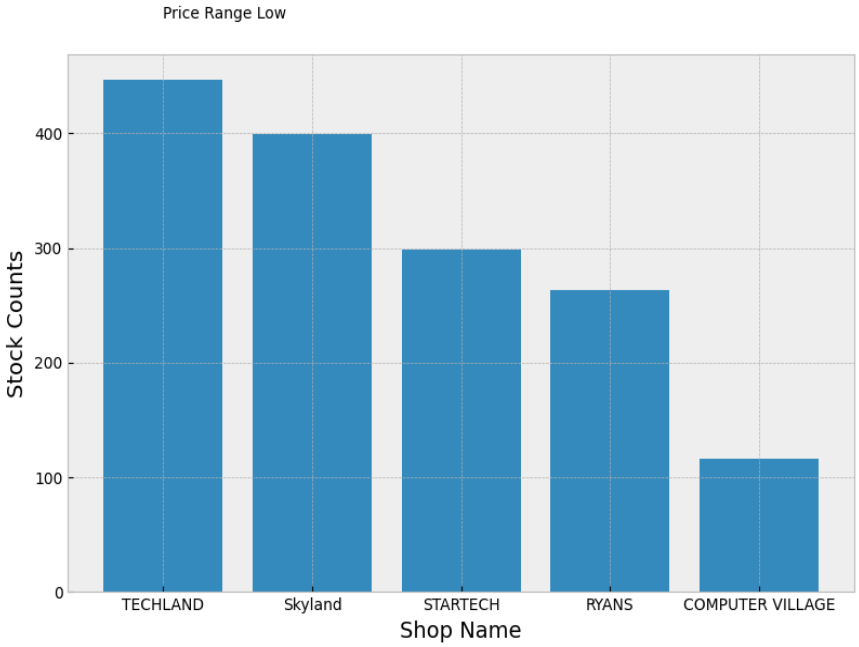


Fig.5 Stock count in low price range

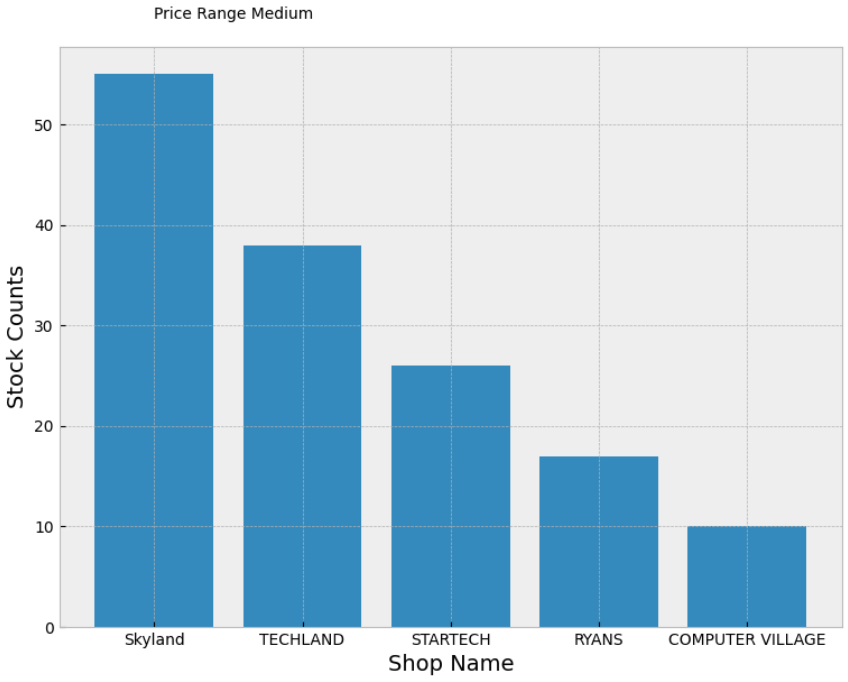


Fig.6 Stock Counts in medium price range

We can understand from the three bar charts that the stocks in the lower price range products are more than the higher price range products. And it is so obvious for any type of market. So we need a specific and more meaningful outcome. And for that i extracted the minimum price range, medium price range and high price range from the data set. And this time I found different results from different product categories. And the outcome will change again over time. Maybe in the future the number of medium price ranges will increase with demand. Because In economics. An increase in supply, all other things unchanged, will cause the equilibrium price to fall, quantity demanded will increase. A decrease in supply will cause the equilibrium price to rise, quantity demanded will decrease.

We have one more variable that will provide us with consumer psychology or trends of any kind of product. If we compare the number of reviews and the number of products within a certain product price range, we will know the consumer's soft spot and which products are more demanded in a specific time period.

Discussion

Initially I tested 5k data, measured the outcomes and then I tested 10k data and measured the outcomes. And what I saw the result changed when the number of data increased. One more thing I tried during this time period is scraping the same data set in two different time periods but I cannot find any changes because the owner of those websites didn’t update their stock and upcoming product information. I can assure that if i can scrape data from two different time periods we will get a more appropriate result.

Design Impacts & Cost

In this section, we discuss about out time cost, improvements and about future work

Time Cost

Task	Working Hours
Theoretical Study	25 Hours
Data Scraping	15 Hours
Data analysis	15 Hours
Pre-processing	5 Hours
Testing and debug	10 Hours
Minor Bug Fix	5 Hours
Total Time	75 Hours

Cost Prediction

For this project we just need a better computer to scrape more data in a short time. I scraped 16k data and my computer takes half an hour because I have a mid-end computer. But if we will scrape over 1m data we definitely need high end devices because when i start to scrape multiple pages from a single website i need to put a delay after every page so that server cannot detect it as a bot. If the server detects it, the server will block it. So if we want to avoid this issue we just need a high end device. After that it won't take too much time to execute.

I am working on a computer that costs around 2lakhs. If we want to run this project for industrial purposes we need a computer that is around 5 -7 lakhs. Because we need faster processor.

Future work

Basically we have data so we can solve different types of problems too. If we compare individual products in different shops we can find out who provided the minimum price of a specific product. It will help the consumer because comparing a single product from different websites is more time consuming. Also everyone knows that some bad businessmen create a syndicate and they increase a certain product's demand and price by holding products. So for this problem I need both price data that is specified by the company and the vendor. It will help to break the unnecessary price hike.

Conclusion

In this project i have made and stated an analytical result from data which is scraped from web based ecommerce sites that offer only viewable data. For this project, i made an actual data set for scraping and find out the current market situation that will influence our newcomers to become entrepreneurs. It will not only help the individuals but also puts a huge impact on our central economy, unemployment and a better environment for consumers to buy products. We believe and hope that our data analysis will be able to eradicate the obstacles that are faced by so many people who have not enough time to spend in competitive market analysis and at the same time it will give mobility in their way of life.

Acknowledgement

First of all, we would like to thank Almighty for all the fate related to our studies and secondly to express our profound gratitude to our honorable course instructor **Muhammad Shafayat Oshman**, for his constant and meticulous supervision, valuable suggestions, his patience and encouragement to complete the project work.

We would also thank the ECE department of North South University for providing us with the opportunity to have an industrial level design experience as part of our curriculum for the undergraduate program.

Finally, we would like to thank our families, friends, classmates and everybody who supported us and provided with guidance for the completion of this project.

Appendix

Code

File Name: allSite.py

```
from bs4 import BeautifulSoup
import requests
import time
import threading
# output csv file declared here
filename = "allData.csv"
f = open(filename, 'w', encoding='utf-8-sig')
headers = "Product_Title,Price,Shop_Name,Category,Availability,G-Price\n"
f.write(headers)
#ramdatasets
def webscrapCvillageRam():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.computervillage.com.bd/ram?page={}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')

        # search element from specified url html

        body = soup.find('body')

        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all('div', class_='grid-view-item')
        # using loop for grabing whole page data
        for product in productInfo:
```

```
            product_name = product.find(
                'h4', class_='h4 grid-view-item__title text-truncate-2')
            title = product_name.a.text.replace(",", "").upper()
            product_price = product.find('span', class_='money')
            tk = product_price.text.replace(",", "").replace("৳", "")
            g_price = tk.replace('Call For Price', 'null')
            #print(tk)
            availability = product.find(
                'span', class_='sticker-stock-l bg-red')
            if availability:
                stock = availability.text
            else:
                stock = 'In Stock'
            #print(stock)

            data = title + "," + tk + "," + "COMPUTER VILLAGE" + \
                "," + "Ram" + "," + stock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)
def webscrapTechlandRam():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
             20, 21, 22, 23, 24, 25]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.techlandbd.com/pc-components/shop-desktop-ram?page={}'.format(page)).text
        # source_link = requests.get('https://www.techlandbd.com/shop-computer-mouse?page=40').text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
```

```
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
    product_thumb = body.find_all('div', class_='product-thumb')
    # using loop for grabing whole page data
    for product in product_thumb:
      product_name = product.find('div', class_='name')
      title = product_name.a.text.upper()
      product_price = product.find('div', class_='price')
      if product_price:
        tk = product_price.span.text
      else:
        tk = 'N/A'
      g_price = tk.replace('N/A', 'null')
      availability = product.find('div', class_='cart-group')
      stock = availability.a.text
      #print(stock)
      data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
          "") + "," + "TECHLAND" + "," + "Ram" + "," + stock.replace("Add to Cart", "In Stock") + "," +
g_price.replace("৳", "").replace(",",
          "") + "\n")
      f.write(data)
    time.sleep(5)
def webscrapStartechRam():
  pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
       13, 14]
  # declared the url directory and store it in a variable
  # techland gpu section
  for page in pages:
    source_link = requests.get(
      'https://www.startech.com.bd/component/ram?page={}'.format(page)).text
    soup = BeautifulSoup(source_link, 'lxml')
    # search element from specified url html
```

```
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
    productInfo = body.find_all('div', class_='p-item')
    # using loop for grabing whole page data
    for product in productInfo:
        product_name = product.find('h4', class_='p-item-name')
        title = product_name.a.text.upper()
        product_price = product.find('div', class_='p-item-price')
        tk = product_price.span.text.replace(",", "").replace("৳", "")
        g_price = tk.replace('TBA', 'null')
        # int(tk)
        availability = product.find('div', class_='actions')
        stock = availability.span.text
        #print(stock)
        data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
             "") + "," + "STARTECH" + "," + "Ram" + "," + stock.replace("shopping_cart Buy Now", "In Stock") + "," +
g_price + "\n")
        f.write(data)
    time.sleep(5)


def webscrapRyansRam():
    pages = [1, 2, 3, 4, 5, 6, 7]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.ryanscomputers.com/category/desktop-component-desktop-ram?page={}'.format(page)).text


        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
```

```
        product_info = body.find_all('div', class_='card h-100')
        # using loop for grabing whole page data
        for product in product_info:
            product_name = product.find(
                'p', class_='card-text p-0 m-0 list-view-text')
            title = product_name.a.text.replace(",", "").upper()

            product_price = product.find(
                'p', class_='pr-text cat-sp-text pb-1')

            tk = product_price.text.replace("Tk", "").replace(",", "").strip()
            g_price = tk
            availability = product.find(
                'button', class_='btn grid-cart-btn cart-btn px-2 cat-cart-btn')
            stock = availability.text
            if stock:
                newstock = 'In Stock'
            else:
                newstock = 'Out of Stock'
            #print(newstock)
            data = title + "," + tk + "," + "RYANS" + "," + \
                "Ram" + "," + newstock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)


def webscrapSkylandRam():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.skyland.com.bd/product-category/components/ram/page/{}'.format(page)).text
```

```
soup = BeautifulSoup(source_link, 'lxml')
# search element from specified url html
body = soup.find('body')
# here product-thumb is a css class so that i used it as a variable for better understanding
productInfo = body.find_all(
    'div', class_='box-text box-text-products text-center grid-style-2')

# using loop for grabing whole page data
for product in productInfo:
    product_name = product.find(
        'p', class_='name product-title woocommerce-loop-product__title')
    title = product_name.a.text.replace(",", "").upper()
    product_price = product.find(
        'span', class_='woocommerce-Price-amount amount')
    if product_price:
        tk = product_price.text.replace(",", "").replace("৳", "")
        tk = int(tk)
    else:
        tk = 'N/A'
    tk = str(tk)
    g_price = tk.replace('N/A', 'null')
    availability = product.find('div', class_='out-of-stock-label')
    if availability:
        stock = availability.text
        #print(stock)
    else:
        stock = 'In Stock'
    data = title + "," + tk + "," + "Skyland" + "," + \
        "Ram" + "," + stock + "," + g_price + "\n"

    f.write(data)
time.sleep(5)
```

```
#mousedatasets
def webscrapCvillageMouse():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.computervillage.com.bd/mouse?page={}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')

        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all('div', class_='grid-view-item')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find(
                'h4', class_='h4 grid-view-item__title text-truncate-2')
            title = product_name.a.text.replace(",", "").upper()
            product_price = product.find('span', class_='money')
            tk = product_price.text.replace(",", "").replace("৳", "")
            g_price = tk.replace('Call For Price', 'null')

            #print(tk)
            availability = product.find(
                'span', class_='sticker-stock-l bg-red')
            if availability:
                stock = availability.text
            else:
```

```
            stock = 'In Stock'
        #print(stock)
        data = title + "," + tk + "," + "COMPUTER VILLAGE" + \
            "," + "Mouse" + "," + stock + "," + g_price + "\n"
        f.write(data)
    time.sleep(5)


def webscrapTechlandMouse():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
             21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.techlandbd.com/shop-computer-mouse?page={}'.format(page)).text
        # source_link = requests.get('https://www.techlandbd.com/shop-computer-mouse?page=40').text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        product_thumb = body.find_all('div', class_='product-thumb')
        # using loop for grabing whole page data
        for product in product_thumb:
            product_name = product.find('div', class_='name')
            title = product_name.a.text.upper()
            product_price = product.find('div', class_='price')
            if product_price:
                tk = product_price.span.text
            else:
                tk = 'N/A'
            g_price = tk.replace('N/A', 'null')
```

```
            availability = product.find('div', class_='cart-group')
            stock = availability.a.text
            #print(stock)
            data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
                    "") + "," + "TECHLAND" + "," + "Mouse" + "," + stock.replace("Add to Cart", "In Stock") + "," +
g_price.replace("৳", "").replace(",",
                    "") + "\n")
            f.write(data)
        time.sleep(5)
def webscrapStartechMouse():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
             21, 22, 23, 24, 25]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.startech.com.bd/accessories/mouse?page={}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all('div', class_='p-item')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find('h4', class_='p-item-name')
            title = product_name.a.text.upper()
            product_price = product.find('div', class_='p-item-price')
            tk = product_price.span.text.replace(",", "").replace("৳", "")
            g_price = tk.replace('TBA', 'null')
            # int(tk)
```

```
        availability = product.find('div', class_='actions')
        stock = availability.span.text
        #print(stock)
        data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
              "") + "," + "STARTECH" + "," + "Mouse" + "," + stock.replace("shopping_cart Buy Now", "In Stock") + "," +
g_price + "\n")
        f.write(data)
    time.sleep(5)


def webscrapRyansMouse():
  pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
  # declared the url directory and store it in a variable
  # techland gpu section
  for page in pages:
    source_link = requests.get(
      'https://www.ryanscomputers.com/category/desktop-component-mouse?page={}'.format(page)).text
    soup = BeautifulSoup(source_link, 'lxml')
    # search element from specified url html
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
    product_info = body.find_all('div', class_='card h-100')
    # using loop for grabing whole page data
    for product in product_info:
      product_name = product.find(
        'p', class_='card-text p-0 m-0 list-view-text')
      title = product_name.a.text.replace(",", "").upper()
      product_price = product.find(
        'p', class_='pr-text cat-sp-text pb-1')
      tk = product_price.text.replace("Tk", "").replace(",", "").strip()
      g_price = tk
      availability = product.find(
        'button', class_='btn grid-cart-btn cart-btn px-2 cat-cart-btn')
```

```
            stock = availability.text
            if stock:
                newstock = 'In Stock'
            else:
                newstock = 'Out of Stock'
            #print(newstock)
            data = title + "," + tk + "," + "RYANS" + "," + \
                "Mouse" + "," + newstock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)
def webscrapSkylandMouse():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
             21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.skyland.com.bd/product-category/accessories/mouse/page/{}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all(
            'div', class_='product-small box')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find(
                'p', class_='name product-title woocommerce-loop-product__title')
            title = product_name.a.text.replace(",", "").upper()
            product_price = product.find(
                'span', class_='woocommerce-Price-amount amount')
```

```
            if product_price:
                tk = product_price.text.replace(",", "").replace("৳", "")
                tk = int(tk)
            else:
                tk = 'N/A'
            tk = str(tk)
            g_price = tk.replace('N/A', 'null')
            availability = product.find('div', class_='out-of-stock-label')
            if availability:
                stock = availability.text
                #print(stock)
            else:
                stock = 'In Stock'
            data = title + "," + tk + "," + "Skyland" + "," + \
                "Mouse" + "," + stock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)

#keyboarddatasets

def webscrapCvillageKeyboard():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.computervillage.com.bd/keyboard?page={}'.format(page)).text

        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
```

```
        productInfo = body.find_all('div', class_='grid-view-item')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find(
                'h4', class_='h4 grid-view-item__title text-truncate-2')
            title = product_name.a.text.replace(",", "").upper()
            product_price = product.find('span', class_='money')
            tk = product_price.text.replace(",", "").replace("৳", "")
            g_price = tk.replace('Call For Price', 'null')
            #print(tk)
            availability = product.find(
                'span', class_='sticker-stock-l bg-red')
            if availability:
                stock = availability.text
            else:
                stock = 'In Stock'
            #print(stock)
            data = title + "," + tk + "," + "COMPUTER VILLAGE" + \
                "," + "Keyboard" + "," + stock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)
def webscrapTechlandKeyboard():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
             20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.techlandbd.com/accessories/computer-keyboard?page={}'.format(page)).text
        # source_link = requests.get('https://www.techlandbd.com/shop-computer-mouse?page=40').text
        soup = BeautifulSoup(source_link, 'lxml')
```

```
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        product_thumb = body.find_all('div', class_='product-thumb')
        # using loop for grabing whole page data
        for product in product_thumb:
            product_name = product.find('div', class_='name')
            title = product_name.a.text.upper()
            product_price = product.find('div', class_='price')
            if product_price:
                tk = product_price.span.text
            else:
                tk = 'N/A'
            g_price = tk.replace('N/A', 'null')
            availability = product.find('div', class_='cart-group')
            stock = availability.a.text
            #print(stock)
            data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
                    "") + "," + "TECHLAND" + "," + "Keyboard" + "," + stock.replace("Add to Cart", "In Stock") + "," +
g_price.replace("৳", "").replace(",",
                    "") + "\n")
            f.write(data)
        time.sleep(5)
def webscrapStartechKeyboard():

    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
             13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.startech.com.bd/accessories/keyboards?page={}'.format(page)).text
```

```
    soup = BeautifulSoup(source_link, 'lxml')
    # search element from specified url html
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
    productInfo = body.find_all('div', class_='p-item')
    # using loop for grabing whole page data
    for product in productInfo:
      product_name = product.find('h4', class_='p-item-name')
      title = product_name.a.text.upper()
      product_price = product.find('div', class_='p-item-price')
      tk = product_price.span.text.replace(",", "").replace("৳", "")
      g_price = tk.replace('TBA', 'null')
      # int(tk)
      availability = product.find('div', class_='actions')
      stock = availability.span.text
      #print(stock)
      data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
          "") + "," + "STARTECH" + "," + "Keyboard" + "," + stock.replace("shopping_cart Buy Now", "In Stock") +
"," + g_price + "\n")
      f.write(data)
    time.sleep(5)
def webscrapRyansKeyboard():
  pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]
  # declared the url directory and store it in a variable
  # techland gpu section
  for page in pages:
    source_link = requests.get(
      'https://www.ryanscomputers.com/category/desktop-component-keyboard?page={}'.format(page)).text
    soup = BeautifulSoup(source_link, 'lxml')
    # search element from specified url html
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
```

```
    product_info = body.find_all('div', class_='card h-100')
    # using loop for grabing whole page data
    for product in product_info:
        product_name = product.find(
            'p', class_='card-text p-0 m-0 list-view-text')

        title = product_name.a.text.replace(",", "").upper()

        product_price = product.find(
            'p', class_='pr-text cat-sp-text pb-1')
        tk = product_price.text.replace("Tk", "").replace(",", "").strip()
        g_price = tk

        availability = product.find(
            'button', class_='btn grid-cart-btn cart-btn px-2 cat-cart-btn')
        stock = availability.text
        if stock:
            newstock = 'In Stock'
        else:
            newstock = 'Out of Stock'
        #print(newstock)
        data = title + "," + tk + "," + "RYANS" + "," + \
            "Keyboard" + "," + newstock + "," + g_price + "\n"
        f.write(data)
    time.sleep(5)


def webscrapSkylandKeyboard():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
             21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34]
    # declared the url directory and store it in a variable
    # techland gpu section
```

```
for page in pages:
    source_link = requests.get(
        'https://www.skyland.com.bd/product-category/accessories/keyboard/page/{}'.format(page)).text
    soup = BeautifulSoup(source_link, 'lxml')
    # search element from specified url html
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
    productInfo = body.find_all(
        'div', class_='box-text box-text-products text-center grid-style-2')
    # using loop for grabing whole page data
    for product in productInfo:
        product_name = product.find(
            'p', class_='name product-title woocommerce-loop-product__title')
        title = product_name.a.text.replace(",", "").upper()
        product_price = product.find(
            'span', class_='woocommerce-Price-amount amount')
        if product_price:
            tk = product_price.text.replace(",", "").replace("৳", "")
            tk = int(tk)
        else:
            tk = 'N/A'
        tk = str(tk)
        g_price = tk.replace('N/A', 'null')
        availability = product.find('div', class_='out-of-stock-label')
        if availability:
            stock = availability.text
            #print(stock)
        else:
            stock = 'In Stock'
        data = title + "," + tk + "," + "Skyland" + "," + \
            "Keyboard" + "," + stock + "," + g_price + "\n"
        f.write(data)
```

```
        time.sleep(5)


#monitordatasets


def webscrapCvillageMonitor():
    pages = [1, 2]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.computervillage.com.bd/monitor?page={}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all('div', class_='grid-view-item')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find(
                'h4', class_='h4 grid-view-item__title text-truncate-2')
            title = product_name.a.text.replace(",", "").upper()
            product_price = product.find('span', class_='money')
            tk = product_price.text.replace(",", "").replace("৳", "")
            g_price = tk.replace('Call For Price', 'null')
            #print(tk)
            availability = product.find(
                'span', class_='sticker-stock-l bg-red')


            if availability:
                stock = availability.text
            else:
                stock = 'In Stock'
```

```
            #print(stock)
            data = title + "," + tk + "," + "COMPUTER VILLAGE" + \
                "," + "Monitor" + "," + stock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)


def webscrapTechlandMonitor():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
             21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.techlandbd.com/computer-monitor?page={}'.format(page)).text
        # source_link = requests.get('https://www.techlandbd.com/shop-computer-mouse?page=40').text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html


        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        product_thumb = body.find_all('div', class_='product-thumb')
        # using loop for grabing whole page data
        for product in product_thumb:
            product_name = product.find('div', class_='name')
            title = product_name.a.text.upper()
            product_price = product.find('div', class_='price')
            if product_price:
                tk = product_price.span.text
            else:
                tk = 'N/A'
            g_price = tk.replace('N/A', 'null')
            availability = product.find('div', class_='cart-group')
```

```
            stock = availability.a.text
            #print(stock)
            data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
                    "") + "," + "TECHLAND" + "," + "Monitor" + "," + stock.replace("Add to Cart", "In Stock") + "," +
g_price.replace("৳", "").replace(",",
                    "") + "\n")
            f.write(data)
        time.sleep(5)
def webscrapStartechMonitor():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
             13, 14, 15, 16]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.startech.com.bd/monitor?page={}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all('div', class_='p-item')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find('h4', class_='p-item-name')
            title = product_name.a.text.upper()
            product_price = product.find('div', class_='p-item-price')
            tk = product_price.span.text.replace(",", "").replace("৳", "")
            g_price = tk.replace('TBA', 'null')
            # int(tk)
            availability = product.find('div', class_='actions')
            stock = availability.span.text
            #print(stock)
```

```
            data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
                    "") + "," + "STARTECH" + "," + "Monitor" + "," + stock.replace("shopping_cart Buy Now", "In Stock") + ","
+ g_price + "\n")
            f.write(data)
        time.sleep(5)
def webscrapRyansMonitor():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.ryanscomputers.com/category/monitor-all-monitor?page={}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        product_info = body.find_all('div', class_='card h-100')
        # using loop for grabing whole page data
        for product in product_info:
            product_name = product.find(
                'p', class_='card-text p-0 m-0 list-view-text')
            title = product_name.a.text.replace(",", "").upper()
            old_price = product.find(
                'del', class_='text-muted')
            product_price = product.find(
                'p', class_='pr-text cat-sp-text pb-1')
            if old_price:
                old_price.clear()
                #print(old_price)
            elif product_price:
                tk = product_price.text.replace(
                    "Tk", "").replace(",", "").strip()
```

```
            else:
                tk = 'N/A'
            #print(old_price)
            #print(tk)
            #tk = product_price.text.replace("Tk", "").replace(",", "").strip()
            g_price = tk
            availability = product.find(
                'button', class_='btn cart-btn')
            #stock = availability.text
            if availability:
                newstock = 'In Stock'
            else:
                newstock = 'Out of Stock'
            #print(newstock)
            data = title + "," + tk + "," + "RYANS" + "," + \
                "Monitor" + "," + newstock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)
def webscrapSkylandMonitor():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
             21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31]

    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.skyland.com.bd/product-category/components/monitor/page/{}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all(
```

```
            'div', class_='product-small box')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find(
                'p', class_='name product-title woocommerce-loop-product__title')
            title = product_name.a.text.replace(",", "").upper()
            product_price = product.find(
                'span', class_='woocommerce-Price-amount amount')
            if product_price:
                tk = product_price.text.replace(",", "").replace("৳", "")
                tk = int(tk)
            else:
                tk = 'N/A'
            tk = str(tk)
            g_price = tk.replace('N/A', 'null')
            availability = product.find('div', class_='out-of-stock-label')
            if availability:
                stock = availability.text
                #print(stock)
            else:
                stock = 'In Stock'
            data = title + "," + tk + "," + "Skyland" + "," + \
                "Monitor" + "," + stock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)
#gpudatasets


def webscrapCvillageGpu():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
```

```
    source_link = requests.get(
        'https://www.computervillage.com.bd/graphics-card?page={}'.format(page)).text
    soup = BeautifulSoup(source_link, 'lxml')
    # search element from specified url html
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
    productInfo = body.find_all('div', class_='grid-view-item')
    # using loop for grabing whole page data
    for product in productInfo:
        product_name = product.find(
            'h4', class_='h4 grid-view-item__title text-truncate-2')
        title = product_name.a.text.replace(",", "").upper()
        product_price = product.find('span', class_='money')
        tk = product_price.text.replace(",", "").replace("৳", "")
        g_price = tk.replace('Call For Price', 'null')
        #print(tk)
        availability = product.find(
            'span', class_='sticker-stock-l bg-red')
        if availability:
            stock = availability.text
        else:
            stock = 'In Stock'
        #print(stock)
        data = title + "," + tk + "," + "COMPUTER VILLAGE" + \
            "," + "GPU" + "," + stock + "," + g_price + "\n"
        f.write(data)
    time.sleep(5)


def webscrapTechlandGpu():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
             20, 21, 22, 23, 24, 25, 26]
    # declared the url directory and store it in a variable
```

```
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.techlandbd.com/pc-components/graphics-card?page={}'.format(page)).text
        # source_link = requests.get('https://www.techlandbd.com/shop-computer-mouse?page=40').text
        soup = BeautifulSoup(source_link, 'lxml')

        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        product_thumb = body.find_all('div', class_='product-thumb')
        # using loop for grabing whole page data
        for product in product_thumb:
            product_name = product.find('div', class_='name')
            title = product_name.a.text.upper()
            product_price = product.find('div', class_='price')
            if product_price:
                tk = product_price.span.text
            else:
                tk = 'N/A'
            g_price = tk.replace('N/A', 'null')
            availability = product.find('div', class_='cart-group')
            stock = availability.a.text
            #print(stock)
            data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
                    "") + "," + "TECHLAND" + "," + "GPU" + "," + stock.replace("Add to Cart", "In Stock") + "," +
g_price.replace("৳", "").replace(",",
                    "") + "\n")
            f.write(data)
        time.sleep(5)
def webscrapStartechGpu():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
```

```
        13, 14, 15, 16, 17, 18, 19, 20, 21]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.startech.com.bd/component/graphics-card?page={}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all('div', class_='p-item')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find('h4', class_='p-item-name')
            title = product_name.a.text.upper()
            product_price = product.find('div', class_='p-item-price')
            tk = product_price.span.text.replace(",", "").replace("৳", "")
            g_price = tk.replace('TBA', 'null')
            # int(tk)
            availability = product.find('div', class_='actions')
            stock = availability.span.text
            #print(stock)
            data = (title.replace(",", "") + "," + tk.replace("৳", "").replace(",",
                "") + "," + "STARTECH" + "," + "GPU" + "," + stock.replace("shopping_cart Buy Now", "In Stock") + "," +
g_price + "\n")
            f.write(data)
        time.sleep(5)
def webscrapRyansGpu():
    pages = [1, 2, 3, 4, 5]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
```

```
    source_link = requests.get(
        'https://www.ryanscomputers.com/category/desktop-component-graphics-
card?page={}'.format(page)).text
    soup = BeautifulSoup(source_link, 'lxml')
    # search element from specified url html
    body = soup.find('body')
    # here product-thumb is a css class so that i used it as a variable for better understanding
    product_info = body.find_all('div', class_='card h-100')
    # using loop for grabing whole page data
    for product in product_info:
        product_name = product.find(
            'p', class_='card-text p-0 m-0 list-view-text')
        title = product_name.a.text.replace(",", "").upper()
        product_price = product.find(
            'p', class_='pr-text cat-sp-text pb-1')
        old_price = product.find(
            'del', class_='text-muted')
        product_price = product.find(
            'p', class_='pr-text cat-sp-text pb-1')
        if old_price:
            old_price.clear()
            #print(old_price)
        elif product_price:
            tk = product_price.text.replace(
                "Tk", "").replace(",", "").strip()
        else:
            tk = 'N/A'
        tk = product_price.text.replace("Tk", "").replace(",", "").strip()
        g_price = tk
        availability = product.find(
            'button', class_='btn cart-btn')
        #stock = availability.text
```

```
            if availability:
                newstock = 'In Stock'
            else:
                newstock = 'Out of Stock'
            #print(newstock)
            data = title + "," + tk + "," + "RYANS" + "," + \
                "GPU" + "," + newstock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)
def webscrapSkylandGpu():
    pages = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17]
    # declared the url directory and store it in a variable
    # techland gpu section
    for page in pages:
        source_link = requests.get(
            'https://www.skyland.com.bd/product-category/components/graphics-card/page/{}'.format(page)).text
        soup = BeautifulSoup(source_link, 'lxml')
        # search element from specified url html
        body = soup.find('body')
        # here product-thumb is a css class so that i used it as a variable for better understanding
        productInfo = body.find_all(
            'div', class_='box-text box-text-products text-center grid-style-2')
        # using loop for grabing whole page data
        for product in productInfo:
            product_name = product.find(
                'p', class_='name product-title woocommerce-loop-product__title')
            title = product_name.a.text.replace(",", "").upper()
            product_price = product.find(
                'span', class_='woocommerce-Price-amount amount')
            if product_price:
                tk = product_price.text.replace(",", "").replace("৳", "")
                tk = int(tk)
```

```
            else:
                tk = 'N/A'
            tk = str(tk)
            g_price = tk.replace('N/A', 'null')
            availability = product.find('div', class_='out-of-stock-label')
            if availability:
                stock = availability.text
                #print(stock)
            else:
                stock = 'In Stock'
            data = title + "," + tk + "," + "Skyland" + "," + \
                "GPU" + "," + stock + "," + g_price + "\n"
            f.write(data)
        time.sleep(5)
#gpu
threading.Thread(target=webscrapCvillageGpu).start()
threading.Thread(target=webscrapTechlandGpu).start()
threading.Thread(target=webscrapStartechGpu).start()
threading.Thread(target=webscrapRyansGpu).start()
threading.Thread(target=webscrapSkylandGpu).start()

#monitor
threading.Thread(target=webscrapCvillageMonitor).start()
threading.Thread(target=webscrapTechlandMonitor).start()
threading.Thread(target=webscrapStartechMonitor).start()
threading.Thread(target=webscrapRyansMonitor).start()
threading.Thread(target=webscrapSkylandMonitor).start()
#keyboard
threading.Thread(target=webscrapCvillageKeyboard).start()
threading.Thread(target=webscrapTechlandKeyboard).start()
threading.Thread(target=webscrapStartechKeyboard).start()
threading.Thread(target=webscrapRyansKeyboard).start()
```

```
threading.Thread(target=webscrapSkylandKeyboard).start()
#mouse
threading.Thread(target=webscrapCvillageMouse).start()
threading.Thread(target=webscrapTechlandMouse).start()
threading.Thread(target=webscrapStartechMouse).start()
threading.Thread(target=webscrapRyansMouse).start()
threading.Thread(target=webscrapSkylandMouse).start()
#ram
threading.Thread(target=webscrapCvillageRam).start()
threading.Thread(target=webscrapTechlandRam).start()
threading.Thread(target=webscrapStartechRam).start()
threading.Thread(target=webscrapRyansRam).start()
threading.Thread(target=webscrapSkylandRam).start()
```

File Name: Filtered_by_Category.py

```
import io
from operator import index
from textwrap import indent
import threading
import pandas as pd
from io import StringIO
#df = pd.read_csv('filteredAllData.csv')
#read = df.groupby(['Category']).mean()
#print(read)
#newread = df[df['Category'].str.contains("Keyboard")]
#print(newread)
#newread = newread.drop('Unnamed: 0', axis=1)
#filename = "productbyKeyboard.csv"
#f = open(filename, 'w', encoding='utf-8-sig')
#newread.to_csv("productbyCategory.csv" , index_col=[0])
#read2 = pd.read_csv(io.StringIO(newread.to_csv()), index_col=[0])
#newfile = newread.to_csv('productbyKeyboard.csv', index=False)
def productbyKeyboard():
```

```
    df = pd.read_csv('filteredAllData.csv')
    read = df.groupby(['Category']).mean()
    #print(read)
    newread = df[df['Category'].str.contains("Keyboard")]
    print(newread)
    newread = newread.drop('Unnamed: 0', axis=1)
    filename = "productbyKeyboard.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    #newread.to_csv("productbyCategory.csv" , index_col=[0])
    read2 = pd.read_csv(io.StringIO(newread.to_csv()), index_col=[0])
    newfile = newread.to_csv('productbyKeyboard.csv', index=False)
def productbyMouse():
    df = pd.read_csv('filteredAllData.csv')
    read = df.groupby(['Category']).mean()
    #print(read)
    newread = df[df['Category'].str.contains("Mouse")]
    print(newread)
    newread = newread.drop('Unnamed: 0', axis=1)
    filename = "productbyMouse.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    #newread.to_csv("productbyCategory.csv" , index_col=[0])
    read2 = pd.read_csv(io.StringIO(newread.to_csv()), index_col=[0])
    newfile = newread.to_csv('productbyMouse.csv', index=False)
def productbyMonitor():
    df = pd.read_csv('filteredAllData.csv')
    read = df.groupby(['Category']).mean()
    #print(read)
    newread = df[df['Category'].str.contains("Monitor")]
    print(newread)
    newread = newread.drop('Unnamed: 0', axis=1)
    filename = "productbyMonitor.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
```

```
    #newread.to_csv("productbyCategory.csv" , index_col=[0])
    read2 = pd.read_csv(io.StringIO(newread.to_csv()), index_col=[0])
    newfile = newread.to_csv('productbyMonitor.csv', index=False)
def productbyRam():
    df = pd.read_csv('filteredAllData.csv')
    read = df.groupby(['Category']).mean()
    #print(read)
    newread = df[df['Category'].str.contains("Ram")]
    print(newread)
    newread = newread.drop('Unnamed: 0', axis=1)
    filename = "productbyRam.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    #newread.to_csv("productbyCategory.csv" , index_col=[0])
    read2 = pd.read_csv(io.StringIO(newread.to_csv()), index_col=[0])
    newfile = newread.to_csv('productbyRam.csv', index=False)
def productbyGPU():
    df = pd.read_csv('filteredAllData.csv')
    read = df.groupby(['Category']).mean()
    #print(read)
    newread = df[df['Category'].str.contains("GPU")]
    print(newread)
    newread = newread.drop('Unnamed: 0', axis=1)
    filename = "productbyGpu.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    #newread.to_csv("productbyCategory.csv" , index_col=[0])
    read2 = pd.read_csv(io.StringIO(newread.to_csv()), index_col=[0])
    newfile = newread.to_csv('productbyGpu.csv', index=False)
threading.Thread(target=productbyGPU).start()
threading.Thread(target=productbyRam).start()
threading.Thread(target=productbyMonitor).start()
threading.Thread(target=productbyMouse).start()
threading.Thread(target=productbyKeyboard).start()
```

File Name: filtered_by_price_range.py

```
import pandas as pd
import threading
def mouseRange():
    df = pd.read_csv('productbyMouse.csv')
    #FINDING MAX AND MIN
    max = df['G-Price'].max()
    min = df['G-Price'].min()
    print(max)
    print(min)
    mid = (max-min)/2
    print(mid)
    low = max/3
    medium = 2*low
    high = max
    print(low)
    print(medium)
    print(high)
    def midRange():
        midRange = df[df['G-Price'].between(min, mid)]
        #print(midrange)
        filename = "midRange_mouse.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        midRange.to_csv("midRange_mouse.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=midRange).start()
    def maxRange():
        maxRange = df[df['G-Price'].between(mid, max)]
        #print(midrange)
        filename = "maxRange_mouse.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
```

```
        maxRange.to_csv("maxRange_mouse.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=maxRange).start()
    def lowRange():
        lowRange = df[df['G-Price'].between(min, low)]
        #print(midrange)
        filename = "lowRange_mouse.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        lowRange.to_csv("lowRange_mouse.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=lowRange).start()
    def mediumRange():
        mediumRange = df[df['G-Price'].between(low, medium)]
        #print(midrange)
        filename = "mediumRange_mouse.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        mediumRange.to_csv("mediumRange_mouse.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=mediumRange).start()
    def highRange():
        highRange = df[df['G-Price'].between(medium, high)]
        #print(midrange)
        filename = "highRange_mouse.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        highRange.to_csv("highRange_mouse.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=highRange).start()
threading.Thread(target=mouseRange).start()
def keyboardRange():
    df = pd.read_csv('productbyKeyboard.csv')
    #FINDING MAX AND MIN
    max = df['G-Price'].max()
```

```
min = df['G-Price'].min()
print(max)
print(min)
mid = (max-min)/2
print(mid)
low = max/3
medium = 2*low
high = max
print(low)
print(medium)
print(high)
def midRange():
    midRange = df[df['G-Price'].between(min, mid)]
    #print(midrange)
    filename = "midRange_keyboard.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    midRange.to_csv("midRange_keyboard.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=midRange).start()
def maxRange():
    maxRange = df[df['G-Price'].between(mid, max)]
    #print(midrange)
    filename = "maxRange_keyboard.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    maxRange.to_csv("maxRange_keyboard.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=maxRange).start()
def lowRange():
    lowRange = df[df['G-Price'].between(min, low)]
    #print(midrange)
    filename = "lowRange_keyboard.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
```

```
        lowRange.to_csv("lowRange_keyboard.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=lowRange).start()
    def mediumRange():
        mediumRange = df[df['G-Price'].between(low, medium)]
        #print(midrange)
        filename = "mediumRange_keyboard.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        mediumRange.to_csv("mediumRange_keyboard.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=mediumRange).start()
    def highRange():
        highRange = df[df['G-Price'].between(medium, high)]
        #print(midrange)
        filename = "highRange_keyboard.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        highRange.to_csv("highRange_keyboard.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=highRange).start()
threading.Thread(target=keyboardRange).start()
def monitorRange():
    df = pd.read_csv('productbyMonitor.csv')
    #FINDING MAX AND MIN
    max = df['G-Price'].max()
    min = df['G-Price'].min()
    print(max)
    print(in)
    mid = (max-min)/2
    print(mid)
    low = max/3
    medium = 2*low
    high = max
```

```
print(low)
print(medium)
print(high)
def midRange():
    midRange = df[df['G-Price'].between(min, mid)]
    #print(midrange)
    filename = "midRange_monitor.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    midRange.to_csv("midRange_monitor.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=midRange).start()
def maxRange():
    maxRange = df[df['G-Price'].between(mid, max)]
    #print(midrange)
    filename = "maxRange_monitor.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    maxRange.to_csv("maxRange_monitor.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=maxRange).start()
def lowRange():
    lowRange = df[df['G-Price'].between(min, low)]
    #print(midrange)
    filename = "lowRange_monitor.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    lowRange.to_csv("lowRange_monitor.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=lowRange).start()
def mediumRange():
    mediumRange = df[df['G-Price'].between(low, medium)]
    #print(midrange)
    filename = "mediumRange_monitor.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
```

```
        mediumRange.to_csv("mediumRange_monitor.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=mediumRange).start()
    def highRange():
        highRange = df[df['G-Price'].between(medium, high)]
        #print(midrange)
        filename = "highRange_monitor.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        highRange.to_csv("highRange_monitor.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=highRange).start()
threading.Thread(target=monitorRange).start()
def ramRange():
    df = pd.read_csv('productbyMonitor.csv')
    #FINDING MAX AND MIN
    max = df['G-Price'].max()
    min = df['G-Price'].min()
    print(max)
    print(min)
    mid = (max-min)/2
    print(mid)
    low = max/3
    medium = 2*low
    high = max
    print(low)
    print(medium)
    print(high)
    def midRange():
        midRange = df[df['G-Price'].between(min, mid)]
        #print(midrange)
        filename = "midRange_Ram.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
```

```
    midRange.to_csv("midRange_Ram.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=midRange).start()
def maxRange():
    maxRange = df[df['G-Price'].between(mid, max)]
    #print(midrange)
    filename = "maxRange_Ram.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    maxRange.to_csv("maxRange_Ram.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=maxRange).start()
def lowRange():
    lowRange = df[df['G-Price'].between(min, low)]
    #print(midrange)
    filename = "lowRange_Ram.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    lowRange.to_csv("lowRange_Ram.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=lowRange).start()
def mediumRange():
    mediumRange = df[df['G-Price'].between(low, medium)]
    #print(midrange)
    filename = "mediumRange_Ram.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    mediumRange.to_csv("mediumRange_Ram.csv")
    #df = pd.read_csv('testmouse.csv')
threading.Thread(target=mediumRange).start()
def highRange():
    highRange = df[df['G-Price'].between(medium, high)]
    #print(midrange)
    filename = "highRange_Ram.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
```

```
        highRange.to_csv("highRange_Ram.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=highRange).start()
threading.Thread(target=ramRange).start()
def gpuRange():
    df = pd.read_csv('productbyMonitor.csv')
    #FINDING MAX AND MIN
    max = df['G-Price'].max()
    min = df['G-Price'].min()
    print(max)
    print(min)
    mid = (max-min)/2
    print(mid)
    low = max/3
    medium = 2*low
    high = max
    print(low)
    print(medium)
    print(high)
    def midRange():
        midRange = df[df['G-Price'].between(min, mid)]
        #print(midrange)
        filename = "midRange_Gpu.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        midRange.to_csv("midRange_Gpu.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=midRange).start()
    def maxRange():
        maxRange = df[df['G-Price'].between(mid, max)]
        #print(midrange)
        filename = "maxRange_Gpu.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
```

```
        maxRange.to_csv("maxRange_Gpu.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=maxRange).start()
    def lowRange():
        lowRange = df[df['G-Price'].between(min, low)]
        #print(midrange)
        filename = "lowRange_Gpu.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        lowRange.to_csv("lowRange_Gpu.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=lowRange).start()
    def mediumRange():
        mediumRange = df[df['G-Price'].between(low, medium)]
        #print(midrange)
        filename = "mediumRange_Gpu.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        mediumRange.to_csv("mediumRange_Gpu.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=mediumRange).start()
    def highRange():
        highRange = df[df['G-Price'].between(medium, high)]


        #print(midrange)
        filename = "highRange_Gpu.csv"
        f = open(filename, 'w', encoding='utf-8-sig')
        highRange.to_csv("highRange_Gpu.csv")
        #df = pd.read_csv('testmouse.csv')
    threading.Thread(target=highRange).start()
threading.Thread(target=gpuRange).start()
```

File Name: stock_count.py

```
import pandas as pd
```

```
import threading
import os
def highRangedMouse_Count():
    df = pd.read_csv('highRange_mouse.csv')
    counts = pd.value_counts(df['Shop_Name'])
    print(counts)
    filename = "./countfile/highRange_mouse_Count.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    #headers = "Shop_Name,Counts\n"
    #f.write(headers)
    counts.to_csv("./countfile/highRange_mouse_Count.csv", header=None)

    newread = pd.read_csv("./countfile/highRange_mouse_Count.csv")

    #print(newread)
    finalread = pd.read_csv("./countfile/highRange_mouse_Count.csv",
                        names=['shop name', 'counts'])
    #print(finalread)
    finalread.to_csv("./countfile/highRange_mouse_Count.csv")
threading.Thread(target=highRangedMouse_Count).start()
def midRangedMouse_Count():
    df = pd.read_csv('mediumRange_mouse.csv')
    counts = pd.value_counts(df['Shop_Name'])
    print(counts)
    filename = "./countfile/mediumRange_mouse_Count.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    #headers = "Shop_Name,Counts\n"
    #f.write(headers)
    counts.to_csv("./countfile/mediumRange_mouse_Count.csv", header=None)
    newread = pd.read_csv("./countfile/mediumRange_mouse_Count.csv")
    #print(newread)
    finalread = pd.read_csv("./countfile/mediumRange_mouse_Count.csv",
```

```
                    names=['shop name', 'counts'])
    #print(finalread)
    finalread.to_csv("./countfile/mediumRange_mouse_Count.csv")
threading.Thread(target=midRangedMouse_Count).start()
def lowRangedMouse_Count():
    df = pd.read_csv('lowRange_mouse.csv')
    counts = pd.value_counts(df['Shop_Name'])
    print(counts)
    filename = "./countfile/lowRange_mouse_Count.csv"
    f = open(filename, 'w', encoding='utf-8-sig')
    #headers = "Shop_Name,Counts\n"
    #f.write(headers)
    counts.to_csv("./countfile/lowRange_mouse_Count.csv", header=None)
    newread = pd.read_csv("./countfile/lowRange_mouse_Count.csv")
    #print(newread)
    finalread = pd.read_csv("./countfile/lowRange_mouse_Count.csv",
                    names=['shop name', 'counts'])
    #print(finalread)
    finalread.to_csv("./countfile/lowRange_mouse_Count.csv")
threading.Thread(target=lowRangedMouse_Count).start()
```

File Name: barchart.py

```
import pandas as pd
import matplotlib.pyplot as plot
import numpy as np
df = pd.read_csv('G:\CSE498R\Final\count\lowRange_Ram.csv')
new_df= df[["Shop_Name", "Availability"]]
condition = [
    (new_df['Availability'] == 'In Stock') & (
        new_df['Shop_Name'] == 'TECHLAND'),
    (new_df['Availability'] == 'In Stock') & (
```

```
        new_df['Shop_Name'] == 'RYANS'),
    (new_df['Availability'] == 'In Stock') & (
        new_df['Shop_Name'] == 'STARTECH'),
    (new_df['Availability'] == 'In Stock') & (
        new_df['Shop_Name'] == 'COMPUTER VILLAGE'),
    (new_df['Availability'] == 'In Stock') & (
        new_df['Shop_Name'] == 'Skyland'),
]
values = ['Techland','Ryans','Startech','Computer Village', 'Skyland']
new_df['Stock Group'] = np.select(condition, values)
#print(new_df.tail(100))
counts = pd.value_counts(new_df['Stock Group'])
print(counts)
counts.to_csv('stockCount.csv',  header=None)
finalcounts = pd.read_csv("stockCount.csv",names=['shop name', 'Stock Counts'])
finalcounts.to_csv('stockCount.csv')
finalcounts.drop(3, axis=0, inplace=True)
finalcounts.to_csv('stockCount.csv')
y = finalcounts['Stock Counts']
x = finalcounts['shop name']
plot.ylabel('Stock Counts', fontsize=14)

plot.xlabel('Shop Name', fontsize=14)
plot.bar(x, y)
plot.text(0.0, 320.0, 'Barchart of Shop vs Stock count from low price ram data')
plot.show()
```