**Amazon Product Scraper**  
A BeautifulSoup4 & Selenium-Based Data Extraction and Price Tracking Tool

B Jayasree

Team-5

**Cybernanut**

***ABSTRACT:***

This project demonstrates the development of an automated Amazon Product Scraper using Python and Selenium. The scraper dynamically accepts user input for any product, retrieves details such as product names and prices, and stores the results in a structured CSV file. The tool simulates human-like browsing to overcome blocking mechanisms and provides an efficient way to collect and analyze product data for research, comparison, and decision-making purposes.

***INTRODUCTION:***

E-commerce platforms like Amazon host millions of products, making manual data collection time-consuming and inefficient. Web scraping offers an automated solution to extract product information, such as names and prices, for further analysis. This project uses Selenium to automate the process of loading Amazon search results, simulating human behavior to bypass restrictions, and extracting structured data. The project bridges data extraction and analysis by exporting results into CSV files for easy handling.

***OBJECTIVE:***

1. To automate the extraction of product names and prices from Amazon.
2. To create a dynamic tool where users can search for any product of interest.
3. To store extracted data into CSV format for analysis and visualization.
4. To implement Selenium to simulate human-like browsing and avoid blocking errors.
5. To provide a foundation for building advanced tools such as price trackers and product comparison systems.

*TECHNOLOGIES TO BE USED:*

* **Programming Language:**

Python 3

* **Libraries & Tools:**

Selenium (browser automation)

Webdriver Manager (ChromeDriver management)

Pandas (data storage and analysis)

Time & Random (delay simulation)

* **Browser:**

Google Chrome

*WORKING AND METHODOLOGY:*

* **User Input:**

The scraper accepts product keywords (e.g., “laptop”, “mobile”).

* **Browser Automation:**

Selenium launches Chrome and loads Amazon’s search page.

* **Dynamic Loading:**

The scraper scrolls the page with delays to simulate a real user.

* **Data Extraction:**

Product names and prices are located using XPath and class selectors.

* **Data Storage:**

Extracted details are stored in Pandas DataFrame and exported as a CSV file.

* **Output:**

Results are displayed in the terminal and stored in amazon\_<product>.csv.

***CODE:***

import requests

from bs4 import BeautifulSoup

import pandas as pd

URL = "https://www.python.org/jobs/"

BASE\_URL = "https://www.python.org"

def scrape\_jobs(pages=2):

jobs = []

for page in range(1, pages + 1):

url = URL if page == 1 else f"{URL}?page={page}"

print(f"[+] Fetching: {url}")

response = requests.get(url)

soup = BeautifulSoup(response.text, "html.parser")

listings = soup.find\_all("h2", class\_="listing-company")

if not listings:

print(" No jobs found on this page.")

continue

for job in listings:

a\_tag = job.find("a")

title = a\_tag.get\_text(strip=True) if a\_tag else "N/A"

link = BASE\_URL + a\_tag["href"] if a\_tag else "N/A"

company = job.find\_next("span", class\_="listing-company-name")

location = job.find\_next("span", class\_="listing-location")

jobs.append({

"Title": title,

"Company": company.get\_text(strip=True) if company else "N/A",

"Location": location.get\_text(strip=True) if location else "N/A",

"Link": link

})

return jobs

if \_\_name\_\_ == "\_\_main\_\_":

job\_list = scrape\_jobs(pages=3)

df = pd.DataFrame(job\_list)

print(df.head())

df.to\_csv("pythons\_job.csv", index=False)

print("\n[+] Saved results to pythons\_job.csv")

***Output:***

Enter the product you want to search on Amazon: laptop

✅ Data saved to amazon\_laptop.csv

Product Price (₹)

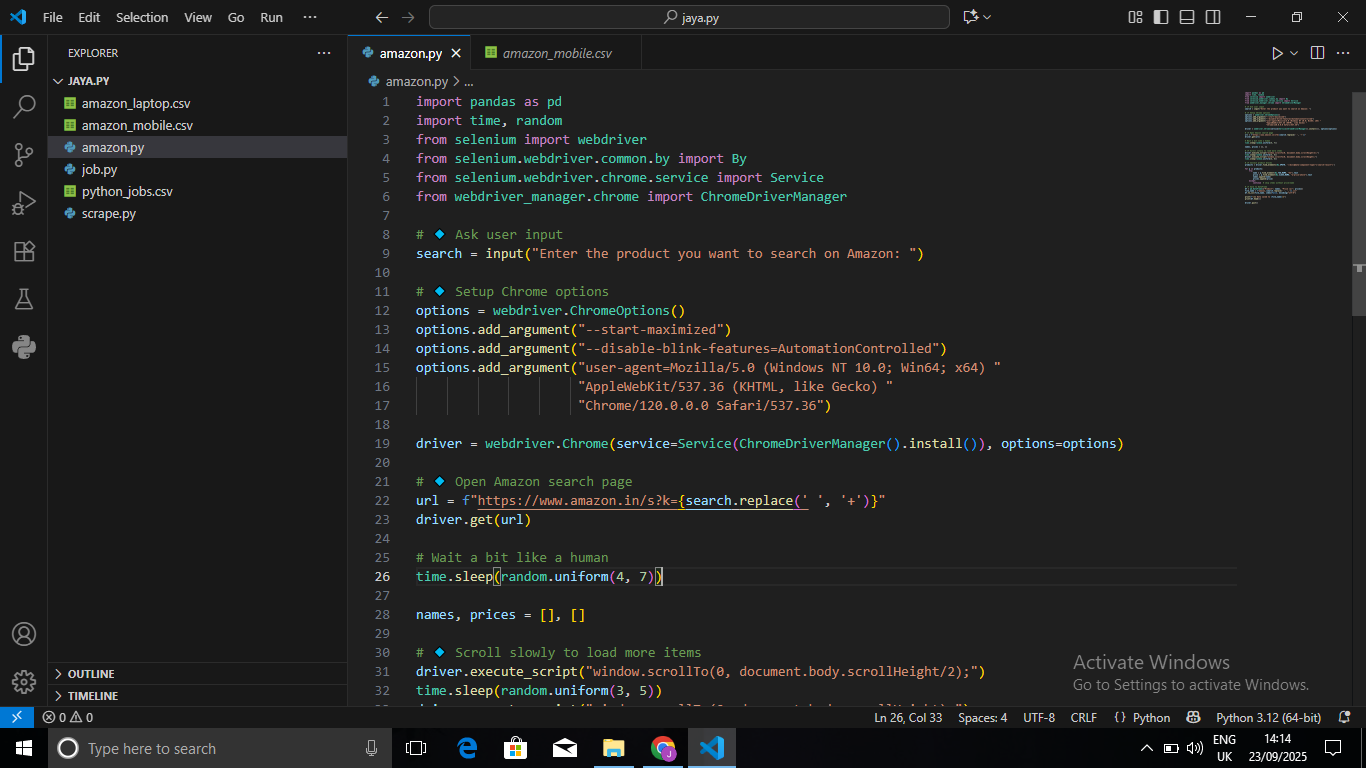
0 HP Laptop 15s, Intel Core i5 12th Gen, 15.6 inch 56,990

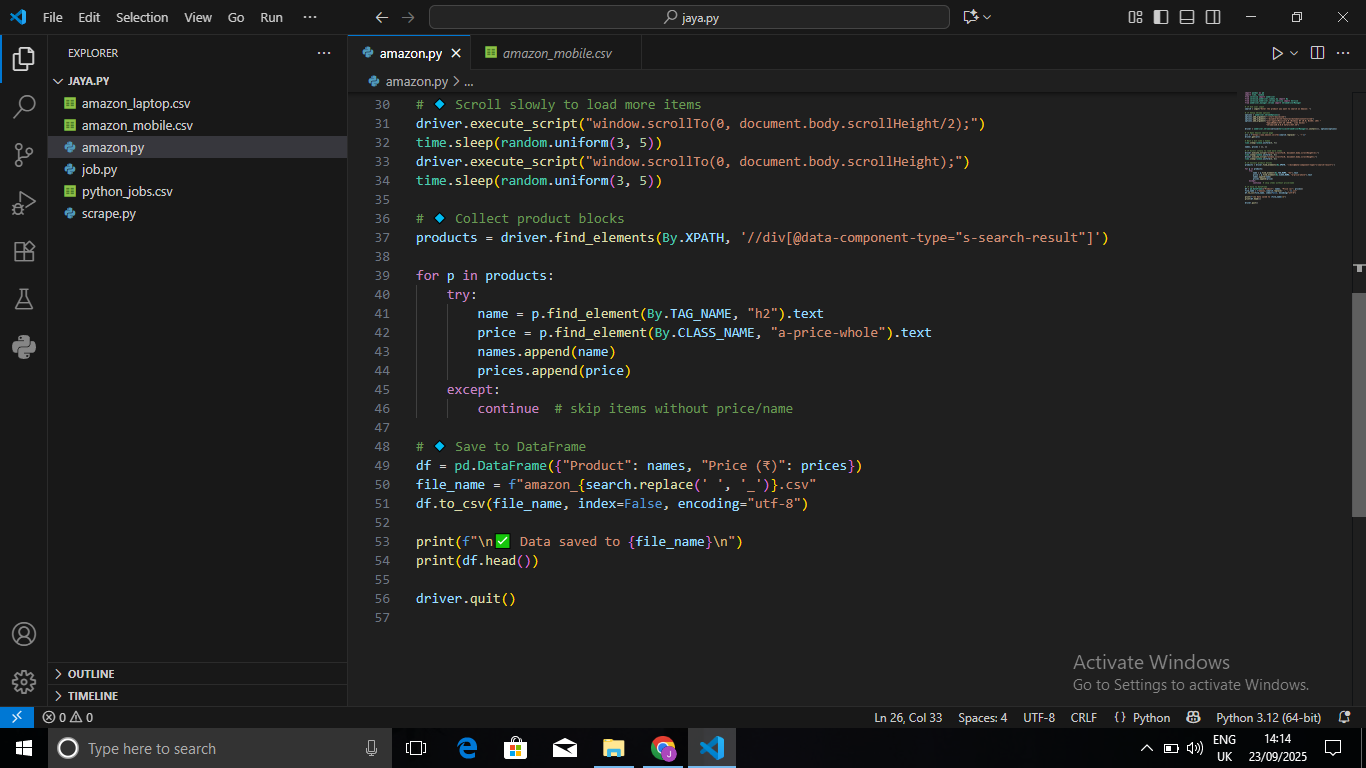
1 Dell Inspiron 14, AMD Ryzen 5, 16GB RAM, 512GB SSD 48,990

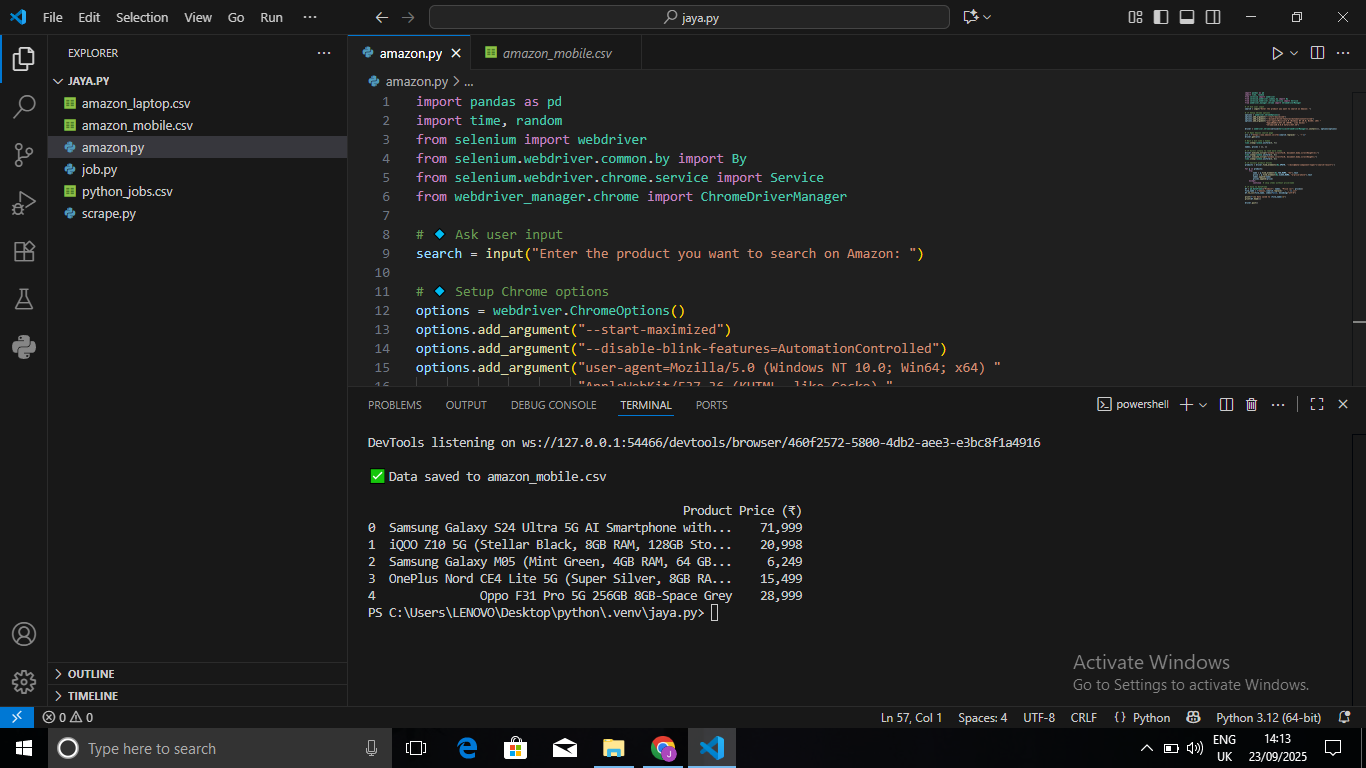
2 ASUS Vivobook 15, Intel i3, 8GB RAM, 512GB SSD 35,990

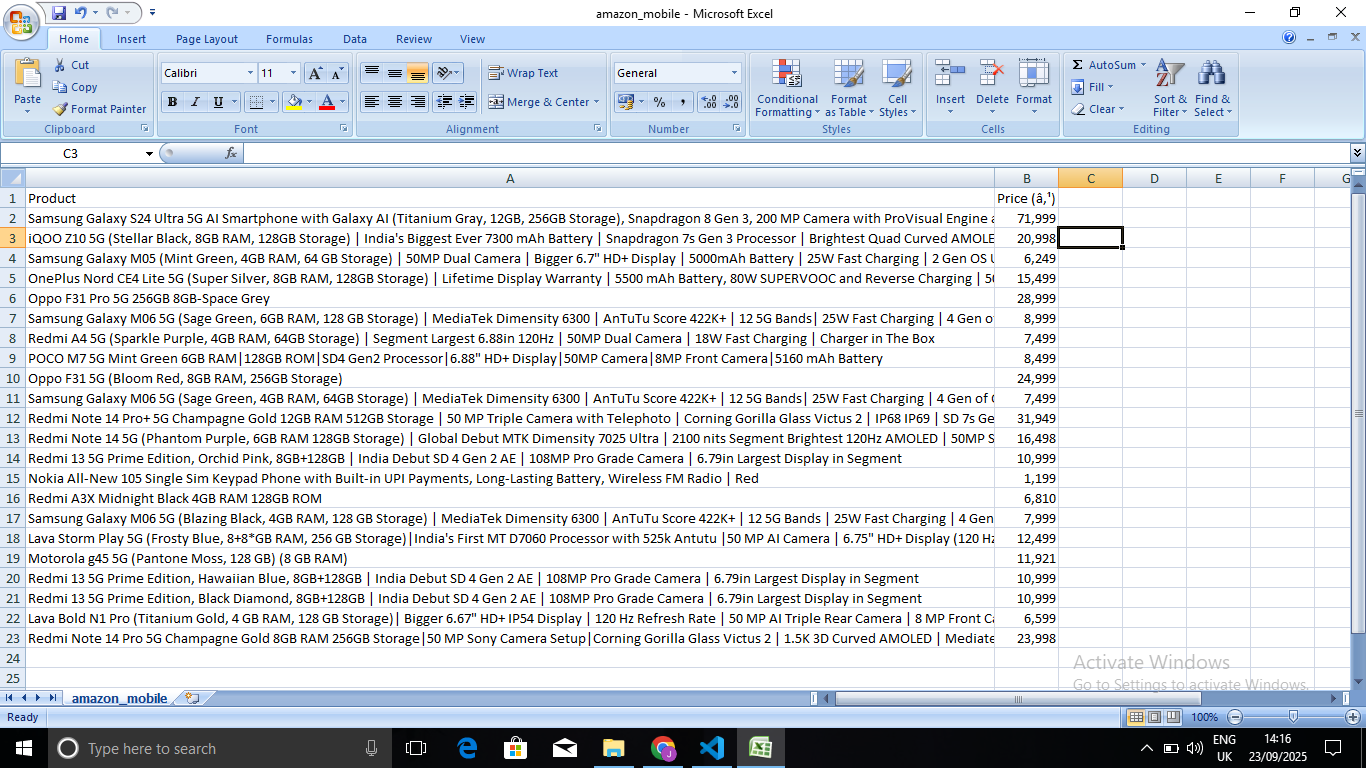
3 Lenovo IdeaPad Slim 3, Intel i5 11th Gen, 8GB RAM 45,490

4 Acer Aspire Lite, Ryzen 3, 8GB RAM, 512GB SSD 32,990

******

******

******



***CONCLUSION:***

### The Amazon Product Scraper successfully automates the process of collecting product details such as names and prices from Amazon. Using Selenium, the scraper bypasses blocking mechanisms by simulating human interactions like scrolling and delays. The extracted data is stored in CSV format, making it suitable for market analysis, price comparison, and research applications. This project demonstrates the effectiveness of web scraping in e-commerce data collection and can be further extended to track multiple pages, ratings, and reviews.

### *****References:*****

 [Amazon India](https://www.amazon.in/" \t "_new)

 Selenium Documentation

 Pandas Documentation

 [Webdriver Manager for Python](https://pypi.org/project/webdriver-manager/)