## horizontal line



JeansScrape

08.06.2023

**─**

Your Name

Cream Cheese

Pakistan

# 

# Overview

JeansScrape is a Python-based web scraping project focused on extracting data from the women's jeans collection on Aquila.pk, an online retail store. It utilizes the BeautifulSoup library to parse the website's HTML structure and retrieve essential details for each item, including the item name, regular price, sale price, and a link to the item's page.

The script sends a GET request to the Aquila.pk website, captures the HTML response, and uses BeautifulSoup to navigate and extract information from specific elements on the page. It iterates through the list of items within the collection, extracts the relevant details, and stores them in a data structure.

The extracted data can be utilized in various ways. The script includes an example of printing the extracted details, providing immediate visibility of the information. Additionally, the data can be processed further, analyzed, or integrated into other applications.

To ensure versatility, the script handles variations in pricing information, considering both regular and sale prices. It also constructs the item's URL by combining the base URL with the extracted link, allowing easy access to the specific item on the Aquila.pk website.

JeansScrape provides a foundation for scraping and extracting data from the Aquila.pk website, demonstrating the ability to parse HTML, handle data structures, and store the extracted information. Its modular design enables flexibility, allowing for customization and extension to other web scraping projects.

# Goals

## Scrape data:

Develop a Python script that utilizes web scraping techniques to extract essential information from the women's jeans collection on Aquila.pk.

## Parse HTML:

Utilize the BeautifulSoup library to parse the HTML structure of the website and navigate to specific elements containing the desired data.

## Extract item details:

Retrieve important details for each item, such as the item name, regular price, sale price, and the link to the item's page.

## Handle pricing variations:

Account for variations in pricing by extracting both regular and sale prices, ensuring comprehensive data extraction.

## Store extracted data:

Create a data structure, such as a list or dictionary, to store the extracted information for further processing and analysis.

## Display extracted data:

Provide the ability to print the extracted details, enabling immediate visibility and verification of the scraped data.

## Enable data processing:

Facilitate further processing of the extracted data, allowing for additional analysis, integration, or manipulation as needed.

## Maintain modularity:

Design the script to be modular and easily adaptable, enabling its extension to other web scraping projects or different websites.

## Promote reusability:

Develop clean and well-organized code, adhering to best practices, to facilitate code reusability and maintainability.

## Demonstrate skills:

Showcase proficiency in web scraping, HTML parsing, data extraction, and handling different data structures through the successful completion of the "JeansScrape" project.

# Specifications

## Target Website: Aquila.pk

The script is designed to scrape data specifically from the women's jeans collection on Aquila.pk.

## Web Scraping Library: BeautifulSoup

Utilize the BeautifulSoup library to parse the HTML structure of the website and extract data.

## Data Extraction:

Extract the following details for each item in the collection:

Item name: Extract the name of the jeans product from the HTML element.

Regular price: Extract the regular price of the item from the HTML element, considering variations in pricing.

Sale price: Extract the sale price of the item from the HTML element, considering variations in pricing.

Item link: Construct the URL to the item's page on the website using the extracted relative link.

## Data Storage:

Store the extracted information in a suitable data structure, such as a list or dictionary, for further processing, analysis, or storage.

## Output:

Print the extracted details to the console as an example of immediate visibility and verification of the scraped data.

## Error Handling:

Implement error handling mechanisms to handle potential exceptions during the scraping process, ensuring graceful handling of errors and preventing script termination.

## Compliance with Web Scraping Policies:

Adhere to the website's terms of service and comply with proper web scraping etiquette, such as respecting the website's robots.txt file and avoiding excessive requests to the server.

## Code Modularity and Reusability:

Design the code with modular components and well-organized functions, promoting code reusability and maintainability for potential extension or reuse in other web scraping projects.

## Scalability:

Ensure the script can handle a varying number of items and adapt to potential changes in the website's HTML structure or design.