**Team Cobra Chatbot**

To start, the chatbot splits the resumes into chunks with RecursiveCharacterTextSplitter and utilizes OpenAIEmbeddings to set up a vector database for each resume so that semantic searches for a user query can be carried out independently on different resumes. Then, the chatbot finds out and records the owner of each resume.

Part 1: Normal Query Processing

The chatbot divides all resumes into groups of 2. For each group, it performs semantic search for each of the two resumes and puts the matched content into the prompt of an LLMChain. As a result, an LLMChain is created for each set of resumes. After the LLMChains generate responses for a given query, the responses are combined by another LLMChain to form the final answer.

**This usually does not take longer than 30 seconds for 15 resumes.** If no answer is generated after a while (this seldom happens), I kindly ask you to halt the execution with the stop button in the top right-hand corner and hit the Ask button again.

Part 2: Resume Comparison

If the user enters some comparison criteria or job requirements in the textbox, the chatbot will tell the user which candidates are most suitable for the job. It will state so if it thinks none of the candidates is a good match. If no job requirement is entered, the chatbot will draw a conclusion on the candidates’ abilities and try to state who are the stronger ones if a fair comparison can be drawn.

All resumes are divided into groups of 3. Before drawing a conclusion, the chatbot creates two LLMChains for each set of resumes comparing the candidates’ work experience and education history respectively. Most relevant contents are extracted through semantic search from the vector stores of every resume in a group. These are combined and fed into the prompts of the two LLMChains mentioned. The responses generated by these chains are fed to a new LLMChain, which is used to draw a conclusion by indicating which candidates in the set fulfil the user specified job requirements, etc. After conclusions are generated for each set of resumes, they are combined to form a final answer.

**If you use this feature on 15 resumes, it will take around 45 seconds to generate an answer. Please be patient.** If no answer is generated after a while (this seldom happens), I kindly ask you to halt the execution with the stop button in the top right-hand corner and hit the Compare Resumes button again.

Part 3: Interview Invitation

**IMPORTANT Tips for testing: Please only use the resumes of someone you know to perform these actions because the chatbot will send emails and Google calendar invites to the selected candidates.**

Utilizing Zapier NLA, the chatbot can invite selected candidates to interviews by sending them emails and Google calendar invites with the specific date and time for the interview. Please enter the date and both start and end time of the interview.

The chatbot is linked to the following Gmail account: [phantomsmithhk@gmail.com](mailto:phantomsmithhk@gmail.com). In my Zapier account, I enabled two actions: “Google Calendar: Create Detailed Event” and “Gmail: Send Email”. The chatbot have two Transformer chains calling ZapierNLARunAction to carry out each action. After that, the calendar event created and email sent are displayed.

If a user wants to perform these actions with their own Gmail account, they must have their Gmail account and Google calendar linked to Zapier, and enable the two actions mentioned above. Also, they will need a Zapier NLA API key for the chatbot to connect to Zapier.