Práctica 1 : ¿Cómo podemos capturar los datos de la web?
Alumnos: Lady Carolina Niño Beltrán (lninob) - Daniel Leonardo Martinez Ruiz

Información de productos de libros en Buscalibre: títulos, precios y más

Enlaces de información:

- Repositorio en github: https://github.com/Lninob/Web-scraping-libreria/
- Dataset en Zenodo: https://zenodo.org/records/14079106 - https://doi.org/10.5281/zenodo.14079106
- Video disponible Google Drive: https://drive.google.com/file/d/187GOLEZyD030NRyMAKHsmkC89-ClrBGi/view?usp=sharing o disponible en youtube https://youtu.be/VFXW1eVO864
- Enlace página web: https://www.buscalibre.com.co/

1. **Contexto**

   Este informe describe el proceso de recolección de datos mediante técnicas de **web scraping** en el sitio web https://www.buscalibre.com.co/, una de las principales plataformas de venta de libros en línea en Colombia. El propósito de este proyecto es extraer información relevante sobre los libros disponibles en la tienda online, específicamente aquellos clasificados en diversas categorías. Los datos recopilados incluyen detalles como el título, autor, precio, ISBN, número de calificaciones, stock disponible y precios con y sin descuento, entre otros.

   La recolección de esta información se llevó a cabo mediante **web scraping**, utilizando herramientas automatizadas que permitieron acceder a las páginas del sitio, navegar por las categorías y extraer los datos de manera estructurada.

2. **Título: Información de productos de libros en Buscalibre: títulos, precios y más**

3. **Descripción del dataset:**

   Este conjunto de datos contiene información detallada sobre los libros disponibles en la plataforma Buscalibre.com, organizados por categoría, se han recolectado datos de cada libro, incluyendo el título, autor, ISBN, precio (con y sin descuento), número de calificaciones, stock disponible y otros atributos relevantes. Cada entrada corresponde a un libro publicado en la tienda en línea, con detalles extraídos de diversas categorías como literatura, tecnología, ciencia, entre otras.

   (Como la tienda cuenta con millones de libros, se decidió tomar los libros de las primeras páginas de cada página para evitar un tiempo muy extendido y un gasto de recurso de máquina a la hora de leer y procesar la información)

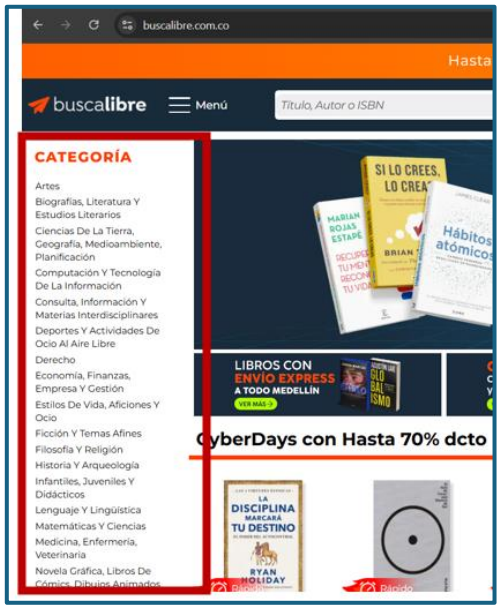
4. **Representación gráfica:**

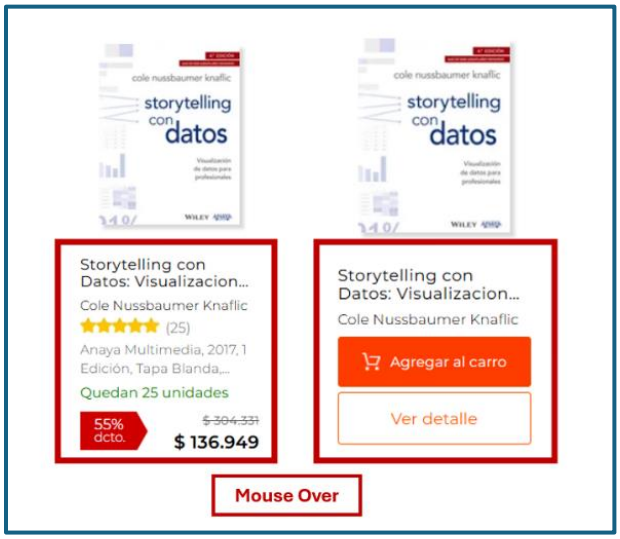
   A continuación, se presenta un ejemplo en esquema de tabla y dos imágenes que ilustran los datos extraídos de los libros en la página de Buscalibre. En la primera imagen se muestra el listado de categorías disponibles en la página principal, mientras que en la segunda imagen se detalla el bloque de un libro específico que contiene la información relevante. Cabe resaltar que al pasar el

mouse por encima de elementos como el precio o las unidades en stock, la información se oculta temporalmente y se despliegan los botones de "Agregar al carrito" o "Ver detalles".

Esquema de columnas

Categoría	ID Producto	Costo Estimado	ISBN	Título	Autor	Calificaciones	Unidades Restantes	Descuento	Precio Anterior	Precio Actual
Literatura	12345	3500	9781234567890	El Gran Libro	Juan Pérez	4.5 (300)	50	10%	4000	3500
Ciencia	67890	4000	9780987654321	La Ciencia Total	María Gómez	5.0 (500)	30	15%	4500	3800





5. Contenido

El conjunto extraído contiene información sobre los libros disponibles en la página web por cada categoría. El data set contiene los siguientes campos:

- Categoría: categoría a la que pertenece el libro
- ID Producto: identificación del producto
- Costo estimado: Se hizo una suposición de que este precio podría corresponder al costo de los libros (para la librería) lo que para fines académicos serviría par calcular márgenes de ganancia, etc)
- ISBN: Identificador isbn
- Título: Nombre del libro
- Autor: Nombre del autor o autores
- Calificaciones: Número de calificaciones o reviews que ha recibido el libro
- Unidades Restantes: Unidades en stock
- Descuento: Porcentaje de descuento
- Precio Anterior: Precio sin descuento
- Precio Actual: Precio actual o con descuento

Los datos se tomaron en noviembre y se realizó la última extracción el 11 d noviembre de 2024 (no varió mucho la información a lo largo del mes).

6. **Propietario:**

   El conjunto de datos utilizado en este proyecto proviene del sitio web Buscalibre (https://www.buscalibre.com.co/), una reconocida librería en línea en Colombia, en cuanto a la propiedad del conjunto de datos, la información recolectada corresponde a los datos públicos disponibles en su página, accesibles para cualquier usuario.

   Para asegurar que la recolección de datos se realice de forma ética y legal, se revisaron los términos y condiciones de uso de la página y no se encontró alguna sección específica que prohíba explícitamente el uso de técnicas de web scraping, por lo tanto al ser datos accesibles públicamente y sin restricciones directas para su uso, se considera que la recolección de esta información no infringe las políticas de la página. Asimismo, se revisaron análisis previos realizados por Buscalibre, pero no se encontró información específica o publicaciones relacionadas que mencionaran restricciones sobre el scraping o el uso de sus datos en un contexto académico o comercial.

   Se actuó conforme a principios éticos de recolección de datos, respetando los límites establecidos por la accesibilidad pública de la información, cabe resaltar que el uso de los datos extraídos es exclusivamente para fines académicos y analíticos, sin fines comerciales, y sin alteración ni distribución de la información original.

   De esta manera, se garantiza que el proyecto se lleva a cabo de acuerdo con las buenas prácticas de respeto a la propiedad intelectual y la legalidad vigente, actuando con responsabilidad en el uso de los datos obtenidos.

   En cuanto a análisis previos, se investigó en la web si la tienda cuenta con informes publicados, análisis previos pero no se encontró mayor información por lo que se buscó análisis similares, un par de estudios sobre analítica de datos en librerías destacan la importancia de recolectar información sobre ventas, stock y preferencias de los clientes para mejorar la oferta y gestión de inventarios. Otro análisis sobre el diseño de bases de datos para librerías subraya la importancia de organizar la información de manera eficiente para facilitar la toma de decisiones.

   Fuentes:

   - https://medium.com/@budharjuna/how-to-use-data-analytics-to-improve-book-sales-0573ada7f57c
   - https://www.showwcase.com/article/5606/Database-Design-and-Bookstore-Sales-Analysis

7. **Inspiración**

   El conjunto de datos extraídos resulta interesante porque proporciona información detallada sobre los libros disponibles en Buscalibre, datos extraídos, como la categoría, el título, el autor, el número de calificaciones, el precio, el stock disponible y los descuentos, son elementos clave que permiten realizar análisis relevantes en el contexto de su tienda en línea.

   Con los datos del dataset, se pueden abordar preguntas como:

- **¿Cuáles son los libros más populares y con mayor volumen de calificaciones?** Al analizar las calificaciones, se pueden identificar los libros con mayor volumen de valoraciones y entender qué géneros o categorías tienen una mayor popularidad entre los clientes.
- **¿Qué efecto tiene el descuento en la cantidad de unidades restantes**? Analizando los descuentos y las unidades restantes, se pueden obtener información útil sobre cómo los precios con descuento afectan la demanda de los productos, y si los descuentos impulsan las ventas. (partiendo de los libros que quedan en stock vs el porcentaje de descuento)
- **¿Cuál es la diferencia entre el precio anterior y el precio actual?** Este análisis puede ayudar a entender la estrategia de precios utilizada por la librería y evaluar cómo los cambios de precio tienen un efecto en la percepción del cliente y las ventas.
- **¿Existen patrones en la distribución de los precios por categoría?** Si se analizan los precios, se puede explorar si ciertos géneros o categorías de libros, cuáles son más costosos, lo que puede ayudar a entender la estructura de precios dentro de la librería.
- **¿Qué categorías de libros tienen mayor descuento y cómo varían los precios dentro de cada categoría?** Esta pregunta puede ayudar a entender si las librerías aplican más descuentos a ciertas categorías de libros, lo que podría ser útil para identificar patrones en la estrategia de precios.
- **¿Cómo varía el stock de los libros en función de su popularidad (volumen de calificaciones y ventas)?** Comparando el volumen de calificaciones y las unidades restantes, se pueden identificar tendencias sobre la demanda de libros. Los libros más populares (con mayores calificaciones) podrían agotarse más rápido y los vendedores podrían ajustar el stock según la demanda anticipada.
- **¿Cuál es el margen de ganancia promedio de los libros en cada categoría, basándose en el costo estimado y el precio actual?** A partir del costo estimado y el precio de venta, se puede calcular el margen de ganancia y determinar qué categorías tienen los márgenes de beneficio más altos. (este con fines académicos ya que no se tiene total certeza de que dicho dato extraído refleje el costo del libro)
- **¿Qué relación existe entre el precio de un libro y el número de calificaciones que recibe?** Esta pregunta permite investigar si los libros más caros tienden a tener más o menos calificaciones, o si hay alguna correlación entre el precio y la popularidad en términos del número de opiniones.
- **¿Existen libros con precios elevados que no estén recibiendo tantas calificaciones o ventas?** Analizar los libros con precios altos pero pocas calificaciones o unidades restantes puede indicar que, aunque sean caros, no están logrando captar el interés de los compradores, lo que podría reflejar una falta de promoción o una mala percepción del precio por parte de los consumidores.

Comparado con los análisis similares previos encontrados sobre ventas de libros y análisis de bases de datos de librerías, se analiza la relación entre precios, descuentos y demanda, lo cual es un tema central también en este conjunto de datos. Al integrar los datos de calificaciones y stock restante, el análisis de este conjunto podría ofrecer una visión más completa de cómo los

precios y la disponibilidad de los mismos (stock) afectan la percepción de los clientes y su disposición a comprar, elemento clave para optimizar la estrategia de ventas de la tienda.

8. Licencia

Para este proyecto y el set de datos resultante se eligió la **CC BY-NC-SA 4.0 License (** Attribution-NonCommercial-ShareAlike 4.0 International) , esta licencia permite que se copie, modifique, distribuya y realice obras derivadas basadas en el conjunto de datos, siempre que:

- Se otorgue crédito al propietario original del conjunto de datos (en este caso, Buscalibre).
- No se utilice con fines comerciales.
- Las obras derivadas se compartan bajo la misma licencia.

Esta opción facilita el uso académico de los datos sin fines comerciales, se protege la propiedad intelectual ya que se exige la atribución y promueve la colaboración abierta.

9. Código

El código se organizó por secciones para facilitar su lectura y comprensión, estas secciones se describen a continuación:

Sección 1: Importación de librerías

La sección del código importa todas las librerías necesarias para la recolección de datos. Se utilizó Selenium para automatizar la interacción con el navegador y BeautifulSoup para analizar el HTML de la página web, por otra parte, se usó Pandas se utiliza para almacenar los datos en un formato adecuado para su exportación, mientras que os y time se empleó para gestionar archivos y controlar los tiempos de espera entre las interacciones con el sitio web.

```
# ************ Sección 1: Importación de librerías ************ #

# Se importan las librerías necesarias para el proyecto

# librerías "estándar" de python
import os  # usado para la gestión de archivos y rutas (exportar csv)
import time # controla tiempos de espera durante la ejecución

# librerías para análisis y procesamiento de datos
import pandas as pd  # Manipulación y exportación de datos en formato CSV

# librerías de scraping
from selenium import webdriver  # Control del navegador (Chrome)
from selenium.webdriver.chrome.options import Options # Configurar opciones del Chrome para Selenium
from bs4 import BeautifulSoup # # Analizar y extraer datos del HTML obtenido
```

Sección 2: Configuración de Selenium y navegación

En esta sección, se configura Selenium para que interactúe con el navegador Chrome, primero, se ajustan las opciones de configuración del navegador para ignorar los errores de certificados

SSL que podrían interrumpir la conexión con la página web. Esto se logra mediante el uso de los parámetros *--ignore-certificate-errors y --ignore-ssl-errors*.
Además, se establece un User-Agent específico para el navegador mediante la opción *chrome_options.add_argument('user-agent=...'),* lo que permite simular una visita desde un navegador común, evitando que el sitio web detecte y bloquee el uso de Selenium.
Finalmente, se inicializa el WebDriver de Selenium con las opciones configuradas, permitiendo que se inicie el navegador con la capacidad de interactuar con la página web de manera automatizada.

```
# *********** Sección 2: Configuración de Selenium y navegación *********** #

# Configuración para ignorar errores de SSL
chrome_options = Options()  # Inicia las opciones de configuración de Chrome
chrome_options.add_argument('--ignore-certificate-errors')
chrome_options.add_argument('--ignore-ssl-errors')

# Establecer el User-Agent  (fuente: https://www.whatismybrowser.com/w/XQ429LH)
chrome_options.add_argument('user-agent=Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.0.0 Safari/537

# Se inicializa el WebDriver con las opciones configuradas
driver = webdriver.Chrome(options=chrome_options) # inicia el navegador con las anteriores configuraciones
```

Sección 3: Navegación en la página y captura de HTML

En esta sección, se accede al sitio web de la librería Buscalibre mediante Selenium, utilizando la URL principal del sitio web: *https://www.buscalibre.com.co*, luego, el navegador simula la apertura de la página web con *driver.get(url_principal),* después de cargar la página, se utiliza *time.sleep(7)* para hacer una pausa de siete segundos y permitir que la página se cargue completamente. (es posible ajustar el tiempo de espera según la velocidad de la conexión a Internet).
Una vez cargada la página, se captura el HTML de la página principal con *driver.page_source*. El código obtiene el código fuente de la página, que es luego almacenado en la variable html.
Para confirmar que la captura fue exitosa, se imprime un mensaje de confirmación: *"HTML capturado correctamente."* Por último el HTML capturado es procesado por BeautifulSoup para crear un objeto soup, que permite analizar el código fuente de la página con el analizador lxml, preparando así los datos para ser extraídos posteriormente.

```
# *********** Sección 3: Navegación en la página y captura de HTML *********** #

# Lectura del sitio web de la tienda de libros Buscalibre
url_principal = "https://www.buscalibre.com.co"
driver.get(url_principal) # Abre el sitio en el navegador (controlado por Selenium)
time.sleep(7)  # Pausa para cargar la página principal (es posible adapatar dependiendo de la veloocidad del internet)

# Captura el HTML de la página principal
html = driver.page_source  # obtiene el html de la págona
print("HTML capturado correctamente.") # mensaje de confirmación de lectura
soup = BeautifulSoup(html, "lxml") # BeautifulSoup crea un objeto para analizar el HTML (parser o analizador)
```

Sección 4: Extracción de categorías

En esta sección se hace la búsqueda y extracción de las categorías disponibles en el sitio web. Primero, se utiliza la función *soup.select* para localizar los enlaces de las categorías dentro del contenedor HTML correspondiente, identificado por el selector *'div#categoriasdeployed ul li a'*, lo

que permite capturar todos los enlaces de las categorías presentes en la página de inicio de Buscalibre.

A continuación, se inicializa una lista vacía, categorias, donde se almacenarán las tuplas que contienen el nombre de la categoría y su respectivo enlace, se recorre cada enlace encontrado y se extrae tanto el href (enlace) como el texto de la categoría, utilizando los métodos *.get('href')* y *.get_text(strip=True),* también se usa strip=True para que se eliminen los espacios adicionales al principio y al final del texto.

Finalmente, cada tupla de nombre y enlace se almacena en la lista categorías, también se inicializa una lista vacía llamada data_total, que servirá para almacenar los datos de los libros extraídos de cada categoría en pasos posteriores del proyecto.

```
# *********** Sección 4: Extracción de categorías *********** #

# Búsqueda y extracción de las categorías dentro de la página
categoria_enlaces = soup.select('div#categoriasdeployed ul li a') # busca todos los enlaces y las etiquetas de categorías
categorias = [] # inicializa una lista vacía donde se almacenarán las categorías y sus enlaces

# Se extrae los enlaces de cada categoría
for enlace in categoria_enlaces:
    link = enlace.get('href') # obtiene el enlace de la categoría
    nombre_categoria = enlace.get_text(strip=True) # obtiene el nombre y elimina espacios demás
    categorias.append((nombre_categoria, link))# se almacena la tupla de nombre y enlace en la lista "categorias"

# se crea una lista para almacenar todos los datos de los libros
data_total = []
```

Sección 5: Navegación por categorías y extracción de datos de los libros

Ahora bien, el proceso de recolección de datos se extiende a cada categoría disponible en la página web, para lo cual primero se itera sobre la lista de categorías extraídas previamente, es decir que para cada categoría, se accede a su respectiva página utilizando su enlace (*link*) y se espera unos segundos para que la página cargue completamente, asegurando que todos los elementos estén disponibles.

Al igual que en la página principal, se captura el HTML de la página de cada categoría y se analiza con BeautifulSoup para extraer la información de los libros accediendo a todas las categorías del sitio y obtener los datos relevantes de los libros que pertenecen a cada una.

De este modo, se automatiza la recolección de datos desde múltiples páginas de categorías sin la necesidad de ingresar manualmente en cada enlace, lo que agiliza el proceso de obtención de los datos.

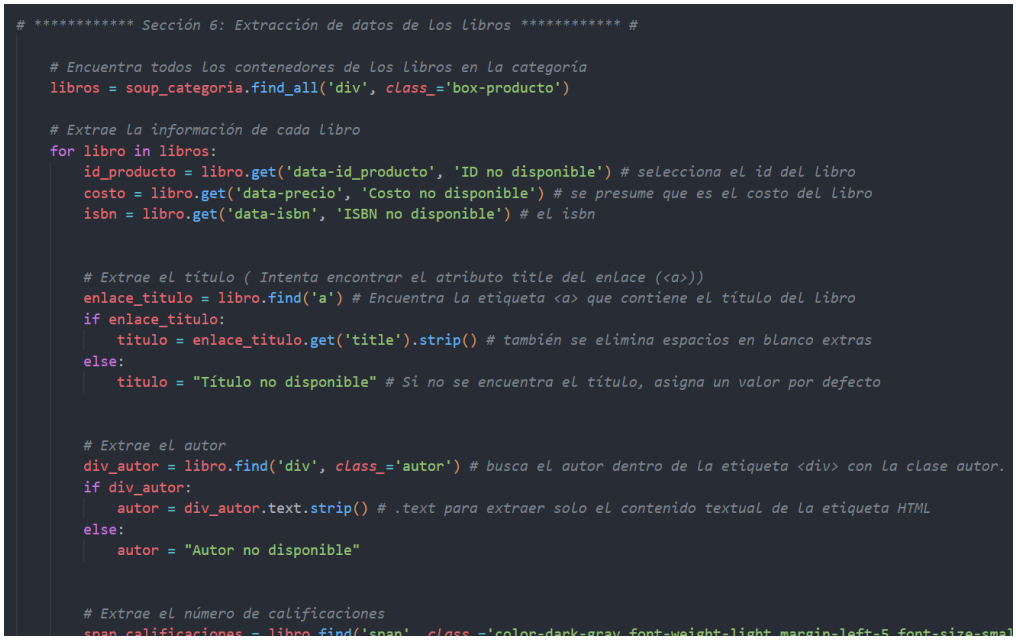
```
# *********** Sección 5: Navegación por categorías y extracción de datos de los libros *********** #

# Se navega por cada categoría y extrae los datos de los libros
for nombre_categoria, link in categorias:
    print(f"Extrayendo datos de la categoría: {nombre_categoria}")
    driver.get(link) # Navega a la URL de la categoría
    time.sleep(5)  # Pausa para cargar la página de la categoría

    html_categoria = driver.page_source # Captura el HTML de la página de la categoría
    soup_categoria = BeautifulSoup(html_categoria, "lxml") # parser del html de la categoría respectiva
```

Sección 6: Extracción de datos de los libros

En esta sección es la más extensa ya que recopila los datos requeridos de los libros, el código utiliza un bucle for para iterar sobre todos los contenedores de libros encontrados en la página de categorías, es decir busca y extrae la información de los libros que aparecen en las páginas de categorías del sitio web, el proceso en general que se emplea es el siguiente:

- **ID Producto**, **Costo estimado** y **ISBN**: Son extraídos directamente de los atributos data-**id_producto, data-precio** y **data-isbn** del contenedor de cada libro.
- **Título**: Se busca dentro del enlace (<a>) del libro, extrayendo el valor del atributo *title*. Si no se encuentra el título, se asigna "*Título no disponible*".
- **Autor**: Se busca dentro de un <*div*> con la clase autor. Si no se encuentra, se asigna "Autor no disponible".
- **Calificaciones**: Se extrae el número de calificaciones de un *span* con una clase específica. Los paréntesis que rodean las calificaciones son eliminados.
- **Unidades restantes**: Se extrae de un <*div*> con la clase stock *hide-on-hover color-green font-size-small margin-top-5*. Si no se encuentra, se asigna "Unidades no disponibles".
- **Descuento**: Se extrae de un <*div*> con la clase descuento-*v2*. Si no se encuentra, se asigna "Descuento no disponible".
- **Precio anterior**: Se obtiene de un contenedor con la clase *precio-antes*. Si el precio está tachado, se extrae el valor. Si no se encuentra, se asigna "Precio sin descuento no disponible".
- **Precio actual**: Se obtiene de un contenedor con la clase precio-ahora. Si se encuentra, se extrae el valor dentro de la etiqueta <*strong*>. Si no se encuentra, se asigna "Precio actual no disponible".

Cada uno de estos datos se guarda en un diccionario y se añade a la lista *data_total*. Al finalizar el bucle, esta lista contiene la información de todos los libros extraídos de la categoría.



```
# ********** Sección 6: Extracción de datos de Los libros ********** #

    # Encuentra todos los contenedores de los libros en la categoría
    libros = soup_categoria.find_all('div', class_='box-producto')

    # Extrae la información de cada libro
    for libro in libros:
        id_producto = libro.get('data-id_producto', 'ID no disponible') # selecciona el id del libro
        costo = libro.get('data-precio', 'Costo no disponible') # se presume que es el costo del libro
        isbn = libro.get('data-isbn', 'ISBN no disponible') # el isbn


        # Extrae el título ( Intenta encontrar el atributo title del enlace (<a>))
        enlace_titulo = libro.find('a') # Encuentra la etiqueta <a> que contiene el título del libro
        if enlace_titulo:
            titulo = enlace_titulo.get('title').strip() # también se elimina espacios en blanco extras
        else:
            titulo = "Título no disponible" # Si no se encuentra el título, asigna un valor por defecto


        # Extrae el autor
        div_autor = libro.find('div', class_='autor') # busca el autor dentro de la etiqueta <div> con la clase autor.
        if div_autor:
            autor = div_autor.text.strip() # .text para extraer solo el contenido textual de la etiqueta HTML
        else:
            autor = "Autor no disponible"


        # Extrae el número de calificaciones
        span_calificaciones = libro.find('span', class_='color-dark-gray font-weight-light margin-left-5 font-size-smal
```

Sección 7: Almacenamiento de los datos

En esta sección, los datos extraídos de cada libro se almacenan en la lista *data_total* como diccionarios, cada diccionario contiene información relevante como la categoría, ID, costo, ISBN, título, autor, calificaciones, unidades, descuento, precio anterior y precio actual. Estos datos se agregan a la lista con el método *append()*. Al finalizar la recolección de datos, el navegador se cierra con *driver.quit(),* asegurando que no queden instancias abiertas.

```
# ************ Sección 7: Almacenamiento de los datos ************ #

# Se añade un diccionario con los datos extraídos a la lista "data_total"
        data_total.append({
            'Categoría': nombre_categoria,
            'ID Producto': id_producto,
            'Costo estimado': costo,
            'ISBN': isbn,
            'Título': titulo,
            'Autor': autor,
            'Calificaciones': calificaciones,
            'Unidades Restantes': unidades,
            'Descuento': descuento,
            'Precio Anterior': precio_sin_dcto,
            'Precio Actual': precio_actual
        })

# Cierra el navegador
driver.quit()
```

Sección 8: *Guardado en CSV*

En esta sección, se crea un dataframe de Pandas a partir de la lista *data_total* que contiene todos los datos recolectados, luego, el archivo CSV se guarda en la misma carpeta donde se encuentra el script, utilizando la función *os.path* para obtener la ubicación del script y definir el nombre del archivo. Paso siguiente, el dataframe se exporta a un archivo CSV con el método *to_csv(),* utilizando la codificación *UTF-8 -sig* para garantizar que los caracteres especiales se guarden correctamente, para finalizar, se imprime un mensaje de confirmación con la ruta donde se guardó el archivo.

```
# ************ Sección 8: Guardado en CSV ************ #

# DataFrame de pandas a partir de los datos recolectados
df = pd.DataFrame(data_total)

# Guardar el archivo CSV en la misma carpeta donde se encuentra el script
script_directory = os.path.dirname(os.path.abspath(__file__))  # Obtiene la ubicación del script
filename = os.path.join(script_directory, "datos_libros_categorias.csv") # "construye" la ruta con la variable

# Guardar el DataFrame en el archivo CSV
df.to_csv(filename, index=False, encoding='utf-8-sig') # convierte el set de datos en csv (se pasa la ruta y la

# Confirmación de la ubicación donde se guardó el archivo
print(f"Archivo CSV generado exitosamente en: {filename}")
```

Indicaciones finales:

Se generó un archivo requirements.txt mediante el comando *pip freeze > requirements.txt*, que contiene todas las librerías y sus versiones necesarias para ejecutar el código en otro entorno.
Si se desea implementar el código en otro computador, basta con seguir estos pasos:

- Asegurarse de tener Python y pip instalados.
- Colocar el archivo *requirements.txt* en el directorio donde se encuentra el código.
- Abra una terminal o consola de comandos en la carpeta donde se encuentra el archivo.
- Ejecute el comando *pip install -r requirements.txt* para instalar las librerías requeridas

Dificultades presentadas:

Durante la interacción con el sitio web, se observó que ciertos elementos, como el precio y las unidades restantes de los libros, solo eran visibles cuando el ratón pasaba sobre ellos, al retirar el puntero, el precio, el descuento y otros datos volvían a desaparecer, lo que dificultaba su identificación en el HTML de la página, para superar esta limitación, se llevó a cabo un análisis exhaustivo del código para identificar los elementos que contenían la información correcta, permitiendo extraerlos sin necesidad de interactuar directamente con la interfaz, lo que garantizó la recopilación de toda la información relevante.
Además, la página de Buscalibre organiza su contenido en bloques HTML con clases específicas para cada tipo de dato. Sin embargo, los datos no siempre se presentaban de forma uniforme. Por ejemplo, el título no siempre seguía una estructura constante entre los diferentes ejemplares. Para resolver esto, se extrajo el título directamente del enlace correspondiente.
Asimismo, para manejar la posible ausencia de ciertos datos, se implementó una solución en el código que asigna valores predeterminados, como "Precio no disponible" o “descuento no disponible”, cuando falta un dato específico, lo que aseguró que el proceso de recolección no se interrumpiera por la falta de valores y que los datos extraídos fueran completos y consistentes.

10. Data set

- El data set generado se subió a Zenodo, el acceso puede hacerse desde https://zenodo.org/records/14079106 o mediante su correspondiente DOI https://doi.org/10.5281/zenodo.14079106

11. Video

- El video explicativo de la práctica se puede acceder desde el enlace: Drive: https://drive.google.com/file/d/187GOLEZyD030NRyMAKHsmkC89-CIrBGi/view?usp=sharing o disponible en youtube https://youtu.be/VFXW1eVO864

12. Contribuciones:

Contribuciones	**Firma**
Investigación previa	LCNB - DLMR
Selección del dataset	LCNB - DLMR
Desarrollo del código	LCNB - DLMR
Revisión código	LCNB - DLMR
Desarrollo respuestas (redacción del informe)	LCNB - DLMR
Participación en el video	LCNB - DLMR