### **LLM Project: Building a News Research Tool**

#### **Project Tasks**

### **Phase 1: Environment Setup**

1. **Task 1.1: Install Required Libraries**
   * Command: pip install langchain openai streamlit newsapi-python
2. **Task 1.2: Obtain API Keys**
   * Get OpenAI API key from the [OpenAI website](https://www.openai.com/). [[Procedure]](https://docs.google.com/document/d/1vCA1XSB_h8lAgDnMiC5cQmmQTk9DJlzSO4LjTjg8LV8/edit#heading=h.6lqcl123jfwc)
   * Get NewsAPI key from the [NewsAPI website](https://newsapi.org/). [[Procedure]](https://docs.google.com/document/d/1tmXeFjOk4BbVn8D1EpSYhBmrYXwWkE2WInfVc2Fle5A/edit#heading=h.5yeay9xcafn1)

### **Phase 2: LangChain Configuration**

**Task 2.1: Create LangChain Configuration File**

* + Create a file named langchain\_config.py.

**Task 2.2: Initialize OpenAI API in langchain\_config.py**  
from langchain import OpenAI, LLMChain, PromptTemplate

openai\_api\_key = 'your-openai-api-key'

openai = OpenAI(api\_key=openai\_api\_key)

template = """

You are an AI assistant helping an equity research analyst. Given the following query, summarize the most relevant news articles.

Query: {query}

"""

prompt = PromptTemplate(template=template, input\_variables=['query'])

llm\_chain = LLMChain(prompt=prompt, llm=openai)

### 

### 

### 

### **Phase 3: Building Streamlit Interface**

**Task 3.1: Create Streamlit App File**

* + Create a file named app.py.

**Task 3.2: Implement Basic Streamlit Interface in app.py**  
import streamlit as st

from langchain\_config import llm\_chain

st.title('Equity Research News Tool')

st.write('Enter your query to get the latest news articles summarized.')

query = st.text\_input('Query')

if st.button('Get News'):

if query:

response = llm\_chain.run({'query': query})

st.write('### Summary:')

st.write(response)

else:

st.write('Please enter a query.')

**Task 3.3: Run Streamlit App**

* + Command: streamlit run app.py

### **Phase 4: Enhancing the Tool**

**Task 4.1: Integrate NewsAPI in langchain\_config.py**  
from newsapi import NewsApiClient

newsapi = NewsApiClient(api\_key='your-newsapi-key')

def get\_news\_articles(query):

articles = newsapi.get\_everything(q=query, language='en', sort\_by='relevancy')

return articles['articles']

def summarize\_articles(articles):

summaries = []

for article in articles:

summaries.append(article['description'])

return ' '.join(summaries)

def get\_summary(query):

articles = get\_news\_articles(query)

summary = summarize\_articles(articles)

return summary

template = """

You are an AI assistant helping an equity research analyst. Given the following query and the provided news article summaries, provide an overall summary.

Query: {query}

Summaries: {summaries}

"""

prompt = PromptTemplate(template=template, input\_variables=['query', 'summaries'])

llm\_chain = LLMChain(prompt=prompt, llm=openai)

**Task 4.2: Update Streamlit App to Use Enhanced LangChain Configuration**  
import streamlit as st

from langchain\_config import llm\_chain, get\_summary

st.title('Equity Research News Tool')

st.write('Enter your query to get the latest news articles summarized.')

query = st.text\_input('Query')

if st.button('Get News'):

if query:

summaries = get\_summary(query)

response = llm\_chain.run({'query': query, 'summaries': summaries})

st.write('### Summary:')

st.write(response)

else:

st.write('Please enter a query.')

### **Phase 5: Testing and Validation**

1. **Task 5.1: Test Basic Functionality**
   * Ensure the app runs without errors.
   * Verify that entering a query returns summarized news articles.
2. **Task 5.2: Validate News Summarization**
   * Check the accuracy and relevance of the summaries generated by the tool.

### **Phase 6: Documentation and Finalization**

1. **Task 6.1: Document Code and Workflow**
   * Add comments and documentation to the code.
   * Write a README file explaining the project, setup instructions, and usage.
2. **Task 6.2: Prepare for Deployment**
   * Ensure all dependencies are listed in requirements.txt.
   * Finalize the project structure and clean up any unused code.

### **Optional Enhancements**

1. **Task 7.1: Add User Authentication**
   * Implement authentication to restrict access to authorized users only.
2. **Task 7.2: Enhance UI/UX**
   * Improve the UI with Streamlit components for a better user experience.
3. **Task 7.3: Extend Functionality**
   * Add features like saving queries, exporting summaries, or historical data analysis.

### **Conclusion**

Following these tasks will guide you through building an end-to-end news research tool using LangChain, OpenAI API, and Streamlit. This project will enhance your skills and add significant value to your portfolio as a data scientist or NLP engineer.