בוא נגיד בשלב זה נניח ואנחנו רוצים להתקין ראג חדש.

mkdir pdfs

mv \*.pdf pdfs/

או להכניס לפנים את כל קבצי הPDF.

יש לנו את הקובץ MAIN

RAG\_PDF

תקייה VENV

REQUIERMENTS.

ובוא נגיד 100 קבצי PDF לא רק 1 אוקי?

mkdir ~/שם התקייה שנרצה..

cd ~/לגשת אלייה..

python3 -m venv venv לפתוח סביבה..

source venv/bin/activate להפעיל סביבה..

pip install -r requirements.txt להתקין דרישות..

הדרישות:

langchain

langchain-ollama

langchain-chroma

langchain-community

pypdf

pymupdf

אולמה צריכה לפעול ברקע עם מודל שמתקשר אני עשיתי עם מודל שנקרא

aya

כי הוא יודע עברית ממש טוב.

ollama pull aya:8b

מאולמה צריך למשוך גם את המודל שעושה אימבדינג:

ollama pull mxbai-embed-large

עכשיו יוצרים rag\_pdf.py

זה התוכן:

# rag\_pdf.py — תומך בתיקיית PDF מלאה

from langchain\_community.document\_loaders import PyPDFDirectoryLoader

from langchain\_community.embeddings import OllamaEmbeddings

from langchain\_chroma import Chroma

import os

# מודל אימבדינג מאולמה

embeddings = OllamaEmbeddings(model="mxbai-embed-large")

# מיקום בסיס הנתונים

db\_location = "./chrome\_langchain\_db"

add\_documents = not os.path.exists(db\_location)

# טען את כל ה־PDF מתיקייה בשם pdfs

loader = PyPDFDirectoryLoader("pdfs")

documents = loader.load()

# יצירת וקטור סטור

vector\_store = Chroma(

collection\_name="hebrew\_documents",

persist\_directory=db\_location,

embedding\_function=embeddings

)

# הוספת מסמכים

if add\_documents:

vector\_store.add\_documents(documents)

print("✅ Documents added to vector store.")

else:

print("ℹ️ Vector store already exists. Skipping addition.")

ואז מריצים אותו בעזרת:

python rag\_pdf.py

זה יוצר את הוקטורים ואימבדים על בסיס המודל שהורדנו מאוללמה.

עכשיו יוצרים את: main.py

# main.py

from langchain\_ollama.llms import OllamaLLM

from langchain\_core.prompts import ChatPromptTemplate

from langchain\_chroma import Chroma

from langchain\_community.embeddings import OllamaEmbeddings

# מודל שיחה

model = OllamaLLM(model="aya")

# Prompt Template

template = """

המסמכים הרלוונטיים הם:

{reviews}

הנה השאלה:

{question}

ענה בצורה ברורה, בעברית, על בסיס המסמכים בלבד.

"""

prompt = ChatPromptTemplate.from\_template(template)

chain = prompt | model

# Vector Store

embeddings = OllamaEmbeddings(model="mxbai-embed-large")

db\_location = "./chrome\_langchain\_db"

vector\_store = Chroma(

collection\_name="hebrew\_documents",

persist\_directory=db\_location,

embedding\_function=embeddings

)

# רטריבר

retriever = vector\_store.as\_retriever()

# לולאת שאלות

while True:

print("\n" + "-"\*40)

question = input("✍️ שאל שאלה (או 'q' כדי לצאת): ")

if question.strip().lower() == "q":

break

reviews = retriever.invoke(question)

result = chain.invoke({"reviews": reviews, "question": question})

print("\n🤖 תשובה:", result)

מריצים אותו:

python main.py

זהו.

אז ברמת העיקרון- כל פעם שאני ארצה להוסיף עוד מסמכים או לשנות מסמכים- אני אטרך למחוק את התקייה chrome\_langchain\_db ואז להריץ מחדש את rag\_pdf

rm -rf chrome\_langchain\_db

הרצה מחדש:  
  
python rag\_pdf.py

לוודא ש[MAIN.PY](http://main.py) נראה ככה::  
  
from langchain\_ollama.llms import OllamaLLM

from langchain\_core.prompts import ChatPromptTemplate

from langchain\_community.embeddings import OllamaEmbeddings

from langchain\_chroma import Chroma

# יצירת מודל שיחה

model = OllamaLLM(model="aya")

# יצירת פונקציית אימבדינג תואמת

embeddings = OllamaEmbeddings(model="mxbai-embed-large")

# טען את המאגר

db\_location = "./chrome\_langchain\_db"

vector\_store = Chroma(

collection\_name="hebrew\_documents",

persist\_directory=db\_location,

embedding\_function=embeddings # ← זה היה חסר!

)

# רטריבר

retriever = vector\_store.as\_retriever()

# תבנית Prompt

template = """

המסמכים הרלוונטיים הם:

{reviews}

השאלה היא:

{question}

ענה תשובה ברורה ומבוססת רק על מה שיש במסמכים.

"""

prompt = ChatPromptTemplate.from\_template(template)

chain = prompt | model

# שיחה

while True:

print("\n" + "-"\*40)

question = input("✍️ שאל שאלה (או 'q' כדי לצאת): ")

if question.strip().lower() == "q":

break

reviews = retriever.invoke(question)

result = chain.invoke({"reviews": reviews, "question": question})

print("\n🤖 תשובה:", result)