Custom training based LLM(Langchain and ChromaDB needed) Project:-

Project Name – Customer Service Chat-bot

File name chat_bot.py

```
from langchain_chroma import Chroma
from langchain_ollama import OllamaLLM,OllamaEmbeddings
from langchain_text_splitters import CharacterTextSplitter

gemma_llm=OllamaLLM(model="gemma3:1b")
gemma_embed=OllamaEmbeddings(model="embeddinggemma")
vs=Chroma(persist_directory='./db_gemma',embedding_function=gemma_embed)

question="who is prince john?"

info=vs.similarity_search(question,k=3)


# print(info)
# print(info[0].page_content)
# print(info[1].page_content)
# print(info[2].page_content)

context=info[0].page_content

# print(context)
```

```
prompt=f'''

Role:

based on following context, please provide a concise information and human-friendly summery or answerthe question.


context:{context}

question:{question}

'''

output=gemma_llm.invoke(prompt)

print(output)
```

File name train.py

```
from langchain_community.document_loaders import TextLoader

from langchain_text_splitters import CharacterTextSplitter

from langchain_ollama import OllamaEmbeddings

from langchain_chroma import Chroma


# Load text file

loader = TextLoader('./data.txt')

documents = loader.load()


# Split text into chunks

text_splitter = CharacterTextSplitter(chunk_size=500, chunk_overlap=0)

docs = text_splitter.split_documents(documents)
```

Embedding using Gemma3:1B model

gemma_embed = OllamaEmbeddings(model="embeddinggemma")

Create and store in Chroma database

vs = Chroma.from_documents(documents=docs, embedding=gemma_embed, persist_directory='./db_gemma')

File name data.txt

Prince John was a greedy king.

He wanted all the gold in the kingdom.

Robin Hood planned to help the poor.

He was the ruler of the Kingdom of Avalon.

He loved music and built a grand palace near the river.

FOLDER NAME INTERNSHIP_PROJECT

Beginner Level (only LLM based) Project:-

Project Name – Spam Email Classifier

File name spam_email_classifier.py

```
from langchain_ollama import OllamaLLM

# Load LLM model (you can replace with "mistral" or "llama3" if "gemma" not installed)
llm = OllamaLLM(model="gemma3:1b")

# Spam classifier prompt template
def classify_email(email_text):
    prompt = f"""
    You are an intelligent email classifier.
    Classify the following email as "Spam" or "Not Spam".
    Only answer with one of these words exactly.

    Email content:
    {email_text}
    """
```

```
    result = llm.invoke(prompt)
    return result.strip()

# Example test emails
emails = [
    "Congratulations! You've won a $10,000 gift card. Click the link to claim your prize!",
    "Your meeting is scheduled for tomorrow at 10 AM. Please confirm your attendance.",
    "Limited time offer!!! Buy 1 get 1 free on all products. Hurry now!",
    "Please find attached your project report and feedback."
]

# Run classification
for i, email in enumerate(emails, start=1):
    result = classify_email(email)
    print(f"Email {i}: {result}")
```

Acknowledgement

I would like to express my sincere gratitude to my Computer Science teacher (Sir) for his continuous guidance, support, and encouragement throughout the completion of my two projects — "Customer Service Chatbot" and "Spam Email Classifier."

Through these projects, I have gained valuable knowledge and hands-on experience in Python programming, Artificial Intelligence (AI), Machine Learning (ML), and Natural Language Processing (NLP).

In the Customer Service Chatbot project, I used LangChain, ChromaDB, and Ollama models, which helped me understand the practical implementation of the Retrieval Augmented Generation (RAG) technique.

On the other hand, in the Spam Email Classifier project, I applied the Naive Bayes Algorithm from Machine Learning to classify emails as spam or non-spam.

I am deeply thankful to my respected teacher for his valuable guidance and continuous motivation, which helped me successfully complete both of these projects.

References

1. LangChain Official Documentation – https://python.langchain.com/

2. Ollama Documentation – https://ollama.ai/docs

3. Python Official Documentation – https://docs.python.org/

4. Wikipedia – Large Language Model (LLM)

5. TutorialsPoint – Natural Language Processing with Python

6.Medium & OpenAI Blogs – LLM-based Chatbot and Email Classification Tutorials

Note:

The above online resources were used as references while developing the Customer Service Chatbot and LLM-based Spam Email Classifier projects using Python, LangChain, and Ollama without using any external database.