CSC361-Artificial intelligence

Cooking chatbot

Student Name: Lama Aldakheelallh
Student ID:444200795
Section: 44092

Introduction

As artificial intelligence and natural language processing came into play, incorporating AI into different apps has become much more feasible. One of the most innovative uses of AI in real life is recipe formulation, where people can provide raw materials, and AI crafts a recipe that is intricate and detailed. This project intends to create an AI recipe generator based on the Google's Gemini API, which takes your ingredients and returns back a recipe, which has a name attached to it, an introduction, a list of ingredients, cooking steps, estimated calories, allergy notes, and any relevant health tips.

The motivation behind this project is to assist users in getting the most value out of everything that they have available at home while exposing them to new and healthy meals. It is quite common for people to face challenges when cooking, especially when they have a few ingredients or specific meals that they are allowed to eat. An AI based recipe generator serves as a tailored approach to help users work around their input.

This document outlines the calls to libraries, credentialing, as well as some of the other issues that were experienced like rate limits, strange model responses, and API combination issues.

LIBRARY CALLS AND MANAGEMENT OF CREDENTIALS.

For hands-on deployment of recipe generation, we needed an artificial intelligence-driven google recipe generator and we had access to the Google Gemini models through the google.generativeai library. Below is how the code library was configured and credentialing was handled in an encrypted manner

- **Installation Of Required Libraries**

Before deploying hands-free recipe generation, AI powered recipe gneration tools were configured on the available libraries. The primary focus of this project was text generation model from Gemini, hence we focused on the key library of this project, google.generativeai. To add the dependencies, the following command was executed:

```
pip install google-generativeai
```

- **Setting Up The API**

Google offers an API Key for the Gemini models based on which requests are authenticated. This key was obtained from the Google AI Developer console to help integrate the API into our python code. genai was set up using this command:

```
import google.generativeai as genai
genai.configure(api_key=”KEY”)
```

To avoid the possibility of exposure, the API Key was stored in an environment variable instead of being hardcoded in the code.

While the application was being executed on a cloud platform, we were able to add extra layers of security when the API Key was saved like other secrets in environment variables in the server, to avoid unauthorized access.

LIBRARY CALLS AND MANAGEMENT OF CREDENTIALS.

- **Defining the Model and Content Creation**

The Gemini model was set up with the capability of converting supplied ingredients into recipes. This required user input in the form of ingredients which were transformed into a detailed prompt and then sent to the model. In return, the model provided a formatted output with the recipe title, ingredients, directions, calorie count, potential allergies, and health advantages. Comprehensive error management was implemented in case of API failures and other erroneous outputs.

- **Maintaining API Rate Limits and Limits on requests to be made.**

As with other services, Google's Gemini API has limits that may affect the number of requests made to the API.

In order to avoid going over the limits, caching strategies were devised for ingredients and their recipes that were requested many times.

Also, the application was built to manage rate limit errors by performing retries with exponential backoff where appropriate.

And so, the AI-based recipe generator was made such that users can come up with creative and well formed recipes based on the ingredients at hand.

Sample run

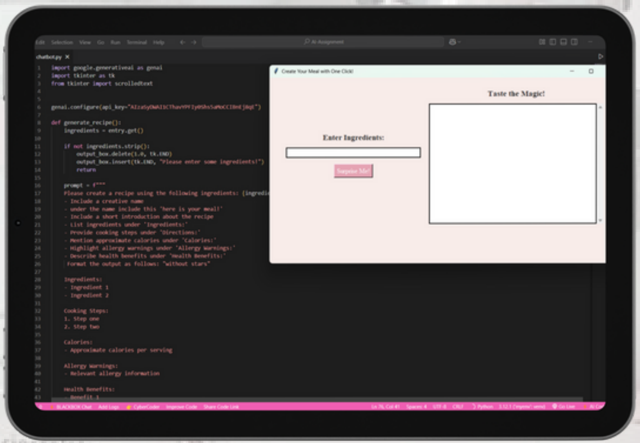

Figure 1: Chatbot running code

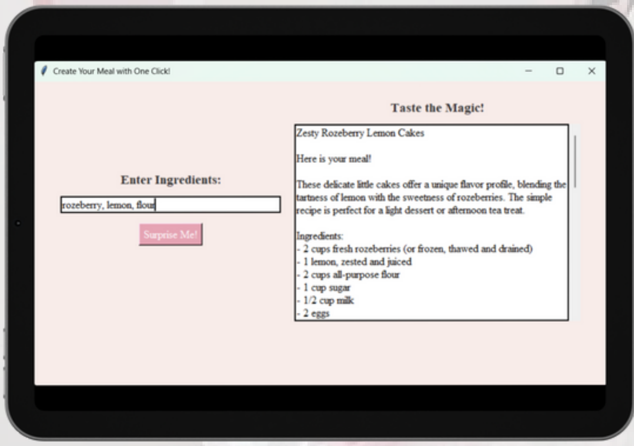

Figure 2: Output Example

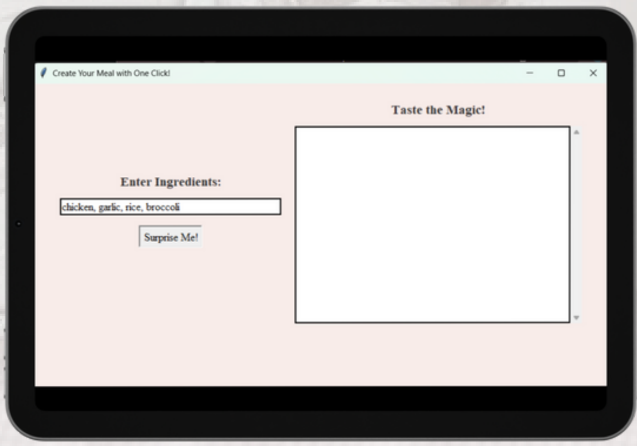

Figure 3: Writing ingredients then press Surprise me to generate recipe

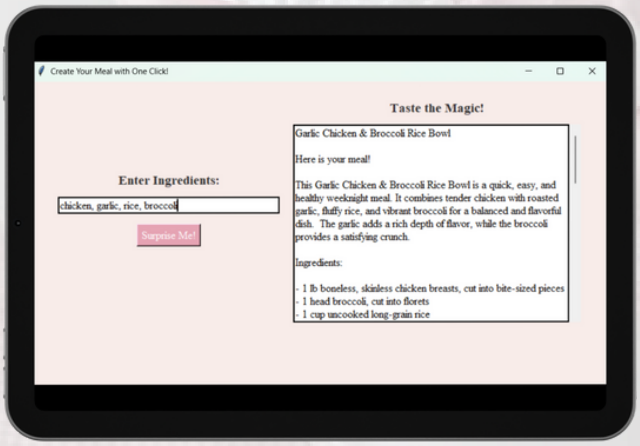

Figure 4: Recipe generated with creative name and introduction followed by ingredients

Sample run

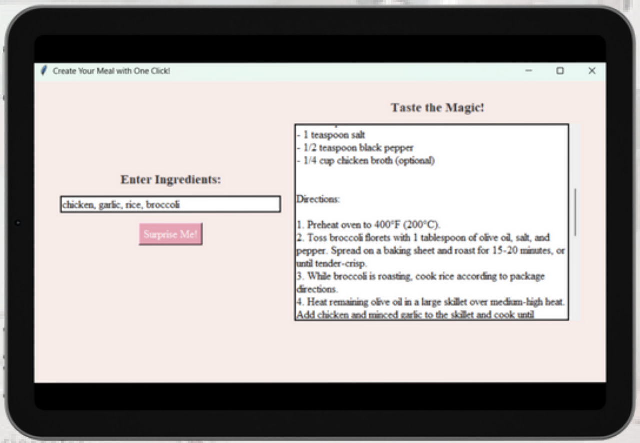

Figure 5: Directions how to prepare the meal

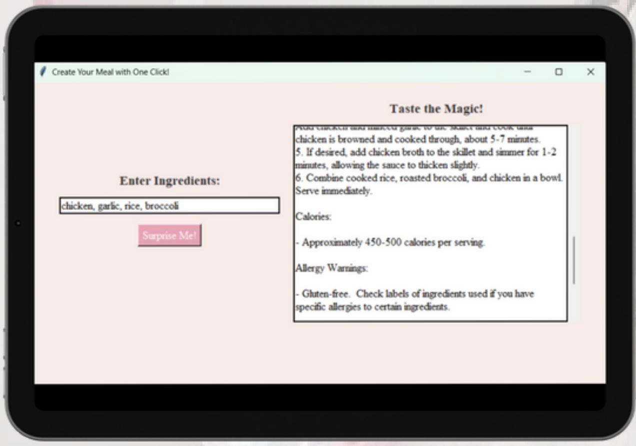

Figure 6: Generated calories and allergy warnings

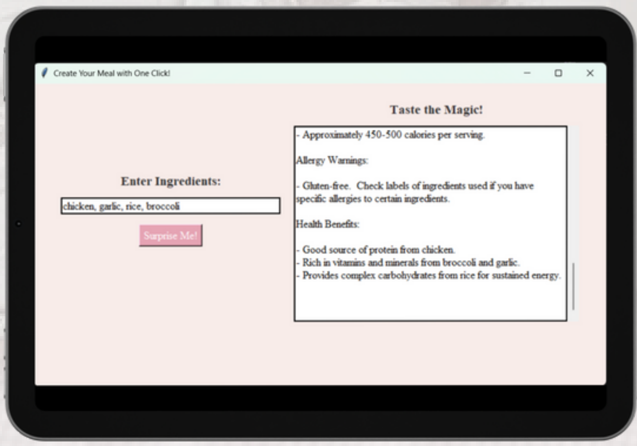

Figure 7: Finally health benefits

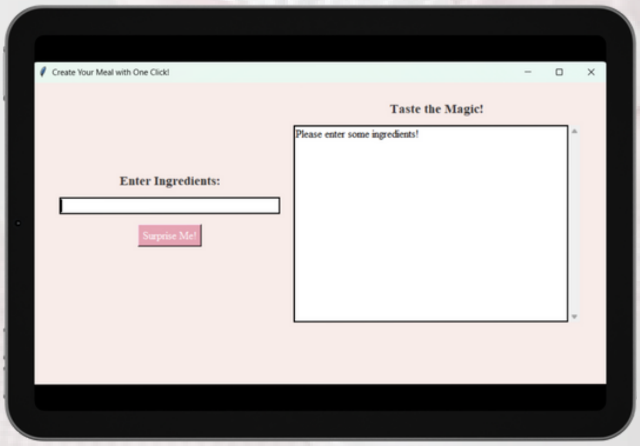

Figure 8: Error handling if the user press the button without writing any thing

Obstacles Faced

Throughout the development phase, a number of difficulties surfaced:

API Rate Limits: There are usage restrictions on the Gemini AI API, which have occasionally caused delays or unsuccessful requests. Error-handling procedures were put in place to notify the user in the event that the request was unsuccessful.

Unexpected Output Formatting: Occasionally, recipes were returned by the AI with erratic formatting. This was addressed by reworking the prompt to require a structured output with sections titled "Ingredients," "Directions," and "Health Benefits."

GUI Integration: Careful handling of multi-line text outputs was necessary to provide a seamless integration between the AI model and Tkinter. To make the text box easier to read, it was scrolled.

Conclusion

This project successfully implemented an AI-powered recipe generator, showing how AI may enhance everyday tasks like meal planning. The potential of AI in the food sector is demonstrated by the usage of Google's Gemini AI, which has an easy-to-use interface. Despite challenges like formatting errors and API constraints, strategic solutions ensured a smooth and engaging user experience. Future advancements could include expanding the AI's ability to offer dietary recommendations based on user preferences and adding the capability to create photos so that users can see the completed dish. This project establishes the foundation for further investigation into AI-powered culinary applications.