from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.edge.service import Service

from selenium.webdriver.edge.options import Options

from selenium.webdriver.common.keys import Keys

import pandas as pd

import time

# Setup Selenium with Proxy

options = Options()

options.add\_argument("--headless")  # Run in headless mode

options.add\_argument("--no-sandbox")

options.add\_argument("--disable-dev-shm-usage")

options.add\_argument('--proxy-server=http://<proxy\_ip>:<proxy\_port>')  # Replace with actual proxy details

# Set up WebDriver

service = Service('C:/Users/amirt/Downloads/edgedriver\_win64/msedgedriver.exe')  # Update with your ChromeDriver path

driver = webdriver.Edge(service=service, options=options)

# url = "https://www.noon.com/uae-en/yoga-barre-socks-non-slip-anti-skid-sticky-silicone-grips-cotton-for-pilates-gym-pure-ballet-dance-barefoot-workout-ankle-multiuse-black-grey-skin-color-3pcs/Z2F9046D53376D3F6A8B7Z/p/?o=z2f9046d53376d3f6a8b7z-1"

# URL to scrape

url = "https://www.noon.com/uae-en/sports-and-outdoors/exercise-and-fitness/yoga-16328/"

driver.get(url)

time.sleep(3)

# # Simulated product data

products = [

    {"Title": "Yoga Mat", "priceNow": "1689", "Brand": "Sparnood Fitness", "Sales": "970"},

    {"Title": "Dumbbells", "priceNow": "100", "Brand": "BrandB", "Sales": "1099"},

    {"Title": "SF-3200:4HP", "priceNow": "1625", "Brand": "Brand s", "Sales": "876"},

    {"Title": "Yoga pants", "priceNow": "8096", "Brand": "Brand B", "Sales": "998"},

    {"Title": "Handgrippers", "priceNow": "3429", "Brand": "Sparnood Fitness", "Sales": "1498"},

]

#Convert to DataFrame and save to CSV

df = pd.DataFrame(products)

df.to\_csv("noon\_products.csv", index=False)

print("Data saved successfully!")

if products:

    df = pd.DataFrame(products)

    df.to\_csv("noon\_products.csv", index=False)

    print(f"Saved {len(products)} products to CSV.")

else:

    print("No products scraped. CSV file not saved.")

print(products)  # Check the content of the products list

#xtract product details

# products = []

while len(products) < 200:

    items = driver.find\_elements(By.CLASS\_NAME, "productContainer")

    for item in items:

        try:

            title = item.find\_element(By.CLASS\_NAME, "data-qa").text

            priceNow = item.find\_element(By.CLASS\_NAME, "priceNow").text

            brand = item.find\_element(By.CLASS\_NAME, "brandname").text

            seller = item.find\_element(By.CLASS\_NAME, "seller").text

            products.append({

                "Title": title,

                "priceNow": priceNow,

                "Brand": brand,

                "Seller": seller

            })

        except Exception as e:

            print("Error extracting item: {e}")

    # Click on "Next Page" if available

    try:

        next\_button = driver.find\_element(By.CLASS\_NAME, "nextButton")

        next\_button.click()

        time.sleep(2)

    except Exception:

        print("No more pages")

        break

driver.quit()

if not products:

    print("No products were scraped.")

else:

    print(f"Saving {len(products)} products to CSV.")

# Convert the list of products to a DataFrame

new\_data = pd.DataFrame(products)

try:

    # Read existing data if the file exists

    existing\_data = pd.read\_csv("noon\_products.csv")

    new\_data = pd.DataFrame(products)

    combined\_data = pd.concat([existing\_data, new\_data]).drop\_duplicates(ignore\_index=True)

except FileNotFoundError:

    combined\_data = new\_data

# Save the combined data to CSV

if not combined\_data.empty:

    combined\_data.to\_csv("noon\_products.csv", index=False)

    print("Data saved to 'noon\_products.csv'")

else:

    print("No data to save to CSV.")

# Save data to the CSV file

combined\_data.to\_csv("noon\_products.csv", index=False)

print("Data saved to 'noon\_products.csv'")

# Save data to CSV

df = pd.DataFrame(products)

df.to\_csv("noon\_products.csv", index=False)

print("Data saved to 'noon\_products.csv'")

print(new\_data.head())

**#Data Analysis**

**import pandas as pd**

**# Load the CSV file**

**data = pd.read\_csv("noon\_products.csv")**

**# Ensure `priceNow` column contains strings and handle missing values**

**if 'priceNow' in data.columns:**

**data['priceNow'] = data['priceNow'].fillna('').astype(str)**

**# Remove 'AED', commas, and convert to float**

**data['priceNow'] = (**

**data['priceNow']**

**.str.replace('AED', '', regex=False)**

**.str.replace(',', '', regex=False)**

**)**

**# Convert the cleaned strings to float**

**try:**

**data['priceNow'] = data['priceNow'].astype(float)**

**except ValueError:**

**print("Some rows in 'priceNow' could not be converted to float. Check the data.")**

**else:**

**print("The 'priceNow' column does not exist in the dataset.")**

**# Display the cleaned data**

**print(data.head())**

**# Save cleaned data back to CSV**

**data.to\_csv("cleaned\_noon\_products.csv", index=False)**

**most\_expensive = data.loc[data['priceNow'].idxmax()]**

**print("Most Expensive Product:")**

**print(most\_expensive)**

**# Cheapest product**

**cheapest = data.loc[data['priceNow'].idxmin()]**

**print("Cheapest Product:")**

**print(cheapest)**

**# Count products by brand**

**brand\_counts = data['Brand'].value\_counts()**

**print("Number of Products from Each Brand:")**

**print(brand\_counts)**