Importing Libraries

In [1]:
```
import requests

from bs4 import BeautifulSoup

import time

import datetime

import smtplib
```

In [2]:
```
#installing selenium to be able more advanced web scraping library like Selenium, which can interact with JavaS
```

In [3]:
```
pip install selenium
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: selenium in c:\users\zju\appdata\roaming\python\python311\site-packages (4.12.0)
Requirement already satisfied: urllib3[socks]<3,>=1.26 in c:\programdata\anaconda3\lib\site-packages (from sele
nium) (1.26.16)
Requirement already satisfied: trio~=0.17 in c:\users\zju\appdata\roaming\python\python311\site-packages (from
selenium) (0.22.2)
Requirement already satisfied: trio-websocket~=0.9 in c:\users\zju\appdata\roaming\python\python311\site-packag
es (from selenium) (0.10.4)
Requirement already satisfied: certifi>=2021.10.8 in c:\programdata\anaconda3\lib\site-packages (from selenium)
(2023.7.22)
Requirement already satisfied: attrs>=20.1.0 in c:\programdata\anaconda3\lib\site-packages (from trio~=0.17->se
lenium) (22.1.0)
Requirement already satisfied: sortedcontainers in c:\programdata\anaconda3\lib\site-packages (from trio~=0.17-
>selenium) (2.4.0)
Requirement already satisfied: idna in c:\programdata\anaconda3\lib\site-packages (from trio~=0.17->selenium) (
3.4)
Requirement already satisfied: outcome in c:\users\zju\appdata\roaming\python\python311\site-packages (from tri
o~=0.17->selenium) (1.2.0)
Requirement already satisfied: sniffio in c:\programdata\anaconda3\lib\site-packages (from trio~=0.17->selenium
) (1.2.0)
Requirement already satisfied: cffi>=1.14 in c:\programdata\anaconda3\lib\site-packages (from trio~=0.17->selen
ium) (1.15.1)
Requirement already satisfied: exceptiongroup in c:\users\zju\appdata\roaming\python\python311\site-packages (f
rom trio-websocket~=0.9->selenium) (1.1.3)
Requirement already satisfied: wsproto>=0.14 in c:\users\zju\appdata\roaming\python\python311\site-packages (fr
om trio-websocket~=0.9->selenium) (1.2.0)
Requirement already satisfied: PySocks!=1.5.7,<2.0,>=1.5.6 in c:\programdata\anaconda3\lib\site-packages (from
urllib3[socks]<3,>=1.26->selenium) (1.7.1)
Requirement already satisfied: pycparser in c:\programdata\anaconda3\lib\site-packages (from cffi>=1.14->trio~=
0.17->selenium) (2.21)
Requirement already satisfied: h11<1,>=0.9.0 in c:\users\zju\appdata\roaming\python\python311\site-packages (fr
om wsproto>=0.14->trio-websocket~=0.9->selenium) (0.14.0)
Note: you may need to restart the kernel to use updated packages.
```

In [4]:
```
pip install webdriver_manager
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: webdriver_manager in c:\users\zju\appdata\roaming\python\python311\site-packages
(4.0.0)
Requirement already satisfied: requests in c:\programdata\anaconda3\lib\site-packages (from webdriver_manager)
(2.31.0)
Requirement already satisfied: python-dotenv in c:\users\zju\appdata\roaming\python\python311\site-packages (fr
om webdriver_manager) (1.0.0)
Requirement already satisfied: packaging in c:\programdata\anaconda3\lib\site-packages (from webdriver_manager)
(23.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\programdata\anaconda3\lib\site-packages (from req
uests->webdriver_manager) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\programdata\anaconda3\lib\site-packages (from requests->webdr
iver_manager) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\programdata\anaconda3\lib\site-packages (from requests-
>webdriver_manager) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda3\lib\site-packages (from requests-
>webdriver_manager) (2023.7.22)
Note: you may need to restart the kernel to use updated packages.
```

In [5]:
```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
```

Displaying Price and Title For Product

In [21]:
```
# Set up Selenium Chrome driver
service = Service(ChromeDriverManager().install())
driver = webdriver.Chrome(service=service)

URL = 'https://www.amazon.com/Funny-Data-Systems-Business-Analyst/dp/B07FNW9FGJ/ref=sr_1_3?dchild=1&keywords=da
```

```python
# Load the page using Selenium
driver.get(URL)

# Extract the page source after JavaScript rendering
page_source = driver.page_source

# Close the Selenium driver
driver.quit()

soup = BeautifulSoup(page_source, "html.parser")

price_element = soup.find(name="span", class_="aok-offscreen")
if price_element is not None:
    price = price_element.get_text().strip()
else:
    price = "Price not found"

title_element = soup.find(name="span", id="productTitle")
if title_element is not None:
    title = title_element.get_text().strip()
else:
    title = "Title not found"

print("Title:", title)
print("Price:", price)
```

```
Title: Funny Got Data MIS Data Systems Business Analyst T-Shirt
Price: Price not found
```

Clean Data

In [7]:

```python
# Set up Selenium Chrome driver
service = Service(ChromeDriverManager().install())
driver = webdriver.Chrome(service=service)

URL = 'https://www.amazon.com/Funny-Data-Systems-Business-Analyst/dp/B07FNW9FGJ/ref=sr_1_3?dchild=1&keywords=da

# Load the page using Selenium
driver.get(URL)

# Extract the page source after JavaScript rendering
page_source = driver.page_source

# Close the Selenium driver
driver.quit()

soup = BeautifulSoup(page_source, "html.parser")

price_element = soup.find(name="span", class_="aok-offscreen")
if price_element is not None:
    price = price_element.text.strip()[1:]  # Accessing text directly and removing the currency symbol
else:
    price = "Price not found"

title_element = soup.find(name="span", id="productTitle")
if title_element is not None:
    title = title_element.text.strip()  # Accessing text directly
else:
    title = "Title not found"

print("Price:", price)
print("Title:", title)
```

```
Price: 16.99
Title: Funny Got Data MIS Data Systems Business Analyst T-Shirt
```

Explanations

The code you provided to attempting to scrape the price and title from the Amazon product page using web scraping techniques. However, the code is not able to locate the necessary elements on the page to extract the price and title information.

There are a few possible reasons for this issue:

1. The structure of the Amazon page might have changed since the code was written, causing the target elements to have different class or id attributes.
2. The code might not be sending the appropriate headers or making the request in a way that Amazon's server accepts. Amazon has measures in place to prevent scraping, so it's important to use proper headers and make requests that mimic a browser.
3. The code might not be handling any potential JavaScript rendering on the page, as some elements may be dynamically loaded after

the initial HTML is received.

To resolve this issue, you can try the following steps:

1. Inspect the Amazon product page to identify the specific HTML elements that contain the price and title information. Make sure that the class or id attributes used in the code match the current structure of the page.
2. Update the headers in the code to mimic a browser request. You can try using the headers provided in the code, or modify them if necessary.
3. Consider using a more advanced web scraping library like Selenium, which can interact with JavaScript elements on the page.
4. If the above steps don't work, you can explore using the official Amazon Product Advertising API, which allows programmatic access to Amazon's product data. This API is designed for accessing Amazon product information in a structured and reliable manner.

Please note that scraping Amazon's website may be against their terms of service, so proceed with caution and make sure to comply with any legal and ethical guidelines when accessing and using their data.

Second T-shirt Product Amazon Web Scrapping

In [8]:
```
# Set up Selenium Chrome driver
service = Service(ChromeDriverManager().install())
driver = webdriver.Chrome(service=service)

URL = 'https://www.amazon.com/JMIERR-Hipster-Longline-Crewneck-T-Shirt/dp/B0BXXR7S5S/?_encoding=UTF8&_ref=dlx_g

# Load the page using Selenium
driver.get(URL)

# Extract the page source after JavaScript rendering
page_source = driver.page_source

# Close the Selenium driver
driver.quit()

soup = BeautifulSoup(page_source, "html.parser")

price_element = soup.find(name="span", class_="a-offscreen")
if price_element is not None:
    price = price_element.text.strip()[1:]  # Accessing text directly and removing the currency symbol
else:
    price = "Price not found"

title_element = soup.find(name="span", id="productTitle")
if title_element is not None:
    title = title_element.text.strip()  # Accessing text directly
else:
    title = "Title not found"

print("Price:", price)
print("Title:", title)
```

```
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
```

Creating Excel Datasets for web Scrapping from Amazon

In [9]:
```
import csv

from datetime import date

product = ['Title', 'Price', 'Date']
p_data = [title, price, date.today().strftime('%Y-%m-%d')]

with open('AmazonTshirtDataset.csv', 'w', newline='', encoding='UTF8') as f:
    writer = csv.writer(f)
    writer.writerow(product)
    writer.writerow(p_data)
```

Read CSV

In [10]:
```
import pandas as pd

df = pd.read_csv(r'C:\Users\ZJU\Downloads\AmazonTshirtDataset.csv')

print(df)
```

```
                                               Title  Price        Date
0  JMIERR Mens 3 Pack Cotton Hipster Hip Hop Long...  19.99  2023-09-14
```

Appending Data to CSV

In [11]:
```
import csv

with open('AmazonTshirtDataset.csv', 'a+', newline='', encoding='UTF8') as f:
    writer = csv.writer(f)
    writer.writerow(p_data)
```

In [12]:
```
def check_price():
    URL = 'https://www.amazon.com/JMIERR-Hipster-Longline-Crewneck-T-Shirt/dp/B0BXXR7S5S/?_encoding=UTF8&_ref=d

    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)

    # Load the page using Selenium
    driver.get(URL)

    # Extract the page source after JavaScript rendering
    page_source = driver.page_source

    # Close the Selenium driver
    driver.quit()

    soup = BeautifulSoup(page_source, "html.parser")

    price_element = soup.find(name="span", class_="a-offscreen")
    if price_element is not None:

        price = price_element.text.strip()[1:]  # Accessing text directly and removing the currency symbol
    else:

        price = "Price not found"

    title_element = soup.find(name="span", id="productTitle")

    if title_element is not None:

        title = title_element.text.strip()  # Accessing text directly

    else:

        title = "Title not found"

    print("Price:", price)

    print("Title:", title)


    import csv

    from datetime import date

    product = ['Title', 'Price', 'Date']
    p_data = [title, price, date.today().strftime('%Y-%m-%d')]

    with open('AmazonTshirtDataset.csv', 'a+', newline='', encoding='UTF8') as f:

        writer = csv.writer(f)

        writer.writerow(p_data)
```

In [14]:
```
import pandas as pd

df = pd.read_csv(r'C:\Users\ZJU\Downloads\AmazonTshirtDataset.csv')

print(df)
```

```
                                               Title  Price        Date
0  JMIERR Mens 3 Pack Cotton Hipster Hip Hop Long...  19.99  2023-09-14
```

In [15]:
```
import csv
import time
from datetime import date
from selenium import webdriver
from bs4 import BeautifulSoup

def check_price():
    URL = 'https://www.amazon.com/JMIERR-Hipster-Longline-Crewneck-T-Shirt/dp/B0BXXR7S5S/?_encoding=UTF8&_ref=d

    # Set up Selenium WebDriver (ChromeDriver)
    driver = webdriver.Chrome()
    driver.get(URL)

    # Extract the page source after JavaScript rendering
    page_source = driver.page_source

    # Close the Selenium driver
    driver.quit()
```

```python
    soup = BeautifulSoup(page_source, "html.parser")

    price_element = soup.find(name="span", class_="a-offscreen")
    if price_element is not None:
        price = price_element.text.strip()[1:]  # Accessing text directly and removing the currency symbol
    else:
        price = "Price not found"

    title_element = soup.find(name="span", id="productTitle")
    if title_element is not None:
        title = title_element.text.strip()  # Accessing text directly
    else:
        title = "Title not found"

    print("Price:", price)
    print("Title:", title)

    product = ['Title', 'Price', 'Date']
    p_data = [title, price, date.today().strftime('%Y-%m-%d')]

    with open('AmazonTshirtDataset.csv', 'a+', newline='', encoding='UTF8') as f:
        writer = csv.writer(f)
        writer.writerow(p_data)

while True:
    check_price()
    time.sleep(15)
```

```
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[15], line 46
     44 while True:
     45     check_price()
---> 46     time.sleep(15)

KeyboardInterrupt:
```

In [16]:

```python
import pandas as pd

df = pd.read_csv(r'C:\Users\ZJU\Downloads\AmazonTshirtDataset.csv')

print(df)
```

```
                                               Title  Price        Date
0  JMIERR Mens 3 Pack Cotton Hipster Hip Hop Long...  19.99  2023-09-14
```

In [17]:

```python
def check_price():
    URL = 'https://www.amazon.com/JMIERR-Hipster-Longline-Crewneck-T-Shirt/dp/B0BXXR7S5S/?_encoding=UTF8&_ref=d

    # Set up Selenium WebDriver (ChromeDriver)
    driver = webdriver.Chrome()
    driver.get(URL)

    # Extract the page source after JavaScript rendering
    page_source = driver.page_source

    # Close the Selenium driver
    driver.quit()

    soup = BeautifulSoup(page_source, "html.parser")

    price_element = soup.find(name="span", class_="a-offscreen")
    if price_element is not None:
        price = price_element.text.strip()[1:]  # Accessing text directly and removing the currency symbol
    else:
        price = "Price not found"

    title_element = soup.find(name="span", id="productTitle")
    if title_element is not None:
        title = title_element.text.strip()  # Accessing text directly
    else:
        title = "Title not found"

    print("Price:", price)
    print("Title:", title)

    product = ['Title', 'Price', 'Date']
    p_data = [title, price, date.today().strftime('%Y-%m-%d')]

    with open('AmazonTshirtDataset.csv', 'a+', newline='', encoding='UTF8') as f:
        writer = csv.writer(f)
        writer.writerow(p_data)

    if (price>10):
        send_mail()
```

```
# Run the script daily
while True:
    check_price()
    time.sleep(4)  # Sleep for 24 hours
```

```
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
Price: 19.99
Title: JMIERR Mens 3 Pack Cotton Hipster Hip Hop Longline Crewneck T-Shirt
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[17], line 41
     39 while True:
     40     check_price()
---> 41     time.sleep(4)

KeyboardInterrupt: 
```

Notice ! They're some page you see error displaying its not an error its because I have Keyboard Interrupt or code cells because time slipe

In [19]:
```
import pandas as pd

df = pd.read_csv(r'C:\Users\ZJU\Downloads\AmazonTshirtDataset.csv')

print(df)
```

```
                                               Title  Price        Date
0  JMIERR Mens 3 Pack Cotton Hipster Hip Hop Long...  19.99  2023-09-14
```

Send Notification Email if Price is more than certain $ Dollar

In [20]:
```
# If uou want to try sending yourself notification email (just for fun) when a price hits below a certain level
# out with this script

def send_mail():
    server = smtplib.SMTP_SSL('smtp.gmail.com',465)
    server.ehlo()
    #server.starttls()
    server.ehlo()
    server.login('turatsinzejunior83@gmail.com','*******************')

    subject = "This Promo from Amazon Your Product you want is below $10! Now is your chance to buy!"
    body = "Junior, This is the moment we have been waiting for. Now is your chance to pick up the product of y

    msg = f"Subject: {subject}\n\n{body}"

    server.sendmail(
        'turatsinzejunior83@gmail.com',
        msg

    )
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js