Model Development Phase Template

Date	26 August 2024
Project Title	Nutrition App Using Gemini Pro : Your Comprehensive Guide to Healthy Eating and Well-being
Maximum Marks	4 Marks

Initial Model Training Code

In this initial model training code, we will focus on developing a foundation for our health and wellness app that integrates two key features: Nutrition Insights and Goal Setting. The Nutrition Insights feature provides users with a detailed analysis of their macronutrient and micronutrient intake, offering suggestions for improvement. The Goal Setting feature enables users to set and track their health and wellness objectives, such as weight loss, muscle gain, or endurance improvement. By combining these features, our app will empower users to make informed decisions about their diet and exercise routine, ultimately driving them towards their health and wellness goals.

Importing Libraries and Setting Up Environment:

```
import streamlit as st
import os
from PIL import Image
import google.generativeai as genai
from dotenv import load_dotenv
```

Linking Google API Key

```
genai.configure(api_key=os.getenv("GOOGLE_API_KEY"))
```

Defining Functions

```
def get_gemini_response(input, image, prompt):
    response = model.generate_content([input, image[0], prompt])
    return response.text

def input_image_details(uploaded_file):
    if uploaded_file is not None:
        bytes_data = uploaded_file.getvalue()
        image_parts = [
            {
                "mime_type": uploaded_file.type,
                "data": bytes_data
            }
        ]
        return image_parts
    else:
        raise FileNotFoundError("No file uploaded")
```

Creating the Streamlit Interface

```
st.set_page_config(page_title="Food Scan")
st.header('Food Scan with Google Gemini')

input = st.text_input("Input prompt: ", key='input')
uploaded_file = st.file_uploader("Choose an image of the food or food table", type=["jpg", 'jpeg', 'png'])
image = ""
```

Processing the Uploaded Image

```
if uploaded_file is not None:
    image = Image.open(uploaded_file)
    st.image(image, caption="Uploaded Image", use_column_width=True)
```

Defining the Submit Button and Processing the Input

```
submit = st.button("Scan the Food(s)")

input_prompt = """You have to identify different types of food in images. The system should accurately detect and label various foods displ

if submit:
    image_data = input_image_details(uploaded_file)
    response = get_gemini_response(input_prompt, image_data, input)
    st.subheader("Food Scan report: ")
    st.write(response)
```

Overall, the code creates a Streamlit interface that allows users to upload an image file and enter a prompt. When the submit button is clicked, the code processes the uploaded image file, generates a response from the Gemini model, and displays the response in the Streamlit interface.