**E-commerce Product Tracker Project Report**

**Introduction**

The E-commerce Product Tracker project automates the monitoring of product information (such as price, ratings, and availability) across online marketplaces like Amazon and Best Buy. By utilizing **web scraping**, **data processing**, and **visualization techniques**, the project offers insights into **pricing trends**, and **cross-platform comparisons**, supporting informed decision-making for market analysis.

**Project Goals**

1. **Automate Data Collection**: Scrape product details from Amazon and Best Buy, including name, price, rating, and category.
2. **Clean and Process Scraped Data**: Ensure consistency, handle missing values, and standardize data formats.
3. **Perform Data Analysis**: Extract actionable insights, such as price distribution trends and cross-platform price comparisons.
4. **Provide Visualizations**: Generate visual insights to facilitate a better understanding of market dynamics.

**Walkthrough of the Project**

**1. Architecture and Components**

The project consists of the following components:

* **Scraper Module**: Uses **Selenium WebDriver** to automate data extraction from Amazon and Best Buy.
* **Data Processor**: Cleans and processes raw data, removing duplicates, handling missing values, and formatting prices.
* **Analysis Module**: Performs statistical analysis and generates visualizations using **matplotlib** and **seaborn**.
* **Scheduler**: Automates periodic scraping and analysis using the **schedule** library, running tasks at regular intervals (default: every 24 hours).

**2. Code Implementation**

**Main Features:**

* **Scraping Data**:
  + **Selenium WebDriver** fetches product details such as names, prices, ratings, and timestamps.
  + Handles anti-bot measures using random delays and other stealth techniques.
* **Data Cleaning**:
  + Standardizes prices, removes duplicates, and handles missing data (e.g., replacing missing prices with the median price).
* **Data Analysis**:
  + Analyzes **price distribution** across categories (e.g., laptops, headphones).
  + Compares **average ratings** and **price differences** across platforms.
  + Aggregates the data into **CSV** files for easier portability and reuse.
* **Visualizations**:
  + Creates **boxplots** for price distributions and **comparative bar charts** for pricing trends.
  + Saves visual outputs as PNG files in the output/ directory for reporting.
* **Scheduler**:
  + Periodically scrapes and processes data at configured intervals, with logs to monitor task progress.

**3. Directories:**

* **logs/**: Stores log files of scraping and processing operations.
* **data/**: Contains the raw and processed product data in CSV format.
* **output/**: Saves visualizations and analysis results.

**Challenges Faced**

**1. Anti-bot Mechanisms**

* **Problem**: Amazon and Best Buy employ anti-bot measures that block automated scraping attempts.
* **Solution**:
  + Used **Selenium WebDriver** with stealth settings to simulate human behavior and reduce detection.
  + Incorporated **random delays** and used a **variety of navigational patterns** to avoid repetitive behavior.

**2. Inconsistent Data Formats**

* **Problem**: Price and rating formats vary across platforms, leading to inconsistencies in scraped data.
* **Solution**:
  + Implemented custom data parsing functions to handle edge cases and standardize formats.
  + Used fallback mechanisms for missing data (e.g., replacing missing prices with median values).

**3. Large Dataset Processing**

* **Problem**: Processing and analyzing large amounts of scraped data within memory limits.
* **Solution**:
  + Used **pandas** for efficient data manipulation and handling large datasets.
  + Limited scraping to the **top 20 products** per category in initial iterations to manage memory usage.

**Solutions Implemented**

**1. Efficient Scraping**

* Optimized the scraper logic to reduce latency while navigating web pages and fetching data.
* Utilized **CSS selectors** for precise data extraction from product listings.

**2. Scalable Data Handling**

* Implemented data processing steps to clean and format the scraped data.
* Stored cleaned datasets in **CSV format** to facilitate further analysis and sharing.

**3. Automation**

* Automated the scraping and analysis tasks to run **daily** using the **schedule library**.
* Added error-handling mechanisms to ensure continuity in case of failures.

**4. Visualizations**

* Generated visual outputs such as **price distribution boxplots** and **price comparison bar charts** to illustrate key trends.
* Saved analysis results in CSV format for further review and reporting.

**Future Enhancements**

**1. Expand Platform Coverage**

* Extend the system to scrape data from additional platforms such as **Walmart**, **eBay**, or **Newegg**.

**2. Advanced Analytics**

* Integrate **predictive analytics** models to forecast future price trends.
* Include sentiment analysis of **user reviews** to complement the rating data.

**3. Real-Time Scraping**

* Transition from periodic scraping to **real-time updates** for time-sensitive products, especially in fast-moving markets.

**4. Cloud Integration**

* Migrate the entire system to a **cloud platform** (e.g., AWS, Google Cloud) for improved scalability, reliability, and remote access.

**5. Interactive Dashboard**

* Develop a **web-based dashboard** for users to explore and interact with the data, offering dynamic insights into market trends and pricing strategies.

**Conclusion**

The **E-commerce Product Tracker** project automates the collection, cleaning, analysis, and visualization of product data across multiple e-commerce platforms. This tool enables businesses and market analysts to track product trends, compare prices, and gain insights into competitor strategies.

The project has tackled key challenges related to anti-bot measures, inconsistent data formats, and large dataset handling, offering a reliable and scalable solution. Future enhancements, such as real-time scraping, cloud integration, and advanced analytics, will further enhance its capabilities.

**Deliverables:**

* **Codebase**: Complete Python scripts for scraping, processing, analysis, and visualization.
* **Processed Data**: Cleaned and structured product data in CSV format.
* **Visualizations**: PNG images of price distribution boxplots and comparison charts.
* **Logs**: Execution logs for error tracking and debugging.

By solving these challenges and implementing scalable solutions, the project is prepared for future expansion, providing valuable insights into the competitive e-commerce landscape.

Git repo link: https://github.com/DavidWarmate/DBU-Final-Project