Coding Challenge: Context Retrieval Script for a RAG-Chatbot

Overview:

Build a python script for information retrieval for a hypothetical chatbot about football clubs in Germany. Use Wikipedia and Wikidata to retrieve up-to-date information about football clubs that are currently part of Germany`s 1. Bundesliga and their current coaches. That data should be processed in a way so that it can be used as a prompt for an LLM.

Scenario and Goal

The user will ask in a colloquial way about the current “coach of a city”, for example: “Who is coaching Berlin?”

The chatbot should then understand that the user wants information about the coach of the respective football club in 1. Bundesliga of that city. The answer should provide the name of the current coach and a little bit of information about the coach.

Additional assumptions:

The chatbot does not need to be able to answer any other question.

You can assume that the user will write the city name correctly except for upper- and lowercase, so you should handle this respectively.

Hamburg has two football clubs in 1. Bundesliga, you can assume for FC St. Pauli that the user will ask for “Pauli” or “pauli”.

Data Requirements

The information about who is coach of the club of that city should come from **Wikidata** and should be retrieved on every question. This should be queried via SPARQL and the Wikidata API.

The information about the coach should come from the intro of their respective **Wikipedia** article and should be retrieved on every question.

The information about which clubs are part of the current season, and which club belongs to which city should come from Wikipedia or Wikidata. Whether you retrieve it on every question or not is up to you.

Imagine these two sources to be your only sources!

Coding Requirements

Input:

String involving the user question.
Examples:

- "Who is coaching Berlin?"
- "What about munich?"
- "Who is heidenheims manager?"
- "Who is it for Pauli?"

Output:

A string that involves the final prompt for an LLM. This should involve the system prompt for the llm, the user question and the additional retrieved information.
The actual answer of the LLM is **not** required, so you do not have to use a commercial API.

Interface:

A frontend is **not** required. Input and output can be provided via the console once the script is started. The interface is the console in this case.

Comments, Logging, Errors:

Please use comments in your code so that it will be easily understood by others.

Please implement logging that enables debugging related to potential false answers of the LLM. Finding out where a false answer came to be, should be possible.

Please handle errors related to data that are not available or missing in a way that makes sense for the user.

Additional Questions:

1. What are advantages and disadvantages of using additional information for a chatbot instead of letting the LLM answer without it.
2. What are advantages and disadvantages of querying for this data on every user question?
3. How would the process change if the information about coaches only were available via pdf?
4. Do you see potential for agents in this process? If so, where and how?
5. How do these kinds of processes profit from a data model that models the specific domain knowledge?

Summary

Show that you can identify entities in an input string, connect them to a data model and infere and retrieve additional information from a Knowledge Graph and from text data. Also show that you can use this data for context in a potential RAG-System.

Submission:

Please submit your code via link to a GitHub repository.

Please include a readme that involves a how to and the answers to the additional questions

Please use Python 3.8+

Estimated time required for the challenge: 3-5 hours
(If you do encounter difficulties that would exceed this time frame by a lot, you are allowed to make sensible simplifying assumptions)

Submission **Deadline**: 1 week

Good luck and have fun! 😊