Coding Task File

Hope to Skill Free AI Basic Course Hackathon

Deadline : 16/01/2024 at 7.00 pm

Welcome to the Hope to Skill Free AI Basic Course Hackathon organized by Xeven Solutions! We are excited to have you participate in this coding challenge. This document outlines the task, coding ethics, general coding conventions in Python, submission guidelines, and important deadlines. So before starting, read the document carefully.

Note: You are not allowed to share and discuss this document with any participant and public.

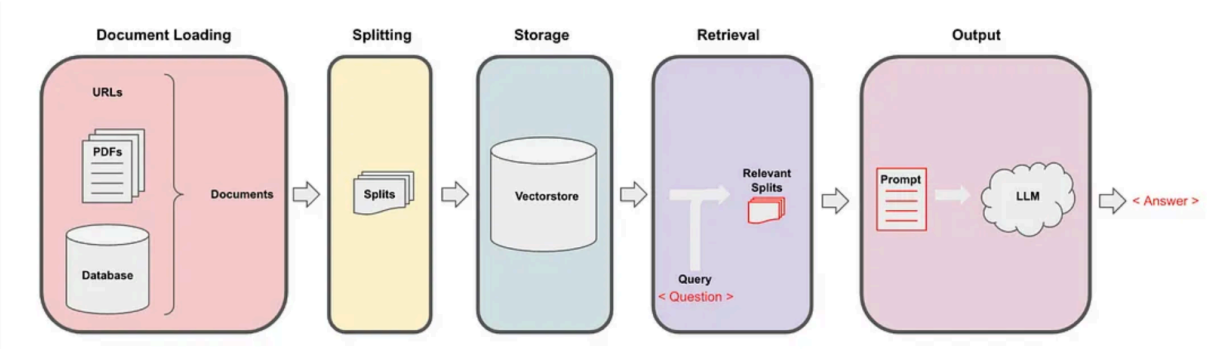
Marks Distribution

No.	Task	Marks
1.	File Loading	10
2.	Document Splitting	10
3.	Embedding Generation	10
4.	Vector Database usage	10
5.	Retrievers	10
6.	Correct Output	05
7.	Documentation	10
8.	Coding Standard	05
9.	Github and Deployment	08
10.	Submission Guidelines Followed	02
11.	Viva	20
Total Marks		100

Task Description

Your challenge is to design and implement a chatbot with the following specifications:

1. File Loading
2. Text Splitting
3. Embedding Generation
4. Vector Database
5. Retrievers
6. Output using LLM



File Loading:

1. The application should provide users with the capability to upload both **PDF** and **DOCX** files. To achieve this functionality, implement separate loaders for **PDF** and **DOCX** files

   Allowed Loaders for PDF (Documentation)

   1. **PyMuPDF**
   2. **PDFPlumber**

   Allowed Loaders for DOCX (Documentation)

   1. **Docx2txt**
   2. **Unstructured**

2. The application should accommodate the submission of a .**ZIP** file containing **PDF** and **DOCX** files. Additionally, you are required to develop a **custom loader** specifically for handling **ZIP** files

Text Splitting:

Use two different text splitters to process the loaded file.

You can use the following text splitters and mentions the results comparison in the document

1. Split by character
2. Recursively split by character

Embedding Models

To generate text embeddings, you must utilize two specified embedding models and conduct a comparative analysis of their results. Additionally, it is crucial to include the vector size information in the documentation for thorough analysis. The permitted embedding models, available on Hugging Face, are provided for your reference

1. 'sentence-transformers/all-MiniLM-L12-v2'
2. "BAAI/bge-small-en-v1.5"

Vector Database:

Utilize two different cloud-based vector databases.
Following vector stores are allowed to you

1. Pinecone
2. Elastic Search

Retrievers:

Implement two different retrievers from LangChain. You are required to use two retrievers and provide the result comparisons for each retriever. Here are the retrievers permitted to use.

1. Vector store-backed retriever
2. Parent Document Retriever

LLM:

You can only use **gpt-3.5-turbo-16k** as LLM. You are not allowed to use any other model in any case. If you face any issue you will mention it in the documentation.
Note: Be careful with the Openai API key. Never Push API key into any github repo.
Here is your OpenAI Key

Your API Key : “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”

Specifications:

1. **Platform**: LangChain
2. **Language:** Python
3. **Additional Tools:** Streamlit.io for deployment
4. **Submission**: You are required to do the following steps
    a. Push the code to your personal GitHub repository and deploy the app on Streamlit.io.
    b. Submit the zip folder in Google Form containing the items mentioned here

Coding Ethics:

Participants are expected to adhere to ethical coding practices. This includes, but is not limited to:

1. **Respect for Users:** Ensure your chatbot is respectful and does not generate offensive or inappropriate content.
2. **Privacy:** Respect user privacy. Do not collect unnecessary personal information.
3. **Transparency**: Clearly disclose the capabilities and limitations of your chatbot.
4. **Bias:** Be mindful of potential biases in your chatbot's responses. Strive for fairness and inclusivity.

General Coding Conventions (Python)

Follow these coding conventions to ensure clean and readable code:

1. **Indentation:** Use 4 spaces or Single Tab equal to 4 spaces for indentation.

2. **Naming:** Use meaningful and descriptive names for variables, functions, and classes (follow PEP 8).
3. **Comments:** Include comments where necessary to explain complex sections of your code.
4. **Modularity:** Break down your code into functions with clear responsibilities. Use separate files for Utility Functions.
5. **Error Handling:** Implement proper error handling to enhance the robustness of your code
6. **Credential security:** Use a separate .env file or secret.toml file to ensure the credentials security.

Submission Guidelines

1. **GitHub Repository:** Push your code to a personal GitHub repository. Ensure that the repository is set to public. Any commit to github after the deadline will result in disqualification. **(Never push any file containing the API key. Pushing keys to any such platform can result in a penalty or in severe case disqualification).**
2. **README.md:** Include a README file with instructions on how to run your chatbot locally.
3. **Documentation:** Provide a brief document in pdf file explaining the functionality of your chatbot and any additional features you've implemented.
4. **Requirement file:** Provide a requirements.txt file to install the dependencies.
5. **Submission:** Make a .zip file of your project code documentation including your code, requirements.txt, documentation.

Submission portal

Click here to access the submission portal. Once you submit the response you will not be able to change it.