```
## Retrieval augmented generation (Minimalistic Code)

# Put the following in requirements.txt and install or install from cli
# llama-index
# openai
# pypdf
# python-dotenv

import os
from dotenv import load_dotenv

import os.path
from llama_index.core.response.pprint_utils import pprint_response

from llama_index.core.retrievers import VectorIndexRetriever
from llama_index.core.query_engine import RetrieverQueryEngine
from llama_index.core.indices.postprocessor import SimilarityPostprocessor

from llama_index.core import (
    VectorStoreIndex,
    SimpleDirectoryReader,
    StorageContext,
    load_index_from_storage,
)

load_dotenv()

# os.environ['OPENAI_API_KEY']=os.getenv("OPENAI_API_KEY")
os.environ['OPENAI_API_KEY']="OPENAI_API_KEY" # Substitute your OpenAI key here or put it in a .env file as above

documents=SimpleDirectoryReader("data").load_data()
index=VectorStoreIndex.from_documents(documents,show_progress=True)
query_engine=index.as_query_engine()
retriever=VectorIndexRetriever(index=index,similarity_top_k=4)    # Give the top 4 results only
postprocessor=SimilarityPostprocessor(similarity_cutoff=0.80)     # Only return results with a similarity score of 0.80 or higher
query_engine=RetrieverQueryEngine(retriever=retriever,
                                  node_postprocessors=[postprocessor])

response=query_engine.query("What is attention is all yopu need?")
print(response)

pprint_response(response,show_source=True)
print(response)

#####################################################################################
# Below code is only if you need to save the index for later use in a file system storage.
# You can use vector DB like MongoDB Atlas, FAISS, Pinecone, ChromaDB, etc.
PERSIST_DIR = "./storage"
if not os.path.exists(PERSIST_DIR):
    # load the documents and create the index
    documents = SimpleDirectoryReader("data").load_data()
    index = VectorStoreIndex.from_documents(documents)
    # store it for later
    index.storage_context.persist(persist_dir=PERSIST_DIR)
else:
    # load the existing index
    storage_context = StorageContext.from_defaults(persist_dir=PERSIST_DIR)
    index = load_index_from_storage(storage_context)

# either way we can now query the index
query_engine = index.as_query_engine()
response = query_engine.query("What are transformers?")
print(response)
```