



Tecnológico de Monterrey

Reporte técnico : Liverpool

Edmundo Iván Hernández Rosales - A01665447

Noel Sebastián Márquez Tovar - A01659730

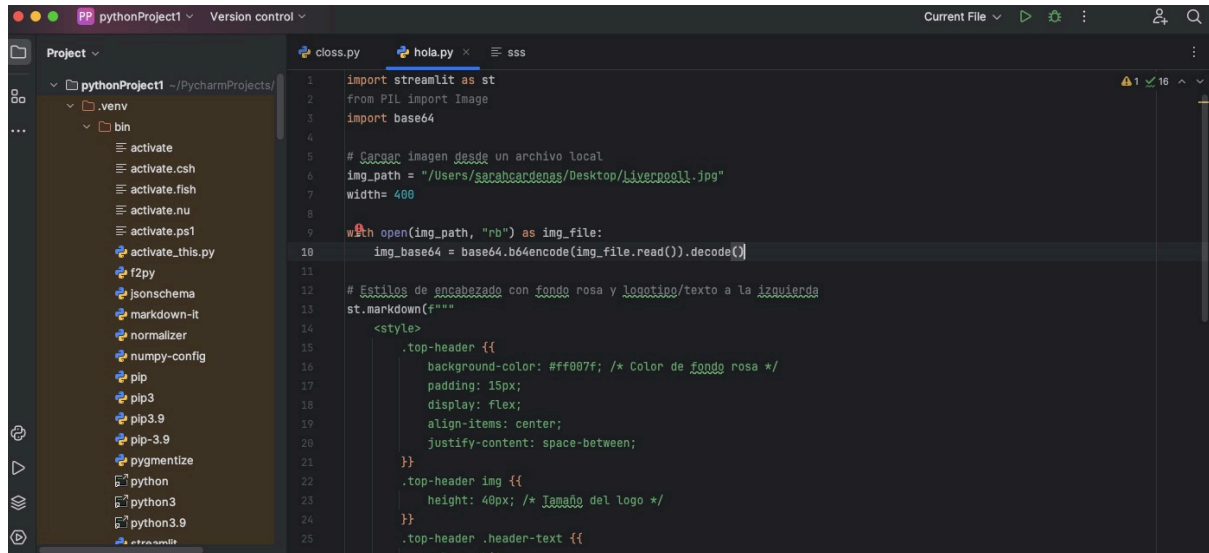
David Isaí Díaz Avilés - A01666448

Sarah Cárdenas Montiel - A01667602

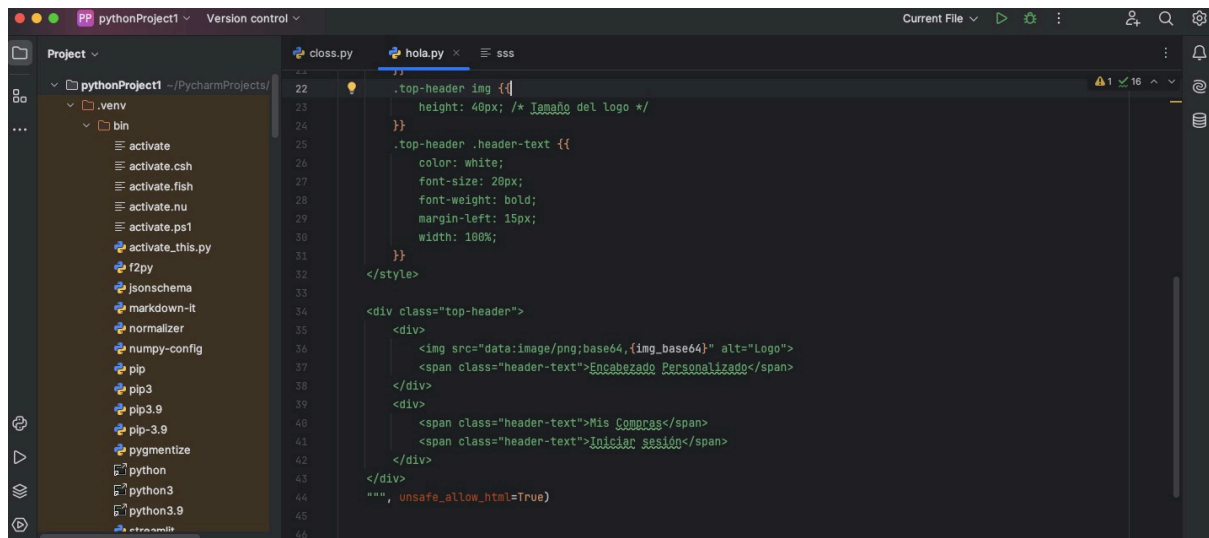
Carolina Pérez Valencia - A01665909

Reporte técnico:

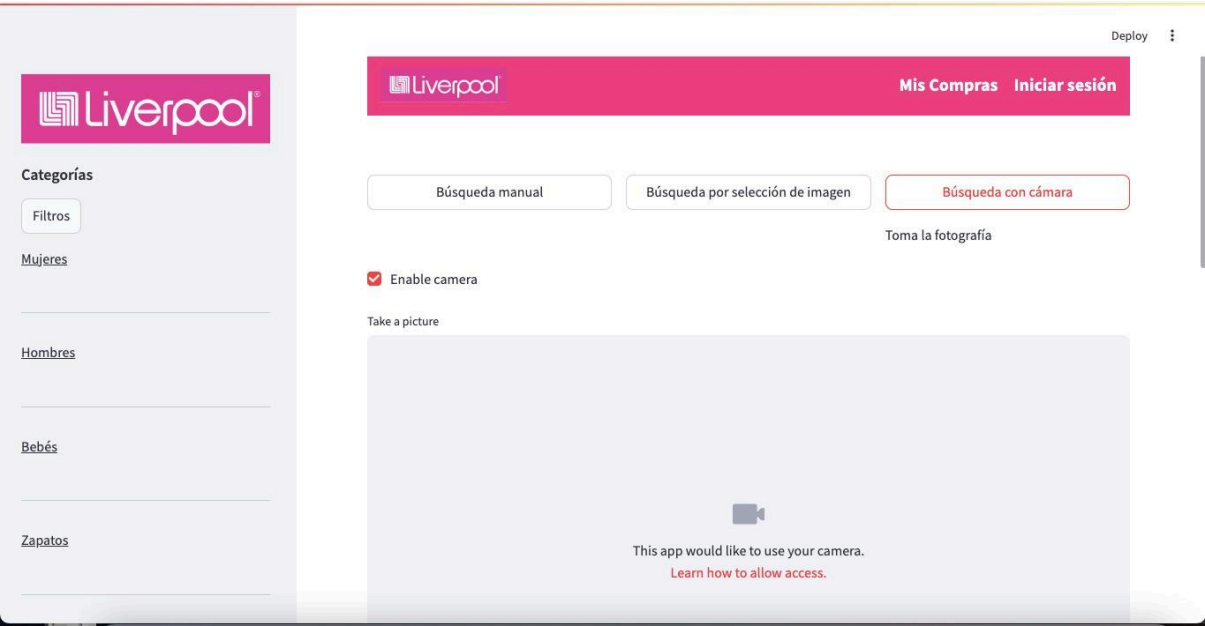
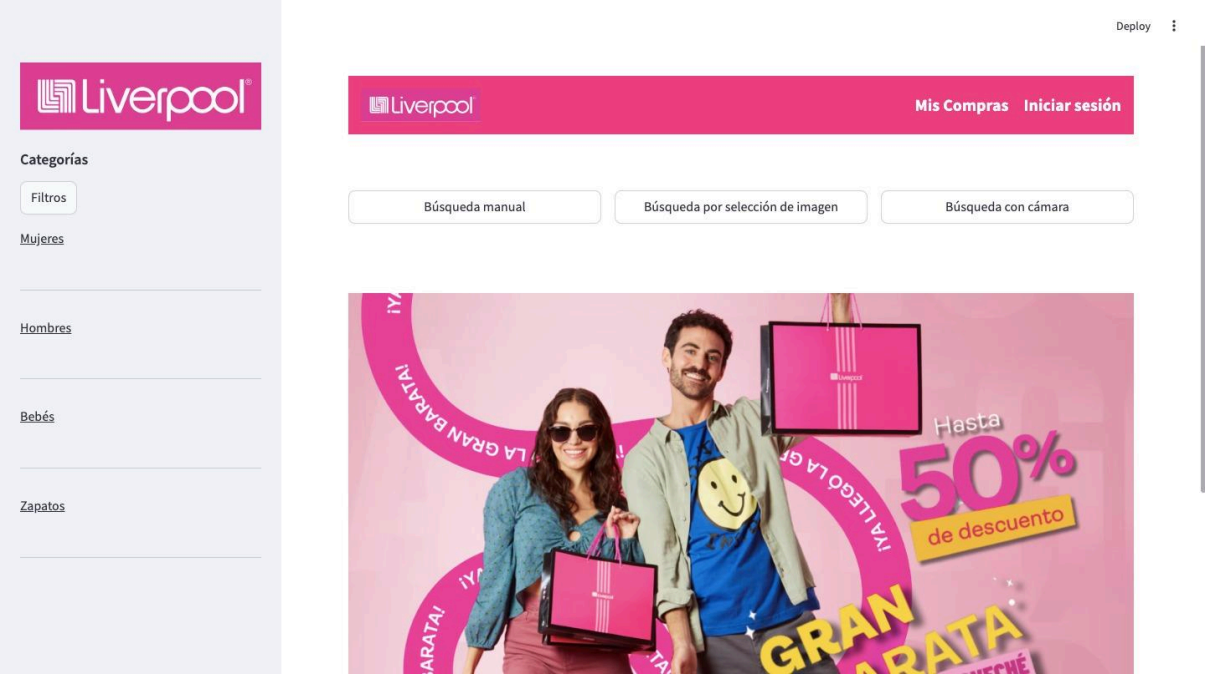
1.- App desarrollada

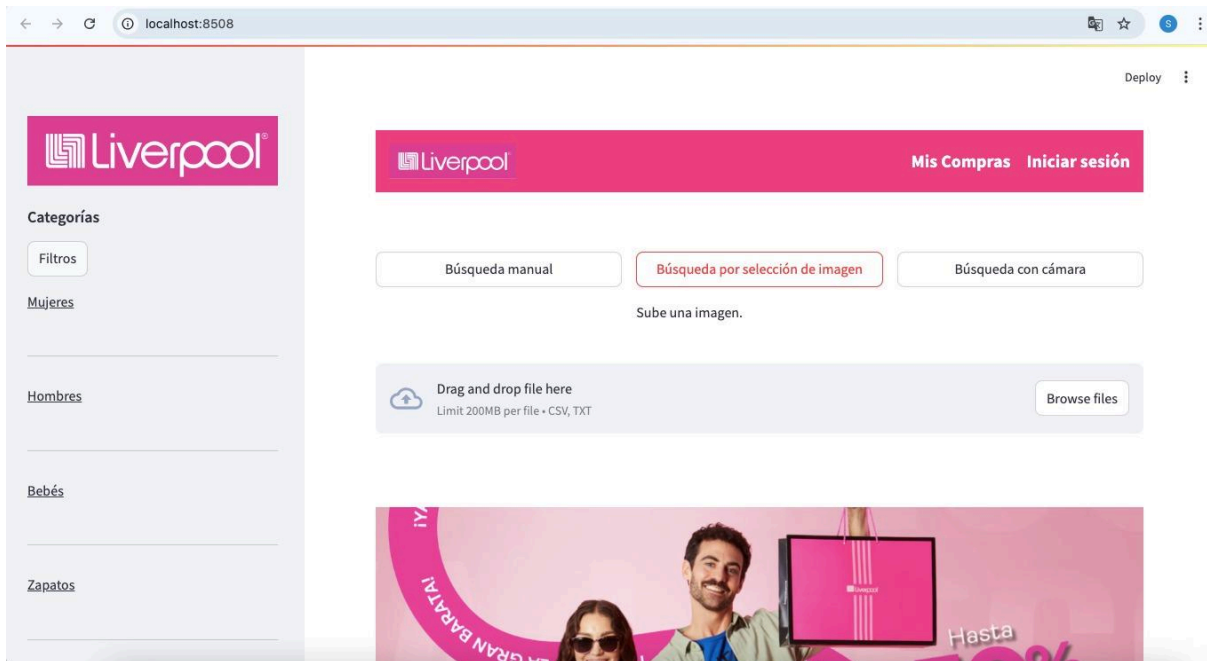


```
1 import streamlit as st
2 from PIL import Image
3 import base64
4
5 # Carga imagen desde un archivo local
6 img_path = "/Users/sarahcardeñas/Desktop/Liverpool.jpg"
7 width= 400
8
9 with open(img_path, "rb") as img_file:
10     img_base64 = base64.b64encode(img_file.read()).decode()
11
12 # Estilos de encabezado con fondo rosa y logo y texto a la izquierda
13 st.markdown("""
14 <style>
15     .top-header {
16         background-color: #ff007f; /* Color de fondo rosa */
17         padding: 15px;
18         display: flex;
19         align-items: center;
20         justify-content: space-between;
21     }
22     .top-header img {
23         height: 40px; /* Tamaño del logo */
24     }
25     .top-header .header-text {
26         color: white;
27         font-size: 28px;
28         font-weight: bold;
29         margin-left: 15px;
30         width: 100%;
31     }
32 </style>
33 <div class="top-header">
34     <div>
35         
36         <span class="header-text">Encabezado Personalizado</span>
37     </div>
38     <div>
39         <span class="header-text">Mis Compras</span>
40         <span class="header-text">Iniciar sesión</span>
41     </div>
42 </div>
43 """, unsafe_allow_html=True)
```



```
22 .top-header img {
23     height: 40px; /* Tamaño del logo */
24 }
25 .top-header .header-text {
26     color: white;
27     font-size: 28px;
28     font-weight: bold;
29     margin-left: 15px;
30     width: 100%;
31 }
32 </style>
33 <div class="top-header">
34     <div>
35         
36         <span class="header-text">Encabezado Personalizado</span>
37     </div>
38     <div>
39         <span class="header-text">Mis Compras</span>
40         <span class="header-text">Iniciar sesión</span>
41     </div>
42 </div>
43 """, unsafe_allow_html=True)
```





2. Carga y Preprocesamiento del Catálogo de Imágenes

```

1 from unittest.mock import Inplace
2
3 #1. Eliminación de URLs duplicadas
4 import pandas as pd
5
6 # Carga directamente el archivo limpio si ya existe
7 file_path_limpio = '/Users/noel/PyCharmProjects/EMB/imgs_limpio.xlsx'
8 df = pd.read_excel(file_path_limpio)
9
10
11 columnas_imagenes = ['Imagen 1', 'Imagen 2', 'Imagen 3', 'Imagen 4', 'Imagen 5', 'Imagen 6', 'Imagen 7',
12 # Paso 1: Crear un conjunto para rastrear URLs únicas
13 url_vistas = set()
14
15 # Paso 2: Eliminar duplicados en cada columna de URLs

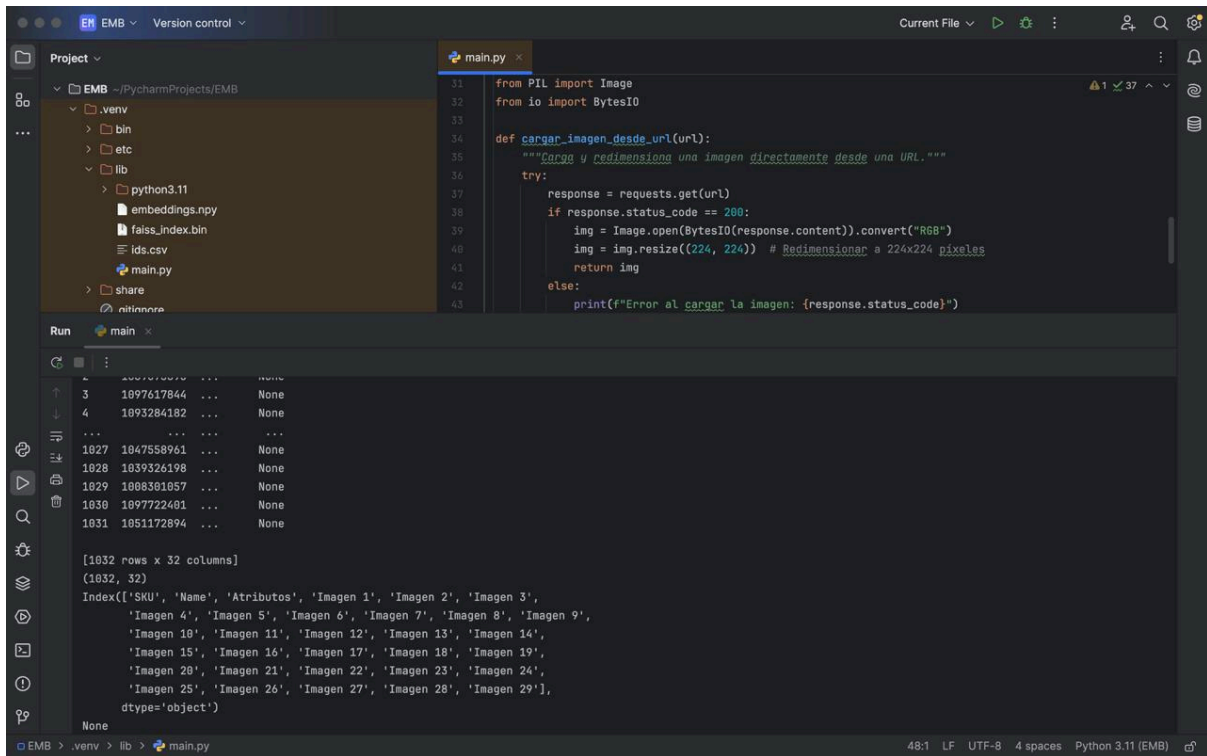
```

```

4 1093284182 ... None
...
1027 1047558961 ... None
1028 1039326198 ... None
1029 1008301057 ... None
1030 1097722401 ... None
1031 1051172894 ... None

[1032 rows x 32 columns]
(1032, 32)
Index(['SKU', 'Name', 'Atributos', 'Imagen 1', 'Imagen 2', 'Imagen 3',
      'Imagen 4', 'Imagen 5', 'Imagen 6', 'Imagen 7', 'Imagen 8', 'Imagen 9',
      'Imagen 10', 'Imagen 11', 'Imagen 12', 'Imagen 13', 'Imagen 14',
      'Imagen 15', 'Imagen 16', 'Imagen 17', 'Imagen 18', 'Imagen 19',
      'Imagen 20', 'Imagen 21', 'Imagen 22', 'Imagen 23', 'Imagen 24',
      'Imagen 25', 'Imagen 26', 'Imagen 27', 'Imagen 28', 'Imagen 29'],
      dtype='object')

```

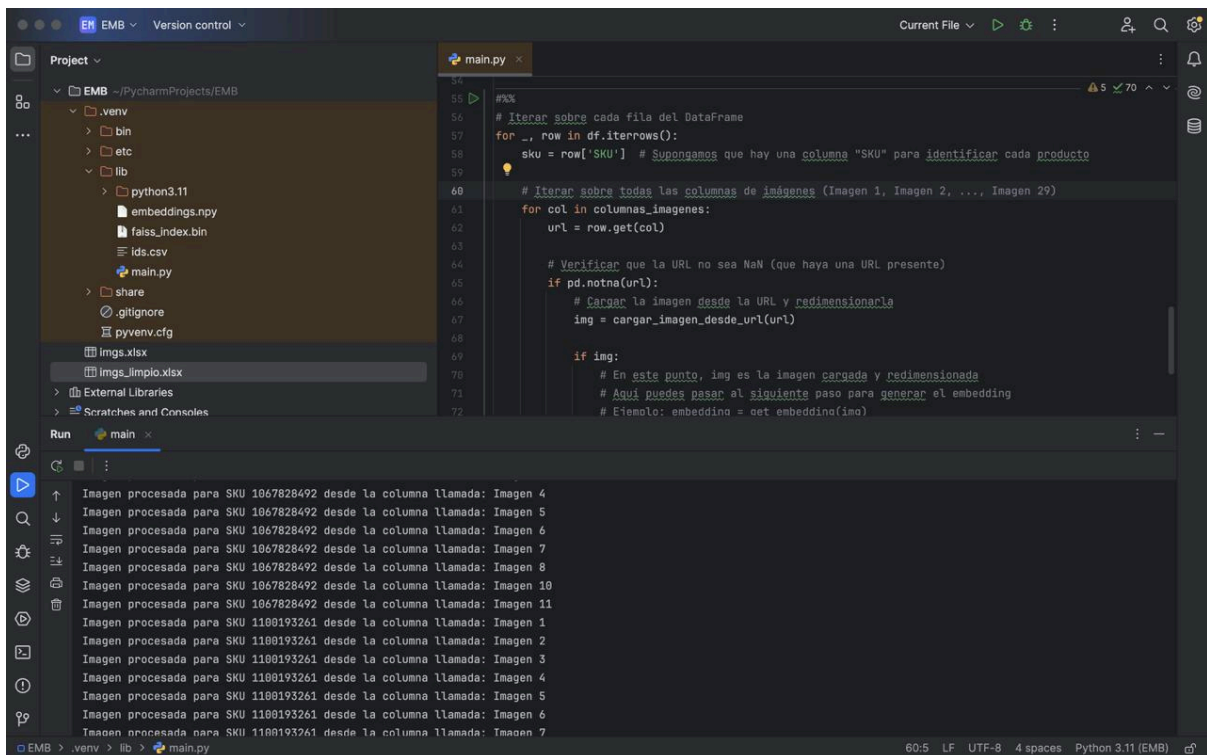


```
31 from PIL import Image
32 from io import BytesIO
33
34 def cargar_imagen_desde_url(url):
35     """Carga y redimensiona una imagen directamente desde una URL."""
36     try:
37         response = requests.get(url)
38         if response.status_code == 200:
39             img = Image.open(BytesIO(response.content)).convert("RGB")
40             img = img.resize((224, 224)) # Redimensionar a 224x224 pixels
41             return img
42         else:
43             print(f"Error al cargar la imagen: {response.status_code}")
```

Run console output:

```
1032 1097617844 ... None
1033 1093284182 ... None
...
1027 1047558961 ... None
1028 1039326198 ... None
1029 1088391057 ... None
1030 1097722481 ... None
1031 1051172894 ... None

[1032 rows x 32 columns]
(1032, 32)
Index(['SKU', 'Name', 'Atributos', 'Imagen 1', 'Imagen 2', 'Imagen 3',
      'Imagen 4', 'Imagen 5', 'Imagen 6', 'Imagen 7', 'Imagen 8', 'Imagen 9',
      'Imagen 10', 'Imagen 11', 'Imagen 12', 'Imagen 13', 'Imagen 14',
      'Imagen 15', 'Imagen 16', 'Imagen 17', 'Imagen 18', 'Imagen 19',
      'Imagen 20', 'Imagen 21', 'Imagen 22', 'Imagen 23', 'Imagen 24',
      'Imagen 25', 'Imagen 26', 'Imagen 27', 'Imagen 28', 'Imagen 29'],
      dtype='object')
```



```
54 #%%
55 # Iterar sobre cada fila del DataFrame
56 for _, row in df.iterrows():
57     sku = row['SKU'] # Supongamos que hay una columna "SKU" para identificar cada producto
58
59     # Iterar sobre todas las columnas de imágenes (Imagen 1, Imagen 2, ..., Imagen 29)
60     for col in columnas_imagenes:
61         url = row.get(col)
62
63         # Verificar que la URL no sea NaN (que haya una URL presente)
64         if pd.isna(url):
65             # Cargar la imagen desde la URL y redimensionarla
66             img = cargar_imagen_desde_url(url)
67
68             # En este punto, img es la imagen cargada y redimensionada
69             # Aquí puedes pasar al siguiente paso para generar el embedding
70             # Ejemplo: embedding = get_embedding(img)
```

Run console output:

```
Imagen procesada para SKU 1067828492 desde la columna llamada: Imagen 4
Imagen procesada para SKU 1067828492 desde la columna llamada: Imagen 5
Imagen procesada para SKU 1067828492 desde la columna llamada: Imagen 6
Imagen procesada para SKU 1067828492 desde la columna llamada: Imagen 7
Imagen procesada para SKU 1067828492 desde la columna llamada: Imagen 8
Imagen procesada para SKU 1067828492 desde la columna llamada: Imagen 10
Imagen procesada para SKU 1067828492 desde la columna llamada: Imagen 11
Imagen procesada para SKU 1100193261 desde la columna llamada: Imagen 1
Imagen procesada para SKU 1100193261 desde la columna llamada: Imagen 2
Imagen procesada para SKU 1100193261 desde la columna llamada: Imagen 3
Imagen procesada para SKU 1100193261 desde la columna llamada: Imagen 4
Imagen procesada para SKU 1100193261 desde la columna llamada: Imagen 5
Imagen procesada para SKU 1100193261 desde la columna llamada: Imagen 6
Imagen procesada para SKU 1100193261 desde la columna llamada: Imagen 7
```

Lo que se deberá realizar:

- Procesar las imágenes ubicadas en data/imágenes.
- Generar embeddings utilizando un modelo pre entrenado.
- Guardar los embeddings en embeddings.npy para su uso en el servidor.
- Cargar los embeddings en el índice FAISS para habilitar búsquedas de similitud.

3. Configuración del Modelo de Embeddings

Objetivo: Entrenar a la IA para la identificación de imágenes en las que podamos encontrar similitudes con las fotos presentadas, para convertir las imágenes en embeddings.

Tareas:

- Crear una función para generar el embedding a partir de una imagen.

4. Creación y Gestión del Índice FAISS

Objetivo: Configurar un índice FAISS que almacene los embeddings del catálogo y permita realizar búsquedas rápidas de similitud.

Tarea:

- Implementar una función de búsqueda de similitud para encontrar los embeddings más cercanos a un vector dado.

5. Desarrollo de la App con streamlit

Objetivo: Desarrollar una App con scanner o mediante la ayuda de una foto

- Recibir imágenes desde la app para generar embeddings.
- Consultar el índice FAISS..

6. Integración con la App Móvil

Objetivo: Permitir que la app móvil capture una foto y la envíe al servidor para su procesamiento.

Tareas:

- Recibir y desplegar los productos similares en la interfaz.
- Implementar validaciones para manejar errores (ej. servidor sin respuesta).

7. Pruebas y Optimización

Objetivo: Verificar la precisión y eficiencia del sistema.

Tareas:

- Probar el sistema con imágenes de prueba y evaluar la relevancia de los resultados.
- Optimizar el tiempo de respuesta y el procesamiento de imágenes.
- Implementar optimizaciones como índices aproximados, para mejorar la velocidad de búsqueda.

8. Documentación:

Objetivo: Proveer documentación clara para uso, despliegue y mantenimiento del sistema.

Tareas:

- Documentar la estructura del proyecto y el flujo de trabajo.

9. Resultados preliminares

Objetivo: Proyectar mejoras a largo plazo para optimizar el sistema.

Ideas:

- Integrar la base de datos proporcionada por la misma empresa.
- Mejorar la precisión mediante técnicas de aprendizaje.
- Implementar un sistema de recomendación que combine la similitud de imágenes con el historial de compras del usuario.