Actividad: Simulacro de ETL con una API pública

Objetivo:

Utilizar Python para interactuar con una API pública, extraer datos, transformarlos utilizando Pandas y guardar el resultado como un archivo CSV.

1. **Configuración inicial**
   - Importar las bibliotecas necesarias:

```
import requests
import pandas as pd
```

2. **Interactuar con la API**
   - Hacer una solicitud GET a la API:

```
response = requests.get('URL_de_la_API')
```

   - Verificar el estado de la solicitud (código de respuesta).
   - Explorar la estructura de los datos en la respuesta:

```
data = response.json()
```

3. **Transformación de datos**
   - Crear un DataFrame de Pandas con los datos obtenidos:

```
df = pd.DataFrame(data)
```

- Realizar cualquier manipulación o limpieza necesaria en los datos utilizando funciones de Pandas (`dropna()`, `fillna()`, etc.).

4. **Guardar como CSV**
   - Guardar el DataFrame resultante como un archivo CSV:

     ```
     df.to_csv('datos_extraidos.csv', index=False)
     ```

5. **Desafío adicional (opcional)**
   - Realizar análisis exploratorio de los datos: contar valores, calcular estadísticas básicas, etc.
   - Visualizar los datos utilizando gráficos simples con la ayuda de la biblioteca pandas.

Ejemplo práctico:

```
import requests
import pandas as pd

#Leer y comprobar que nos devuelva datos:
response = requests.get('https://api.example.com/data')
if response.status_code == 200:
    data = response.json()

    df = pd.DataFrame(data)
    # Realizar cualquier manipulación o limpieza necesaria

    # Paso 5: Guardar como CSV
    df.to_csv('datos_extraidos.csv', index=False)
    print("Datos guardados correctamente como datos_extraidos.cs
else:
    print("Error al obtener los datos de la API")
```

Opciones de API:

1. **API de JSONPlaceholder:** Una API de prueba que simula un servicio RESTful con datos en formato JSON. Puedes obtener datos ficticios de usuarios, publicaciones, comentarios, etc. JSONPlaceholder
2. **API de PokeAPI:** Proporciona información sobre Pokémon, como detalles sobre especies, movimientos, tipos, habilidades, etc. PokeAPI
3. **API de SpaceX:** Ofrece datos sobre misiones, cohetes, naves espaciales, etc., relacionados con SpaceX. SpaceX API
4. **API de REST Countries:** Proporciona información sobre países, incluyendo detalles como moneda, idioma, capital, población, etc. REST Countries
5. **API de COVID-19:** Diversas APIs ofrecen datos en tiempo real o históricos sobre la pandemia de COVID-19, como datos de casos, pruebas, vacunaciones, etc. COVID-19 Data API o COVID-19 API

```
import requests
import pandas as pd
import matplotlib.pyplot as plt

def obtener_info_pokemon(numero):
    url = f"https://pokeapi.co/api/v2/pokemon/{numero}"
    response = requests.get(url)
    if response.status_code == 200:
        data = response.json()
        return {
            'Nombre': data['name'],
            'Altura': data['height'],
            'Peso': data['weight'],
            'Tipo(s)': [tipo['type']['name'] for tipo in data['t
        }
    else:
        print(f"No se pudo obtener la información del Pokémon {n
        return None

# Obtener información de 10 Pokémon
pokemones = []
```

```
for numero in range(1, 11):
    info = obtener_info_pokemon(numero)
    if info:
        pokemones.append(info)

# Convertir la lista de diccionarios en un DataFrame de Pandas
df = pd.DataFrame(pokemones)

# Contar la cantidad de cada tipo de Pokémon
tipo_counts = df['Tipo(s)'].explode().value_counts()

# Mostrar un gráfico de barras
tipo_counts.plot(kind='bar', figsize=(10, 6), color='skyblue')
plt.title('Cantidad de cada tipo de Pokémon')
plt.xlabel('Tipo de Pokémon')
plt.ylabel('Cantidad')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```