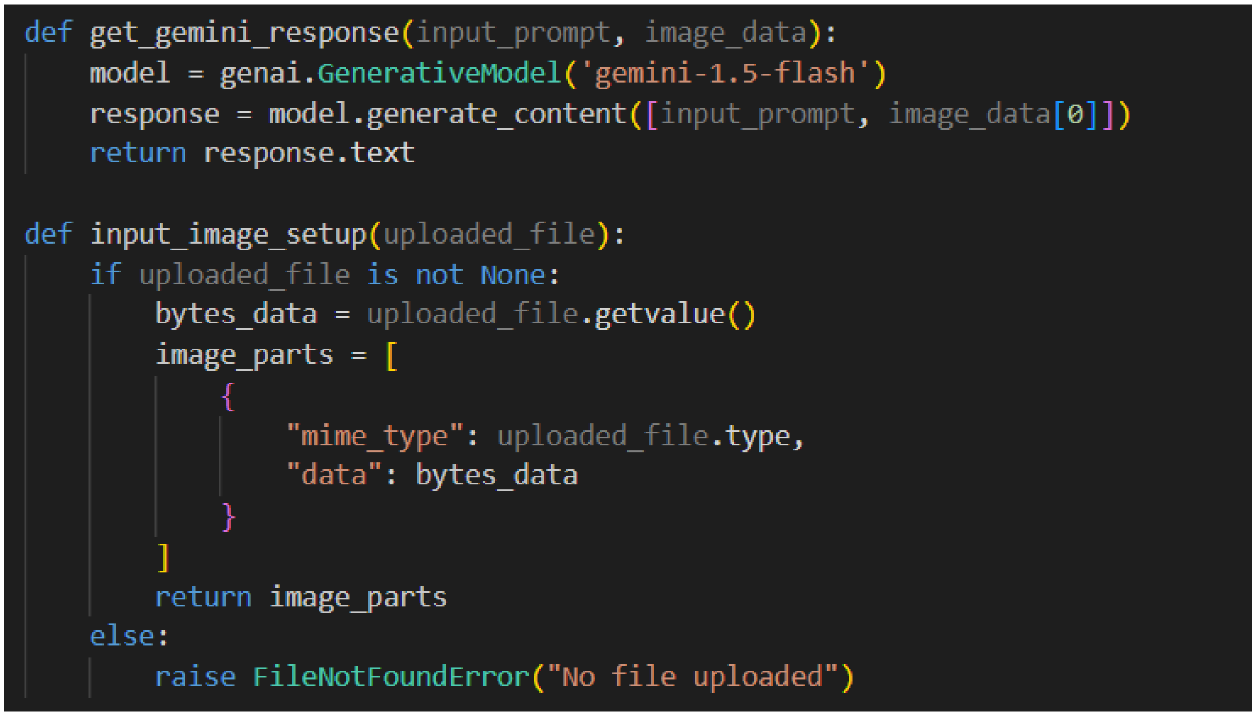
Model Development Phase Template

Date	16 July 2024
Team ID	SWTID1720159923
Project Title	Nutrition App Using Gemini Pro : Your Comprehensive Guide To Healthy Eating And Well-Being
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code:



```python
def get_gemini_response(input_prompt, image_data):
    model = genai.GenerativeModel('gemini-1.5-flash')
    response = model.generate_content([input_prompt, image_data[0]])
    return response.text

def input_image_setup(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")
```

Overview

This project sets up a Streamlit application that allows users to upload an image of their meal, enter a prompt, and receive nutritional information generated by Google's Generative AI model. The app also features a custom background and various CSS styles.

Main Functions

1. Loading Environment Variables

Purpose: Loads environment variables from a .env file, particularly the Google API key.

Key Function: load_dotenv() reads the .env file and loads the environment variables into the application.

2. Configuring Google Generative AI

Purpose: Configures the Google Generative AI model using the API key retrieved from the environment variables.

Key Functions:

os.getenv("GOOGLE_API_KEY"): Retrieves the API key from environment variables.

genai.configure(api_key=google_api_key): Configures the generative AI model with the API key.

st.error(): Displays an error message if the API key is not found.

3. Setting Up Background Image

Purpose: Sets a custom background image for the Streamlit app.

Key Functions:

base64.b64encode(): Encodes the image file into a base64 string.

st.markdown(): Applies custom CSS to the Streamlit app, setting the background image.

4. Generating AI Response

Purpose: Generates content using Google's Generative AI model based on the input prompt and image

Key Functions:

genai.GenerativeModel('gemini1.5flash'): Initializes the generative AI model.

model.generate_content([input_prompt, image_data[0]]): Generates

content using the model.

response.text: Extracts the generated text from the response.

5. Handling Image Upload

Purpose: Processes the uploaded image file and prepares it for use with the AI model.

Key Functions:

uploaded_file.getvalue(): Retrieves the raw bytes of the uploaded file.

Constructs a list of image parts with the MIME type and data of the image.

Raises a FileNotFoundError if no file is uploaded.

6.Setting Up Streamlit App

Purpose: Configures the page title of the Streamlit app.

Key Function:st.set_page_config(): Sets the page title and other configurations for the Streamlit app.

7. Custom CSS Styles

Purpose: Defines and applies custom CSS styles to various elements in the Streamlit app.

Key Function: st.markdown() : Applies the CSS styles to the app.

8. Layout and User Input

Purpose: Defines the layout of the app, including columns, titles, headers, and input fields for prompts and file uploads.

Key Functions:

st.markdown() : Adds styled text elements to the app.

st.text_area() : Provides a text area for user input.

st.file_uploader() : Provides a file uploader for image uploads.

Image.open() : Opens the uploaded image.

st.image() : Displays the uploaded image in the app.

9. Submission and Response Handling

Purpose: Handles the submission of user inputs and displays the generated response.

Key Functions:

st.button(): Creates a submission button.

st.error(): Displays error messages.

st.spinner(): Displays a spinner while processing.

st.subheader(): Adds a subheader to the app.

st.write(): Displays the generated response text.

Summary

The project creates a Streamlit app where users can upload an image of their meal and enter a prompt to get nutritional information generated by Google's Generative AI model. The app includes custom styling, error handling, and realtime feedback to enhance user experience.