AI Recipe Generator Documentation

Overview

The **AI Recipe Generator** is a Python application built using the Streamlit framework and Google's Generative AI API. This application allows users to generate recipes based on a topic and a specified word count. Additionally, it provides a humorous touch by displaying random programming-related jokes while the recipe is being generated.

Features

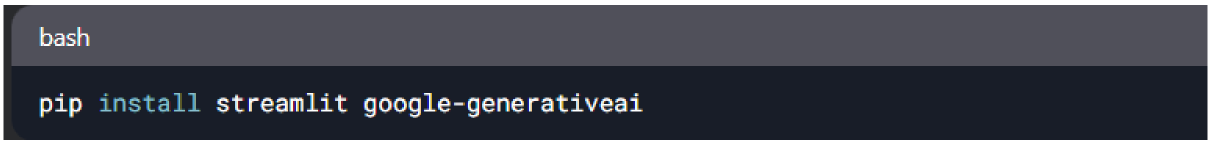
1. **Recipe Generation**: Users can input a topic and a word count to generate a recipe.
2. **Programming Jokes**: While the recipe is being generated, a random programming-related joke is displayed to keep the user entertained.
3. **User-Friendly Interface**: The application is built using Streamlit, providing a simple and intuitive interface.

Requirements

- Python 3.13.11
- Streamlit (pip install streamlit)
- Google Generative AI API (pip install google-generativeai)
- An API key for Google's Generative AI service

Installation

Install the required Python packages



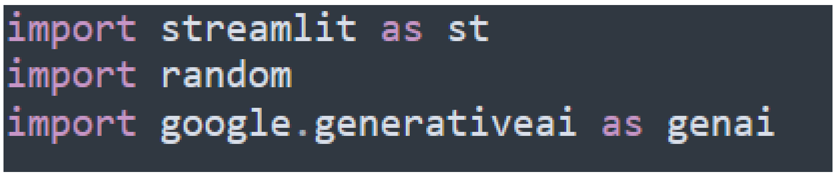
bash

```bash
pip install streamlit google-generativeai
```

Code Structure

❖ **Imports**

- **Streamlit**: Used for creating the web interface.
- **Random**: Used for selecting random jokes.
- **Google Generative AI**: Used for interacting with the Generative AI model.

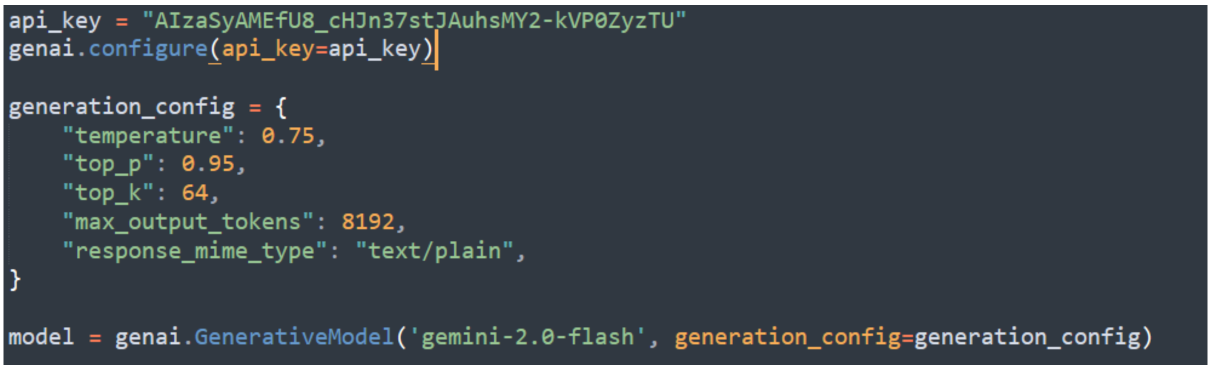


```python
import streamlit as st
import random
import google.generativeai as genai
```

API Configuration

The API key is configured, and the Generative AI model is set up with specific generation parameters.

We need to Use our own Unique API key, and we can also use other available Gemini Models Like gemini-1.0, gemini-1.5-flash, gemini-pro.



```python
api_key = "AIzaSyAMEfU8_cHJn37stJAuhsMY2-kVP0ZyzTU"
genai.configure(api_key=api_key)

generation_config = {
    "temperature": 0.75,
    "top_p": 0.95,
    "top_k": 64,
    "max_output_tokens": 8192,
    "response_mime_type": "text/plain",
}

model = genai.GenerativeModel('gemini-2.0-flash', generation_config=generation_config)
```

Joke Generation

A list of programming-related jokes is defined, and a random joke is selected and returned.

"Why Don't Programmers like Nature...!? It has too many bugs.",

"Why do JAVA Developers Wear Glasses? Because They Don't See Sharp."

"Why was the JavaScript developer sad? Because he didn't know how to 'null' his feelings.",

"Why don't programmers like nature? It has too many bugs.",

"Why do programmers prefer dark mode? Because light attracts bugs!",

"Why do Java developers wear glasses? Because they don't see sharp.", "Why was the JavaScript developer sad? Because he didn't know how to 'null' his feelings.",

"Why do Python programmers prefer using snake case? Because it's easier to read!",

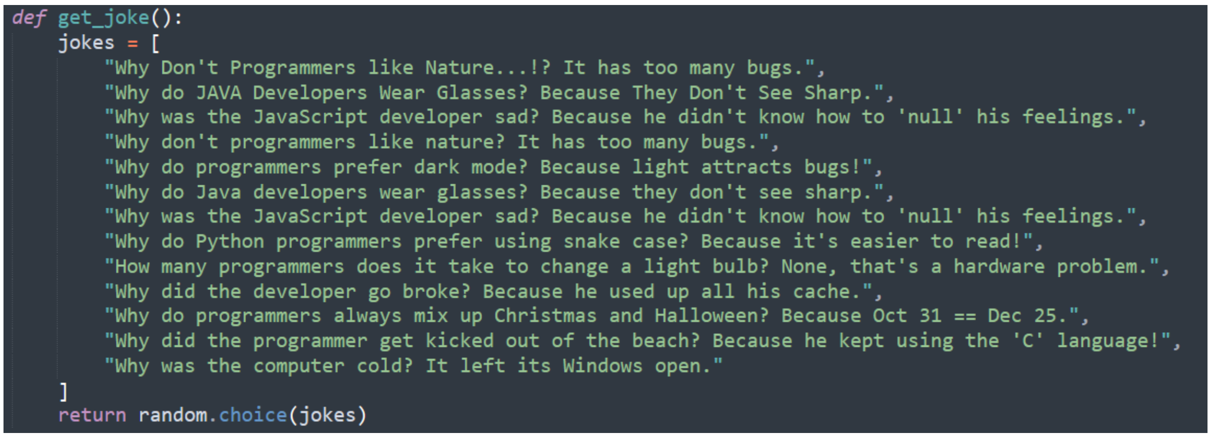
"How many programmers does it take to change a light bulb? None, that's a hardware problem.",

"Why did the developer go broke? Because he used up all his cache.",

"Why do programmers always mix up Christmas and Halloween? Because Oct 31 == Dec 25.",

"Why did the programmer get kicked out of the beach? Because he kept using the 'C' language!",

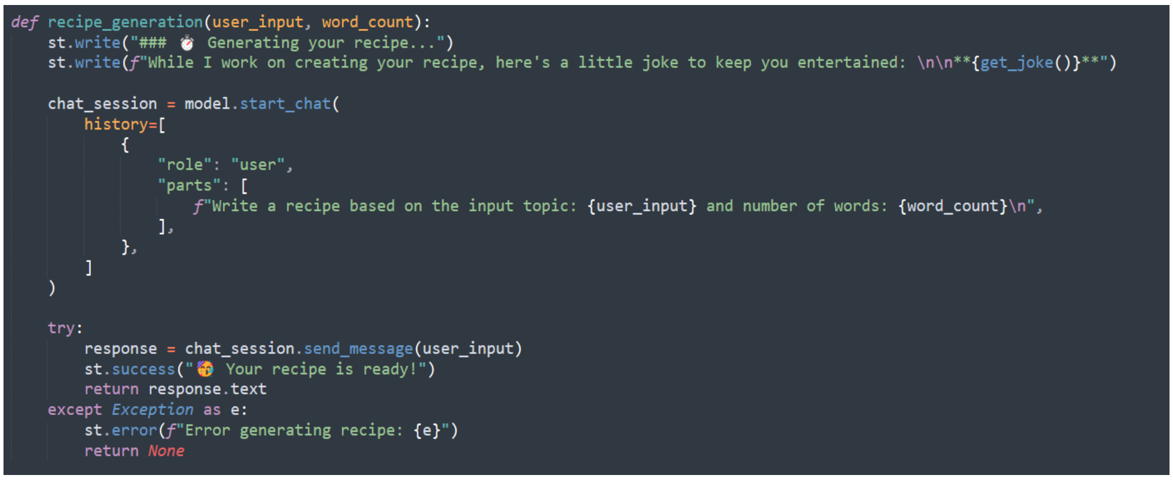
"Why was the computer cold? It left its Windows open."



```python
def get_joke():
    jokes = [
        "Why Don't Programmers like Nature...!? It has too many bugs.",
        "Why do JAVA Developers Wear Glasses? Because They Don't See Sharp.",
        "Why was the JavaScript developer sad? Because he didn't know how to 'null' his feelings.",
        "Why don't programmers like nature? It has too many bugs.",
        "Why do programmers prefer dark mode? Because light attracts bugs!",
        "Why do Java developers wear glasses? Because they don't see sharp.",
        "Why was the JavaScript developer sad? Because he didn't know how to 'null' his feelings.",
        "Why do Python programmers prefer using snake case? Because it's easier to read!",
        "How many programmers does it take to change a light bulb? None, that's a hardware problem.",
        "Why did the developer go broke? Because he used up all his cache.",
        "Why do programmers always mix up Christmas and Halloween? Because Oct 31 == Dec 25.",
        "Why did the programmer get kicked out of the beach? Because he kept using the 'C' language!",
        "Why was the computer cold? It left its Windows open."
    ]
    return random.choice(jokes)
```

Recipe Generation

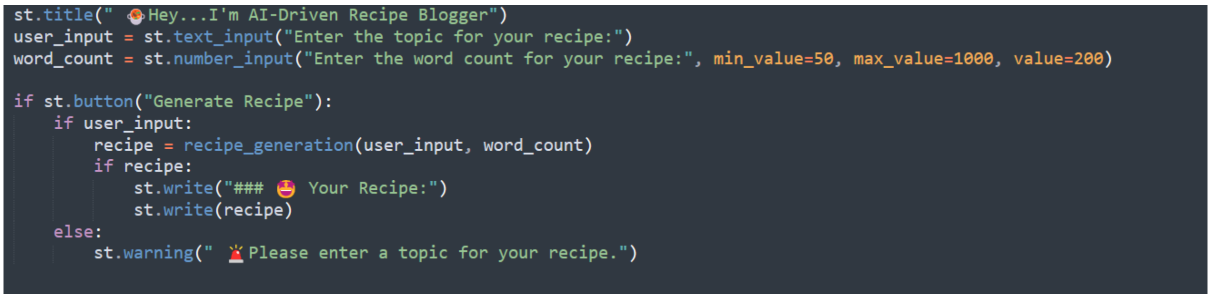
The recipe_generation function takes user input and word count, starts a chat session with the Generative AI model, and returns the generated recipe.



```python
def recipe_generation(user_input, word_count):
    st.write("### ⏱️ Generating your recipe...")
    st.write(f"While I work on creating your recipe, here's a little joke to keep you entertained: \n\n**{get_joke()}**")

    chat_session = model.start_chat(
        history=[
            {
                "role": "user",
                "parts": [
                    f"Write a recipe based on the input topic: {user_input} and number of words: {word_count}\n",
                ],
            },
        ]
    )

    try:
        response = chat_session.send_message(user_input)
        st.success("🥳 Your recipe is ready!")
        return response.text
    except Exception as e:
        st.error(f"Error generating recipe: {e}")
        return None
```

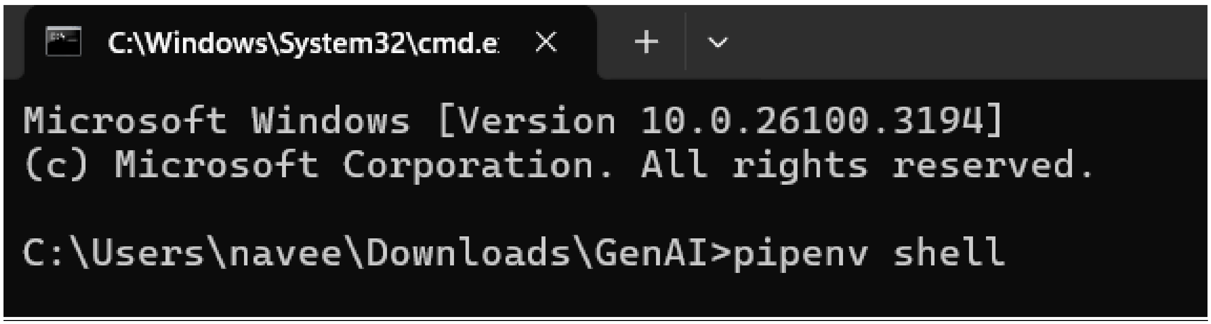
Streamlit Interface

The Streamlit interface is set up with a title, text input for the recipe topic, and a number input for the word count. A button triggers the recipe generation process.



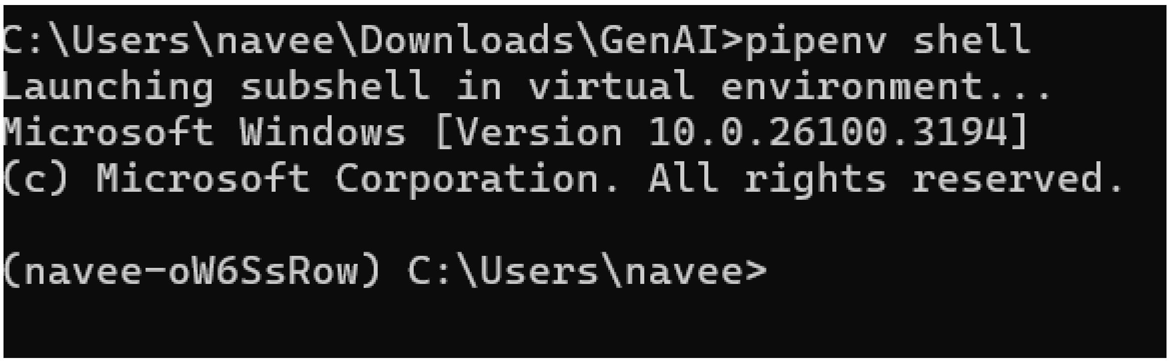
```python
st.title(" 🍲Hey...I'm AI-Driven Recipe Blogger")
user_input = st.text_input("Enter the topic for your recipe:")
word_count = st.number_input("Enter the word count for your recipe:", min_value=50, max_value=1000, value=200)

if st.button("Generate Recipe"):
    if user_input:
        recipe = recipe_generation(user_input, word_count)
        if recipe:
            st.write("### 🤩 Your Recipe:")
            st.write(recipe)
    else:
        st.warning(" 🚨Please enter a topic for your recipe.")
```

How to Run the Code:



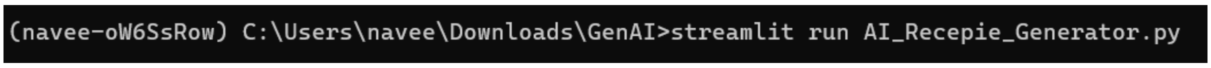
```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\navee\Downloads\GenAI>pipenv shell
```

After running the above a Virtual Environment will be created.



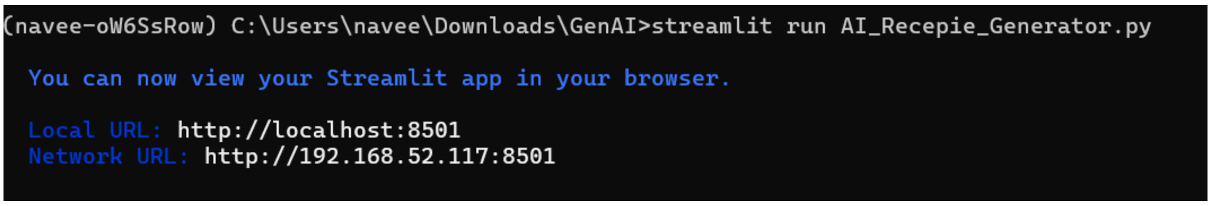
```
C:\Users\navee\Downloads\GenAI>pipenv shell
Launching subshell in virtual environment...
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

(navee-oW6SsRow) C:\Users\navee>
```

To get the Output we need to Run the Following command



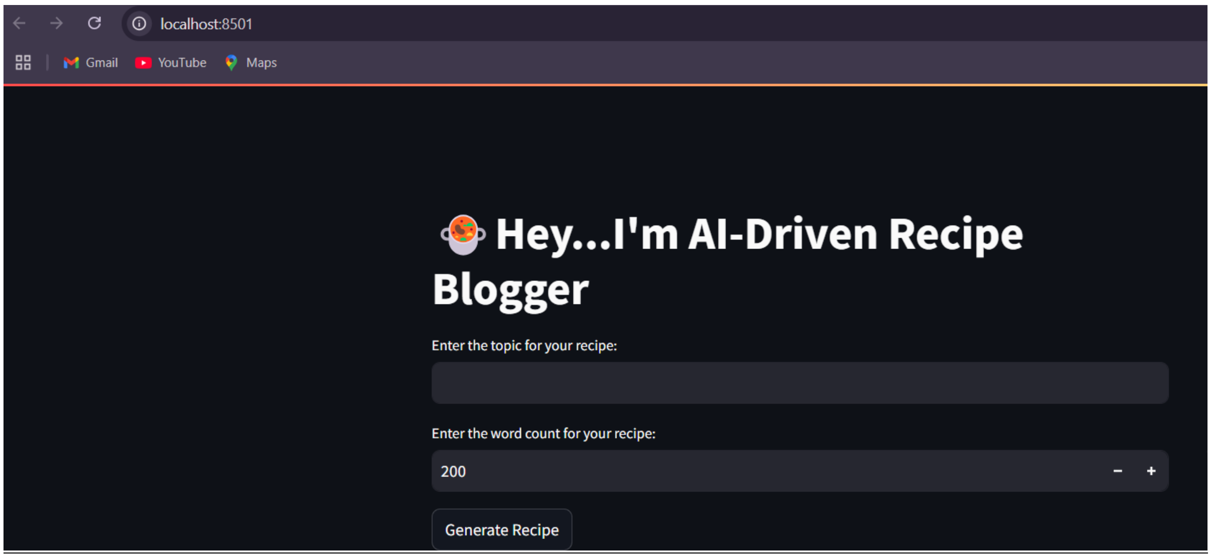
```
(navee-oW6SsRow) C:\Users\navee\Downloads\GenAI>streamlit run AI_Recepie_Generator.py
```

Now we can view our Streamlit Application in our Web Browser



```
(navee-oW6SsRow) C:\Users\navee\Downloads\GenAI>streamlit run AI_Recepie_Generator.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.52.117:8501
```

Final Output:


localhost:8501

Gmail YouTube Maps

Hey...I'm AI-Driven Recipe Blogger

Enter the topic for your recipe:

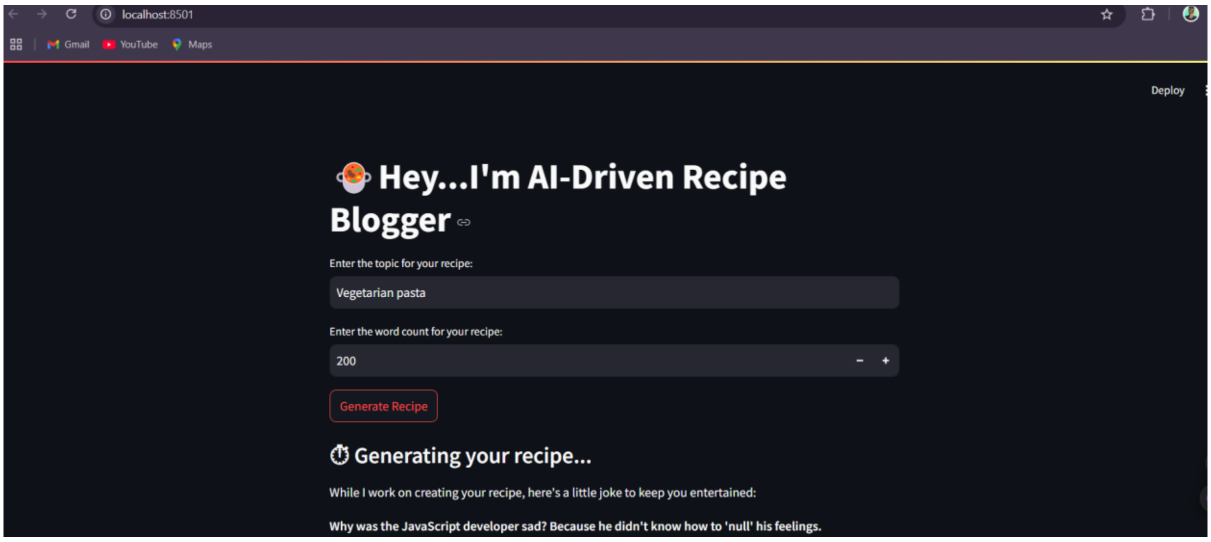
Enter the word count for your recipe:

200

Generate Recipe

Input:

- Topic: "Vegetarian pasta"
- Word Count: 200


localhost:8501

Gmail YouTube Maps

Deploy

Hey...I'm AI-Driven Recipe Blogger

Enter the topic for your recipe:

Vegetarian pasta

Enter the word count for your recipe:

200

Generate Recipe

⏱ Generating your recipe...

While I work on creating your recipe, here's a little joke to keep you entertained:

Why was the JavaScript developer sad? Because he didn't know how to 'null' his feelings.

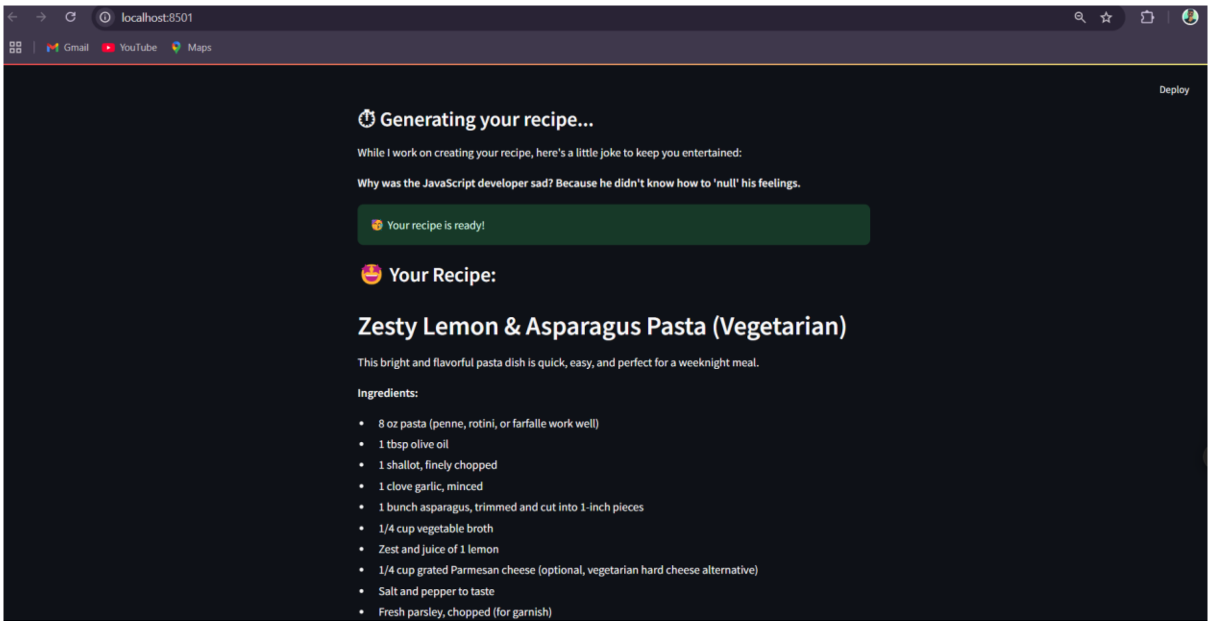
It Shows a Prompt Like Below

Generating your recipe...

While I work on creating your recipe, here's a little joke to keep you entertained:

Why was the JavaScript developer sad? Because he didn't know how to 'null' his feelings.

Now we Can Finally get the Output of our Mention Recepie.



⏱ Generating your recipe...

While I work on creating your recipe, here's a little joke to keep you entertained:

Why was the JavaScript developer sad? Because he didn't know how to 'null' his feelings.

Your recipe is ready!

🤩 Your Recipe:

Zesty Lemon & Asparagus Pasta (Vegetarian)

This bright and flavorful pasta dish is quick, easy, and perfect for a weeknight meal.

Ingredients:

- 8 oz pasta (penne, rotini, or farfalle work well)
- 1 tbsp olive oil
- 1 shallot, finely chopped
- 1 clove garlic, minced
- 1 bunch asparagus, trimmed and cut into 1-inch pieces
- 1/4 cup vegetable broth
- Zest and juice of 1 lemon
- 1/4 cup grated Parmesan cheese (optional, vegetarian hard cheese alternative)
- Salt and pepper to taste
- Fresh parsley, chopped (for garnish)