0306_LangGraph와 LLM을 사용한 프로그램을 작성하기

Concept

사용자의 고민을 들어주는 ChatBot

특징

LangGraph를 이용해서 단계를 나눠 관리하기 용이하게 구성했다.

ChatBot class를 활용해서 history를 저장할 수 있게 했다.

Node 소개

- user_login : 사용자의 이름을 확인한다. 두 번 이상의 상담시에는 history 폴더에 상담 내용이 저장되어있는데, user_login에서 상담내용이 있는지를 파악하고, 있다면 이전 상담 내용을 불러올 수 있다. 추후 아이디와 비밀번호, DB를 활용한다면, 이를 확장할 수 있다.

- init_model : LLM에게 역할을 부여한다. 역할은 settings.py에 저장되어 있다. 역할을 부여받은 LLM은 상담자로서의 역할을 수행하게 된다.

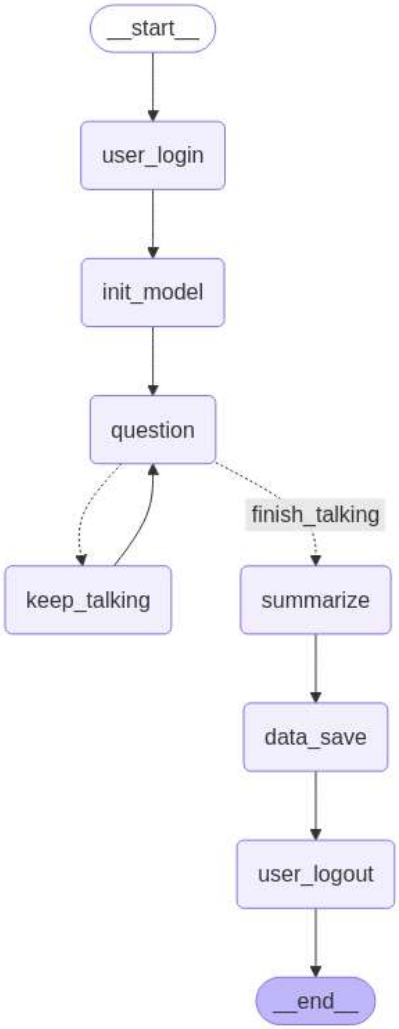
- question : 사용자와 ChatBot간의 대화를 진행하기 위해 사용자로부터 input data를 받는다. 만약 대화를 종료하기 원한다면 '대화 종료'를 입력해서 대화를 끝낼 수 있다. 추후 대화를 종료하는 버튼을 누르거나, 긴 문장 속에서 대화를 종료한다는 의미를 가질 경우 대화를 끝낼 수 있도록 확장할 수 있다.

- keep_talking : 대화가 진행되는 노드이다. 사용자의 입력을 받은 후에 history에 저장하고, 기존에 저장된 대화내용과 새로 입력받은 문장을 가지고 LLM에게서 응답을 받는다. 응답을 다시 history에 저장한 후에 사용자에게 보여지게 된다. 이후 다시 question 노드로 돌아가 사용자에게 입력을 받는다. [Tokenizer.apply_chat_template]를 사용해서 LLM이 문장의 흐름을 이해하기 쉽도록 했다.

- summarize : 사용자가 '대화 종료'를 입력해서 대화를 끝낸 후에 일어나는 이벤트이다. 얼마나 대화를 이뤘는지, 대화의 요점은 무엇인지를 요약해 사용자에게 보여준다.

- data_save : 사용자에게 대화 내용을 저장할지 여부를 물은 후에 대화를 저장한다. 다음 대화가 이루어질 때 저장된 대화내용을 LLM이 참조하게 된다. 추후 DB에 대화를 저장하거나, 사용자가 원하는 부분만 저장, 또는 기존 데이터를 삭제하는 등의 확장이 가능할 것이다.

- user_logout : 대화를 종료하고, user_id를 초기화한다. 이후 프로그램을 종료한다.



왼쪽 사진은 Graph의 형태이다.