**Project Documentation: E-Commerce Data Analysis**

**Overview**

This project, the **E-Commerce Data Analysis Script**, is designed to collect and analyze product information from a single online marketplace. The script performs a one-time data scrape to capture a snapshot of current product prices, brands, and discounts. The primary goal is to use this data to perform a competitive analysis focused on brands and discount patterns, rather than tracking historical price changes.

**Questions to Answer**

* Which brand has the lowest average product price?
* What is the average discount percentage across all products in the dataset?
* Which brands have the highest average discount percentage?

**Methodology**

The analysis is performed using a Python script that leverages several popular libraries. The process can be broken down into the following steps:

1. **Web Scraping with requests and BeautifulSoup:** The script uses the requests library to send an HTTP GET request to a specified Jumia URL, retrieving the HTML content of the main product listing page. This HTML is then parsed using BeautifulSoup to identify and isolate individual product items.
2. **Nested Data Extraction:** For each product item found, the script extracts key details directly from the main listing page, such as the product name and current price. To get more granular information like the product rating, the script performs a nested scrape by following the product's unique link to its dedicated detail page. This approach ensures a comprehensive set of data points for each item.
3. **Data Cleaning and Structuring:** As data is extracted, it undergoes a cleaning process. For example, numerical values like prices and ratings are converted from text strings into appropriate data types (int and float, respectively). The brand name is also extracted from the product title. All this information is then organized into a structured list of dictionaries.
4. **Data Analysis with pandas:** The final list of product dictionaries is converted into a pandas DataFrame. This DataFrame provides a robust structure for performing the required analyses, such as calculating average prices and discounts, which can be used to answer the project's core questions.
5. **Output:** The script prints the resulting DataFrame to the console and can optionally save the structured data to a CSV file for further analysis or record-keeping.

**Source**

This methodology is based on standard web scraping and data analysis techniques using the following open-source Python libraries:

* **requests**: A fundamental library for making HTTP requests.
* **Beautiful Soup**: A library designed for parsing HTML and XML documents.
* **pandas**: A fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool.