



Productos-anuncios



MERCADO LIBRE

UNALM - LP 2

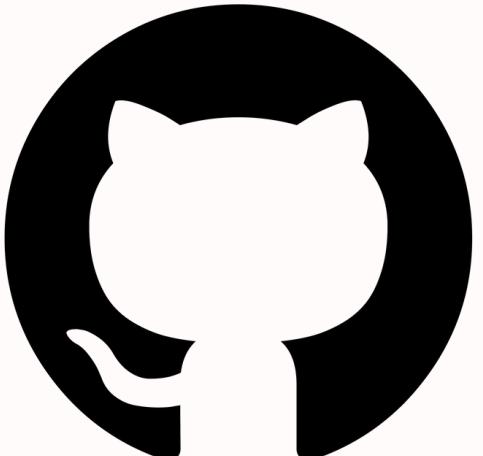


Participantes

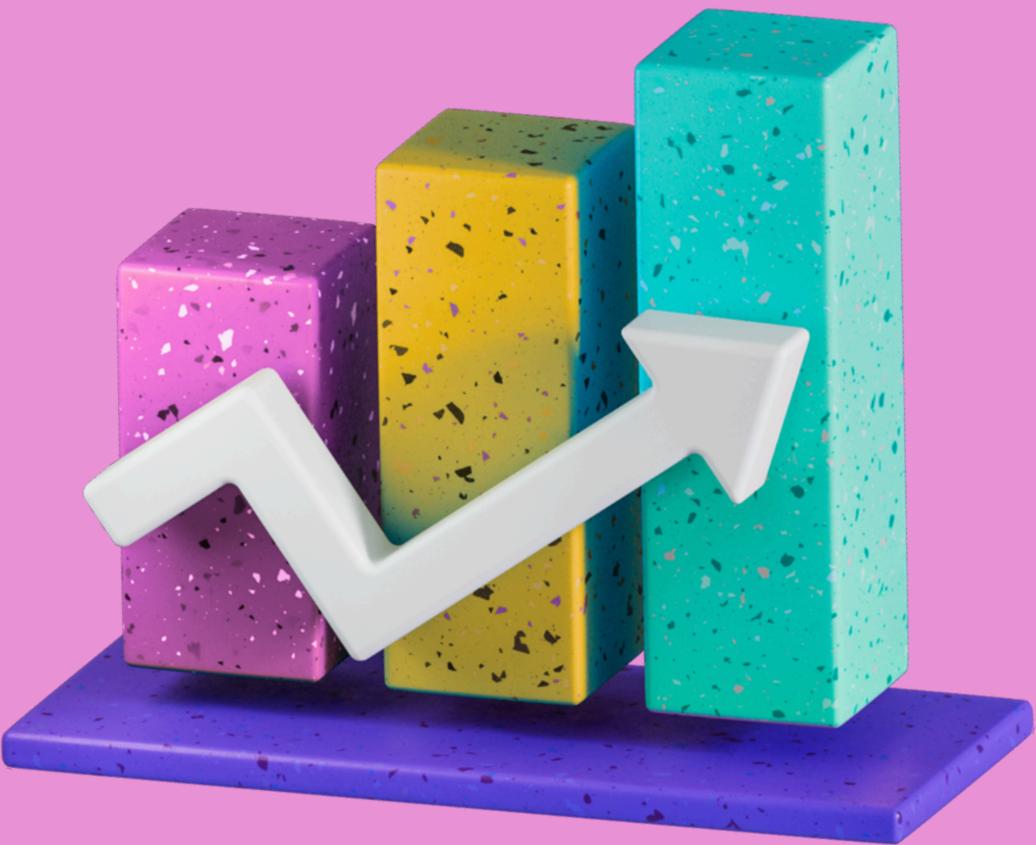
Los integrantes de este trabajo son:

- ✓ Venegas Mendez Janice
- ✓ victoria Zarate Alexandra
- ✓ Paucar Arango Marcos

- ✓ Janice2812
- ✓ Alexandra-2803
- ✓ MarcosIAP



ÍNDICE



01. Presentación del trabajo y objetivos
02. Parámetros
03. Sintaxis del trabajo
04. Extracción, procesamiento y almacenamiento de productos
05. Cálculos estadísticos
06. Acceso y almacenamiento de anuncios (API y csv)
07. Gráficas
08. HTML





Proyecto

Mercado Libre es una empresa multinacional líder en comercio electrónico en Iberoamérica. Su rica información comercial y su enfoque en el comercio electrónico la hacen ideal para analizar tendencias de mercado, y ciertos análisis de sus productos.





Objetivos

Datos de los productos

- Implementar funciones para limpiar y validar los datos obtenidos.
- Calcular estadísticas adicionales sobre el promedio de precios, la disponibilidad promedio de productos, o la cantidad total de productos.
- Agregar funcionalidades de ordenamiento para que los usuarios puedan clasificar los productos por precio, popularidad o cualquier otra métrica relevante.

Datos de los anuncios

- Análisis de Impresiones y Vistas.
- Evaluación del Tamaño del Público Estimado.
- Análisis Temporal.



Parámetros de la API Mercado_Libre



- **q:** Para realizar una búsqueda por palabra clave.
- **category:** Filtrar por una categoría específica usando su ID.
- **limit:** Especifica el número de resultados por página.
- **offset:** Define el punto de partida para la lista de resultados.
- **sort:** Ordena los resultados según diferentes criterios.
- **price:** Filtra los resultados dentro de un rango de precios especificado.
- **official_store:** Filtra para mostrar solo productos de tiendas oficiales.
- **shipping_cost:** Filtra según el costo de envío (gratis o con costo).
- **attributes:** Permite filtrar por atributos específicos del producto.
- **state:** Filtra por el estado de la publicación.

Parámetros de nuestra



- **fecha_final:** Fecha de finalización del anuncio.
- **fecha_inicio:** Fecha de inicio del anuncio.
- **id:** Identificador único del anuncio.
- **importe_gastado:** Importe gastado en el anuncio.
- **impresiones:** Número de impresiones del anuncio.
- **numero_vistas:** Número de vistas del anuncio.
- **tamano_publico_estimado:** Tamaño estimado del público objetivo.
- **titulo:** Título del anuncio del producto.
- **url_imagen:** URL de la imagen del producto.
- **url_producto:** URL del producto en MercadoLibre.

Sintaxis general

Código:

```
import requests
import random

# URL de la API de MercadoLibre para el sitio de México
url = 'https://api.mercadolibre.com/sites/MLM/search'

# Tasa de cambio aproximada de MXN a PEN (solo como ejemplo)
# Esta tasa debe ser actualizada según la tasa de cambio real
tasa_cambio_mx_n_pen = 0.5 # Ejemplo: 1 MXN = 0.5 PEN

# Parámetros de la solicitud para buscar laptops en México
params = {
    'q': 'laptop',
    'currency': 'MXN', # Especificar la moneda en pesos mexicanos
    'limit': 5
}

# Hacer la solicitud GET
response = requests.get(url, params=params)

# Verificar el estado de la respuesta
if response.status_code == 200:
    data = response.json()

    # Procesar los datos
    for item in data['results']:
        title = item['title']

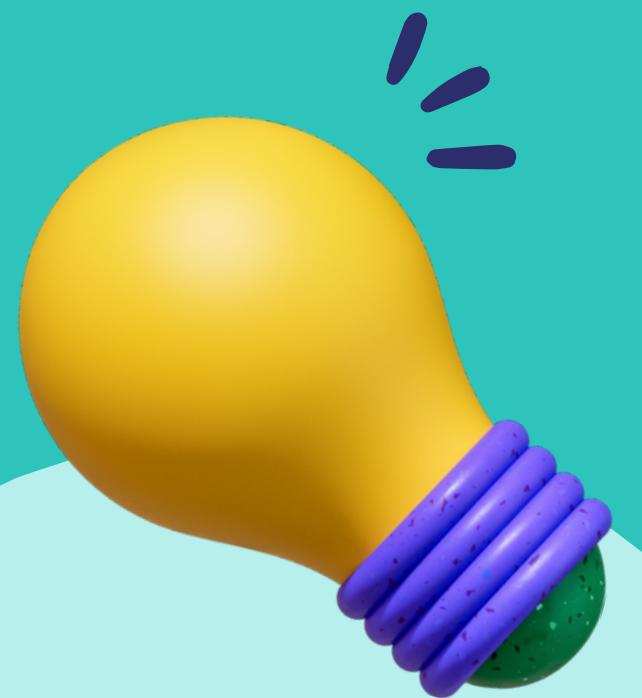
        # Generar un número aleatorio de vistas (por ejemplo, entre 100 y 5000)
        views = random.randint(100, 5000)

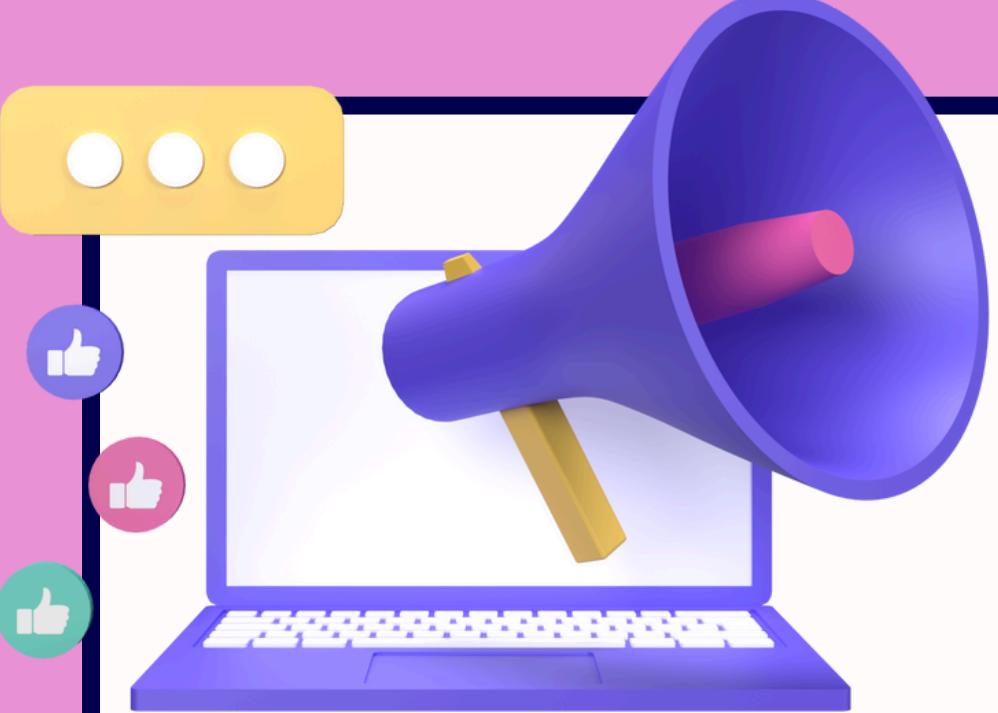
        condition = item.get('condition', 'No disponible')
        available_quantity = item.get('available_quantity', 'No disponible')

        # Imprimir la información del producto
        print(f"Nombre del producto: {title}")
        print(f"Número de vistas: {views}")
        print(f"Condición: {condition}")
        print(f"Cantidad disponible: {available_quantity}")
```

Resultado:

```
Nombre del producto: Apple Macbook Air (13 Pulgadas, 2020, Chip M1, 256 Gb De Ssd, 8 Gb De Ram) - Gris Espacial - Distribuido
r Autorizado
Número de vistas: 2472
Condición: new
Cantidad disponible: 150
---
Nombre del producto: Laptop Acer Aspire 3 15.6 Ryzen 7, 16gb/512gb, Windows 11 Color Plateado
Número de vistas: 813
Condición: new
Cantidad disponible: 50
---
Nombre del producto: Laptop Lenovo Ideapad Celeron 4gb + 128ssd + Office Regalo Color Gris
Número de vistas: 3833
Condición: new
Cantidad disponible: 50
---
Nombre del producto: Laptop Dell 3525, 15.6 Fhd , Ryzen 7, 16gb, 1tb Ssd, W11h Color Black
Número de vistas: 4371
Condición: new
Cantidad disponible: 1
---
Nombre del producto: Laptop Gamer Thunderobot 911mt 12th Intel Core I7 12650h 16gb De Ram 512gb Ssd, Nvidia Geforce Rtx 30
50 165 Hz 1920x1080px Windows 11 Pro
Número de vistas: 942
Condición: new
Cantidad disponible: 50
---
```





Estrategias del trabajo

Los datos que lograremos extraer lo guardaremos en formato json.

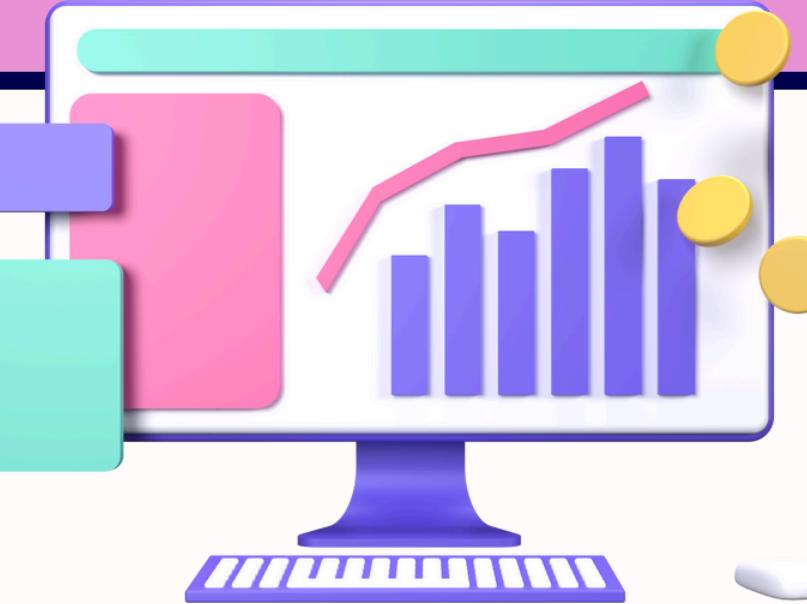
```
import json

# Nombre del archivo JSON
json_filename = 'productos_mercadolibre.json'

# Escribir datos en el archivo JSON
with open(json_filename, 'w', encoding='utf-8') as json_file:
    json.dump(data['results'], json_file, ensure_ascii=False, indent=4)

print(f'Datos guardados exitosamente en {json_filename}')
```

Datos guardados exitosamente en productos_mercadolibre.json



Extracción, procesamiento y almacenamiento de datos

Los datos que lograremos extraer lo limpiaremos , luego se almacenan en un archivo JSON para su posterior análisis y visualización

```
: import json
import pandas as pd
import requests
import random

# URL de La API de MercadoLibre para el sitio de México
url = 'https://api.mercadolibre.com/sites/MLM/search'

# Tasa de cambio aproximada de MXN a PEN (solo como ejemplo)
# Esta tasa debe ser actualizada según La tasa de cambio real
tasa_cambio_mx_n_pen = 0.5 # Ejemplo: 1 MXN = 0.5 PEN

# Parámetros de la solicitud para buscar Laptops en México
params = {
    'q': 'laptop',
    'currency': 'MXN', # Especificar la moneda en pesos mexicanos
    'limit': 10
}

# Hacer la solicitud GET
response = requests.get(url, params=params)

# Verificar el estado de la respuesta
if response.status_code == 200:
    data = response.json()['results']

    # Procesar y limpiar los datos
    cleaned_data = []
    for item in data:
        title = item['title']

        # Verificar si el precio está presente y convertirlo a soles peruanos
        if 'price' in item and isinstance(item['price'], (int, float)):
            price_mx_n = item['price']
            price_pen = price_mx_n * tasa_cambio_mx_n_pen
            price = round(price_pen, 2) # Redondear a dos decimales

            # Convertir a string con símbolo de soles peruanos
            price_str = f"S/ {price:.2f}"
        else:
            price_str = 'No disponible'

        # Generar un número aleatorio de vistas (por ejemplo, entre 100 y 5000)
        views = random.randint(100, 5000)
```

```
views = random.randint(100, 5000)

# Extraer otros datos
image_url = item['thumbnail']
item_url = item['permalink']
condition = item.get('condition', 'No disponible')
available_quantity = item.get('available_quantity', 'No disponible')

# Limpiar y estructurar datos
cleaned_item = {
    'Nombre del producto': title,
    'Precio': price_str,
    'Número de vistas': views,
    'Condición': condition,
    'Cantidad disponible': available_quantity,
    'URL de la imagen': image_url,
    'Enlace al producto': item_url
}

cleaned_data.append(cleaned_item)

# Guardar los datos limpios en un archivo JSON
json_filename = 'productos_mercadolibre_limpios.json'
with open(json_filename, 'w', encoding='utf-8') as json_file:
    json.dump(cleaned_data, json_file, ensure_ascii=False, indent=4)

# print(f'Datos procesados y limpios guardados exitosamente en {json_filename}')

# Crear e imprimir un DataFrame con los datos obtenidos
df = pd.DataFrame(cleaned_data)

else:
    print(f'Error en la solicitud: {response.status_code}')
    print(response.text)

df
```

[3]:	Nombre del producto	Precio	Número de vistas	Condición	Cantidad disponible	URL de la imagen	Enlace al producto
0	Apple Macbook Air (13 Pulgadas, 2020, Chip M1,...	S/ 5999.50	3739	new	150	http://http2.mlstatic.com/D_801112-MLA46516512...	https://www.mercadolibre.com.mx/apple-macbook-...
1	Laptop Acer Aspire 3 15.6 Ryzen 7, 16gb/512gb,...	S/ 4187.00	4000	new	50	http://http2.mlstatic.com/D_701272-MLU75124279...	https://www.mercadolibre.com.mx/laptop-acer-as...
2	Laptop Lenovo Ideapad Celeron 4gb + 128ssd + O...	S/ 2499.50	2573	new	50	http://http2.mlstatic.com/D_640104-MLU72721294...	https://www.mercadolibre.com.mx/laptop-lenovo-...

Cálculos estadísticos básicos

```
[12]: import requests
import json

if response.status_code == 200:
    data = response.json()['results']

    # Lista para almacenar los precios
    prices = []
    cleaned_data = []

    # Extraer precios y limpiar datos
    for item in data:
        # Aquí puedes incluir la lógica para limpiar los datos y extraer los precios
        # Supongamos que el precio está en el campo 'price'
        price = item.get('price')
        if price is not None:
            prices.append(price)
            cleaned_data.append(item)

    # Calcular estadísticas básicas
    if prices:
        avg_price = sum(prices) / len(prices)
        min_price = min(prices)
        max_price = max(prices)
    else:
        avg_price = 0
        min_price = 'No disponible'
        max_price = 'No disponible'

    # Guardar los datos limpios y estadísticas en un archivo JSON
    json_filename = 'productos_mercadolibre_limpios_estadisticas.json'
    data_to_save = {
        'productos': cleaned_data,
        'estadisticas': {
            'Precio promedio': f"${avg_price:.2f}" if avg_price else 'No disponible',
            'Precio mínimo': f"${min_price:.2f}" if isinstance(min_price, (int, float)) else min_price,
            'Precio máximo': f"${max_price:.2f}" if isinstance(max_price, (int, float)) else max_price
        }
    }

    with open(json_filename, 'w', encoding='utf-8') as json_file:
        json.dump(data_to_save, json_file, ensure_ascii=False, indent=4)

    print(f'Datos procesados y estadísticas guardadas exitosamente en {json_filename}')

else:
    print(f'Error en la solicitud: {response.status_code}')
    print(response.text)
```

Datos procesados y estadísticas guardadas exitosamente en productos_mercadolibre_limpios_estadisticas.json

calculamos algunas estadísticas básicas sobre los precios, incluyendo el promedio, mínimo y máximo. Para que finalmente, los datos procesados y las estadísticas se almacenen en un archivo JSON



Accediendo a la api

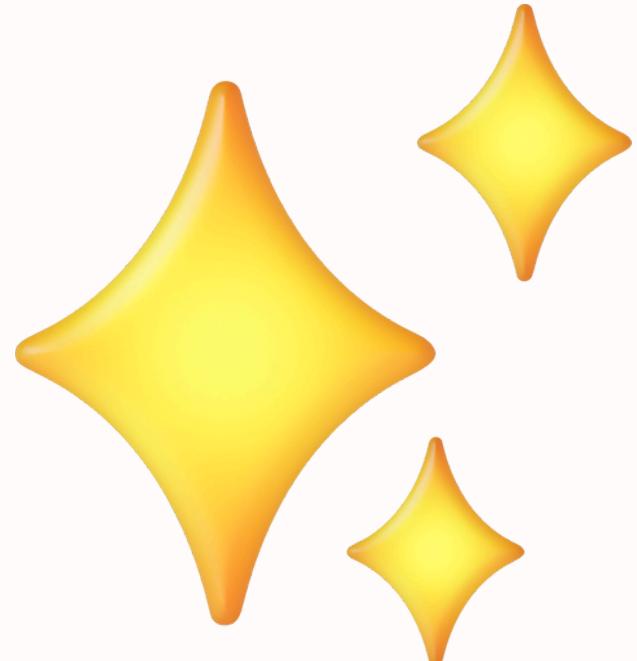
```
] import requests

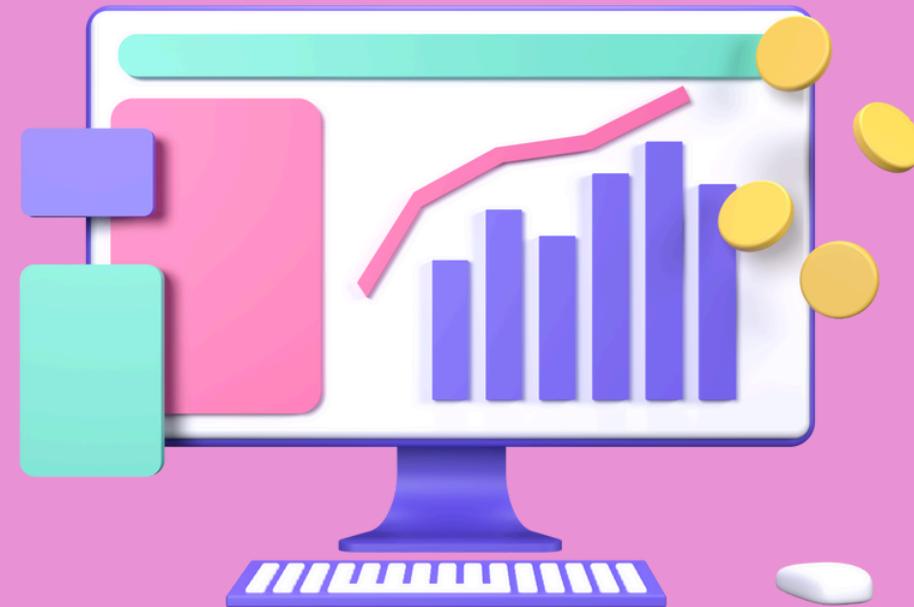
# URL de la API
api_url = 'http://127.0.0.1:5000/anuncios'

# Hacer la solicitud GET a la API
response = requests.get(api_url)

# Verificar si la solicitud fue exitosa
if response.status_code == 200:
    # Convertir la respuesta JSON en un diccionario de Python
    data = response.json()

    # Procesar y mostrar los datos
    for anuncio in data:
        print(f"ID: {anuncio['id']}")
        print(f"Titulo: {anuncio['titulo']}")
        print(f"Tamaño de Público Estimado: {anuncio['tamaño_publico_estimado']}")
        print(f"Importe Gasto: {anuncio['importe_gasto']}")
        print(f"Impresiones: {anuncio['impresiones']}")
        print(f"Fecha Inicio: {anuncio['fecha_inicio']}")
        print(f"Fecha Final: {anuncio['fecha_final']}")
        print(f"URL Imagen: {anuncio['url_imagen']}")
        print(f"URL Producto: {anuncio['url_producto']}")
        print(f"Número de Vistas: {anuncio['numero_vistas']}")
        print("\n")
else:
    print(f"Error al acceder a la API: {response.status_code}")
```





ALMACENAMOS DATOS EN CSV



Nombramos el archivo y definimos los nombres de los campos

```
# Nombre del archivo CSV
csv_filename = 'data_api.csv'

# Definir los nombres de campo para el encabezado del CSV
fieldnames = ['id', 'titulo', 'tamano_publico_estimado', 'importe_gastado',
              'impresiones', 'fecha_inicio', 'fecha_final', 'url_imagen',
              'url_producto', 'numero_vistas']
```



Colocamos los datos en el csv

```
# Guardar en un archivo CSV
with open(csv_filename, 'w', newline='', encoding='utf-8') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

    # Escribir el encabezado
    writer.writeheader()

    # Escribir los datos
    for anuncio in data:
        # Asegurar que el índice sea el campo 'id'
        writer.writerow({'id': anuncio['id'],
                        'titulo': anuncio['titulo'],
                        'tamano_publico_estimado': anuncio['tamano_publico_estimado'],
                        'importe_gastado': anuncio['importe_gastado'],
                        'impresiones': anuncio['impresiones'],
                        'fecha_inicio': anuncio['fecha_inicio'],
                        'fecha_final': anuncio['fecha_final'],
                        'url_imagen': anuncio['url_imagen'],
                        'url_producto': anuncio['url_producto'],
                        'numero_vistas': anuncio['numero_vistas']}))

    print(f"Se ha guardado la información en {csv_filename}")
else:
    print(f"Error al acceder a la API: {response.status_code}")
```

Se ha guardado la información en data_api.csv

MOSTRAMOS LOS DATOS

Utilizamos la librería pandas para leer el csv.

```
[27]: import pandas as pd
df= pd.read_csv('data_api.csv',index_col=0) #index_col=0 para asegurarnos que el indice sea la primera columna
df
```

	titulo	tamano_publico_estimado	importe_gastado	impresiones	fecha_inicio	fecha_final	url_imagen	url_prc
1	Laptop Asus Asus Tuf Gaming A15 Amd R5 Rtx2050...	10 mil	\$./600	4 mil	2023-01-01	2023-01-31	http://http2.mlstatic.com/D_774375-MLU76509333...	https://www.mercadolibre.com.mx/l...
2	Notebook 245 G9 14in Negro 16gb De Ram - 512gb...	20 mil	\$./1000	5 mil	2023-02-01	2023-02-28	http://http2.mlstatic.com/D_676170-MLA74065928...	https://www.mercadolibre.com.mx/nob...
3	Laptop Gamer Thunderobot 911mt 12th Intel Core...	15 mil	\$./800	4 mil	2023-03-01	2023-03-31	http://http2.mlstatic.com/D_901042-MLU76249680...	https://www.mercadolibre.com.mx/l...
4	Laptop Hp 245 G9 Amd Ryzen 3 3250u Hasta	10 mil	\$./600	2 mil	2023-04-01	2023-04-30	http://http2.mlstatic.com/D_661047-MLU73159073...	https://www.mercadolibre.com.mx/l...



RAZÓN ENTRE IMPRESIONES Y VISTAS



Evaluamos la relación entre el número de impresiones y el número de vistas para determinar la efectividad de las campañas publicitarias.

```
dicir, cuantas veces se mostro un anuncio) se comparan con las vistas (cuantas veces se visualizo realmente el anuncio por los usuarios).

10]: data = response.json()
# Calcular métricas clave para cada anuncio
for anuncio in data:

    # Convertir impresiones a int
    impresiones = int(anuncio['impresiones'].replace(' mil', '000').replace(',', ''))

    # Convertir numero_vistas a int
    numero_vistas = int(anuncio['numero_vistas'])

    # Razón impresiones a vistas
    razon_impresiones_vistas = impresiones / numero_vistas if numero_vistas else 0

    # Imprimir resultados para cada anuncio
    print(f"ID: {anuncio['id']}")
    print(f"Titulo: {anuncio['titulo']}")
    print(f"Impresiones: {anuncio['impresiones']}")
    print(f"Razón Impresiones a Vistas: {razon_impresiones_vistas:.2f}")
    print("-----")

import matplotlib.pyplot as plt

# Graficar La relación entre Impresiones y Vistas
plt.figure(figsize=(10, 6))
plt.scatter(df['impresiones'], df['numero_vistas'])
plt.title('Relación entre Impresiones y Vistas')
plt.xlabel('Impresiones')
plt.ylabel('Vistas')
plt.grid(True)
plt.show()

ID: 1
Título: Laptop Asus Asus Tuf Gaming A15 Amd R5 Rtx2050 16gb 512gb Color Negro
Impresiones: 4 mil
Razón Impresiones a Vistas: 2.43
-----
ID: 2
Título: Notebook 245 G9 14in Negro 16gb De Ram - 512gb Ssd - Amd Ryzen 3
Impresiones: 5 mil
Razón Impresiones a Vistas: 3.33
-----
ID: 3
Título: Portátil Acer Nitro 5 AN515-55-57HJ i5-11300H 15.6" FHD 144Hz 16GB RAM 512GB SSD RTX3060 6GB GDDR6
```

MOSTRAMOS LA RELACION MEDIANTE ESTE GRAFICO Y ADICIONAMOS LOS DATOS



ID: 1
Título: Laptop Asus Asus Tuf Gaming A15 Amd R5 Rtx2050 16gb 512gb Color Negro
Impresiones: 4 mil
Razón Impresiones a Vistas: 2.43

ID: 2
Título: Notebook 245 G9 14in Negro 16gb De Ram - 512gb Ssd - Amd Ryzen 3
Impresiones: 5 mil
Razón Impresiones a Vistas: 3.33

ID: 3
Título: Laptop Gamer Thunderobot 911mt 12th Intel Core I7 12650h 16gb De Ram 512gb Ssd, Nvidia Geforce Rtx 3050 165 Hz 1920x1080px Windows 11 Pro
Impresiones: 4 mil
Razón Impresiones a Vistas: 8.16

ID: 4
Título: Laptop Hp 245 G9 Amd Ryzen 3 3250u Hasta 3,5 Ghz, Memoria Ram De 16 Gb Ddr4, Ssd 256 Gb, Windows 11 Home 64-bit, Teclado En Español, 14 Pulgadas, Negro
Impresiones: 2 mil
Razón Impresiones a Vistas: 7.58

ID: 5
Título: Laptop Lenovo Ideapad Slim 3 15.6'' Ci5 8gb + 512gb Ssd
Impresiones: 5 mil
Razón Impresiones a Vistas: 8.55

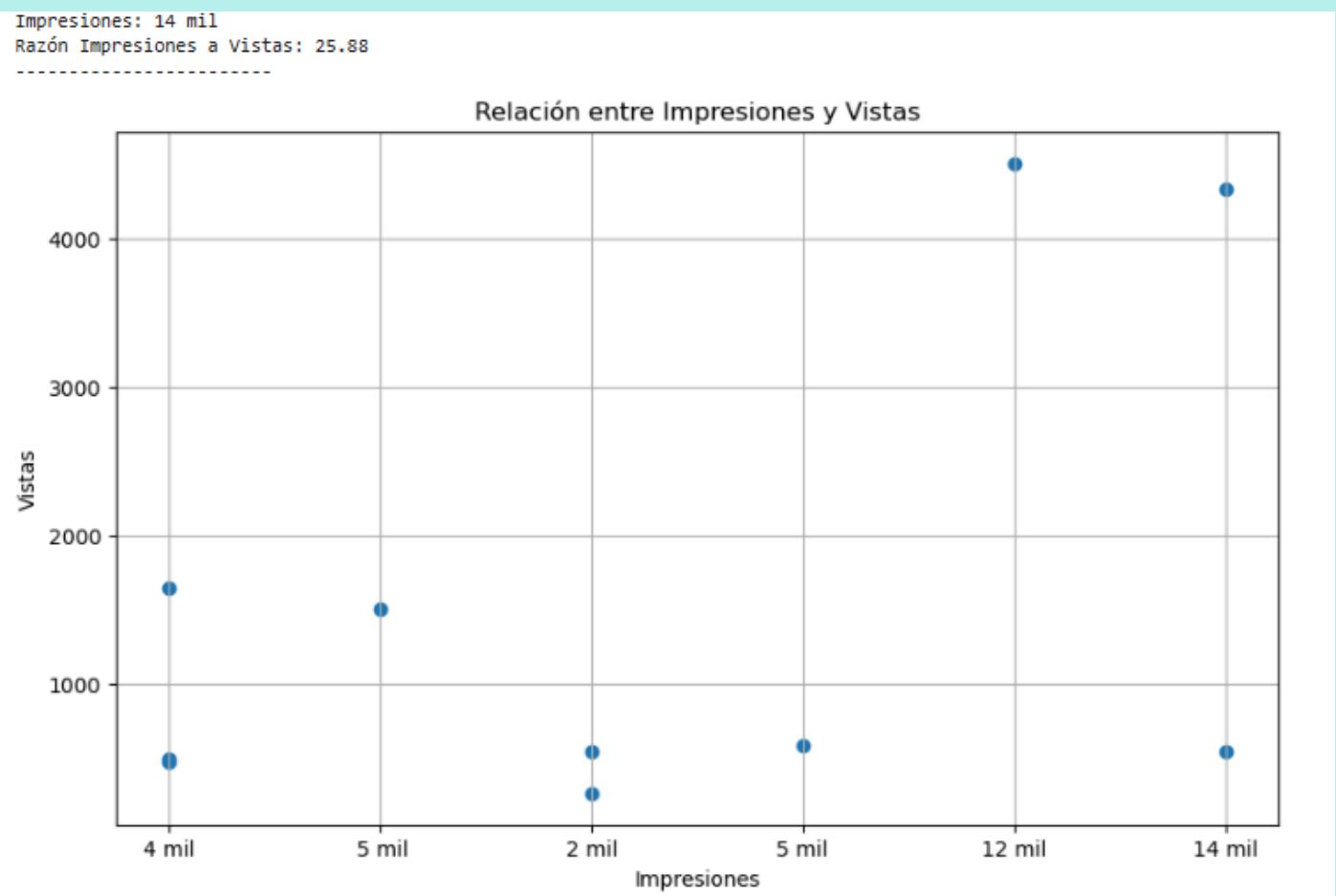
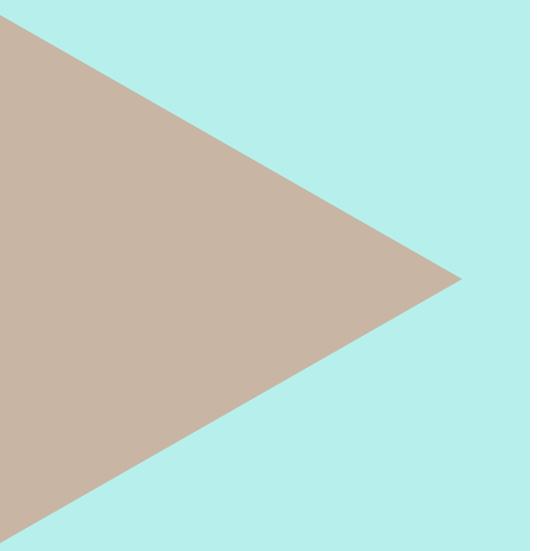
ID: 6
Título: Laptop Acer Aspire 3 15.6 Ryzen 7, 16gb/512gb, Windows 11 Color Plateado
Impresiones: 12 mil
Razón Impresiones a Vistas: 2.66

ID: 7
Título: Laptop Asus Vivobook F15 15.6 Core I7 1255u 16g 512g Ssd W11 Color Azul
Impresiones: 14 mil
Razón Impresiones a Vistas: 3.23

ID: 8
Título: Laptop Hp 240 G9 Intel Core I3 1215u 512gb Ssd 16gb Ram 1366x768px Windows 11 Home + Antivirus Norton 360
Impresiones: 4 mil
Razón Impresiones a Vistas: 8.53

ID: 9
Título: 14.1'' Laptop 8 Ram+512gb Ssd Intel Celeron N4000 Windows 11
Impresiones: 2 mil
Razón Impresiones a Vistas: 3.70

ID: 10
Título: Notebook Victus 15-fb1013dx 15.6 Gris 16gb De Ram - 512gb Ssd - Amd Ryzen 5
Impresiones: 14 mil
Razón Impresiones a Vistas: 25.88





ANÁLISIS TEMPORAL



Este análisis nos ayudará a ver cómo evoluciona el rendimiento a lo largo de diferentes campañas en el tiempo, y para ello usaremos un gráfico de líneas

```
import plotly.graph_objs as go
from datetime import datetime

# Preparar datos para el gráfico de Líneas
fechas = []
impresiones = []
vistas = []

for anuncio in data:
    fecha_inicio = datetime.strptime(anuncio['fecha_inicio'], '%Y-%m-%d')
    fecha_final = datetime.strptime(anuncio['fecha_final'], '%Y-%m-%d')
    duracion_campana = (fecha_final - fecha_inicio).days
    fechas.append(fecha_inicio)
    vistas.append(int(anuncio['numero_vistas']))
    impresiones.append(int(anuncio['impresiones'].replace(' mil', '000')))

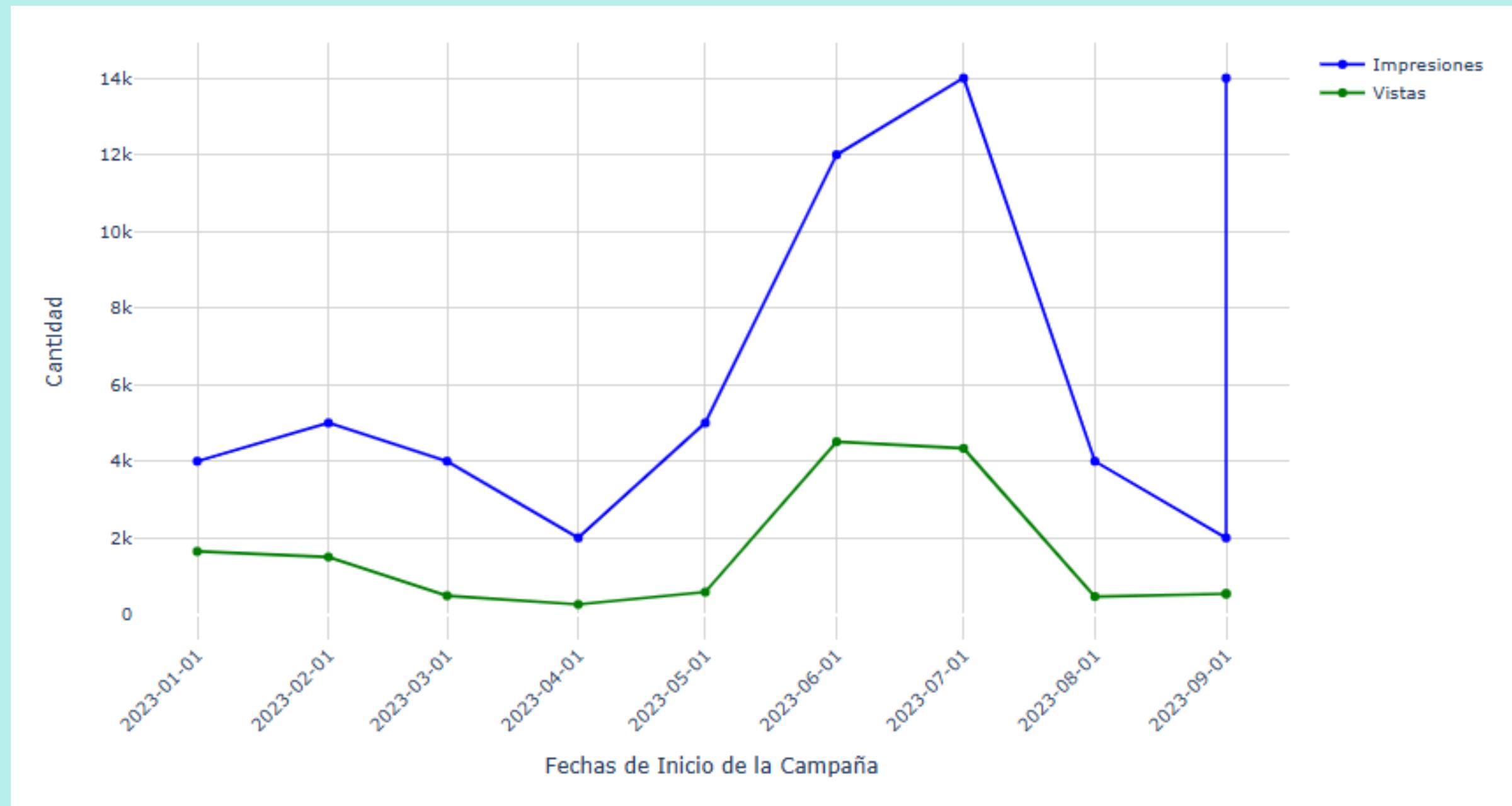
# Crear La figura interactiva con Plotly
fig = go.Figure()

# Añadir Las Líneas de impresiones y vistas con colores personalizados
fig.add_trace(go.Scatter(x=fechas, y=impresiones, mode='lines+markers', name='Impresiones', line=dict(color='blue')))
fig.add_trace(go.Scatter(x=fechas, y=vistas, mode='lines+markers', name='Vistas', line=dict(color='green')))

# Configurar diseño y etiquetas
fig.update_layout(
    title='<b>Evolución de Impresiones y Vistas por Fecha de Inicio de la Campaña</b>',
    title_x=0.5, # Centrar el título
    xaxis_title='Fechas de Inicio de la Campaña',
    yaxis_title='Cantidad',
    xaxis_tickformat='%Y-%m-%d',
    xaxis_tickangle=-45,
    hovermode='x',
    width=1000, # Ancho del gráfico
    height=600, # Alto del gráfico
    plot_bgcolor='white' , # Color de fondo del gráfico
    xaxis=dict(showgrid=True, gridcolor='lightgray'), # Mostrar cuadriculas en el eje x
    yaxis=dict(showgrid=True, gridcolor='lightgray') # Mostrar cuadriculas en el eje y
)

# Mostrar La figura interactiva
fig.show()
```

GRAFICO DE EVALUACION DE IMPRESIONES Y VISTAS POR FECHA DE INICIO DE CAMPAÑA



Creando un html



Importamos librerías y leemos el archivo json

```
[2]: import json
from jinja2 import Template
json_filename = 'anuncios.json'
with open(json_filename, 'r', encoding='utf-8') as json_file:
    data = json.load(json_file)
```



Aplicamos template y guardamos el html

```
# Crear un objeto de template Jinja2
template = Template(template_html)

# Renderizar la plantilla con los datos
rendered_html = template.render(data=data)

# Guardar el HTML generado en un archivo
with open('biblioteca_anuncios.html', 'w', encoding='utf-8') as file:
    file.write(rendered_html)

print("Archivo HTML generado exitosamente.")

Archivo HTML generado exitosamente.
```



Creamos la plantilla html

```
# Plantilla HTML con Bootstrap integrado
template_html = """
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Biblioteca de Anuncios JAM Nexus</title>
    <!-- Bootstrap CSS -->
    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">
    <!-- Estilos personalizados -->
    <style>
        body {
            font-family: 'Roboto', sans-serif;
            margin: 20px;
            background-color: #f4f4f9; /* Fondo gris claro para toda la página */
            transition: background-color 0.3s; /* Transición suave del color de fondo */
        }
        body:hover {
            background-color: #e9ecef; /* Cambia el color de fondo al pasar el mouse */
        }
        h2 {
            text-align: center;
            color: #007bff; /* Color azul */
            margin-bottom: 20px;
            transition: color 0.3s; /* Transición suave del color del título */
        }
        h2:hover {
```

MOSTRANDO HTML

En este html podemos ver la información de cada anuncio junto con sus métricas correspondientes.

Biblioteca de Anuncios JAM Nexu

Esta biblioteca contiene información de anuncios sobre laptops. Cada tarjeta de anuncio muestra la imagen de la laptop junto con detalles como la fecha de inicio, fecha de fin, costo, impresiones y número de vistas. Las transiciones suaves de color y efectos de escala y sombra hacen que la experiencia de usuario sea más dinámica al interactuar con los elementos de la página. .



Laptop Asus Tuf Gaming
A15 Amd R5 Rtx2050 16gb
512gb Color Negro
Impresiones: 4 mil
Número de Vistas: 1647
Tamaño Público Estimado: 10 mil
Fecha Inicio: 2023-01-01
Fecha Final: 2023-01-31
Importe Gasto: S/.600

[Ver Producto](#)



Notebook 245 G9 14in Negro
16gb De Ram - 512gb Ssd - Amd Ryzen 3
Impresiones: 5 mil
Número de Vistas: 1501
Tamaño Público Estimado: 20 mil
Fecha Inicio: 2023-02-01
Fecha Final: 2023-02-28
Importe Gasto: S/.1000

[Ver Producto](#)



Laptop Gamer Thunderobot
911mt 12th Intel Core I7 12650h
16gb De Ram 512gb Ssd, Nvidia Geforce Rtx 3050 165 Hz
1920x1080px Windows 11 Pro
Impresiones: 4 mil
Número de Vistas: 490
Tamaño Público Estimado: 15 mil
Fecha Inicio: 2023-03-01
Fecha Final: 2023-03-31
Importe Gasto: S/.800

**Muchas
GRACIAS**

