Build an Interview Practice App

Description:

In this project, you will need to build an Interview Practice app with prompt engineering, an OpenAI API, and a user interface.

Later, you could use this for your own needs to practice for interviews.

Topics:

- Interview Prep
- ChatGPT
- Prompt Engineering
- Web applications

Prerequisites:

- Python or JavaScript knowledge
- Knowledge of ChatGPT and OpenAI API
- Basic knowledge of building a front-end application

Estimated time to complete: 5 hours

Task Description

A huge round of applause comes to you for reaching this step. You have started with sprint 1 on prompt-engineering. You are just at the start of the AI engineering course, but you know the very basics, so let's get that into action.

This project will be a good starter. You will build your own Interview Preparation app. I would imagine that you have participated in several interviews at some point. You have been asked questions. You were given exercises or some personality tests to complete.

Imagine if there is a way to practice for all of that in an efficient and interesting way.

Fortunately, ChatGPT comes to help. With it, you can do pretty much everything, including preparing for your next dream position.

Your task will be to implement a single-page website using **VS Code** (or Cursor) editor, and either a Python library called Streamlit or a JavaScript framework called Next.js.

You would need to call OpenAI, write a system prompt as the instructions for ChatGPT, and your own prompt with the interview prep instructions.

You will have a lot of freedom in the things you want to practice for your interview. We don't want you to put it in the box. Interview Questions? Specific programming language questions? Asking questions at the end of the interview? Analyzing the job description to come up with the interview preparation strategy? Experiment!

If you feel confident and have experience in software engineering, feel free to over-engineer the app by adding different things from the Optional Tasks, plus coming up with your own things. You can try making it as a portfolio project!

Remember, you have all of your tools at your disposal if, for some reason, you get stuck or need inspiration: ChatGPT, StackOverflow, or your friend!

Task Requirements

The exact task requirements are as follows:

1. Research the exact nature of interview preparation you want to do. This is intended as we want you to explore and get creative!
2. Figure out how you are going to build the front-end, either using specific Streamlit components, or HTML/CSS in Next.js.
3. Create an OpenAI API Key for this project.
4. Write at least 5 system prompts with different techniques (few-shot learning, Chain-of-Thought, Zero-Shot Prompting, etc) and check which one works best for you.
5. Tune at least one OpenAI setting (temperature, Top-p, frequency, etc).
6. Add at least one security guard to your app to prevent misuse.

Optional Tasks

After the main functionality is implemented and your code works correctly, and you feel that you want to upgrade your project, choose one or more improvements from this list. The list is sorted by difficulty levels.

Caution: Some of the tasks in medium or hard categories may contain tasks with concepts or libraries that may be introduced in later sections or even require outside knowledge/time to research outside of the course.

Easy:

1. Ask ChatGPT to critique your solution from the usability, security, and prompt-engineering sides.
2. Improve ChatGPT prompts for your personal domain in which you would like to interview prep: IT, finance, HR, communication, etc.
3. Implement more security constraints, like user input validation and system prompt validation. Could we consider utilizing ChatGPT to verify these aspects?
4. Simulate different difficulty levels – Adjust the complexity of interview questions (easy, medium, hard).
5. Optimize prompts for concise vs. detailed responses – Experiment with prompting ChatGPT to give short or in-depth answers.
6. Generate interviewer guidelines – Ask ChatGPT to create structured evaluation criteria for technical and behavioral interviews.
7. Simulate a mock interview with AI personas – Ask ChatGPT to role-play as a strict, neutral, or friendly interviewer.

Medium:

1. Add all of the OpenAI settings (models, temperature, frequency, etc.) for the user to tune as sliders/fields.
2. Implement at least two structured JSON output formats for the interview preparation.
3. Deploy your app to the Internet.

4.Calculate and provide output to the user on the price of the prompt.

5.Read OpenAI API documentation, think of your own improvement, and implement it.

6.Use Gemini, Claude or a different LLM to act as LLM 2 that would validate the output of the main LLM 1 that is used to generate interview preparation (LLM as a judge).

7.Try to jailbreak (break into) your own application by trying to provide an invalid prompt, message, job file, etc. Provide the results of this experiment in an Excel sheet.

8.Add a separate text field or another field to include the job description (the position) you are applying for and getting interview preparation for that particular position (RAG).

9.Provide the user with the ability to choose from a list of LLMs (Gemini, OpenAI, etc.) for this project.

10.Think of a creative way; how could you use image generation inside this project and implement it using code?

Hard:

1.Using Streamlit (Python) or React (JS) components, implement a full-fledged chatbot application instead of a one-time call to the OpenAI API.

2.Deploy your app to one of these cloud providers: Gemini, AWS, or Azure.

3.Use LangChain packages to implement the app using chains or agents.

4.Add a vector database to check if the provided interview preparation data was seen before, prompting LLM to generate new interview preparation data.

5.Use open-source LLMs (not Gemini, OpenAI, etc.) for the project.

6.Fine-tune an LLM to have an interview preparation focus.

Evaluation Criteria

•The learner understands the basic principles of how the program works.

•The learner knows how to build a single-page front-end website using their knowledge and/or external resources (internet, Chat-GPT, etc.)

•The learner's project works as intended; you can prepare for an interview by asking for help from the ChatGPT.

•The learner understands the potential problems with the application. They are able to detect the potential improvements.

•The learner understands differences between user, system, and assistant roles.

•The learner understands basic prompt engineering techniques.

•For the maximum mark, complete **at least 2 medium and hard optional tasks**.

If you want to push yourself even higher, implement all of the optional tasks!