Amazon Scraper
Mini Project - 2

Rajalakshmi Engineering College
Cybernaut Intern
Thanalaxmi S

Abstract

The **Amazon Scraper** is a Python-based data extraction tool designed to collect product information from Amazon using the SerpAPI Amazon Engine. With the rapid growth of e-commerce, analyzing product data has become essential for businesses and researchers. This scraper automates the process of gathering details such as product titles, ASINs, prices, ratings, reviews, and images. The data is stored in CSV format, enabling further processing, visualization, and market trend analysis.

The project demonstrates the integration of external APIs with Python for real-world data collection, highlighting the importance of automation, scalability, and compliance with legal web scraping practices. The results can be applied to price monitoring, competitive analysis, and consumer behavior research.

Introduction

E-commerce platforms like Amazon host millions of products, each with dynamic pricing, reviews, and availability. Manually tracking this information is time-consuming and inefficient. The **Amazon Scraper** addresses this problem by automating product data collection.

Unlike traditional HTML scrapers, this project leverages the **SerpAPI Amazon Engine**, which provides structured results without directly parsing Amazon's HTML. This ensures accuracy, reliability, and legal compliance. The project provides a foundation for tasks such as **price comparison websites, market research reports, and trend analysis tools**.

Objectives

The main objectives of the Amazon Scraper are:

- To **automatically extract product details** (title, ASIN, price, rating, reviews, image).
- To **store structured product data** in CSV format for future use.
- To demonstrate the use of **Python and APIs** for real-world data collection.
- To provide a **lightweight and scalable solution** compared to manual methods.
- To enable **analysis and visualization** of e-commerce data for business intelligence.

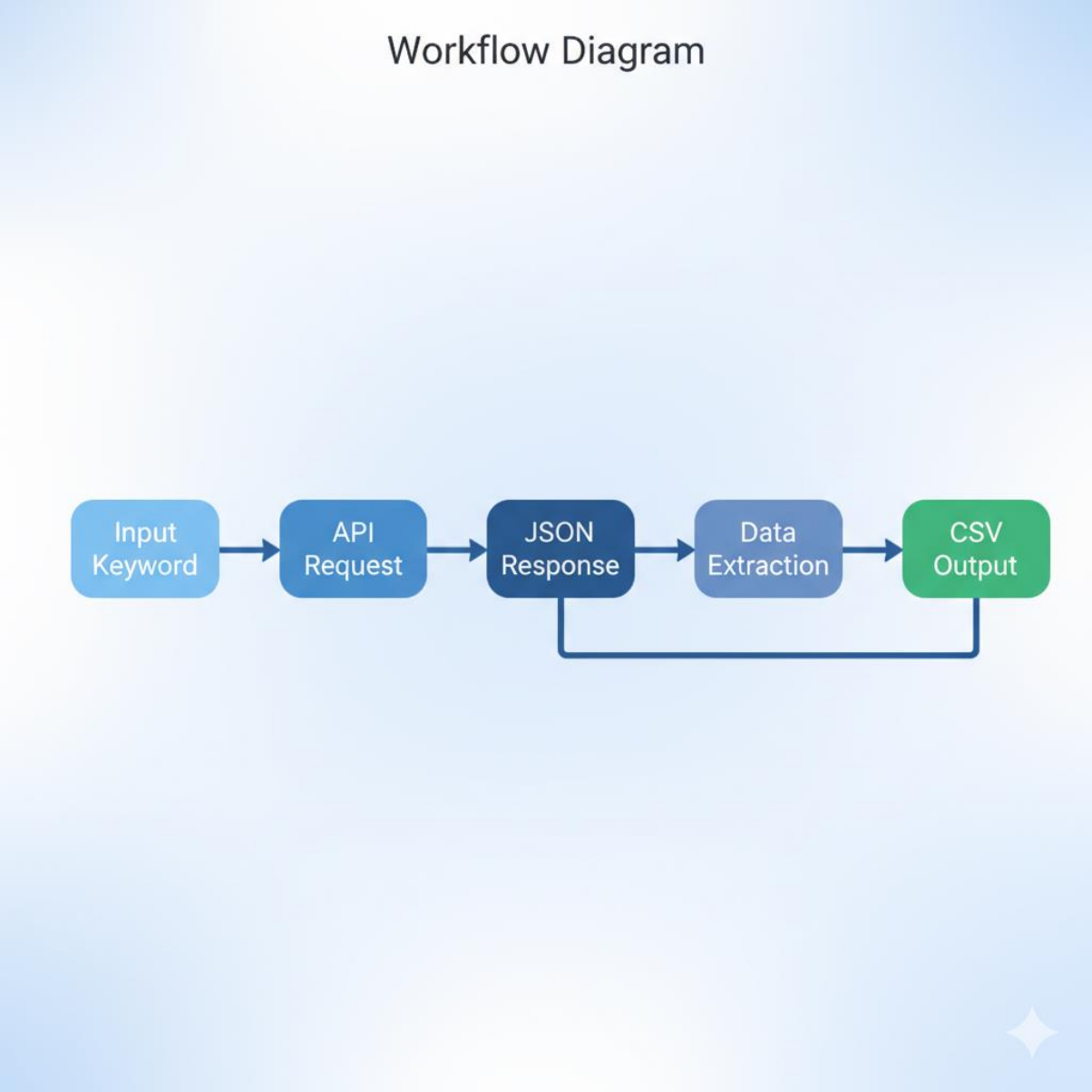
Technology Used

- **Programming Language:** Python
- **API Service:** SerpAPI (Amazon Engine) – provides structured product data via JSON response.
- **Libraries:**
    - requests → for sending HTTP requests.
    - os → for handling environment variables (API key).
    - json → for parsing API responses.
    - csv → for saving product details in tabular format.
- **Storage:** CSV file output (amazon_results.csv).

System Design and Workflow

Workflow Steps:

1. User specifies a **search keyword** (e.g., *Mac Mini M4*).
2. Script sends a **request to SerpAPI** with API key and keyword.
3. SerpAPI returns structured **JSON response** containing Amazon search results.
4. The script extracts relevant fields:
    - Title
    - ASIN
    - Price
    - Rating
    - Number of Reviews
    - Thumbnail URL
5. Extracted data is stored in **CSV format** for further analysis.



Advantages

- Automates repetitive data collection tasks.
- Provides structured and clean CSV output.
- Lightweight and requires minimal dependencies.
- More reliable than traditional HTML scraping.
- Can be extended for multi-page results.

Limitations

- Limited by SerpAPI request quota.
- Internet connectivity required.
- Can only extract fields available in SerpAPI's response.
- Not suitable for extremely large-scale scraping without scaling infrastructure.

Conclusion

The **Amazon Scraper** successfully demonstrates how Python and APIs can automate the collection of e-commerce data. By leveraging SerpAPI, it avoids common issues with web scraping such as blocking and complex HTML parsing. The tool outputs structured data in CSV format, which can be integrated into analysis workflows for business intelligence, research, and automation.

This project highlights the importance of **data-driven decision-making** and serves as a foundation for more advanced systems like **price comparison platforms** or **e-commerce analytics dashboards**.