Amazon Best Sellers Web Scraper Documentation

Overview

This script is a Python-based web scraper designed to extract information from Amazon's Best Sellers section using Selenium. The scraper authenticates using Amazon credentials and collects details of products on sale across ten categories, focusing on products with discounts greater than 50%. The extracted data is stored in a structured CSV format.

Script Functionality

1. **Authentication:**
    - Logs into Amazon using valid credentials (email and password) through Selenium.
    - Verifies login success by checking the presence of account-related elements.
2. **Data Collection:**
    - Scrapes product details for the specified categories from Amazon's Best Sellers section.
    - Filters products with discounts greater than 50%.
    - Extracts the following details:
        - Product Name
        - Product Price
        - Sale Discount
        - Best Seller Rating
        - Ship From
        - Sold By
        - Rating
        - Product Description
        - Number Bought in the Past Month (if available)
        - Category Name
        - All Available Images

3. **Data Storage:**
    - Saves the extracted data into a CSV file (amazon_best_sellers.csv) in the current working directory.
4. **Error Handling:**
    - Handles common exceptions during product detail extraction and navigation.
    - Logs scraping errors and continues processing the next product or page.
5. **Pagination:**
    - Navigates through multiple pages within a category to collect up to 1500 products or until no more pages are available.

Setup Instructions

Prerequisites

- Python 3.7+
- Google Chrome browser
- ChromeDriver (matching your Chrome version)
- Python libraries:
    - selenium
    - beautifulsoup4
    - pandas

Installation Steps

1. **Install Required Libraries:**
2. pip install selenium beautifulsoup4 pandas
3. **Download ChromeDriver:**
    - Visit ChromeDriver Downloads and download the version matching your installed Chrome browser.
    - Place the chromedriver executable in a directory (e.g., C:\Windows\).
4. **Configure the Script:**
    - Update the driver_path variable with the full path to your ChromeDriver executable.

- Replace amazon_username and amazon_password with your Amazon credentials.

5. **Run the Script:**
    - Save the script as amazon_scraper.py.
    - Execute the script:
    - python amazon_scraper.py

Usage Guidelines

1. **Login Process:**
    - The script navigates to the Amazon login page and authenticates using the provided credentials.
    - Ensure the credentials are valid and have access to the specified Amazon region.
2. **Customizing Categories:**
    - The script processes ten predefined categories.
    - To add or modify categories, update the categories list with the URLs of the desired Best Sellers categories.
3. **Extracted Data:**
    - The script filters products based on discounts greater than 50% and stores only relevant products.
    - Data is saved in amazon_best_sellers.csv in the current working directory.
4. **Error Handling:**
    - Errors during scraping are logged in the console for review.
    - If scraping for a specific product fails, the script skips it and continues.
5. **Output:**
    - The output file contains columns for all specified product details.

Limitations and Notes

1. **Amazon's Terms of Service:**
    - Direct scraping may violate Amazon's terms of service. It is recommended to use the Amazon Product Advertising API for compliance.
2. **Dynamic Page Structures:**
    - The script may fail if Amazon changes its website structure. Regular updates may be required.
3. **Headless Mode:**
    - Headless mode is currently disabled to allow debugging. You can enable it by setting options.headless = True.
4. **Rate Limiting and IP Blocking:**
    - Scraping at high speeds may trigger Amazon's rate limiting or IP blocking mechanisms. Use delays (time.sleep) between requests to mitigate this.

Troubleshooting

1. **Login Issues:**
    - Ensure the email and password are correct.
    - Check if additional authentication steps (e.g., CAPTCHA, OTP) are required.
2. **ChromeDriver Errors:**
    - Verify that ChromeDriver matches your Chrome version.
    - Ensure the driver_path is correctly set.
3. **Data Missing or Incorrect:**
    - Check if the product fields exist on the page. Amazon's structure may vary across categories or products.
    - Review the console logs for errors during data extraction.

Future Enhancements

1. **API Integration:**
    - Replace scraping with the Amazon Product Advertising API for compliance and reliability.

2. **Improved Filtering:**
    - Add user-configurable filters for discount percentages or product ratings.
3. **Logging:**
    - Implement logging to a file for easier debugging and monitoring.
4. **Multi-Threading:**
    - Optimize scraping speed by implementing multi-threaded processing.

End of Documentation