**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

**Introduction 8**

Background 8

Problem Statement8

Contribution9

**Feasibility Study9**

Problem Solution9

Dataset Representation9

**Description10**

**Proposed Solution11**

Platform11

Software11

Packages11

Proposed Solution Flowchart12

Dataset Selection13

Pre-Processing13

Punctuation Removal13

**Impacts14**

Technical Analysis14

Financial Analysis14

Environmental Analysis14

Societal Analysis14

**Result Analysis15**

Methodology15

Discussion19

**Design Impacts & Cost20**

Time Cost20

Cost Prediction 20

Future Work21

**Conclusion21**

**Acknowledgement22**

**Appendix 23**

Code 23-66

**List of Figures**

1. Proposed Solution Block Diagram
2. Corrupted Data
3. Dataset
4. Stock counts in high price range
5. Stock count in low price range
6. Stock Counts in medium price range

## 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

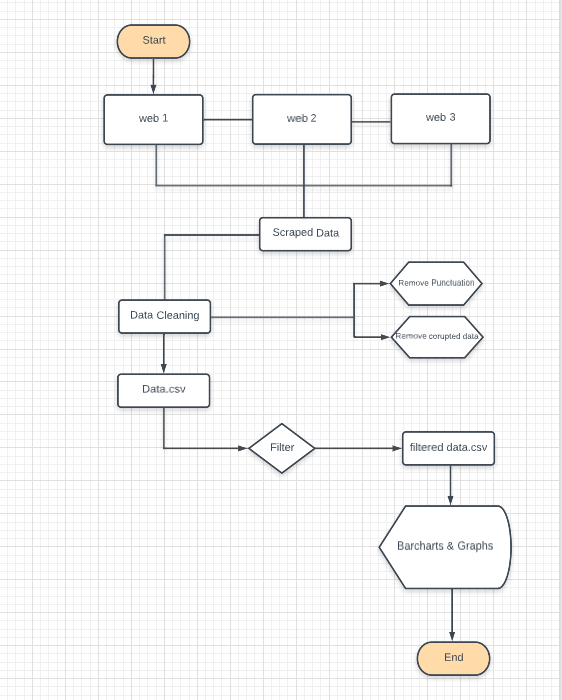


Fig 1: Proposed Solution Block Diagram

## 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.



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.

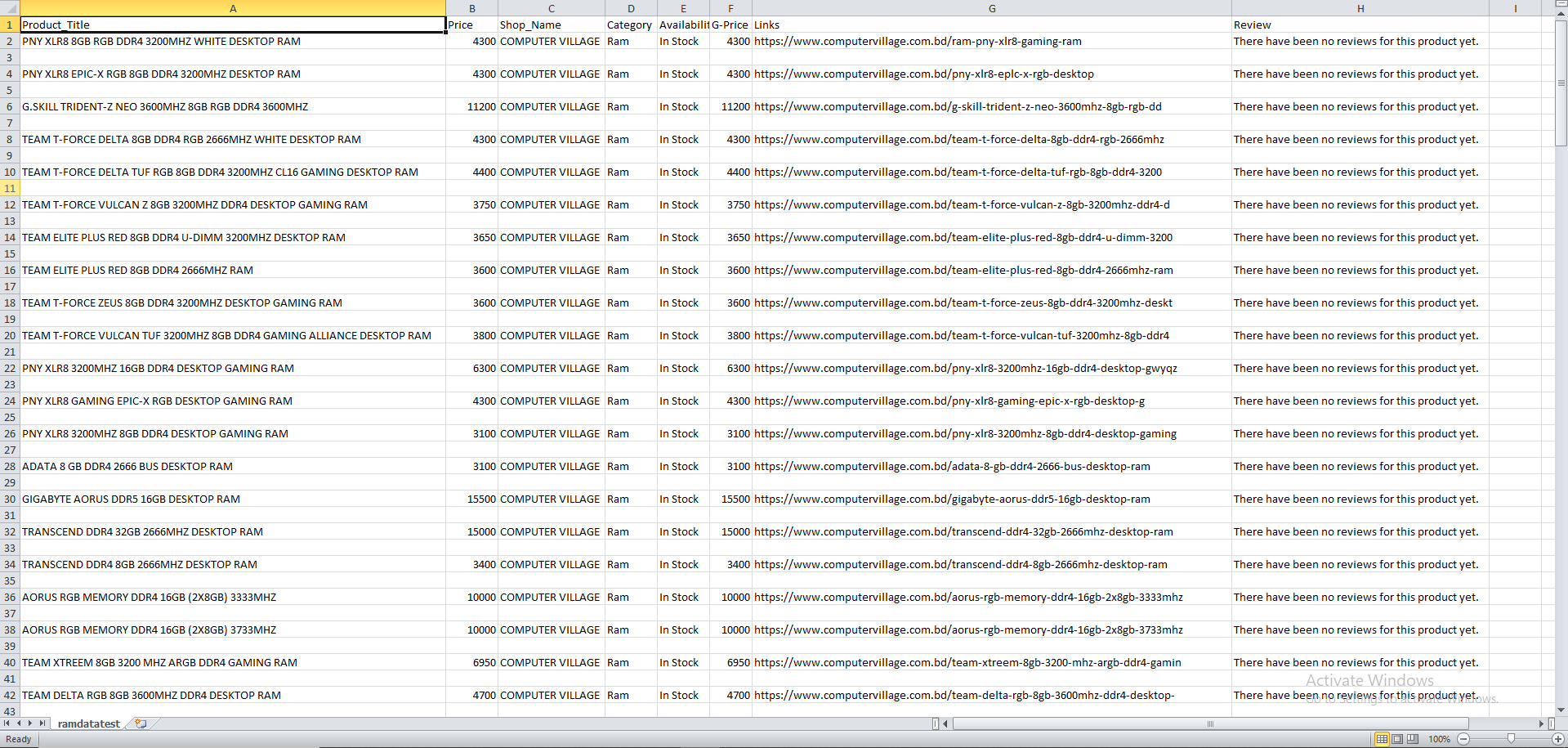


Fig.3 Dataset

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. 

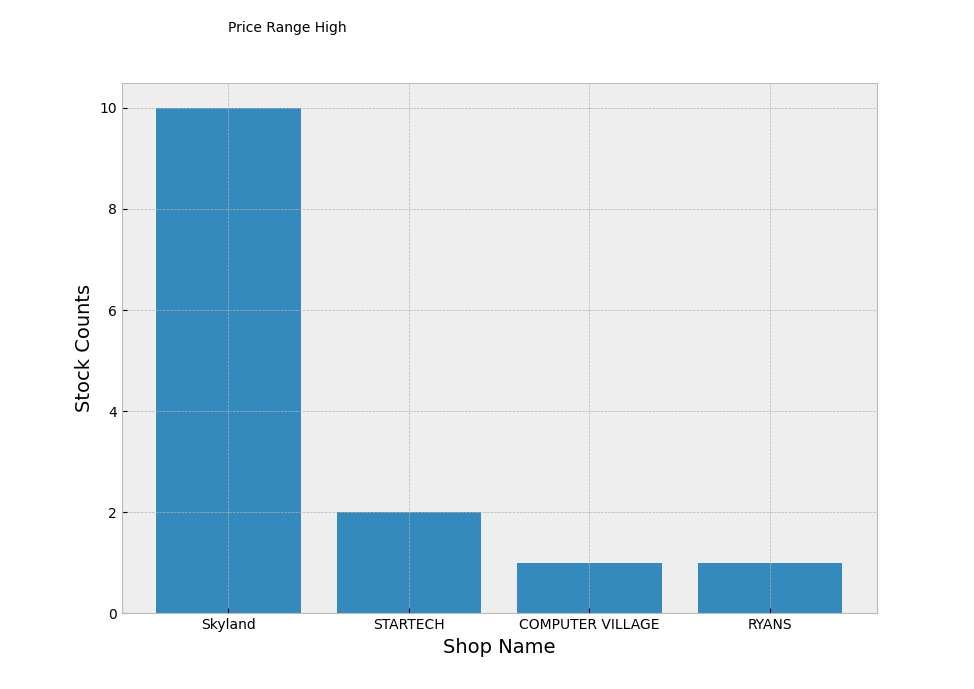


Fig.4 Stock counts in high price range

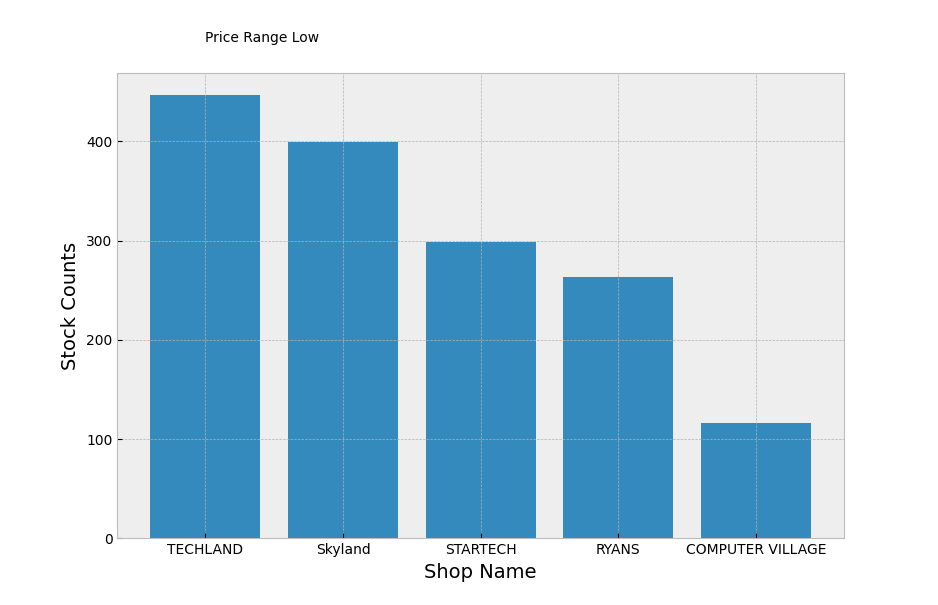


Fig.5 Stock count in low price range

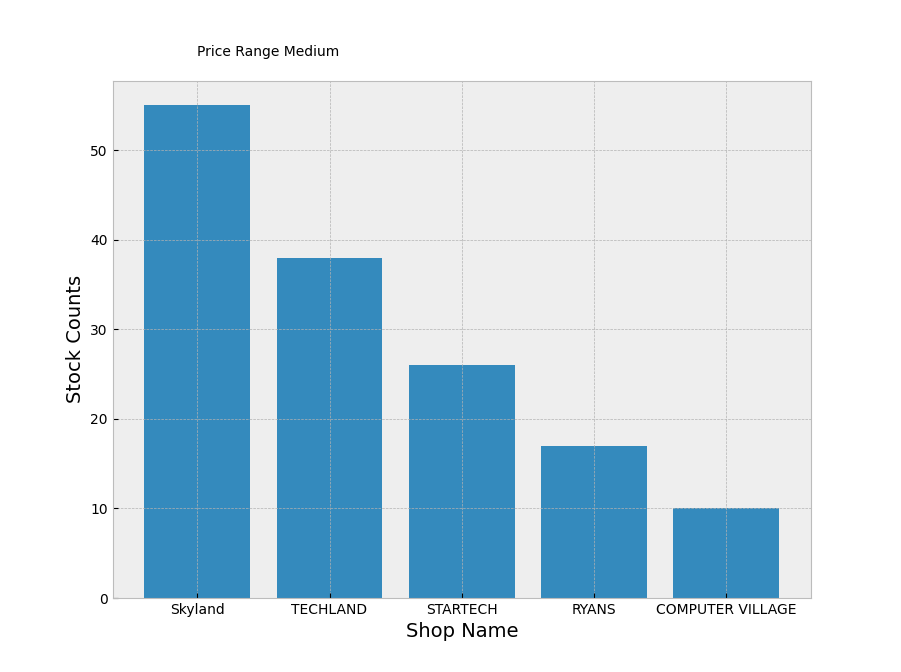


Fig.6 Stock Counts in medium price range

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()