Luke McMeans

Data Project 2 Reflection

May 1, 2025

Flavors, Failures, and Fixes: Guy, the Recipe Chatbot

Dataset Selection: As a college student who's focused on eating healthy and gaining muscle, I tend to meal prep for the week. LLMs, like ChatGPT, have been helpful throughout this process, creating recipes and portioning properly. So, I took inspiration from this to make a chatbot that was solely focused on this. I choose spoonacular for my API due to its complex search. This way, we can input the prompt directly as the API's header, and it can still return a recipe from it. If this weren't the case, we would've either had to parse through the relevant words/phrases in the prompt, or just have the user input the name of a recipe. Both situations would've been unsatisfactory to the foundational goals of this chatbot, so I'm glad it was able to work out. For the imported data, I just needed to find a dataset that had proper recipes. I first found a Kaggle dataset with about two million entries, which would've been great in creating diversity. However, this would've been a massive file, over two gigabytes in size. The amount of time it would've taken to traverse this CSV file would've been unnecessarily long, especially given the use of an API. I instead choose to have a smaller dataset, with thirteen thousand recipes, and have spoonacular do the heavy lifting if needed.

Challenges/Problems: My biggest challenge in this process was trying to transverse the CSV/SQL file once the user prompted the bot. We can't send the prompt through the dataset as we can with spoonacular's complex search, so I instead sent each word in at a time to find matches. Along with this, the dataset has a string of ingredients, but the string is structured like an array. I eventually had to account for this formatting in my recipe service file, which led to the file reading portion functioning well. Finally, my last issue was finding the right Gemini model to use. I had used Gemini for another Python project before, so I simply used the same API key. However, the model I used for that project wasn't compatible here for some reason. To solve this, I just brute-forced my way through it, iterating through all available models and printing while the ones were available under my API key. From this, I selected Gemini 2.0 Flash, which is still in use for this project.

Key Learnings/Discoveries: This entire process gave me great insight into the development process. Working solo, I got the chance to interact with every process: frontend, backend, services, styling, cloud deployment, etc. It felt like a full circle moment, bringing together aspects that I've worked on before to create a fun project. This project also gave great insight into the power that APIs have. I know we've worked before with it, whether it's finance or flight data, but the complex search feature shows how useful these are. I'm excited to see where my software development journey goes from here with everything in mind.

Potential Enhancements: In its current state, Guy is only able to remember messages from the current conversation. It doesn't have a memory like most modern LLMs use. So, once the user leaves the conversation, all memory should be forgotten. If I had more time, I would've loved to be able to include that. It also would've been nice to have a user system, leading to more personalized details. From here, we could've had the bot remember their dietary restrictions, nutritional goals, and more. As a final note, spoonacular has support for nutritional details. I would've loved to be able to incorporate that, adding a level of information and customization.