```
import os
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import DeepLake
from langchain.text_splitter import CharacterTextSplitter
from langchain import OpenAI
from langchain.document_loaders import SeleniumURLLoader
from langchain import PromptTemplate
import streamlit as st
from watsonxlangchain import LangChainInterface
from langchain.indexes import VectorstoreIndexCreator
from langchain.chains import RetrievalQA


os.environ['OPENAI_API_KEY'] = ''
os.environ["ACTIVELOOP_TOKEN"] = ""

urls = ['https://beebom.com/what-is-nft-explained/',
        'https://beebom.com/how-delete-spotify-account/',
        'https://beebom.com/how-download-gif-twitter/',
        'https://beebom.com/how-use-chatgpt-linux-terminal/',
        'https://beebom.com/how-delete-spotify-account/',
        'https://beebom.com/how-save-instagram-story-with-music/',
        'https://beebom.com/how-install-pip-windows/',
        'https://beebom.com/how-check-disk-usage-linux/']

loader = SeleniumURLLoader(urls=urls)
docs_not_splitted = loader.load()

text_splitter = CharacterTextSplitter(chunk_size=1000, chunk_overlap=0)
docs_splitted = text_splitter.split_documents(docs_not_splitted)

embeddings = OpenAIEmbeddings(model='text-embedding-ada-002')

dataset_path = "hub://ihamzakhan89/langchain_course_fewshot_selector"
db = DeepLake(dataset_path=dataset_path, embedding_function=embeddings)

db.add_documents(docs_splitted)

template = """You are an exceptional customer support chatbot that gently answer questions.

You know the following context information.

{chunks_formatted}

Answer to the following question from a customer. Use only information from the previous context information. Do not invent stuff.

Question: {query}

Answer:"""

prompt = PromptTemplate(
    input_variables=["chunks_formatted", "query"],
    template=template,
)

llm = OpenAI(model="gpt-3.5-turbo-instruct", temperature=0)




def get_formatted_chunks(query):
    docs = db.similarity_search(query)
    retrieved_chunks = [doc.page_content for doc in docs]
    chunks_formatted = "\n\n".join(retrieved_chunks)
    return chunks_formatted

# Streamlit UI
'''def main():
    st.title("Customer Support Chatbot")

    # User input for the question
    user_question = st.text_input("Ask a question:")

    if st.button("Get Answer"):
        # Retrieve and format chunks
        chunks_formatted = get_formatted_chunks(user_question)

        # Format the prompt
        prompt_formatted = prompt.format(chunks_formatted=chunks_formatted, query=user_question)

        # Generate answer
        answer = llm(prompt_formatted)

        # Display the answer
        st.subheader("Answer:")
        st.write(answer)
```

```
if __name__ == "__main__":
    main()
'''

st.title('Ask Us')

prompt = st.chat_input('Pass your prompt here')
```