Python Coding

LLM Model API

LMStudio

```
from langchain.llms import OpenAI

#set llm for langchain using model from lmstudio
llm = OpenAI(
       openai_api_base='http://localhost:1234/v1',
       openai_api_key='NULL'
       )
```

```
import streamlit as st
from openai import OpenAI

# Set up the Streamlit App
st.title("ChatGPT Clone using Llama-3 □□")
st.caption("Chat with locally hosted Llama-3 using the LM Studio □□")

# Point to the local server setup using LM Studio
client = OpenAI(base_url="http://localhost:1234/v1", api_key="lm-studio")

# Initialize the chat history
if "messages" not in st.session_state:
    st.session_state.messages = []

# Display the chat history
for message in st.session_state.messages:
    with st.chat_message(message["role"]):
        st.markdown(message["content"])

# Accept user input
if prompt := st.chat_input("What is up?"):
```

```
    # Add user message to chat history
    st.session_state.messages.append({"role": "user", "content": prompt})
    # Display user message in chat message container
    with st.chat_message("user"):
        st.markdown(prompt)
    # Generate response
    response = client.chat.completions.create(
        model="lmstudio-community/Meta-Llama-3-8B-Instruct-GGUF",
        messages=st.session_state.messages, temperature=0.7
    )
    # Add assistant response to chat history
    st.session_state.messages.append({"role": "assistant", "content":
response.choices[0].message.content})
    # Display assistant response in chat message container
    with st.chat_message("assistant"):
        st.markdown(response.choices[0].message.content)
```

GPT

```
from langchain_openai import ChatOpenAI

llm = ChatOpenAI(
    model="gpt-4o",
    temperature=0,
    max_tokens=None,
    timeout=None,
    max_retries=2,
    # api_key="...",
    # base_url="...",
    # organization="...",
    # other params...
)
```

Ollama

```
from langchain_community.llms import Ollama

llm = Ollama(model="llama2:13b")
llm.invoke("The first man on the moon was ... think step by step")
```

Chunking/Splitting

??????

```
# Unicode 編碼
#   \u3002 全形句號
#   \uff0c 全形逗號
# Get Unicode for specific character
# >>> ',','.encode('unicode-escape') # for py3
# >>> list(u',') # for py2

import re
text = "這是中文句子。第一段，第二段，第三段。"
chunks = re.split('[\u3002\uff0c]', text)
#print("\n\n".join([chunk for chunk in chunks]))
for chunk in chunks:
    print("---" * 10)
    print(chunk)
```

??????

```
# \s+ 單或多個空白
chunks = re.split(r'(?<=[.?!])\s+', text)
```

PDF to Markdown

- ??????????????????????
- pdf_to_md.ipynb

```
import os
import pymupdf.layout
import pymupdf4llm
from pathlib import Path
import glob

os.environ["TOKENIZERS_PARALLELISM"] = "false"

def pdf_to_markdown(pdf_path, output_dir):
    doc = pymupdf.open(pdf_path)
    md = pymupdf4llm.to_markdown(doc, header=False, footer=False, page_separators=True,
```

```
ignore_images=True, write_images=False, image_path=None)
    md_cleaned = md.encode('utf-8', errors='surrogatepass').decode('utf-8', errors='ignore')
    output_path = Path(output_dir) / Path(doc.name).stem
    Path(output_path).with_suffix(".md").write_bytes(md_cleaned.encode('utf-8'))

def pdfs_to_markdowns(path_pattern, overwrite: bool = False):
    output_dir = Path(MARKDOWN_DIR)
    output_dir.mkdir(parents=True, exist_ok=True)

    for pdf_path in map(Path, glob.glob(path_pattern)):
        md_path = (output_dir / pdf_path.stem).with_suffix(".md")
        if overwrite or not md_path.exists():
            pdf_to_markdown(pdf_path, output_dir)

pdfs_to_markdowns(f"{DOCS_DIR}/*.pdf")
```

Prompt

```
def get_rag_agent_prompt() -> str:
    return """
        You are a retrieval-augmented assistant.

        You are NOT allowed to answer immediately.

        Before producing ANY final answer, you must first perform a document search
        and observe retrieved content.

        If you have not searched, the answer is invalid.

        Workflow:
        1. Search the documents using the user query.
        2. Inspect retrieved excerpts and keep only relevant ones.
        ...
        """
```

Revision #12
Created 2024-06-03 13:53:11 CST by A-Lang (Admin)
Updated 2025-12-19 14:39:12 CST by A-Lang (Admin)