



# E - Commerce Price Comparison Tool

Bharath Kumar J N  
Team 4

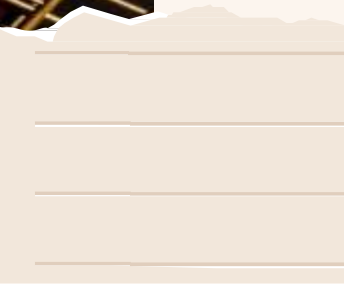
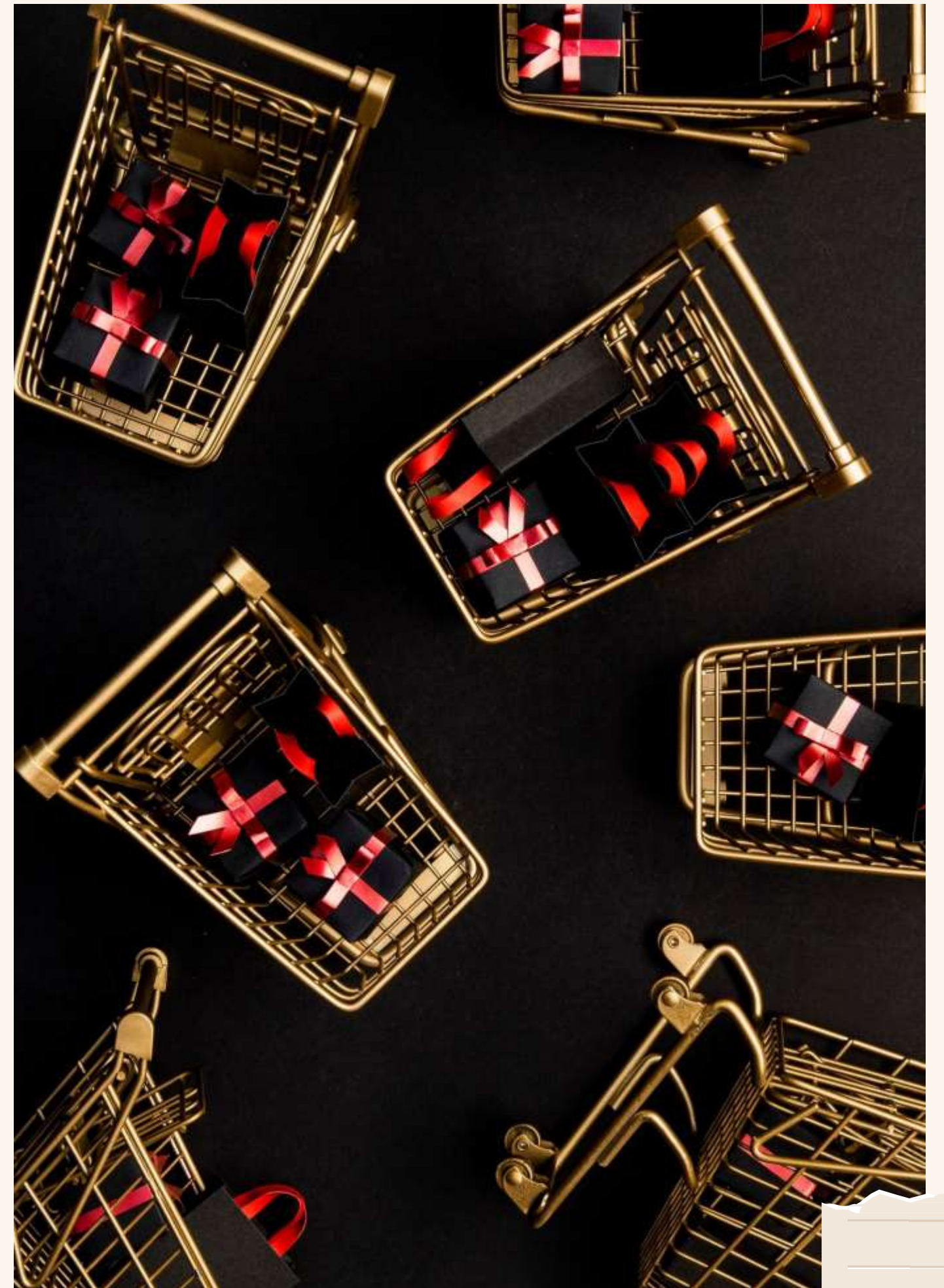


## Introduction to E-commerce Tools

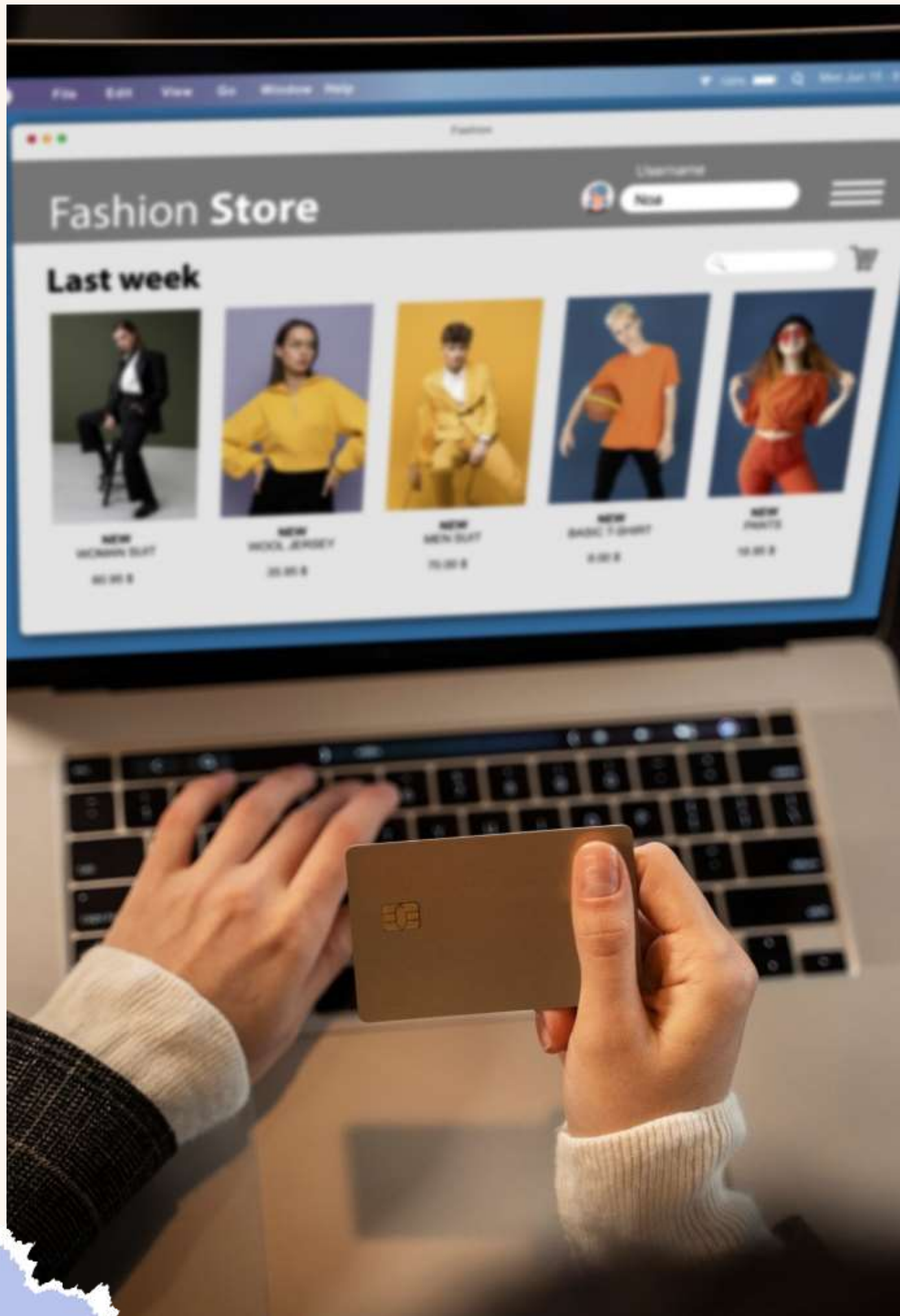
A software or platform that compares prices of similar products from different online stores or sellers.

Helps consumers find the best deals and make informed purchasing decisions.

Provides businesses with competitive insights into pricing strategies.







## Technologies Used:

Languages: Python

Web Scraping Libraries:  
BeautifulSoup, Scrapy, Selenium

Data Storage: MongoDB

Framework: Flask (Backend)

Data Processing: Pandas, NumPy

Visualization: Chart.js or Plotly



# Benefits of Using Comparison Tools

## Consumer Benefits:

- Helps save money by finding the best prices.
- Time-saving—no need to visit multiple websites.
- Transparency in pricing across different platforms.

## Business Benefits:

- Insights into competitors' pricing strategies.
- Opportunity to adjust prices to stay competitive.
- Increase sales by attracting price-conscious customers.





# Challenges and Limitations

## **Data Accuracy:**

Some stores might not update their prices in real time. Limited Product

## **Availability:**

Certain products may be out of stock or unavailable in different regions.

## **Customer Trust:**

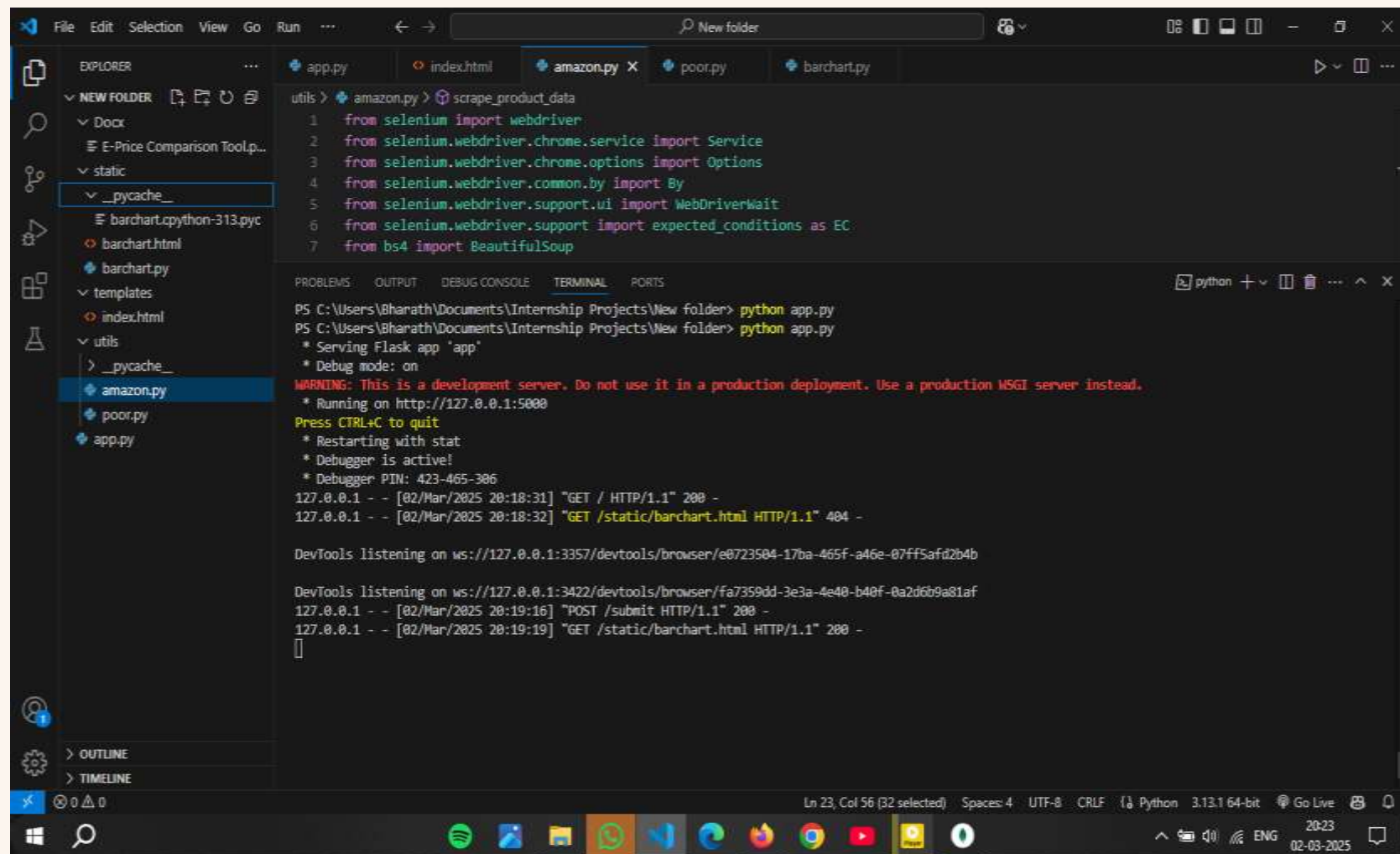
Ensuring consumers trust the tool's data and price information.





# Output

## Terminal in VS code



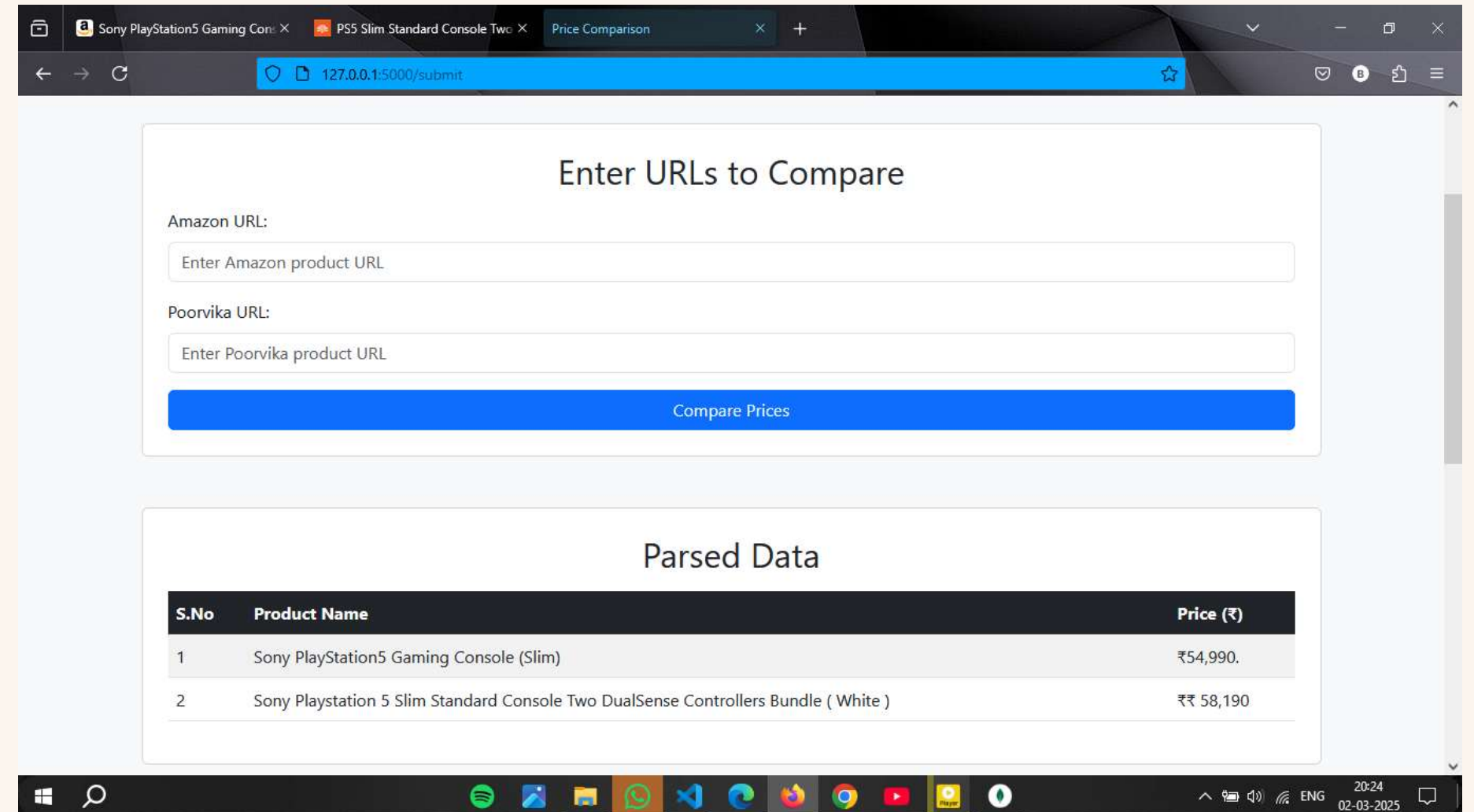
The screenshot shows the VS Code interface with a Python file named `amazon.py` open. The code uses Selenium and BeautifulSoup to scrape product data from Amazon. The terminal output shows the Flask application running on `http://127.0.0.1:5000` and receiving requests from a browser. The code in `amazon.py` is as follows:

```
1 from selenium import webdriver
2 from selenium.webdriver.chrome.service import Service
3 from selenium.webdriver.chrome.options import Options
4 from selenium.webdriver.common.by import By
5 from selenium.webdriver.support.ui import WebDriverWait
6 from selenium.webdriver.support import expected_conditions as EC
7 from bs4 import BeautifulSoup
```

The terminal output shows the following messages:

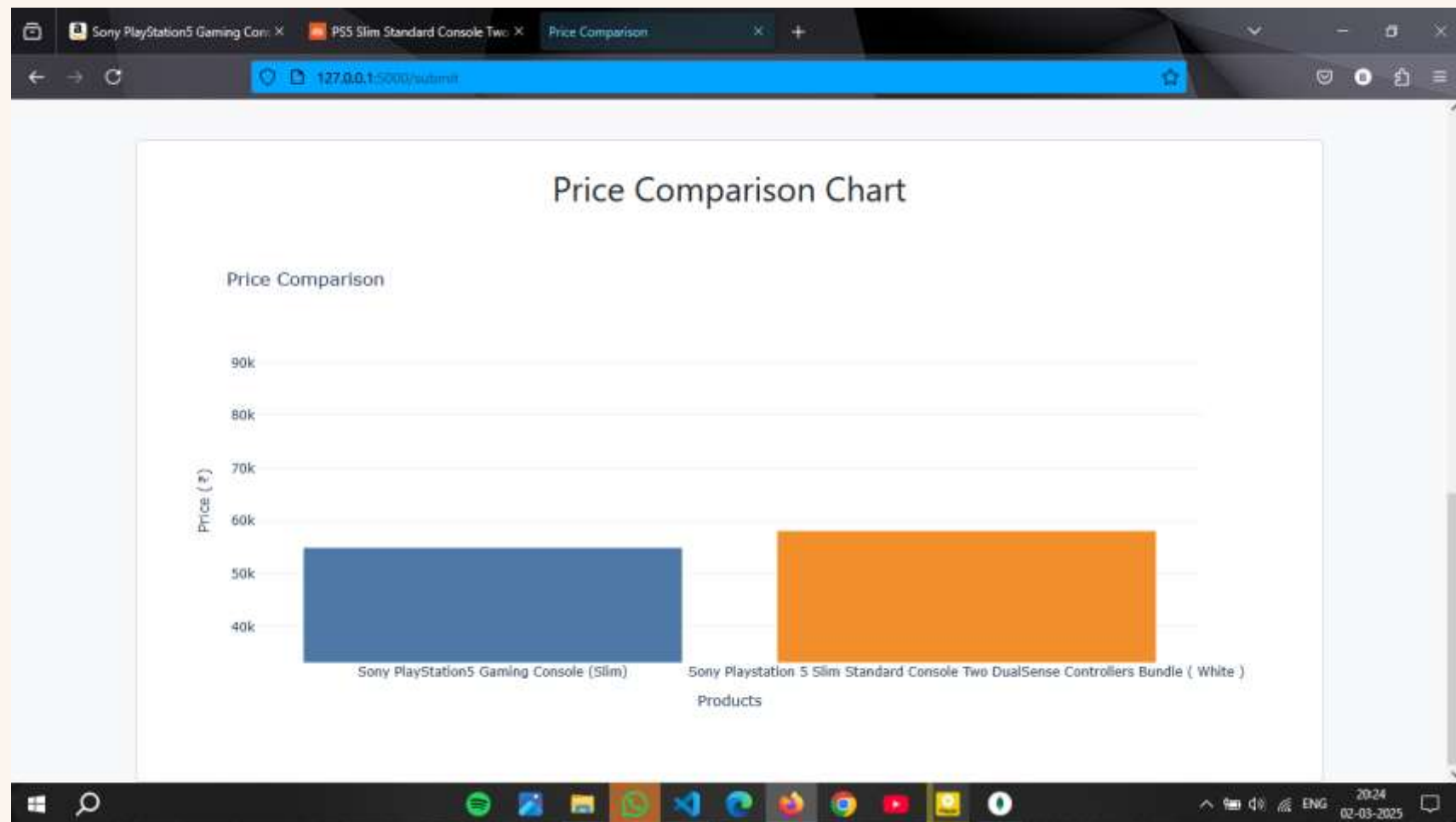
```
PS C:\Users\Bharath\Documents\Internship Projects\New folder> python app.py
PS C:\Users\Bharath\Documents\Internship Projects\New folder> python app.py
* Serving Flask app "app"
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 423-465-386
127.0.0.1 - - [02/Mar/2025 20:18:31] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [02/Mar/2025 20:18:32] "GET /static/barchart.html HTTP/1.1" 404 -
DevTools listening on ws://127.0.0.1:3357/devtools/browser/e0723504-17ba-465f-a46e-07ff5afd2b4b
DevTools listening on ws://127.0.0.1:3422/devtools/browser/fa7359dd-3e3a-4e40-b40f-0a2d6b9a81af
127.0.0.1 - - [02/Mar/2025 20:19:16] "POST /submit HTTP/1.1" 200 -
127.0.0.1 - - [02/Mar/2025 20:19:19] "GET /static/barchart.html HTTP/1.1" 200 -
```

## Webpage in browser

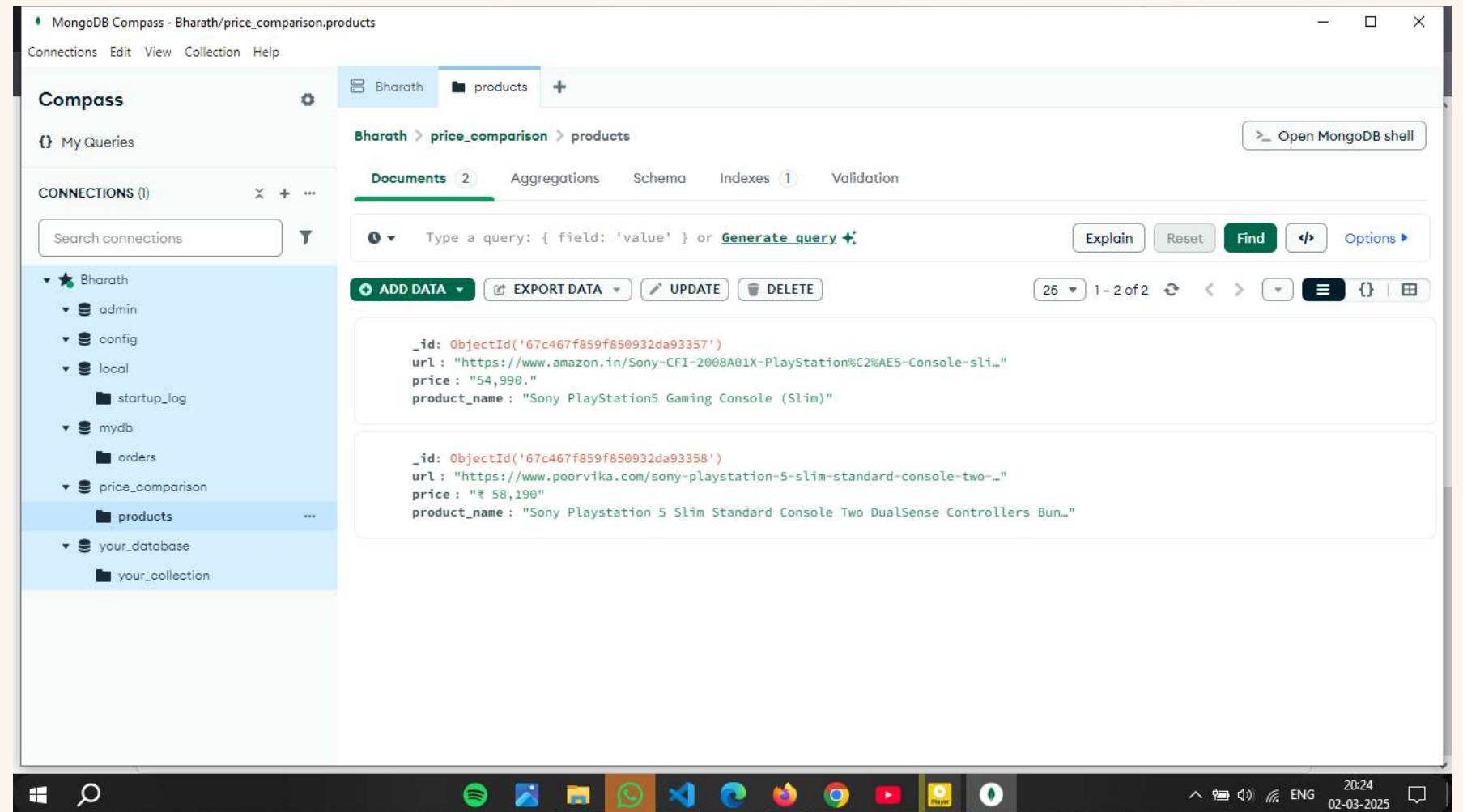


# Output

## Comparison Chart



## MongoDB Compass





**Thanks!**