Install Required Libraries:

```
# @title Install Required Libraries:
!pip install transformers scikit-learn PyMuPDF
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.46.3)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.5.2)
Requirement already satisfied: PyMuPDF in /usr/local/lib/python3.10/dist-packages (1.25.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.16.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.26.5)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2024.9.11)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: tokenizers<0.21,>=0.20 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.20.3)
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.4.5)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.6)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.23.2->trans
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.2
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2024.8.3
```

Extract Text from PDFs:

```
# @title Extract Text from PDFs:
import fitz  # PyMuPDF

def extract_text_from_pdf(pdf_path):
    doc = fitz.open(pdf_path)
    text = ""
    for page_num in range(len(doc)):
        page = doc.load_page(page_num)
        text += page.get_text()
    return text

# Example PDF paths
pdf_paths = ["butterflies.pdf", "docker.pdf", "flowers.pdf","rockmusic.pdf"]

# Extract text from each PDF
texts = [extract_text_from_pdf(pdf_path) for pdf_path in pdf_paths]
```

Generate Embeddings:

```
# @title Generate Embeddings:
from transformers import AutoTokenizer, AutoModel
import torch

# Load pre-trained model and tokenizer
model_name = 'sentence-transformers/all-MiniLM-L6-v2'
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModel.from_pretrained(model_name)

# Function to generate embeddings
def get_embeddings(texts):
    inputs = tokenizer(texts, padding=True, truncation=True, return_tensors='pt')
    with torch.no_grad():
        outputs = model(**inputs)
    return outputs.last_hidden_state.mean(dim=1).numpy()

# Generate embeddings for the extracted texts
embeddings = get_embeddings(texts)
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_auth.py:94: 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 s
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(
```

Cluster Embeddings:

```
# @title Cluster Embeddings:
from sklearn.cluster import KMeans

# Number of clusters
num_clusters = 3

# Apply KMeans clustering
kmeans = KMeans(n_clusters=num_clusters, random_state=0)
clusters = kmeans.fit_predict(embeddings)

# Print cluster assignments
for pdf_path, cluster in zip(pdf_paths, clusters):
    print(f"PDF: {pdf_path} -> Cluster: {cluster}")
```

```
PDF: butterflies.pdf -> Cluster: 0
PDF: docker.pdf -> Cluster: 2
PDF: flowers.pdf -> Cluster: 0
PDF: rockmusic.pdf -> Cluster: 1
```

Prepare Data for Visualization: Save the clustering results to a JSON file that can be used by the D3.js script.

```
# @title Prepare Data for Visualization: Save the clustering results to a JSON file that can be used by the D3.js script.
import json

# Prepare data for visualization
data = [{"pdf": pdf_path, "cluster": int(cluster)} for pdf_path, cluster in zip(pdf_paths, clusters)]

# Save data to a JSON file
with open("clustering_results.json", "w") as f:
    json.dump(data, f)
```

```
import numpy as np
import plotly.express as px
import pandas as pd

# Prepare data for visualization
data = [{"pdf": pdf_path, "cluster": int(cluster)} for pdf_path, cluster in zip(pdf_paths, clusters)]

# Generate random coordinates for demonstration purposes
for d in data:
    d['x'] = np.random.rand()
    d['y'] = np.random.rand()
    d['z'] = np.random.rand()

# Convert data to a DataFrame
df = pd.DataFrame(data)

# Create a 3D scatter plot
fig = px.scatter_3d(df, x='x', y='y', z='z', color='cluster', hover_data=['pdf'], title='PDF Clustering Visualization')

# Add data labels
fig.update_traces(marker=dict(size=5),
                  selector=dict(mode='markers'))

# Show the plot
fig.show()
```

PDF Clustering Visualization