**Streamlit Wikipedia Question Answering Application**

**Overview**

This application allows users to query Wikipedia content interactively. It utilizes Streamlit for the user interface, LangChain for document loading and processing, and OpenAI's GPT-3.5-turbo for natural language understanding and generation.

**Features**

* **Load Wikipedia Content**: Users can specify a Wikipedia topic, and the app will fetch and embed the relevant content.
* **Interactive Q&A**: Users can ask questions about the loaded Wikipedia content, and the app will provide answers based on the embedded content.
* **Session State Management**: The app maintains a history of questions and answers within the session for reference.

**Installation**

To run this application, you need to have Python installed along with the necessary libraries. You can install the required libraries using the following command:

pip install streamlit langchain chroma-client openai python-dotenv

**Environment Variables**

Create a .env file in the root directory of your project and add your OpenAI API key:

API\_KEY=your\_openai\_api\_key

**Code Explanation**

**1. Importing Necessary Libraries**

The application imports several libraries necessary for its functionality:

import streamlit as st

from langchain.embeddings.openai import OpenAIEmbeddings

from langchain.vectorstores import Chroma

from dotenv import load\_dotenv

import os

**2. Loading Environment Variables**

Load the API key from the .env file:

load\_dotenv()

api\_key = os.getenv('API\_KEY')

**3. Loading Wikipedia Content**

This function fetches Wikipedia content based on the user's query:

def load\_wikipedia(query, lang='en', load\_max\_docs=10):

from langchain.document\_loaders import WikipediaLoader

loader = WikipediaLoader(query=query, lang=lang, load\_max\_docs=load\_max\_docs)

data = loader.load()

return data

**4. Splitting Text into Chunks**

To process large documents, the text is split into smaller chunks:

def split\_text\_into\_chunks(data, chunk\_size=256, chunk\_overlap=20):

from langchain.text\_splitter import RecursiveCharacterTextSplitter

text\_splitter = RecursiveCharacterTextSplitter(chunk\_size=chunk\_size, chunk\_overlap=chunk\_overlap)

chunks = text\_splitter.split\_documents(data)

return chunks

**5. Creating Embeddings**

Generate embeddings from the text chunks and store them in a vector store:

def create\_embeddings(chunks):

embeddings = OpenAIEmbeddings()

vector\_store = Chroma.from\_documents(chunks, embeddings)

return vector\_store

**6. Chat with Wikipedia**

This function interacts with the loaded Wikipedia content using OpenAI's model:

def chat\_with\_wikipedia(vector\_store, query, k=3):

from langchain.chains import RetrievalQA

from langchain\_openai import ChatOpenAI

llm = ChatOpenAI(model='gpt-3.5-turbo', temperature=1)

retriever = vector\_store.as\_retriever(search\_type='similarity', search\_kwargs={'k': k})

chain = RetrievalQA.from\_chain\_type(llm=llm, chain\_type="stuff", retriever=retriever)

answer = chain.invoke(query)

return answer

**7. Clear Chat History**

Reset the chat history in the session state:

def clear\_history():

if 'history' in st.session\_state:

del st.session\_state['history']

**8. Streamlit Application Entry Point**

The main function to run the Streamlit application:

if \_\_name\_\_ == '\_\_main\_\_':

import os

from dotenv import load\_dotenv, find\_dotenv

load\_dotenv(find\_dotenv(), override=True)

st.subheader('Question Wikipedia content')

with st.sidebar:

subject = st.text\_input('Please pick your topic from wikipedia')

chunk\_size = 512

k = 3

add\_data = st.button('Load Data', on\_click=clear\_history)

if subject and add\_data and api\_key:

with st.spinner('Loading, Chunking and embedding file ...'):

os.environ['OPENAI\_API\_KEY'] = api\_key

data = load\_wikipedia(subject, lang='en')

chunks = split\_text\_into\_chunks(data)

st.write(f'Chunk size: {chunk\_size}, chunks: {len(chunks)}')

vector\_store = create\_embeddings(chunks)

st.session\_state.vs = vector\_store

st.success('Wikipedia information loaded, chunked, and embedded successfully!')

else:

pass

q = st.text\_input('Ask a question about the content of your file:')

if q:

if 'vs' in st.session\_state:

vector\_store = st.session\_state.vs

with st.spinner('Getting the information ...'):

answer = chat\_with\_wikipedia(vector\_store, q, k)

st.text\_area('LLM Answer: ', value=answer['result'])

st.divider()

if 'history' not in st.session\_state:

st.session\_state.history = ''

value = f'Q: {q} \nA:{answer}'

st.session\_state.history = f'{value} \n {"-" \* 100} \n {st.session\_state.history}'

h = st.session\_state.history

st.text\_area('Chat history:', value=h, key='history', height=400)

**Running the Application**

To run the application, execute the following command in your terminal:

streamlit run app.py

Replace app.py with the filename of your script. The application will open in your default web browser.

**Usage**

1. Enter a topic in the sidebar and click "Load Data".
2. Ask questions about the loaded Wikipedia content in the text input field.
3. View the answer and chat history displayed on the main page.

This documentation provides an overview and detailed explanation of the application's functionality and code structure, ensuring you can understand and modify it as needed.