WEB SCRAPING BEST BUY FOR TV INDUSTRY INSIGHTS:

A MARKET RESEARCH PROJECT

TEAM MEMBERS:

Chi-En (Ian), Hwang

Sanka Naga Nitesh

Jyothika Mohan

EXECUTIVE SUMMARY

As a market researcher in a digital appliances company "XYZ", I want to explore which technologies in the TV industry are the most popular in the US. To fulfill the task, my team and I are initiating the project of collecting relevant information from the internet automatically, to make sure the decisions and strategies XYZ makes are data-driven and fit the market preferences.

This project involved scraping data from BestBuy on various TVs. The data was scraped at two levels, with level 1 scraping involving collecting information such as the name, brand, features, model number, etc of all TVs. Level 2 scraping involved accessing each product link to gather data such as the product name, Skuid, return period, warranty, and installment payment options.

The data was primarily scraped using Beautiful Soup and stored in MongoDB. The level 1 data was stored in a collection called "best_buy_collection" and the level 2 data was stored in a new collection called "best_buy_product_level_collection". The collected data can be used to gain insights into consumer preferences, trends, and patterns in the TV market, as well as to inform pricing strategies and identify opportunities for innovation.

<u>BACKGROUND AND DOMAIN KNOWLEDGE</u>

As a market researcher in the digital appliances company "XYZ", we are planning to launch a new TV product under their digital devices vertical in the US as a part of our new initiatives. However, before introducing any new product to the market, it's essential to understand the industry trends and customer preferences to develop a reliable and competitive product that will meet the market demand.

The TV industry is a vast market valued at billions of dollars, with the largest markets in North America, Europe, and Asia. The industry includes various stakeholders, including manufacturers, retailers, content creators, and service providers. The past decade has seen many changes in TV technology, including the rise of smart TVs, 4K and 8K resolutions, and HDR imaging, which have improved the viewing experience for consumers.

The pricing data of existing TVs in the market can also be analyzed to observe the price range preferred by the public in general. This data will help your company price the products accordingly and compete with other brands in the market. Moreover, it will also help our firm to understand the customers and the segment that our company has to target, which will aid in forming your market strategy.

Based on this data, we can advise the product design and development team to create a TV product that meets the market demand, such as incorporating popular technologies and features that customers are seeking. This may involve cost optimization or introducing some differentiation in your product to help you stand out from the competition.

In conclusion, understanding the TV industry's trends, customer preferences, and the competitive landscape is crucial to developing a reliable and competitive product. Through market research, we can gather valuable insights that will guide your product development and market strategy and help your company succeed in the highly

DATA SOURCES

Scraping TV data from the Best Buy website is a promising strategy for gathering a wide range of information about TVs. With this approach, the firm can extract detailed data on various aspects of TVs, including brand, model, price, size, features, and customer reviews. This data can provide valuable insights into the competitive landscape and help the firm make informed decisions about its TV-related business strategies.

Scraping the Best Buy website can give the firm access to a vast database of TV-related information. The website features an extensive range of TVs from various manufacturers, each with detailed product descriptions and specifications. By scraping this data, the firm can collect information on the different brands and models of TVs available, along with their prices and features.

Moreover, the Best Buy website also features customer reviews and ratings for each product. This information can be particularly valuable, as it provides insight into customer satisfaction levels, common issues, and areas for improvement. By analyzing these reviews, the firm can gain a better understanding of the needs and preferences of its target audience and make informed decisions about product development and marketing strategies.

Overall, scraping TV data from the Best Buy website can provide the firm with a wealth of information that can help it make data-driven decisions about its TV-related business strategies. By analyzing this data, the firm can identify opportunities for growth, assess the competitive landscape, and gain a deeper understanding of its target audience.

DESCRIPTION OF WEB-SCRAPING ROUTINES

1. The team started by creating an object “Best_buy_scrap” for the entire scraping and storing process. The user only has to input the name of the product to initiate an object, for instance, “TV” for this project. [1]
2. After initiation, the user can apply the function “execute_scrapping” to [2]

* The user will have to specify the folder name (directory) to store all the information.

1) Access all the search results of the underlying product on Best Buy and store them in the folder “best_buy_folder” [3].

   * We utilize Best Buy’s **get request** to activate the search feature and parse information into HTML format with **Beautifulsoup,** after parsing, we store each page in a folder to keep track of the information.

2) Access each individual product shown on the search result pages and store their advanced information (information not shown on the search results) in a designated folder. [4]

   * We use the information stored from the search page(web URL) to request more information from each product and store the information in a designated folder.

3) Handle information: To process the information, we create functions that utilize **str functions** to filter out redundant symbols [5]

On top of the current routine, we can access more information about the expert rating, expert summary review, expert reviews, ratings for picture, sound and brightness quality, customer reviews and many more[6]

EXPLANATION OF THE DATASET

To gather relevant information about the TVs sold on the Best Buy website, we have employed web scraping techniques. In the first level of scraping, we extracted data from all the pages containing the information we were interested in. Specifically, we scraped the TV name, brand, features such as resolution and voice assist, model number, SKU, product link, star rating, number of reviews, price, original price, and discount.

- **header**

  The header is basically the name of the TV that can help us identify the brand, model, and other information about the product. This can be useful for tracking sales and popularity over time and also helps us understand the naming strategies among different brands, for example, which traits to place in the front of the name to capture customer eyesight.
- **brand**

  The brand of the TV can give us insights into consumer preferences and brand loyalty. It can also help identify trends and patterns across different brands.
- **features (like resolution, Voice Assist, Smart or not Smart etc)**

  The features can give us insights into which technologies are most popular among consumers. It also helps us identify gaps in the market and opportunities for innovation.
- **model**

  The model number can help us identify between different models, combining it with SKU, we'll be able to create unique keys to label different products.
- **sku (a unique ID)**

The SKU can help us identify between different products, combining it with “Model number”, we’ll be able to create unique keys to label different products.

- **product link**

  The product link can help us access the profile of the product easily, we can learn more about the underlying product with it, for example, the outlook and other graphics-related features that are not being scraped.

- **stars**

  The stars is basically the ratings that can give us insights into consumer satisfaction and the quality of the product. It can also be useful for identifying which products or brands are most popular and which features are most important to consumers.

- **number_of_reviews**

  The number of reviews can give us insights into consumer preferences and the popularity of the product, we can combine it with a star rating to see if the product is popular in a good way or the other way around. It can also be useful for identifying trends and patterns over time.

- **price**

  The price can give us insights into consumer preferences and the value of the product. It helps us formulate a pricing model in regard to product specifications and brands. It can also be useful for identifying trends and patterns across different brands and models.

- **original_price**

  The original price can be useful for tracking sales and discounts over time.

- **price_reduction (aka discount)**

  Discount = Original price - Price. This entry helps us identify the price deduction at first glance.

As a part of second-level crapping, we accessed each product's individual page using the product link and extracted additional information such as the product name, return period, warranty, and payment through installment option (if available)

- **product_name**

  We keep “Product name” to make this advanced information table more readable, with Skuid as the primary key, users can still have some fundamental understanding of the underlying product from “Product name”.
- **skuid**

  The Skuid will be the primary key for us to perform mapping with the level 1 product information table.
- **installment**

  Installment indicates if the payment can be done in installments. This option can give us insights into how willing consumers are to pay for a product over time, as well as how much they are willing to pay overall. This can be useful for identifying pricing trends and patterns, as well as opportunities for offering financing options to consumers who may not be able to afford the product upfront.
- **return period**

  The return period can give us insights into how confident the retailer is in the product, as well as how easy it is for consumers to return the product if they are not satisfied. This can be useful for identifying popular products that are also reliable and well-regarded by retailers and consumers alike.
- **warrant**

  The warranty can give us insights into the level of confidence the manufacturer has in the product, as well as how much support consumers can expect if something goes wrong.

This data allowed us to gain a more comprehensive understanding of each product, including factors that might impact consumer decision-making such as the return policy or warranty.

DATABASE DESIGN CHOICES

For the purpose of storing our scraped data, we have chosen MongoDB as our database because MongoDB's flexible data model allows data to be stored in a JSON-like format that aligns with the structure of the scraped data, making it easy to store and manipulate. Secondly, MongoDB is designed to scale horizontally, which means it can handle large amounts of data and traffic by adding more servers to a cluster, making it ideal for storing scraped data that may grow over time. Also, MongoDB is designed to be fast and efficient, making it well-suited for handling high-volume, read-heavy workloads. Moreover, its integration with many popular programming languages is seamless because of the presence of drivers. Overall, MongoDB's flexibility, scalability, speed, ease of integration, and ad hoc querying capabilities make it a good choice for storing scraped data from websites.

For our project, we have created a MongoDB database called “best_buy_db” which houses 2 collections. The level 1 scraping data is stored in the "best_buy_collection" and the level 2 scraping data is stored in the "best_buy_product_level_collection" within the same database [7]. Additionally, we downloaded all the views and individual product pages as HTML files for reference.

Also, a point to note that the header and sku features in “best_buy_collection” and the product name and skuid features in the “best_buy_product_level_collection” can be used to join both collections which can be helpful during further analysis.

<u>BUSINESS VALUE</u>

The data we have extracted can help answer fundamental questions about customer preferences and market trends, which are critical to developing and launching new products. The business value that it will create is as follows

Studying demand for specific features: Understanding the features that customers value the most is crucial to create products that cater to their needs and preferences. By analyzing the data we have collected, we can identify which technologies are most important to customers and which ones are driving more engagement. This will help us make informed decisions about which features to prioritize in our products and services, ultimately increasing their market appeal and customer satisfaction.

Pricing model: Pricing a product correctly is essential for maximizing profits and staying competitive in the market. By leveraging the information we have collected, we can develop a pricing model that takes into account the different specifications of a product and outputs a reasonable price range. This will enable the company to make pricing decisions based on the value that the product offers to customers, rather than simply relying on intuition or guesswork.

Marketing research: In the ever-evolving consumer electronics market, it is important to stay on top of changing customer preferences and market trends. By analyzing the data we have collected, we can gain insights into the changes in trends related to TV brands, screen sizes, resolutions, and other TV-related features. This can help us identify opportunities for innovation and the development of new products that align with customer preferences. Additionally, it can also help us fine-tune our marketing strategies to better reach and engage our target audience.

Overall, the data can be used to inform business decisions that can lead to increased market share and profitability.

SUMMARY AND CONCLUSIONS

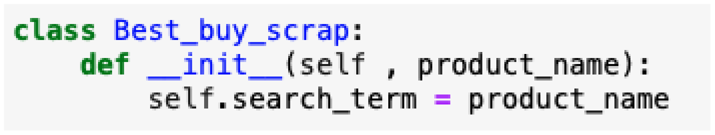
Overall, our web scraping strategy has provided us with a wealth of information that can inform our business decisions, such as product development and marketing strategies. By analyzing this data, we can identify patterns and trends, understand consumer preferences, and make data-driven decisions.

Having this data will allow us to identify trends and patterns across brands and models, gain insights into consumer preferences and brand loyalty, and identify gaps and opportunities in the market. For example, we could track sales and popularity over time and determine which features were most important to consumers.

As a market researcher in a digital appliances company "XYZ", my goal is to explore the most popular TV technologies in the US and observe the preferred price range of the general public. The data collected will help the product design and development team create a reliable and competitive product, form a strategy, and develop a pricing model. The research data will also capture customer preferences and changing trends to stay competitive in the market.

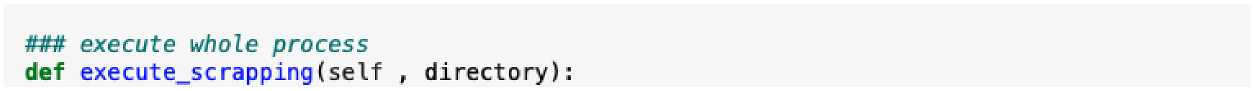
APPENDIX

[1]



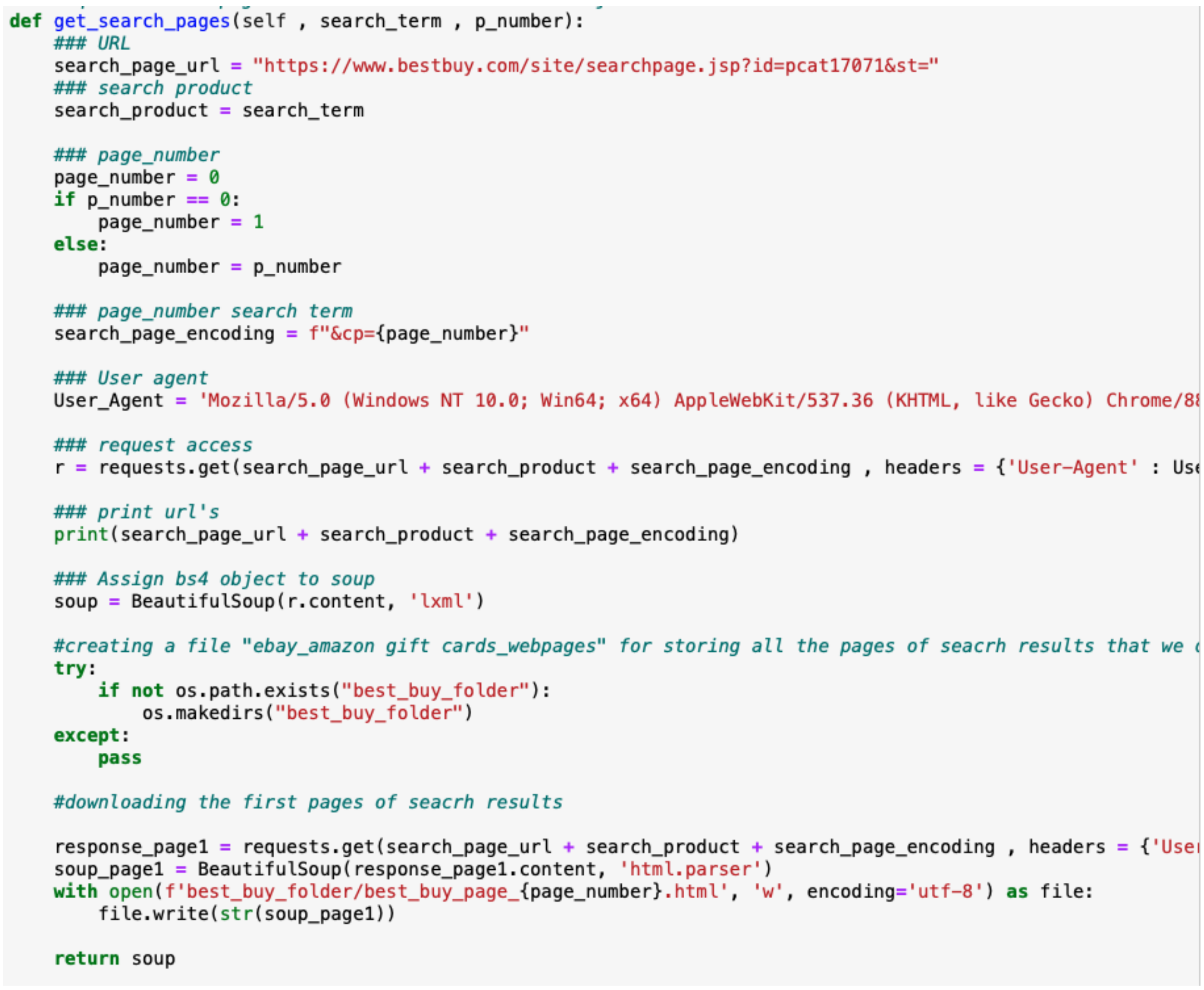
```
class Best_buy_scrap:
    def __init__(self , product_name):
        self.search_term = product_name
```

[2]



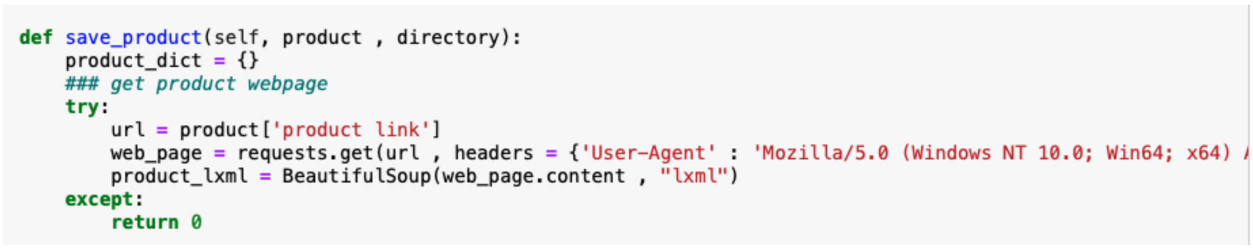
```
### execute whole process
def execute_scrapping(self , directory):
```

[3]



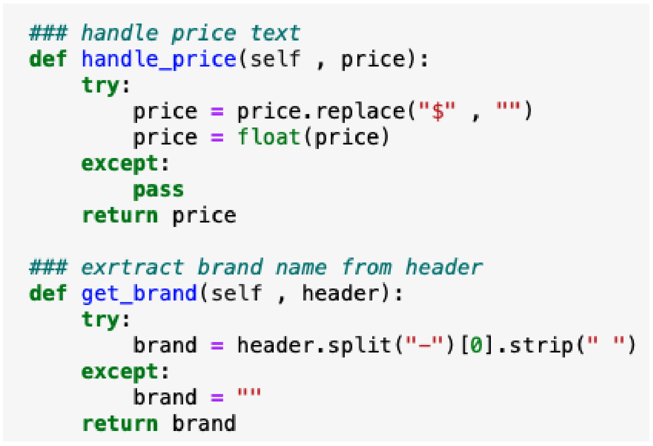
```python
def get_search_pages(self , search_term , p_number):
    ### URL
    search_page_url = "https://www.bestbuy.com/site/searchpage.jsp?id=pcat17071&st="
    ### search product
    search_product = search_term

    ### page_number
    page_number = 0
    if p_number == 0:
        page_number = 1
    else:
        page_number = p_number

    ### page_number search term
    search_page_encoding = f"&cp={page_number}"

    ### User agent
    User_Agent = 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/8

    ### request access
    r = requests.get(search_page_url + search_product + search_page_encoding , headers = {'User-Agent' : Use

    ### print url's
    print(search_page_url + search_product + search_page_encoding)

    ### Assign bs4 object to soup
    soup = BeautifulSoup(r.content, 'lxml')

    #creating a file "ebay_amazon gift cards_webpages" for storing all the pages of seacrh results that we 
    try:
        if not os.path.exists("best_buy_folder"):
            os.makedirs("best_buy_folder")
    except:
        pass

    #downloading the first pages of seacrh results

    response_page1 = requests.get(search_page_url + search_product + search_page_encoding , headers = {'Use
    soup_page1 = BeautifulSoup(response_page1.content, 'html.parser')
    with open(f'best_buy_folder/best_buy_page_{page_number}.html', 'w', encoding='utf-8') as file:
        file.write(str(soup_page1))

    return soup
```

[4]



```python
def save_product(self, product , directory):
    product_dict = {}
    ### get product webpage
    try:
        url = product['product link']
        web_page = requests.get(url , headers = {'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) 
        product_lxml = BeautifulSoup(web_page.content , "lxml")
    except:
        return 0
```

[5]

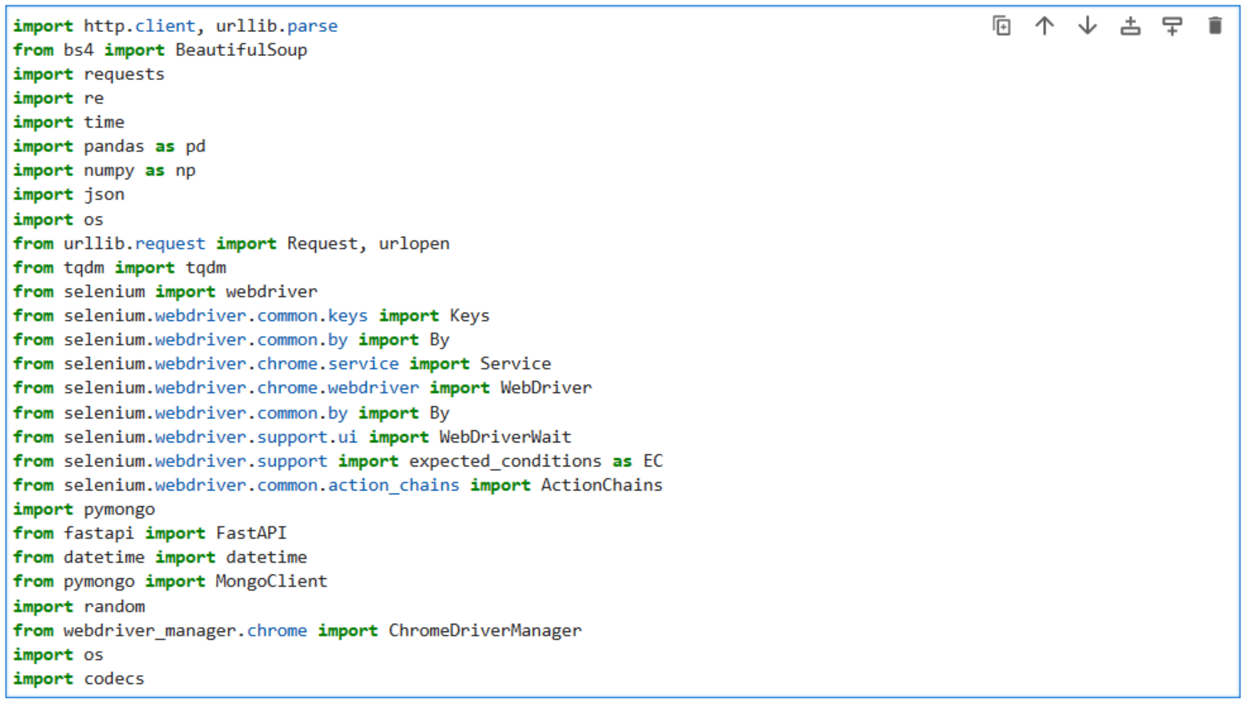


```
### handle price text
def handle_price(self , price):
    try:
        price = price.replace("$" , "")
        price = float(price)
    except:
        pass
    return price

### exrtract brand name from header
def get_brand(self , header):
    try:
        brand = header.split("-")[0].strip(" ")
    except:
        brand = ""
    return brand
```

[6]

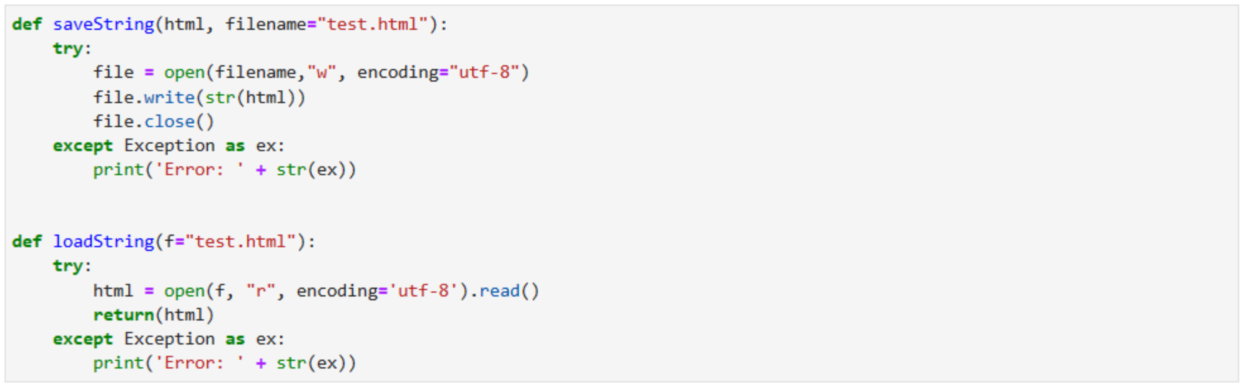
We tried to extract expert ratings, expert summary reviews, expert reviews, ratings for picture, sound and brightness quality, and customer reviews but these details were not scrapable using Beautiful soup because of empty classes. We also tried using tags present above and below the empty tag, but the HTML code post the empty class was not being displayed. However, we found a workaround using selenium where we click on the “reviews” button and then download the page and use this to scrape the required information. But this approach did not always give us the results. We increased our wait time from 20 seconds to 50 seconds to see if the HTML would be downloaded without any overlaps, but then the output varied a lot. We didn’t want to submit an incomplete file hence we are pasting the code for the same in the appendix.

Step 1: Imported the necessary libraries



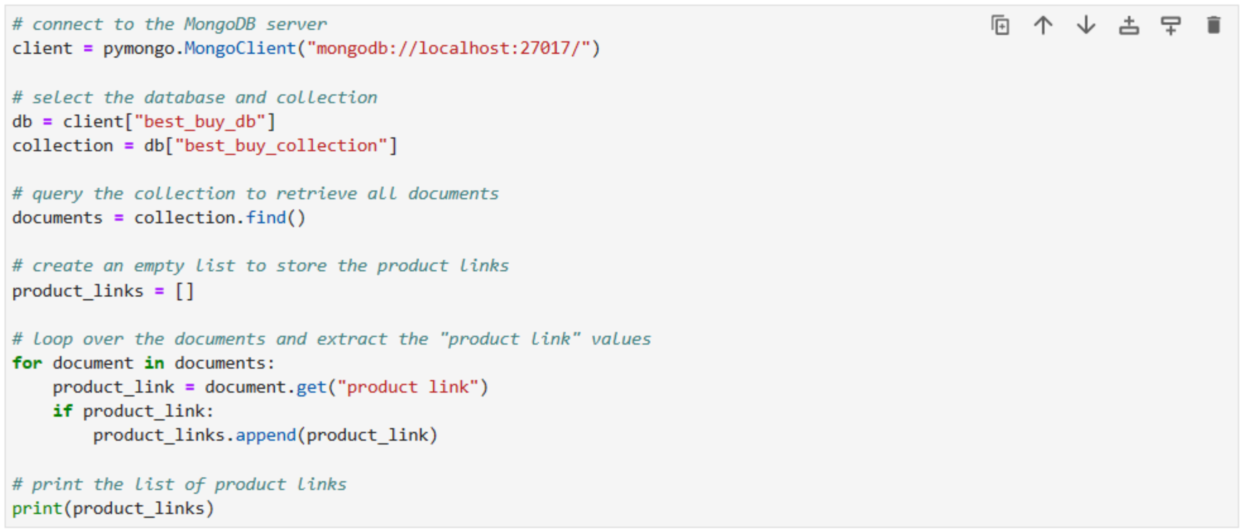
```python
import http.client, urllib.parse
from bs4 import BeautifulSoup
import requests
import re
import time
import pandas as pd
import numpy as np
import json
import os
from urllib.request import Request, urlopen
from tqdm import tqdm
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.webdriver import WebDriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.action_chains import ActionChains
import pymongo
from fastapi import FastAPI
from datetime import datetime
from pymongo import MongoClient
import random
from webdriver_manager.chrome import ChromeDriverManager
import os
import codecs
```

Step 2: Functions to save and read HTML pages



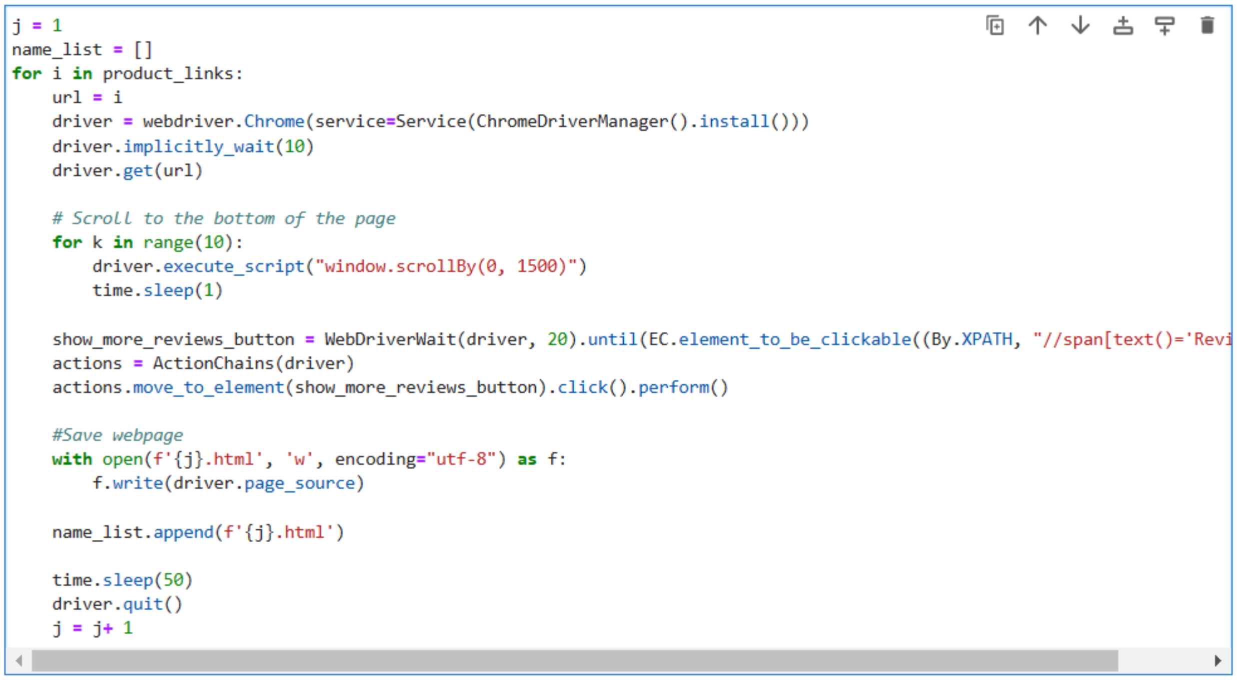
```python
def saveString(html, filename="test.html"):
    try:
        file = open(filename,"w", encoding="utf-8")
        file.write(str(html))
        file.close()
    except Exception as ex:
        print('Error: ' + str(ex))


def loadString(f="test.html"):
    try:
        html = open(f, "r", encoding='utf-8').read()
        return(html)
    except Exception as ex:
        print('Error: ' + str(ex))
```

Step 3: Getting the product links from the “best_buy_collection”



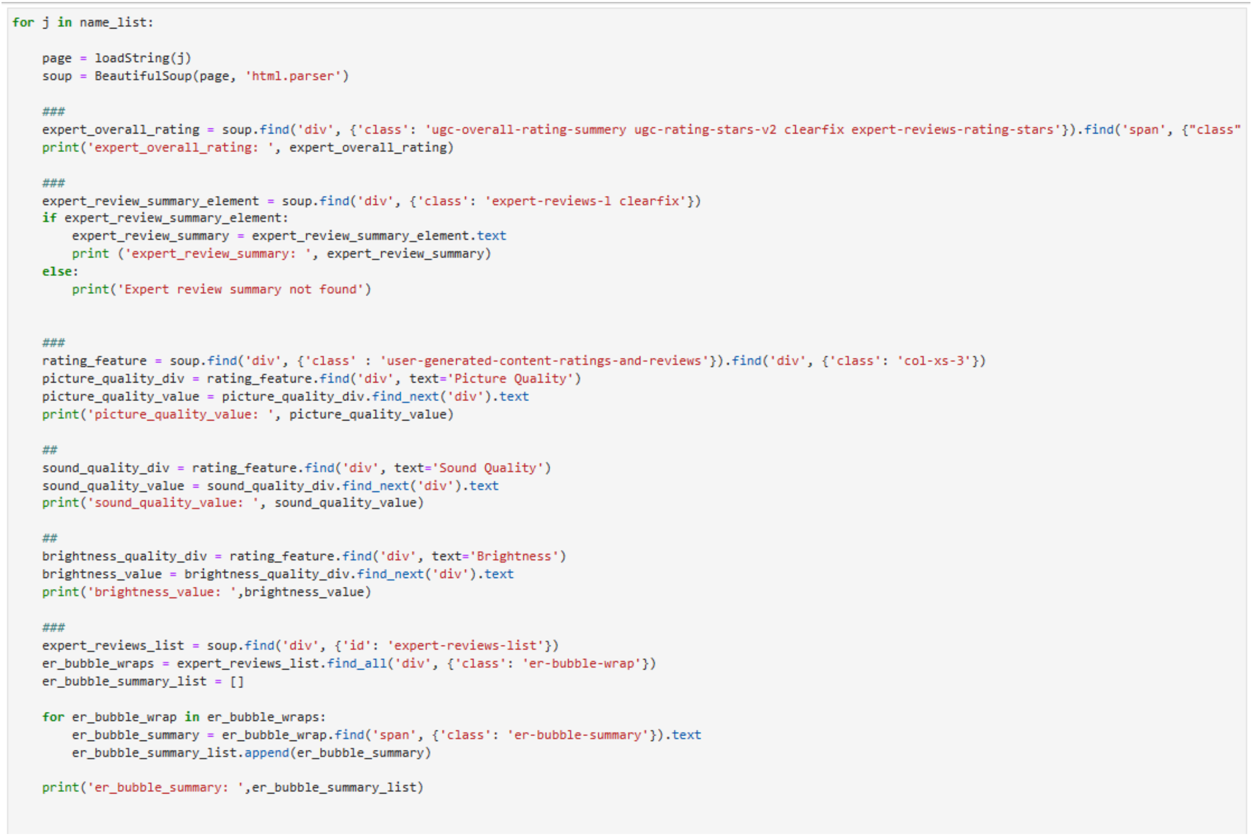
```python
# connect to the MongoDB server
client = pymongo.MongoClient("mongodb://localhost:27017/")

# select the database and collection
db = client["best_buy_db"]
collection = db["best_buy_collection"]

# query the collection to retrieve all documents
documents = collection.find()

# create an empty list to store the product links
product_links = []

# loop over the documents and extract the "product link" values
for document in documents:
    product_link = document.get("product link")
    if product_link:
        product_links.append(product_link)

# print the list of product links
print(product_links)
```

Step 4: Code snippet to loop through all the product links and use selenium to click on the reviews tab. Once that is done, we will call the download function and store loaded the HTML page



```python
j = 1
name_list = []
for i in product_links:
    url = i
    driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()))
    driver.implicitly_wait(10)
    driver.get(url)

    # Scroll to the bottom of the page
    for k in range(10):
        driver.execute_script("window.scrollBy(0, 1500)")
        time.sleep(1)

    show_more_reviews_button = WebDriverWait(driver, 20).until(EC.element_to_be_clickable((By.XPATH, "//span[text()='Revi
    actions = ActionChains(driver)
    actions.move_to_element(show_more_reviews_button).click().perform()

    #Save webpage
    with open(f'{j}.html', 'w', encoding="utf-8") as f:
        f.write(driver.page_source)

    name_list.append(f'{j}.html')

    time.sleep(50)
    driver.quit()
    j = j+ 1
```

Step 5: Code snippet to extract the mentioned elements and just print them on screen

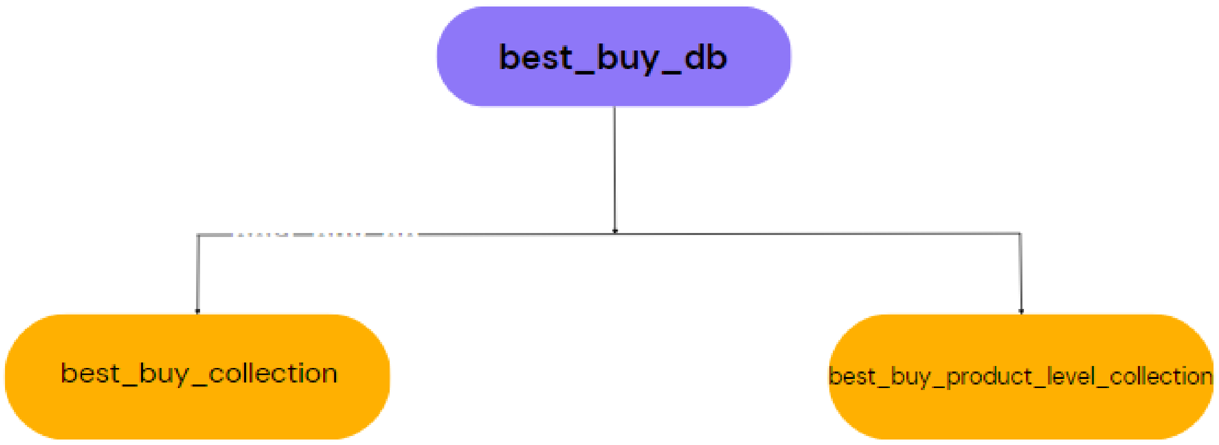


```
for j in name_list:

    page = loadString(j)
    soup = BeautifulSoup(page, 'html.parser')

    ###
    expert_overall_rating = soup.find('div', {'class': 'ugc-overall-rating-summery ugc-rating-stars-v2 clearfix expert-reviews-rating-stars'}).find('span', {"class"
    print('expert_overall_rating: ', expert_overall_rating)

    ###
    expert_review_summary_element = soup.find('div', {'class': 'expert-reviews-1 clearfix'})
    if expert_review_summary_element:
        expert_review_summary = expert_review_summary_element.text
        print ('expert_review_summary: ', expert_review_summary)
    else:
        print('Expert review summary not found')


    ###
    rating_feature = soup.find('div', {'class' : 'user-generated-content-ratings-and-reviews'}).find('div', {'class': 'col-xs-3'})
    picture_quality_div = rating_feature.find('div', text='Picture Quality')
    picture_quality_value = picture_quality_div.find_next('div').text
    print('picture_quality_value: ', picture_quality_value)

    ##
    sound_quality_div = rating_feature.find('div', text='Sound Quality')
    sound_quality_value = sound_quality_div.find_next('div').text
    print('sound_quality_value: ', sound_quality_value)

    ##
    brightness_quality_div = rating_feature.find('div', text='Brightness')
    brightness_value = brightness_quality_div.find_next('div').text
    print('brightness_value: ',brightness_value)

    ###
    expert_reviews_list = soup.find('div', {'id': 'expert-reviews-list'})
    er_bubble_wraps = expert_reviews_list.find_all('div', {'class': 'er-bubble-wrap'})
    er_bubble_summary_list = []

    for er_bubble_wrap in er_bubble_wraps:
        er_bubble_summary = er_bubble_wrap.find('span', {'class': 'er-bubble-summary'}).text
        er_bubble_summary_list.append(er_bubble_summary)

    print('er_bubble_summary: ',er_bubble_summary_list)
```

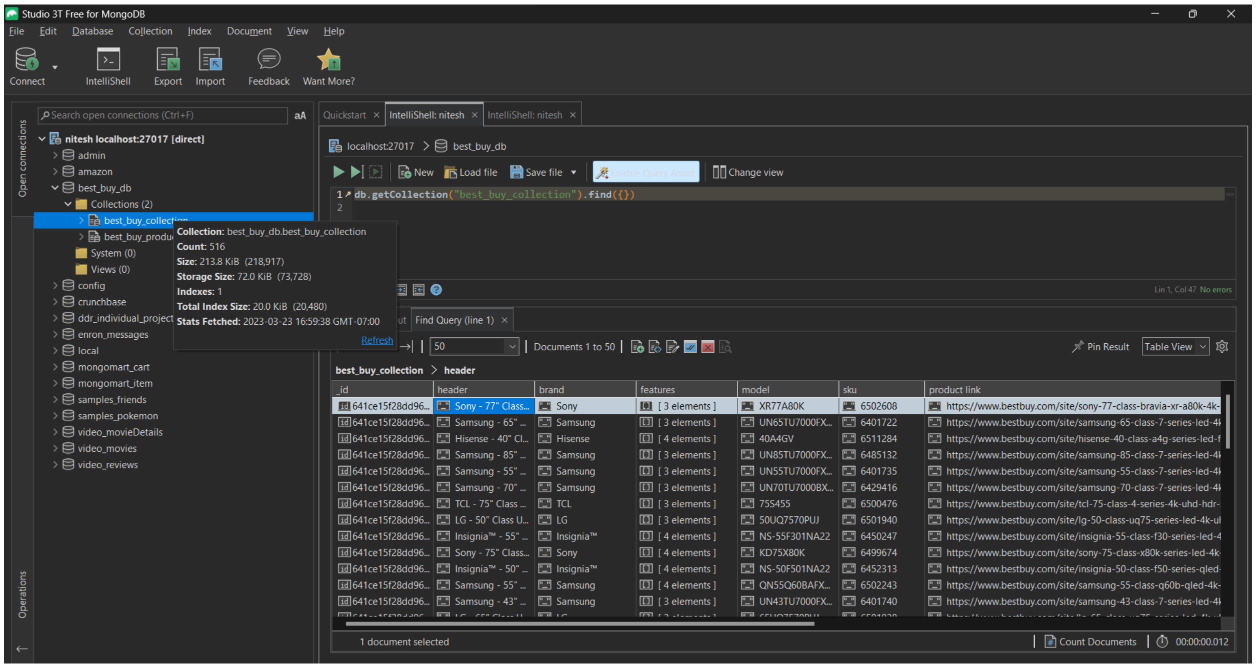
Post this we can store all the details that we extracted in a dictionary and update the “best_buy_product_level_collections” using the sku id.

[7]

General Outline:



best_buy_collection



Collection: best_buy_db.best_buy_collection
Count: 516
Size: 213.8 KiB (218,917)
Storage Size: 72.0 KiB (73,728)
Indexes: 1
Total Index Size: 20.0 KiB (20,480)
Stats Fetched: 2023-03-23 16:59:38 GMT-07:00
Refresh

```
db.getCollection("best_buy_collection").find({})
```

Lin 1, Col 47 No errors

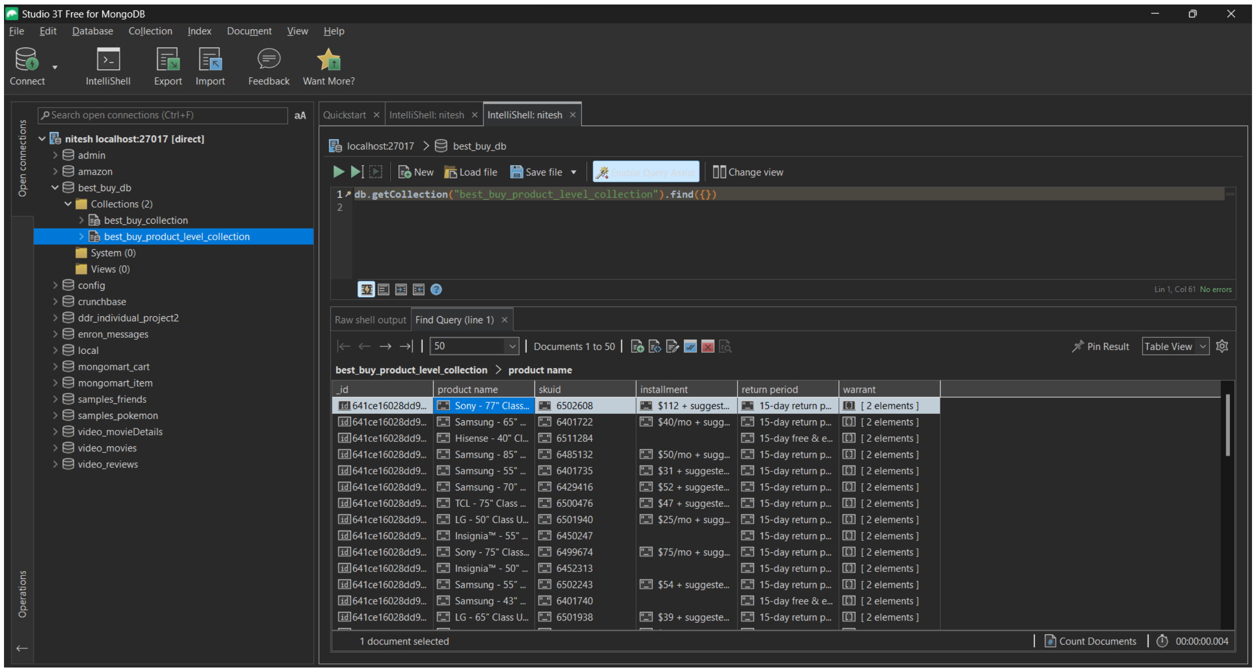
50 | Documents 1 to 50

best_buy_collection > header

_id	header	brand	features	model	sku	product link
641ce15f28dd96...	Sony - 77" Class...	Sony	[3 elements]	XR77A80K	6502608	https://www.bestbuy.com/site/sony-77-class-bravia-xr-a80k-4k-
641ce15f28dd96...	Samsung - 65" ...	Samsung	[3 elements]	UN65TU7000FX...	6401722	https://www.bestbuy.com/site/samsung-65-class-7-series-led-4k
641ce15f28dd96...	Hisense - 40" Cl...	Hisense	[4 elements]	40A4GV	6511284	https://www.bestbuy.com/site/hisense-40-class-a4g-series-led-f
641ce15f28dd96...	Samsung - 85" ...	Samsung	[3 elements]	UN85TU7000FX...	6485132	https://www.bestbuy.com/site/samsung-85-class-7-series-led-4k
641ce15f28dd96...	Samsung - 55" ...	Samsung	[3 elements]	UN55TU7000FX...	6401735	https://www.bestbuy.com/site/samsung-55-class-7-series-led-4k
641ce15f28dd96...	Samsung - 70" ...	Samsung	[3 elements]	UN70TU7000BX...	6429416	https://www.bestbuy.com/site/samsung-70-class-7-series-led-4k
641ce15f28dd96...	TCL - 75" Class ...	TCL	[3 elements]	75S455	6500476	https://www.bestbuy.com/site/tcl-75-class-4-series-4k-uhd-hdr-
641ce15f28dd96...	LG - 50" Class U...	LG	[3 elements]	50UQ7570PUJ	6501940	https://www.bestbuy.com/site/lg-50-class-uq75-series-led-4k-ul
641ce15f28dd96...	Insignia™ - 55" ...	Insignia™	[4 elements]	NS-55F301NA22	6450247	https://www.bestbuy.com/site/insignia-55-class-f30-series-led-4
641ce15f28dd96...	Sony - 75" Class...	Sony	[4 elements]	KD75X80K	6499674	https://www.bestbuy.com/site/sony-75-class-x80k-series-led-4k-
641ce15f28dd96...	Insignia™ - 50" ...	Insignia™	[4 elements]	NS-50F501NA22	6452313	https://www.bestbuy.com/site/insignia-50-class-f50-series-qled-
641ce15f28dd96...	Samsung - 55" ...	Samsung	[4 elements]	QN55Q60BAFX...	6502243	https://www.bestbuy.com/site/samsung-55-class-q60b-qled-4k-
641ce15f28dd96...	Samsung - 43" ...	Samsung	[3 elements]	UN43TU7000FX...	6401740	https://www.bestbuy.com/site/samsung-43-class-7-series-led-4k

1 document selected | Count Documents | 00:00:00.012

Best_buy_product_level_collection



```
db.getCollection("best_buy_product_level_collection").find({})
```

Lin 1, Col 61 No errors

50 | Documents 1 to 50

best_buy_product_level_collection > product name

_id	product name	skuid	installment	return period	warrant
641ce16028dd9...	Sony - 77" Class...	6502608	$112 + suggest...	15-day return p...	[2 elements]
641ce16028dd9...	Samsung - 65" ...	6401722	$40/mo + sugg...	15-day return p...	[2 elements]
641ce16028dd9...	Hisense - 40" Cl...	6511284		15-day free & e...	[2 elements]
641ce16028dd9...	Samsung - 85" ...	6485132	$50/mo + sugg...	15-day return p...	[2 elements]
641ce16028dd9...	Samsung - 55" ...	6401735	$31 + suggeste...	15-day return p...	[2 elements]
641ce16028dd9...	Samsung - 70" ...	6429416	$52 + suggeste...	15-day return p...	[2 elements]
641ce16028dd9...	TCL - 75" Class ...	6500476	$47 + suggeste...	15-day return p...	[2 elements]
641ce16028dd9...	LG - 50" Class U...	6501940	$25/mo + sugg...	15-day return p...	[2 elements]
641ce16028dd9...	Insignia™ - 55" ...	6450247		15-day return p...	[2 elements]
641ce16028dd9...	Sony - 75" Class...	6499674	$75/mo + sugg...	15-day return p...	[2 elements]
641ce16028dd9...	Insignia™ - 50" ...	6452313		15-day return p...	[2 elements]
641ce16028dd9...	Samsung - 55" ...	6502243	$54 + suggeste...	15-day return p...	[2 elements]
641ce16028dd9...	Samsung - 43" ...	6401740		15-day free & e...	[2 elements]
641ce16028dd9...	LG - 65" Class U...	6501938	$39 + suggeste...	15-day return p...	[2 elements]

1 document selected | Count Documents | 00:00:00.004