Implementacion de RAG con Mistral y LlamaIndex. El LLM utilizado es Mistral mistral-medium-latest. Se utiliza UnstructuredPDFLoader para cargar el archivo .pdf con información de un contexto específico. Se puede responder a la pregunta de un usuario sobre ese dominio de aplicación utilizando el poder de las respuestas generativas con dicho LLM.

```
%pip install --q unstructured langchain
%pip install --q "unstructured[all-docs]"
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.0/2.0 MB 8.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 974.6/974.6 kB 16.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 431.4/431.4 kB 14.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 274.7/274.7 kB 14.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 981.5/981.5 kB 24.8 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.4/3.4 MB 16.4 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41.0/41.0 kB 1.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 321.8/321.8 kB 14.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.1/127.1 kB 6.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 145.0/145.0 kB 3.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.2/49.2 kB 3.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 80.8/80.8 kB 5.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.6/75.6 kB 4.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 290.4/290.4 kB 16.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.5/54.5 kB 3.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 77.9/77.9 kB 5.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 5.7 MB/s eta 0:00:00
  Building wheel for langdetect (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.5/7.5 MB 19.8 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 471.6/471.6 kB 34.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.4/56.4 kB 5.4 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 15.9/15.9 MB 50.4 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 244.3/244.3 kB 23.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 459.6/459.6 kB 33.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 112.5/112.5 kB 12.4 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.6/5.6 MB 62.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.4/2.4 MB 44.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19.2/19.2 MB 33.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.8/6.8 MB 43.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.3/2.3 MB 51.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 159.9/159.9 kB 17.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.5/79.5 kB 9.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.5/4.5 MB 23.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 114.6/114.6 kB 11.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 117.0/117.0 kB 13.4 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 46.0/46.0 kB 4.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.3/21.3 MB 59.3 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 42.2/42.2 kB 4.3 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 57.9/57.9 kB 3.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 86.8/86.8 kB 9.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.8/2.8 MB 74.2 MB/s eta 0:00:00
  Building wheel for antlr4-python3-runtime (setup.py) ... done
  Building wheel for iopath (setup.py) ... done
```

```
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts
imageio 2.31.6 requires pillow<10.1.0,>=8.3.2, but you have pillow 10.3.0 which is incompatible.
```

```
!pip install langchain-community
```

```
Collecting langchain-community
  Downloading langchain_community-0.2.5-py3-none-any.whl (2.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.2/2.2 MB 10.1 MB/s eta 0:00:00
Requirement already satisfied: PyYAML>=5.3 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (6.0.1)
Requirement already satisfied: SQLAlchemy<3,>=1.4 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (2.0.30)
Requirement already satisfied: aiohttp<4.0.0,>=3.8.3 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (3.9.5)
Requirement already satisfied: dataclasses-json<0.7,>=0.5.7 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.6.7)
Requirement already satisfied: langchain<0.3.0,>=0.2.5 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.2.5)
Requirement already satisfied: langchain-core<0.3.0,>=0.2.7 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.2.9)
Requirement already satisfied: langsmith<0.2.0,>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.1.81)
Requirement already satisfied: numpy<2,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (1.25.2)
Requirement already satisfied: requests<3,>=2 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (2.31.0)
Requirement already satisfied: tenacity<9.0.0,>=8.1.0 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (8.4.1)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (23.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (4.0.3)
Requirement already satisfied: marshmallow<4.0.0,>=3.18.0 in /usr/local/lib/python3.10/dist-packages (from dataclasses-json<0.7,>=0.5.7->langchain-community) (3.21.3)
Requirement already satisfied: typing-inspect<1,>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from dataclasses-json<0.7,>=0.5.7->langchain-community) (0.9.0)
Requirement already satisfied: langchain-text-splitters<0.3.0,>=0.2.0 in /usr/local/lib/python3.10/dist-packages (from langchain<0.3.0,>=0.2.5->langchain-community) (0.2.1)
Requirement already satisfied: pydantic<3,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain<0.3.0,>=0.2.5->langchain-community) (2.7.4)
Requirement already satisfied: jsonpatch<2.0,>=1.33 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.3.0,>=0.2.7->langchain-community) (1.33)
Requirement already satisfied: packaging<25,>=23.2 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.3.0,>=0.2.7->langchain-community) (24.1)
Requirement already satisfied: orjson<4.0.0,>=3.9.14 in /usr/local/lib/python3.10/dist-packages (from langsmith<0.2.0,>=0.1.0->langchain-community) (3.10.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (2024.6.2)
Requirement already satisfied: typing-extensions>=4.6.0 in /usr/local/lib/python3.10/dist-packages (from SQLAlchemy<3,>=1.4->langchain-community) (4.12.2)
Requirement already satisfied: greenlet!=0.4.17 in /usr/local/lib/python3.10/dist-packages (from SQLAlchemy<3,>=1.4->langchain-community) (3.0.3)
Requirement already satisfied: jsonpointer>=1.9 in /usr/local/lib/python3.10/dist-packages (from jsonpatch<2.0,>=1.33->langchain-core<0.3.0,>=0.2.7->langchain-community) (3.
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain<0.3.0,>=0.2.5->langchain-community) (0.7.0)
Requirement already satisfied: pydantic-core==2.18.4 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain<0.3.0,>=0.2.5->langchain-community) (2.18.4)
Requirement already satisfied: mypy-extensions>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from typing-inspect<1,>=0.4.0->dataclasses-json<0.7,>=0.5.7->langchain-comm
Installing collected packages: langchain-community
Successfully installed langchain-community-0.2.5
```

```
from langchain_community.document_loaders import UnstructuredPDFLoader
from langchain_community.document_loaders import OnlinePDFLoader
```

```
# Instalación de PyPDF2
!pip install PyPDF2

# Importar la biblioteca necesaria
import PyPDF2

# Ruta local del archivo PDF
local_path = "/content/00f6e6_ManualMEGAMIG500Spanish.pdf"

# Función para unir todas las páginas en un archivo .txt
def unir_paginas_en_txt(file_path):
    # Verificar que se haya proporcionado un archivo PDF
    if file_path:
        # Cargar el archivo PDF usando UnstructuredPDFLoader
        loader = UnstructuredPDFLoader(file_path=file_path)
        pdf_obj = loader.load()  # Suponiendo que esto devuelve un objeto que permite acceder a cada página

        # Crear un objeto PDF de PyPDF2 para manejar el archivo PDF
        pdf_reader = PyPDF2.PdfReader(open(file_path, "rb"))

        # Ruta de salida para el archivo .txt combinado
        output_file = "/content/combined_text.txt"

        # Abrir el archivo de texto para escribir
        with open(output_file, "w", encoding="utf-8") as txt_file:
            # Recorrer todas las páginas del PDF
            for page_num in range(len(pdf_reader.pages)): # Use len(pdf_reader.pages) instead of pdf_reader.numPages
                page = pdf_reader.pages[page_num] # Access pages using index
                page_text = page.extract_text()

                # Escribir el contenido de la página en el archivo de texto
                txt_file.write(f"Página {page_num + 1}\n\n")
                txt_file.write(page_text)
                txt_file.write("\n\n---\n\n")  # Separador entre páginas

        print(f"Se ha creado el archivo combinado: {output_file}")
    else:
        print("Sube un archivo PDF")

# Llamar a la función para unir todas las páginas en un archivo .txt
unir_paginas_en_txt(local_path)
```

```
Requirement already satisfied: PyPDF2 in /usr/local/lib/python3.10/dist-packages (3.0.1)
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
Se ha creado el archivo combinado: /content/combined_text.txt
```

```
!pip install llama-index llama-index-llms-mistralai llama-index-embeddings-mistralai
```

```
Requirement already satisfied: networkx>=3.0 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (3.3)
Requirement already satisfied: nltk<4.0.0,>=3.8.1 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (3.8.1)
Requirement already satisfied: numpy<2.0.0 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (1.25.2)
Collecting openai>=1.1.0 (from llama-index-core==0.10.48->llama-index)
  Downloading openai-1.35.3-py3-none-any.whl (327 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 327.4/327.4 kB 24.9 MB/s eta 0:00:00
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (2.0.3)
Requirement already satisfied: pillow>=9.0.0 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (10.3.0)
Requirement already satisfied: requests>=2.31.0 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (2.31.0)
Requirement already satisfied: tenacity!=8.4.0,<9.0.0,>=8.2.0 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (8.4.1)
Collecting tiktoken>=0.3.3 (from llama-index-core==0.10.48->llama-index)
  Downloading tiktoken-0.7.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.1/1.1 MB 46.5 MB/s eta 0:00:00
Requirement already satisfied: tqdm<5.0.0,>=4.66.1 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (4.66.4)
Requirement already satisfied: typing-extensions>=4.5.0 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (4.12.2)
Requirement already satisfied: typing-inspect>=0.8.0 in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (0.9.0)
Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from llama-index-core==0.10.48->llama-index) (1.14.1)
Collecting mistralai>=0.4.0 (from llama-index-llms-mistralai)
  Downloading mistralai-0.4.1-py3-none-any.whl (19 kB)
Requirement already satisfied: beautifulsoup4<5.0.0,>=4.12.3 in /usr/local/lib/python3.10/dist-packages (from llama-index-readers-file<0.2.0,>=0.1.4->llama-index) (4.12.3)
Requirement already satisfied: pypdf<5.0.0,>=4.0.1 in /usr/local/lib/python3.10/dist-packages (from llama-index-readers-file<0.2.0,>=0.1.4->llama-index) (4.2.0)
Collecting striprtf<0.0.27,>=0.0.26 (from llama-index-readers-file<0.2.0,>=0.1.4->llama-index)
  Downloading striprtf-0.0.26-py3-none-any.whl (6.9 kB)
Collecting llama-parse<0.5.0,>=0.4.0 (from llama-index-readers-llama-parse<0.2.0,>=0.1.2->llama-index)
  Downloading llama_parse-0.4.4-py3-none-any.whl (8.0 kB)
Requirement already satisfied: orjson<3.11,>=3.9.10 in /usr/local/lib/python3.10/dist-packages (from mistralai>=0.4.0->llama-index-llms-mistralai) (3.10.5)
Requirement already satisfied: pydantic<3,>=2.5.2 in /usr/local/lib/python3.10/dist-packages (from mistralai>=0.4.0->llama-index-llms-mistralai) (2.7.4)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.6->llama-index-core==0.10.48->llama-index) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.6->llama-index-core==0.10.48->llama-index) (23.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.6->llama-index-core==0.10.48->llama-index) (1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.6->llama-index-core==0.10.48->llama-index) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.6->llama-index-core==0.10.48->llama-index) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.6->llama-index-core==0.10.48->llama-index) (4.0
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4<5.0.0,>=4.12.3->llama-index-readers-file<0.2.0,>=0.1.4->llama-
Requirement already satisfied: anyio in /usr/local/lib/python3.10/dist-packages (from httpx->llama-index-core==0.10.48->llama-index) (3.7.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from httpx->llama-index-core==0.10.48->llama-index) (2024.6.2)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.10/dist-packages (from httpx->llama-index-core==0.10.48->llama-index) (1.0.5)
Requirement already satisfied: idna in /usr/local/lib/python3.10/dist-packages (from httpx->llama-index-core==0.10.48->llama-index) (3.7)
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-packages (from httpx->llama-index-core==0.10.48->llama-index) (1.3.1)
Requirement already satisfied: h11<0.15,>=0.13 in /usr/local/lib/python3.10/dist-packages (from httpcore==1.*->httpx->llama-index-core==0.10.48->llama-index) (0.14.0)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk<4.0.0,>=3.8.1->llama-index-core==0.10.48->llama-index) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk<4.0.0,>=3.8.1->llama-index-core==0.10.48->llama-index) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk<4.0.0,>=3.8.1->llama-index-core==0.10.48->llama-index) (2024.5.15)
Requirement already satisfied: distro<2,>=1.7.0 in /usr/lib/python3/dist-packages (from openai>=1.1.0->llama-index-core==0.10.48->llama-index) (1.7.0)
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=2.5.2->mistralai>=0.4.0->llama-index-llms-mistralai) (0
Requirement already satisfied: pydantic-core==2.18.4 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=2.5.2->mistralai>=0.4.0->llama-index-llms-mistralai) (2.
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests>=2.31.0->llama-index-core==0.10.48->llama-index) (3.3.2)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.31.0->llama-index-core==0.10.48->llama-index) (2.0.7)
Requirement already satisfied: greenlet!=0.4.17 in /usr/local/lib/python3.10/dist-packages (from SQLAlchemy[asyncio]>=1.4.49->llama-index-core==0.10.48->llama-index) (3.0.
Requirement already satisfied: mypy-extensions>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from typing-inspect>=0.8.0->llama-index-core==0.10.48->llama-index) (1.0.
Requirement already satisfied: marshmallow<4.0.0,>=3.18.0 in /usr/local/lib/python3.10/dist-packages (from dataclasses-json->llama-index-core==0.10.48->llama-index) (3.21.
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas->llama-index-core==0.10.48->llama-index) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->llama-index-core==0.10.48->llama-index) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas->llama-index-core==0.10.48->llama-index) (2024.1)
```

```
from mistralai.client import MistralClient
from mistralai.models.chat_completion import ChatMessage
from getpass import getpass
```

```
api_key= getpass("Type your API Key")
client = MistralClient(api_key=api_key)
```

```
Type your API Key··········
```

```
# Función para ejecutar el modelo de Mistral
def run_mistral(user_message, model="mistral-large-latest"):
    messages = [
        {"role": "user", "content": user_message}
    ]
    try:
        chat_response = client.chat(
            model=model,
            messages=messages
        )
        return chat_response.choices[0].message.content
    except mistral_ai.MistralAPIException as e:
        return f"Error: {e}"
```

```
import os
from llama_index.core import Settings, SimpleDirectoryReader, VectorStoreIndex
from llama_index.llms.mistralai import MistralAI
from llama_index.embeddings.mistralai import MistralAIEmbedding

# Carga la información
reader = SimpleDirectoryReader(input_files=["/content/combined_text.txt"])
documents = reader.load_data()
# Define LLM y el modelo de embedding
Settings.llm = MistralAI(model="mistral-large-latest", api_key=api_key)
Settings.embed_model = MistralAIEmbedding(model_name='mistral-embed', api_key=api_key)
# Create vector store index
index = VectorStoreIndex.from_documents(documents)
# Crea el motor de la query
query_engine = index.as_query_engine(similarity_top_k=2)
response = query_engine.query(
    "¿Qué se debe hacer antes de soldar?"
)
print(str(response))
```

```
Antes de comenzar a soldar, se deben verificar varios aspectos para asegurar una operación segura y efectiva. Primero, asegúrese de que la conexión de tierra de la toma eléc
```

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
!pip install rouge-score
from rouge_score import rouge_scorer
```

```
Collecting rouge-score
  Downloading rouge_score-0.1.2.tar.gz (17 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: absl-py in /usr/local/lib/python3.10/dist-packages (from rouge-score) (1.4.0)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (from rouge-score) (3.8.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from rouge-score) (1.25.2)
Requirement already satisfied: six>=1.14.0 in /usr/local/lib/python3.10/dist-packages (from rouge-score) (1.16.0)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (2024.5.15)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (4.66.4)
Building wheels for collected packages: rouge-score
  Building wheel for rouge-score (setup.py) ... done
  Created wheel for rouge-score: filename=rouge_score-0.1.2-py3-none-any.whl size=24933 sha256=261e362304a5e6da1c48707787f2a29b24966092fd2dd4a6cb5ae83d5e4005d8
  Stored in directory: /root/.cache/pip/wheels/5f/dd/89/461065a73be61a532ff8599a28e9beef17985c9e9c31e541b4
Successfully built rouge-score
Installing collected packages: rouge-score
Successfully installed rouge-score-0.1.2
```

```
!pip install bert-score
from bert_score import score
```

```
Collecting bert-score
  Downloading bert_score-0.3.13-py3-none-any.whl (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.1/61.1 kB 2.2 MB/s eta 0:00:00
Requirement already satisfied: torch>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from bert-score) (2.3.0+cu121)
Requirement already satisfied: pandas>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from bert-score) (2.0.3)
Requirement already satisfied: transformers>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from bert-score) (4.41.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from bert-score) (1.25.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from bert-score) (2.31.0)
Requirement already satisfied: tqdm>=4.31.1 in /usr/local/lib/python3.10/dist-packages (from bert-score) (4.66.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from bert-score) (3.7.1)
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.10/dist-packages (from bert-score) (24.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.1->bert-score) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.1->bert-score) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.1->bert-score) (2024.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (3.15.1)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (1.12.1)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (2023.6.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (11.0.2.54)
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.20.5 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (2.20.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: triton==2.3.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (2.3.0)
Requirement already satisfied: nvidia-nvjitlink-cu12 in /usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-cu12==11.4.5.107->torch>=1.0.0->bert-score) (12.5.40)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (0.23.4)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (2024.5.15)
Requirement already satisfied: tokenizers<0.20,>=0.19 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (0.19.1)
```

```
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (0.4.3)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (4.53.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (1.4.5)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (3.1.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (2024.6.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas>=1.0.1->bert-score) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.0.0->bert-score) (2.1.5)
Requirement already satisfied: mpmath<1.4.0,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.0.0->bert-score) (1.3.0)
Installing collected packages: bert-score
Successfully installed bert-score-0.3.13
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """Antes de soldar verifique el estado de las conexiones.
Revise que la conexión de tierra del enchufe esté correcta.
Utilice ropa y herramientas apropiadas para evitar dañar la vista y la piel.
Cuando se este soldando se debe usar la máscara de soldar cubriendo toda la cabeza, sólo se puede realizar la observación visual del arco eléctrico a través del visor de la másc
Evite la sobrecarga de su equipo revisando previamente el ciclo de trabajo de este. Tome la precaución de que la manguera de gas nunca se encuentre presionada o doblada. Siempre
Asegúrese de que no exista riesgo potencial, tanto para el operador como para la máquina, de caída de cualquier objeto extraño.
El polvo, ácido o material corrosivo en el ambiente del lugar de trabajo y medio ambiente corrosivo no deben sobrepasar los parámetros exigidos (exceptuándose los provocados a c
El equipo de soldar debe instalarse protegido del sol, la lluvia, humedad excesiva o temperaturas por bajo los -10° C o por sobre los 40° C.
Deben existir al menos 50 cms. libres alrededor del equipo de soldar para asegurar una adecuada ventilación.
No debe introducir piezas extrañas al equipo de soldar.
No deben existir vibraciones excesivas en el área alrededor del lugar de trabajo de la máquina.
Elija un área sin interferencias electromagnéticas.
Asegúrese de instalar un interruptor automático de protección en la instalación eléctrica de acuerdo a las especificaciones indicadas por el manual de servicio del equipo.
El equipo de soldar debe instalarse horizontalmente, en una superficie plana.
Si la inclinación fuera superior a 15° se le debe adicionar elementos anti vuelco para evitar la inclinación del equipo.
No se recomienda conectar su equipo de soldar a un grupo generador.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart yo
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
tokenizer_config.json: 100%  49.0/49.0 [00:00<00:00, 2.58kB/s]
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:1132: FutureWarning: `resume_download` is deprecated and will be removed in version 1.0.0. Downloads
  warnings.warn(
config.json: 100%  625/625 [00:00<00:00, 32.6kB/s]
vocab.txt: 100%  996k/996k [00:00<00:00, 6.36MB/s]
tokenizer.json: 100%  1.96M/1.96M [00:00<00:00, 12.4MB/s]
model.safetensors: 100%  714M/714M [00:07<00:00, 137MB/s]
calculating scores...
computing bert embedding.
100%  1/1 [00:03<00:00, 3.29s/it]
computing greedy matching.
100%  1/1 [00:00<00:00, 22.64it/s]
done in 3.34 seconds, 0.30 sentences/sec
ROUGE-1 score: Score(precision=0.6129032258064516, recall=0.2932098765432099, fmeasure=0.3966597077244259)
ROUGE-2 score: Score(precision=0.21428571428571427, recall=0.1021671826625387, fmeasure=0.13836477987421386)
ROUGE-L score: Score(precision=0.32903225806451614, recall=0.1574074074074074, fmeasure=0.2129436325678497)
BERTScore - Precision: 0.7237033843994141, Recall: 0.6898183822631836, F1: 0.7063547372817993
```

```
response = query_engine.query(
    "¿Qué ocurre si sucede Shock Eléctrico?"
)
print(str(response))
```

```
Si una persona sufre un shock eléctrico y queda inconsciente, es importante no tocarla si todavía está en contacto con algún cuerpo posiblemente energizado. Lo primero que s
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """Si la persona accidentada esta inconsciente y se sospecha un shock eléctrico, tenga la precaución de
no tocarla si ha quedado en contacto con algún cuerpo posiblemente energizado.
Corte el suministro eléctrico que alimenta el equipo y recurra a los cuidados de primeros auxilios.
Para alejar los cables y/o partes energizadas de la víctima, se recomienda utilizar trozos de madera
bien seca, como una escoba o cualquier otro material aislante.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1 [00:00<00:00,  1.61it/s]
computing greedy matching.
100%                                                  1/1 [00:00<00:00, 31.43it/s]
done in 0.66 seconds, 1.52 sentences/sec
ROUGE-1 score: Score(precision=0.7875, recall=0.8289473684210527, fmeasure=0.8076923076923076)
ROUGE-2 score: Score(precision=0.6708860759493671, recall=0.7066666666666667, fmeasure=0.6883116883116883)
ROUGE-L score: Score(precision=0.75, recall=0.7894736842105263, fmeasure=0.7692307692307692)
BERTScore - Precision: 0.9254627227783203, Recall: 0.9266050457954407, F1: 0.9260335564613342
```

```
response = query_engine.query(
    "¿Cuáles son los requerimientos para la fuente de poder?"
)
print(str(response))
```

```
sobre los 40° C. También deben existir al menos 50 cms. libres alrededor del equipo para asegurar una adecuada ventilación. Si la ventilación del área de trabajo no es adecu
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """El oscilograma de voltaje debe mostrar una onda de seno, la oscilación de la frecuencia no
debe exceder ± 1% del valor indicado. La oscilación del voltaje no debe superar ± 10 % del valor indicado.
El desbalance de la alimentación trifásica no debe exceder el 5%.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1 [00:01<00:00,  1.74s/it]
computing greedy matching.
100%                                                  1/1 [00:00<00:00, 25.76it/s]
done in 1.78 seconds, 0.56 sentences/sec
ROUGE-1 score: Score(precision=0.15527950310559005, recall=0.5208333333333334, fmeasure=0.23923444976076555)
ROUGE-2 score: Score(precision=0.01875, recall=0.06382978723404255, fmeasure=0.02898550724637681)
ROUGE-L score: Score(precision=0.09316770186335403, recall=0.3125, fmeasure=0.14354066985645933)
BERTScore - Precision: 0.623805820941925, Recall: 0.6832370162010193, F1: 0.6521702408790588
```

```
response = query_engine.query(
    "¿Cuáles son las principales tareas de mantenimiento?"
)
print(str(response))
```

```
Las principales tareas de mantenimiento para el equipo de soldar Mega Mig 500 incluyen:

1. Remoción de Polvo: Asegúrate de que la máquina no esté conectada a una fuente de poder antes de proceder a la remoción de polvo.

2. Almacenamiento seguro: Cuando dejes tu equipo por un largo periodo de inactividad, hazlo en un ambiente seco, limpio y con buena ventilación.

3. Limpieza periódica: A través de personal especializado, se debe realizar limpieza por medio de aire comprimido seco para limpiar el interior del equipo de soldar. También

4. Mantenimiento del cable de poder: Es necesario chequear periódicamente el cable de alimentación del equipo de soldar.

5. Revisión de Cables de soldar: Al momento de utilizar el equipo de soldar, el operador debe revisar que los cables de soldar no estén desconectados y no presenten cortes e

6. Revisión del circuito de gas: Revise regularmente que el circuito de gas no presente fugas y se mantenga completamente sellado.

7. Limpieza de la boquilla: Limpie periódicamente la boquilla de escoria y suciedad.

8. Utilización de alambre Mig de calidad: Utilice alambre Mig de calidad, que cuente con certificaciones.

9. Mantenimiento de los rodillos: Reemplace los rodillos si se encuentran gastados o deteriorados. Apriete los rodillos contra el alambre sin sobrepresionar para asegurar un
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """Remoción de Polvo .Debe asegurarse que la máquina no esté conectada a una fuente de
poder antes de proceder a la remoción de polvo.
Asegurece que al dejar su equipo por un largo periodo de inactividad, lo haga en un ambiente seco, limpio y con buena ventilación.
Limpieza periódica. A través de personal especializado, se debe realizar limpieza por medio de aire comprimido seco para limpiar el interior del equipo de soldar. También se
debe revisar si existen componentes o cables sueltos, si los hubiera, proceda a la reparación o reemplazo correspondiente. Generalmente cuando no existe polvo en demasía la limp
se debe realizar periódicamente, y cuando existe mucha acumulación de polvo la limpieza se debe realizar con mayor frecuencia.Mantenga el cable de poder en buen estado Es necesa
de alimentación del equipo de soldar, sin embargo, es aconsejable revisar cada vez que la máquina sea utilizada de forma portátil.
Revise Cables de soldar. Al momento de utilizar el equipo de soldar, el operador debe revisar que los cables de soldar no estén desconectados, no presenten cortes en el
aislamiento, etc. En caso de que se presente alguno de estos problemas se debe proceder al cambio o reparación correspondiente.
Revise regularmente que el circuito de gas no presente fugas y se mantenga completamente sellado. Verifique ademas que el ventilador y el motor alimentador estén funcionando
correctamente sin sonidos anormales ni cables sueltos. Limpie periódicamente la boquilla de escoria y suciedad.
Utilice alambre Mig de calidad, que cuente con certificaciones. Reemplace los rodillos si se encuentran gastados o deteriorados. Apriete los rodillos contra
el alambre sin sobrepresionar para asegurar un buen desplazamiento de éste."""  # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1 [00:02<00:00,  2.09s/it]
computing greedy matching.
100%                                                  1/1 [00:00<00:00, 23.07it/s]
done in 2.14 seconds, 0.47 sentences/sec
ROUGE-1 score: Score(precision=0.8174603174603174, recall=0.6936026936026936, fmeasure=0.7504553734061931)
ROUGE-2 score: Score(precision=0.7131474103585658, recall=0.6047297297297297, fmeasure=0.6544789762340038)
ROUGE-L score: Score(precision=0.7658730158730159, recall=0.6498316498316499, fmeasure=0.7030965391621129)
BERTScore - Precision: 0.8059012293815613, Recall: 0.800682544708252, F1: 0.8032833933830261
```

```
response = query_engine.query(
    "¿Antes de armar o desarmar la pistola que se debe hacer?"
)
print(str(response))
```

```
Antes de armar o desarmar la pistola, es importante desconectarla de la energía eléctrica. Esto garantiza la seguridad durante el proceso.
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = "Antes de armar o desarmar la pistola se la debe desconectar de la energía eléctrica." # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1 [00:00<00:00,  1.91it/s]
computing greedy matching.
100%                                                  1/1 [00:00<00:00, 18.77it/s]
done in 0.58 seconds, 1.71 sentences/sec
ROUGE-1 score: Score(precision=0.6086956521739131, recall=0.8235294117647058, fmeasure=0.7)
ROUGE-2 score: Score(precision=0.5, recall=0.6875, fmeasure=0.5789473684210527)
ROUGE-L score: Score(precision=0.5652173913043478, recall=0.7647058823529411, fmeasure=0.65)
BERTScore - Precision: 0.8530756831169128, Recall: 0.9336416125297546, F1: 0.8915421962738037
```

```
response = query_engine.query(
    "¿Cuándo pueden resultar peligrosas las operaciones de soldadura?"
)
print(str(response))
```

```
arga el equipo sin revisar previamente su ciclo de trabajo, se puede acortar la vida útil de los componentes y del equipo en general. Finalmente, si el cable de soldar se enc
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response ="""Si las normas de seguridad y de utilización no se observan atentamente, las operaciones de
soldadura pueden resultar peligrosas no solo para el operador, sino incluso para las personas que
se encuentren próximas al área de trabajo.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                1/1 [00:01<00:00,  1.48s/it]
computing greedy matching.
100%                                                1/1 [00:00<00:00, 27.08it/s]
done in 1.54 seconds, 0.65 sentences/sec
ROUGE-1 score: Score(precision=0.17142857142857143, recall=0.6153846153846154, fmeasure=0.2681564245810056)
ROUGE-2 score: Score(precision=0.050359712230215826, recall=0.18421052631578946, fmeasure=0.0790960451977401)
ROUGE-L score: Score(precision=0.08571428571428572, recall=0.3076923076923077, fmeasure=0.1340782122905028)
BERTScore - Precision: 0.6921112537384033, Recall: 0.7591825127601624, F1: 0.7240971326828003
```

```
response = query_engine.query(
    "¿Cómo influye la velocidad del alambre?"
)
print(str(response))
```

```
La velocidad del alambre influye directamente en la corriente de soldadura. A un mismo voltaje, cuanto mayor sea la velocidad del alambre, mayor será la corriente de soldadu
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response ="""La velocidad del alambre influye directamente en la corriente de soldadura. A un
mismo voltaje, a mayor velocidad del alambre mayor es la corriente de soldadura.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
```

```
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                1/1 [00:01<00:00,  1.19s/it]
computing greedy matching.
100%                                                1/1 [00:00<00:00, 17.50it/s]
done in 1.27 seconds, 0.79 sentences/sec
ROUGE-1 score: Score(precision=0.8571428571428571, recall=0.9230769230769231, fmeasure=0.888888888888889)
ROUGE-2 score: Score(precision=0.7407407407407407, recall=0.8, fmeasure=0.7692307692307692)
ROUGE-L score: Score(precision=0.8571428571428571, recall=0.9230769230769231, fmeasure=0.888888888888889)
BERTScore - Precision: 0.9501052498817444, Recall: 0.9623439908027649, F1: 0.9561854600906372
```

```
response = query_engine.query(
    "¿Cómo se debe conectar el equipo de soldar y el sistema de gas?"
)
print(str(response))
```

```
or a 150 mm para evitar daños en su interior. La pureza del CO2 utilizado no debe ser menor al 99,5% y su contenido de agua no debe superar el 0,005%. Siempre revisa que el c
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response ="""Asegúrese que el regulador de gas este firmemente atornillado a su conexión para evitar fugas de
gas. Luego conecte y fije firmemente el cable de calefacción a la conexión del calefactor del panel trasero
de la fuente de poder.
Conecte un extremo de la manguera de gas a la salida del regulador de presión y el otro extremo a la
entrada de gas del alimentador de alambre. """ # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
```

100% 1/1 [00:01<00:00, 1.01s/it]

```
computing greedy matching.
```

100% 1/1 [00:00<00:00, 35.67it/s]

```
done in 1.05 seconds, 0.96 sentences/sec
ROUGE-1 score: Score(precision=0.3037037037037037, recall=0.5694444444444444, fmeasure=0.3961352657004831)
ROUGE-2 score: Score(precision=0.1044776119402985, recall=0.19718309859154928, fmeasure=0.13658536585365855)
ROUGE-L score: Score(precision=0.2, recall=0.375, fmeasure=0.26086956521739135)
BERTScore - Precision: 0.711998701095581, Recall: 0.7835975289344788, F1: 0.7460842728614807
```

```
response = query_engine.query(
    "¿Cómo se debe realizar la instalacion y conexión del alimentador de alambre SUPER TRACK?"
)
print(str(response))
```

```
:orrespondiente del panel frontal y aprieta firmemente. Conecta un extremo del cable de tierra al conector correspondiente en el panel frontal de la fuente de poder y atornil
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response ="""Seleccione el diámetro del alambre según el trabajo a ejecutar. En función de este diámetro elija el
carrete, flexible y boquilla adecuados.
Abra la cubierta del alimentador de alambre SUPER TRACK e instale en el eje del carrete.
Introduzca el alambre en el tubo de entrada del mecanismo de arrastre y alínielo en la ranura del
rodillo de empuje adecuado. Vuelva a centrarlo en la boquilla de salidad y active el mecanismo de
presión. """ # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
```

```
response = query_engine.query(
```