Q1. Create a rudimentary Question Answering system that can answer questions related to the Israel Hamas war.

1. You can use any open-source LLM (LLAMA, T5, BERT), library (langchain (https://www.langchain.com/), LLamaIndex (https://www.llamaindex.ai/)) or algorithm (BM25, Maximal Marginal Relevance) that you want.
2. Also, you can augment the existing articles by extracting additional information from Wikipedia or DBpedia

You will be judged based on the relevancy and readability of the answers generated.

Example question can be "**What happened at the Al-Shifa Hospital**"?

IDEA OF THE CODE

The system integrates both information retrieval and natural language processing to answer user queries based on a given dataset.The system operates in two main stages:

1. **Document Retrieval**
2. **Question Answering**

Detailed Explanation

1. Data Loading:
    - The system begins by first loading the dataset which was provided and was a JSON file. Each document contains an `articleBody` that is used for retrieval and answering purposes.
2. Document Retrieval (BM25Retriever):
    - `BM25Retriever` class is initialized with the list of documents. It converts these documents into TF-IDF matrix using `CountVectorizer` and `TfidfTransformer` from `sklearn`.
    - Query Processing: When query comes, the `retrieve` method converts the question into a TF-IDF vector and calculates cosine similarity between this vector and the document matrix…. It will then identify and return the top `n` most similar documents. The similarity score will indicates how relevant each document is to the query.
3. Question Answering (LangChainQA):

- The `LangChainQA` class is initialized with the API URL and an API token for authentication.
- Querying the API: The `query` method sends an HTTP POST request to the external API with the question and the retrieved document (context). The API returns an answer extracted from the context. This method also includes error handling to manage any issues with the API request.
- Api for the model (DistilBERT)–https://api-inference.huggingface.co/models/deepset/bert-basecased-squad2 –This is a smaller, faster, and lighter version of BERT fine-tuned on the SQuAD dataset for question answering.

4. Main Function:
   - Data Loading: Loads the dataset from the specified file path.
   - Document Retrieval: Initializes the `BM25Retriever` with the dataset and retrieves the most relevant document based on the user's question.
   - Question Answering: it will Initialize the `LangChainQA` with the API details and will send the question along with the retrieved document to the API to get an answer.
   - Output: it Prints the question and the answer obtained from the API.
5. Entry Point:
   - Ensures the script is executed with a question provided as a command-line argument. If not, it displays usage instructions. If a question is provided, it calls the `main` function with the question.

CODE EXPLAINATION

1) **Imported Necessary Libraries**:

The libraries are for data handling (`os`, `json`), natural language processing (`nltk`), machine learning (`sklearn`), HTTP requests (`requests`), and numerical computations (`numpy`).

2) **Downloaded NLTK Data** (nltk.download('punkt'))

3) **BM25Retriever Class:**(class BM25Retriever:)

Took a list of documents and initialized a `CountVectorizer` and `TfidfTransformer` to create a TF-IDF matrix.

Took a query, which will convert it to a TF-IDF vector, then computes cosine similarity with the document matrix, and retrieves the top n documents.

4) **LangChainQA Class**:(class LangChainQA:)

It will Store the API URL and token.

It Sends a POST request to the API with the question and context, handling any errors and returning the answer or an error message.

5) **load_data Function**: (def load_data(file_path):)

Loads data from a JSON file, ensuring the file exists and reading its content. Returns a list of document bodies.

6) **main Function**:

It will Load the dataset, initialize the BM25Retriever, and will retrieve the most relevant document as context, and use LangChainQA to query the external API for an answer, then print the question and answer.

7) **Entry Point**:(if __name__ == "__main__":)

It Checks if the script is run as the main program, verifies command-line arguments, and runs the `main` function with the provided question.

Example Workflow:

1. User inputs a question, e.g., "What happened at the Al-Shifa Hospital?"
2. The code loads the JSON data containing documents related to the Israel-Hamas war.
3. The BM25 retriever identifies the most relevant document based on the question.
4. The LangChainQA class sends the question and relevant context to the Hugging Face model.
5. The model generates an answer based on the provided context.
6. The answer is displayed to the user.

Summary:

- The code combines document retrieval using the BM25 algorithm and question answering using a Hugging Face model.

- It provides a simple yet effective way to generate answers to questions related to the Israel-Hamas war based on available documents.
- By following the workflow outlined above, the system can efficiently process user queries and provide relevant answers.