Yarnit Case Study Round - Submission

Case study 1) Webpage Question Answering

The Webpage Question Answering API is built using FastAPI. FastAPI provides automatic generation of interactive API documentation, making it easy to understand and use.

Input Parameters

To use the Webpage Question Answering API, clients need to send a POST request to the /url endpoint with the following parameters:

- url (str): The URL of the webpage from which the answer will be extracted.
- question (str): The question to be answered based on the webpage content.

Response Generation:

Upon receiving the request parameters, the API leverages ***Language model by OpenAI (GPT)"*** model to process the input and generate an answer to the provided question based on the content of the specified webpage. The model uses advanced natural language processing techniques to analyse the webpage content and extract relevant information to formulate an accurate response.

Deployment Instructions (Local)

1. Install the required dependencies by running
   ```
   pip install -r requirements.txt
   ```
2. Set the OPENAI_API_KEY environment variable with your OpenAI API key.
3. Run the FastAPI application using
   ```
   uvicorn url-bot:app --reload
   ```

Model Workflow: The below is the model diagram to get an idea of how each function is working and what each technique is doing.
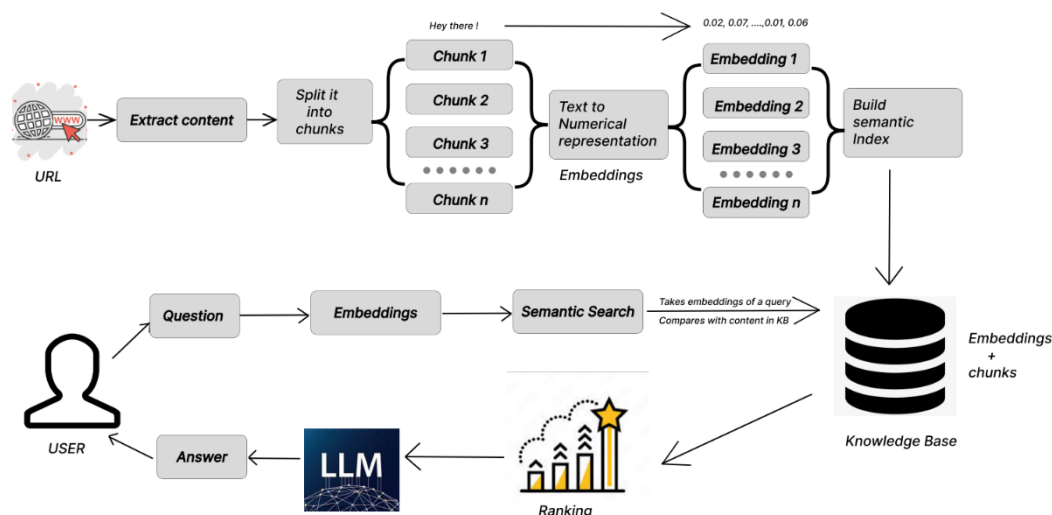
Sample Input:


```json
{
  "url": "https://en.wikipedia.org/wiki/Generative_artificial_intelligence",
  "question": "What are the concerns around Generative AI?"
}
```

Sample Output:

Code	Details
200	**Response body**

```json
{
  "Output": " Concerns around generative AI include potential misuse for cybercrime, the creation of fake news and deepfakes, and the replacement of human jobs. Bias in training data and resulting models is also a concern, as well as the potential for existential risks from artificial general intelligence. Guidelines and regulations are being developed to address these concerns."
}
```

Response headers

```
content-length: 381
content-type: application/json
date: Mon,06 May 2024 11:26:05 GMT
server: uvicorn
```

Case study 2) Generate Marketing Content

This API enables users to generate content for various platforms based on specified topics and tones. It leverages "*gpt-3.5 turbo*" model provided by OpenAI to generate well-structured, informative, and engaging content tailored to the target platform and topic.

Input Parameters

To use the API, send a POST request to the /generate_content endpoint with the following parameters:

- **platform name (str)**: The name of the platform for which the content is being generated (e.g., Twitter, Instagram, LinkedIn, Snapchat, etc.)
- **topic (str):** The topic or subject matter for the content.
- **tone (str)**: The desired tone or style for the content.
  Available options: "Funny", "Informative", "Inspirational", "Personal ", "Conversational", "Formal", "Engaging ", "Visual ", "Storytelling", "Thought-provoking", "Other".
  If the tone parameter is set to "Other", the user can provide a custom tone string through this parameter.
- **length (str)**: The desired length of the content. Available options: "short", "medium", "long".

Response Generation

- The API leverages GPT model by OpenAI. The prompt for the language model is constructed using the ChatPromptTemplate class, which takes the input parameters and generates a prompt instructing the model to generate content based on the specified platform, topic, tone, and length.
- The LLMChain class is used to combine the prompt template and the language model, and the StrOutputParser is used to parse the generated output into a string format.

Deployment Instructions (Local)

1. Install the required dependencies by running
   ```
   pip install -r requirements.txt
   ```
2. Set the OPENAI_API_KEY environment variable with your OpenAI API key.
3. Run the FastAPI application using
   ```
   uvicorn content-generator:app --reload
   ```

Sample Input:

default

POST /generate_content Generate Content Api

Parameters

Cancel

Name	Description
platform_name * required string (query)	Linkedin
topic * required string (query)	GenerativeAI
tone * required string (query)	Informative posts
length * required string (query)	short

Sample Output:

Code	Details

200	**Response body**
	{
	  "content": "Exciting developments in the field of Generative AI are revolutionizing industries across the board! From creating realistic images and videos to generating human-like text, the possibilities are endless. As a content writer, I am fascinated by the potential of Generative AI to enhance creativity and streamline workflows.\n\nIf you're curious about how Generative AI can benefit your business or creative projects, I highly recommend diving into this cutting-edge technology. Stay tuned for more updates on the latest trends and applications in Generative AI\n\n#GenerativeAI #AI #Innovation #Technology #Creativity"
	}
	
	Response headers
	content-length: 632
	content-type: application/json
	date: Mon,06 May 2024 11:32:35 GMT
	server: uvicorn
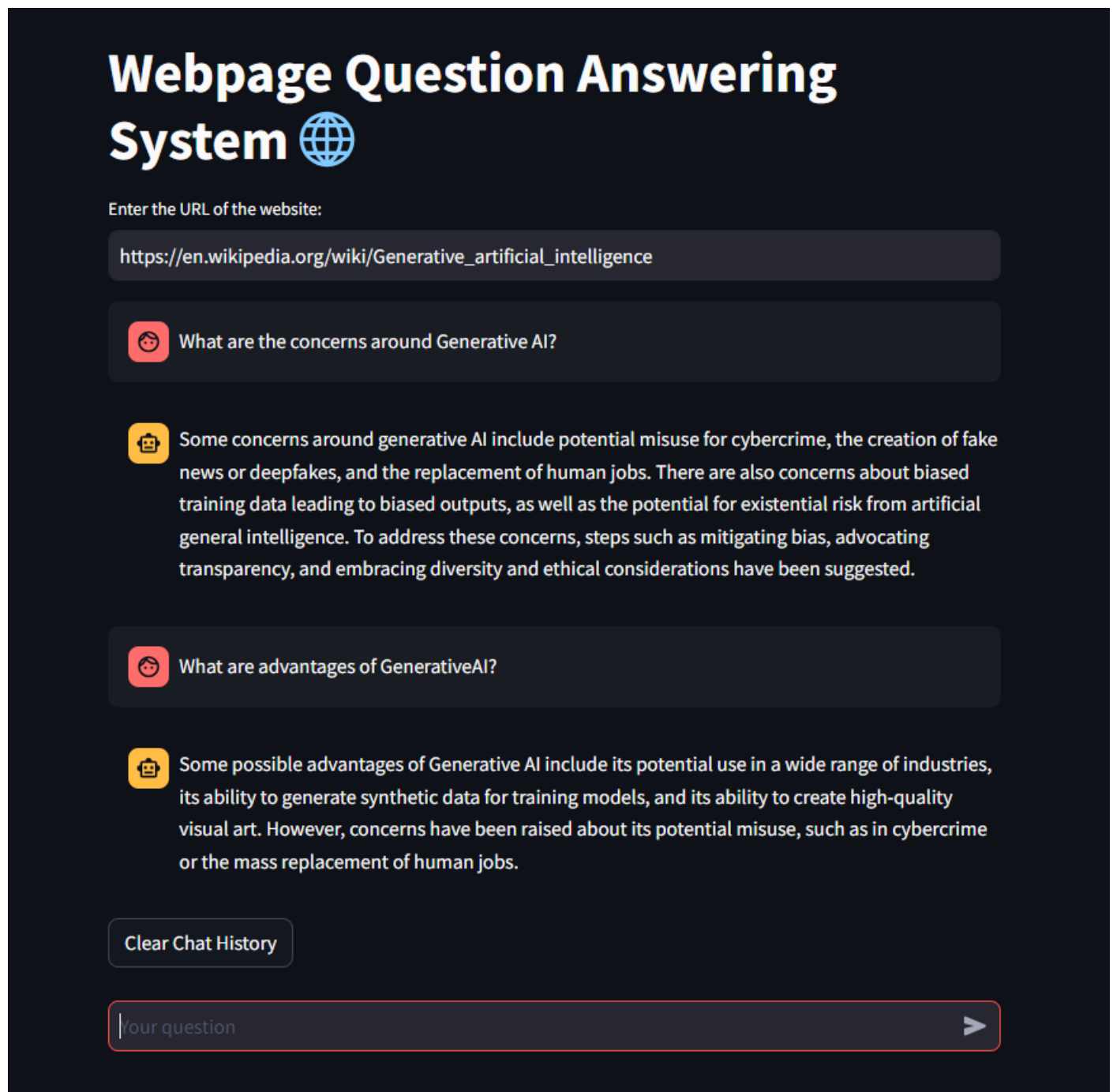
For both the case studies a sample UI is done using *Streamlit*. These are the screenshots of it.

If you want to check out the code, please visit the following Colab notebook links.

Link: Webpage Question Answering

Link: <u>Generate Marketing Content</u>