Nutrition App Using Gemini Pro : Your Comprehensive Guide To Healthy Eating And Well-Being

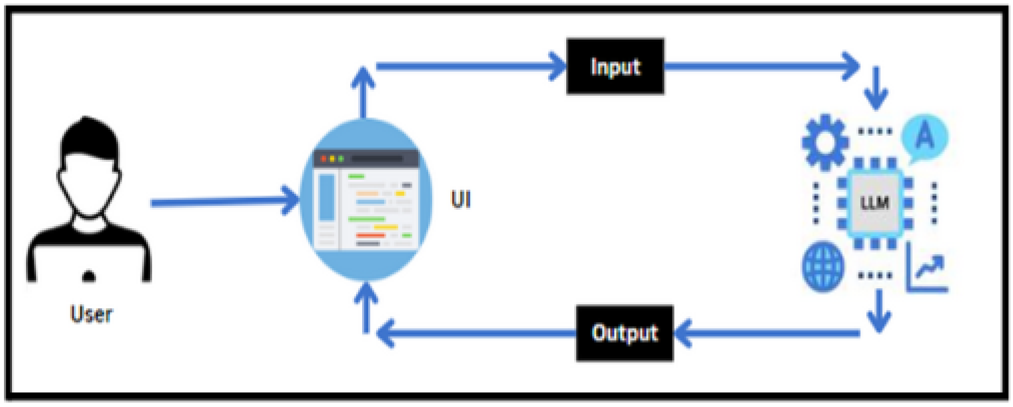
Nutritionist AI is an innovative mobile application designed to provide personalized dietary recommendations and nutritional advice using the advanced capabilities of the Gemini Pro model. The app leverages artificial intelligence to analyze user data, dietary preferences, and health goals, delivering tailored meal plans, nutritional insights, and wellness tips. The primary aim of Nutritionist AI is to promote healthier eating habits and improve overall well-being through intelligent and data-driven recommendations.

Scenario 1: Weight Loss Journey
Sarah, a 28-year-old with a goal to lose 15 pounds, uses Nutritionist AI to aid her in her weight loss journey. As a vegetarian with a moderate activity level, she inputs her dietary preferences and health goals into the app. Nutritionist AI creates a calorie-controlled, nutrient-dense meal plan tailored to her vegetarian diet. Sarah logs her meals by taking photos or scanning barcodes, and the app provides feedback on her calorie intake and nutritional balance, suggesting necessary adjustments. By syncing her fitness tracker, the app integrates her physical activity data, offering comprehensive insights to help Sarah stay on track with her weight loss while maintaining proper nutrition.

Scenario 2: Managing Diabetes
John, a 45-year-old with Type 2 Diabetes, relies on Nutritionist AI to manage his condition through diet. He inputs his low-carb dietary preference and diabetes condition, and the app generates meal plans that focus on low carbohydrate and high fiber content to help control his blood sugar levels. John uses the app to log his meals, receiving immediate feedback on their suitability for his diabetes management. Detailed nutritional breakdowns highlight carbohydrate content and glycemic index, aiding John in making informed food choices. Additionally, the app provides educational resources about managing diabetes through diet, keeping John well-informed and empowered to handle his condition better.

Scenario 3: Building the Muscle
Emily, a 30-year-old strength training enthusiast, uses Nutritionist AI to support her goal of gaining muscle mass. With a preference for high-protein meals and an intense workout regime, she inputs her dietary preferences and fitness goals into the app. Nutritionist AI generates meal plans rich in protein and essential nutrients necessary for muscle growth. Emily benefits from a variety of high-protein recipes that cater to her needs, with each recipe including detailed instructions and nutritional information. By connecting her fitness tracker, the app accounts for her caloric expenditure and provides insights on balancing her protein intake with her workouts, optimizing her muscle-building efforts.

Technical Architecture



Project Flow

- User interacts with the UI to enter the input.
- User input is collected from the UI and transmitted to the backend using the Google API key.
- The input is then forwarded to the Gemini Pro pre-trained model via an API call.
- The Gemini Pro pre-trained model processes the input and generates the output.
- The results are returned to the frontend for formatting and display.

To accomplish this, we have to complete all the activities listed below:

- Requirements Specification
    - Create a requirements.txt file to list the required libraries.
    - Install the required libraries
- Initialization of Google API Key
    - Generate Google API Key
    - Initialize Google API Key
- Interfacing with Pre-trained Model
    - Load the Gemini Pro pre-trained model
    - Implement a function to get gemini response
    - Implement a function to read PDF content
    - Write a prompt for gemini model
- Model Deployment
    - Integrate with Web Framework
    - Host the Application

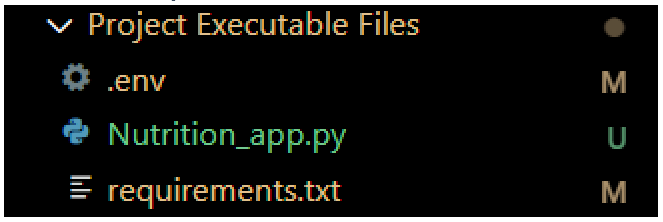
Prior Knowledge

You must have the prior knowledge of the following topics to complete this project.

- Generative AI Concepts
- NLP: https://www.tutorialspoint.com/natural_language_processing/index.htm
- Generative AI: https://en.wikipedia.org/wiki/Generative_artificial_intelligence
- About Gemini: https://deepmind.google/technologies/gemini/#introduction
- Gemini API: https://ai.google.dev/gemini-api/docs/get-started/python
- Gemini Demo: https://colab.research.google.com/github/google/generative-ai-docs/blob/main/site/en/gemini-api/docs/get-started/python.ipynb
- Streamlit: https://www.geeksforgeeks.org/a-beginners-guide-to-streamlit/

Project Structure

Create the Project folder which contains files as shown below:



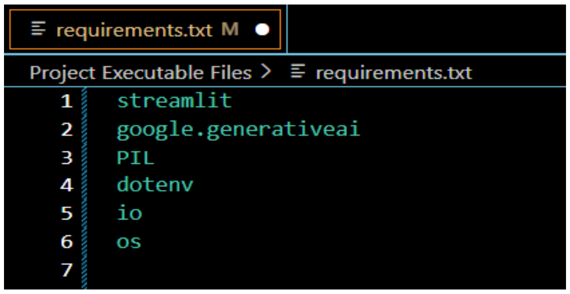
```
Project Executable Files
    .env                 M
    Nutrition_app.py     U
    requirements.txt     M
```

- .env file: It securely stores the Google API key.
- Nutrition_app.py: It serves as the primary application file housing both the model and Streamlit UI code.
- requirements.txt: It enumerates the libraries necessary for installation to ensure proper functioning.
- Additionally, ensure proper file organization and adhere to best practices for version control.

Milestone1 : Requirements Specification

Specifying the required libraries in the requirements.txt file ensures seamless setup and reproducibility of the project environment, making it easier for others to replicate the development environment.

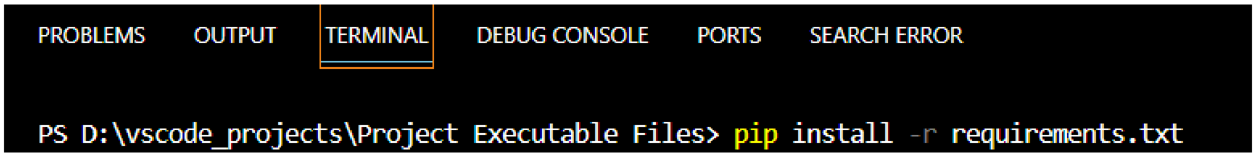
Activity 1 : Create A Requirements.Txt File To List The Required Libraries.



```
streamlit
google.generativeai
PIL
dotenv
io
os
```

- streamlit: Streamlit is a powerful framework for building interactive web applications with Python.
- streamlit_extras: Additional utilities and enhancements for Streamlit applications.
- google-generativeai: Python client library for accessing the GenerativeAI API, facilitating interactions with pre-trained language models like Gemini Pro.
- python-dotenv: Python-dotenv allows you to manage environment variables stored in a .env file for your Python projects.
- io: It is a Python library for I/0 related functionalities
- os: python library to work os related functionalities incuding the .env path variable

Activity 2 : Install The Required Libraries



```
PS D:\vscode_projects\Project Executable Files> pip install -r requirements.txt
```

- Open the terminal.
- Run the command: pip install -r requirements.txt
- This command installs all the libraries listed in the requirements.txt file

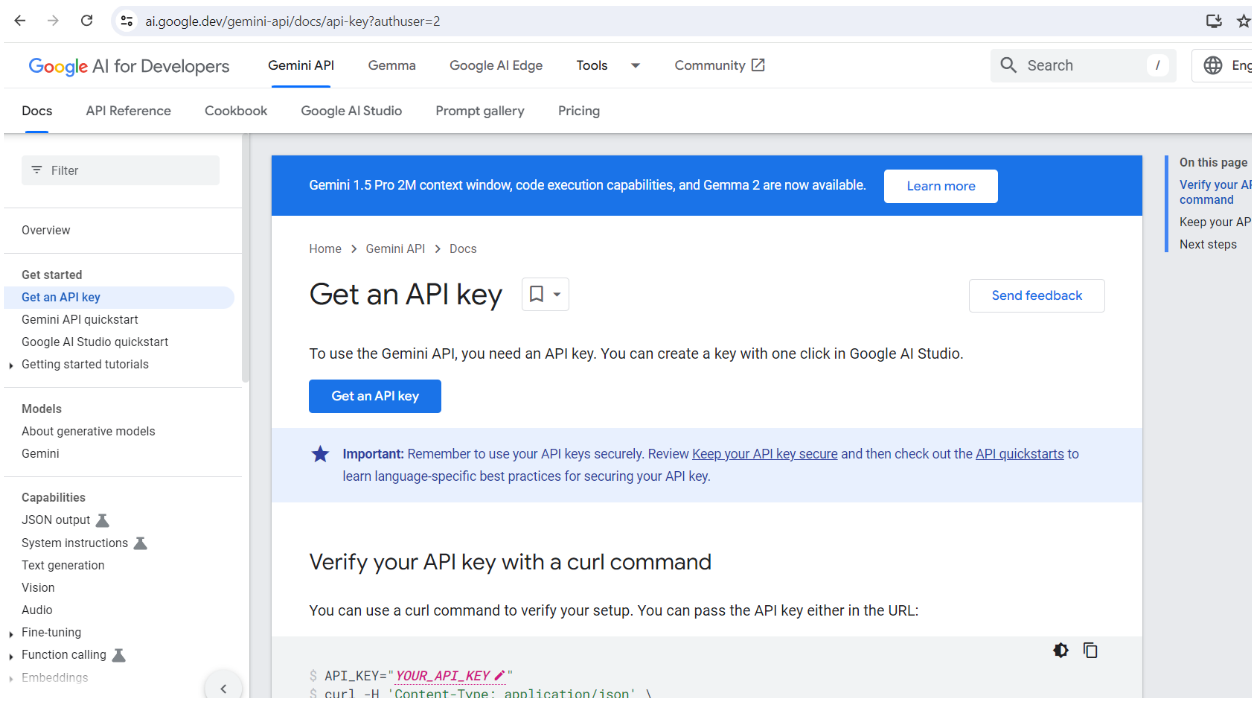
Milestone 2 : Initialization Of Google API Key

The Google API key is a secure access token provided by Google, enabling developers to authenticate and interact with various Google APIs. It acts as a form of identification, allowing users to access specific Google services and resources. This key plays a crucial role in authorizing and securing API requests, ensuring that only authorized users can access and utilize Google's services.

Activity 1 : Generate Google API Key

Click the provided link to access the following webpage.

Link: https://ai.google.dev/gemini-api/docs/api-key



Gemini 1.5 Pro 2M context window, code execution capabilities, and Gemma 2 are now available.

Get an API key

To use the Gemini API, you need an API key. You can create a key with one click in Google AI Studio.

Get an API key

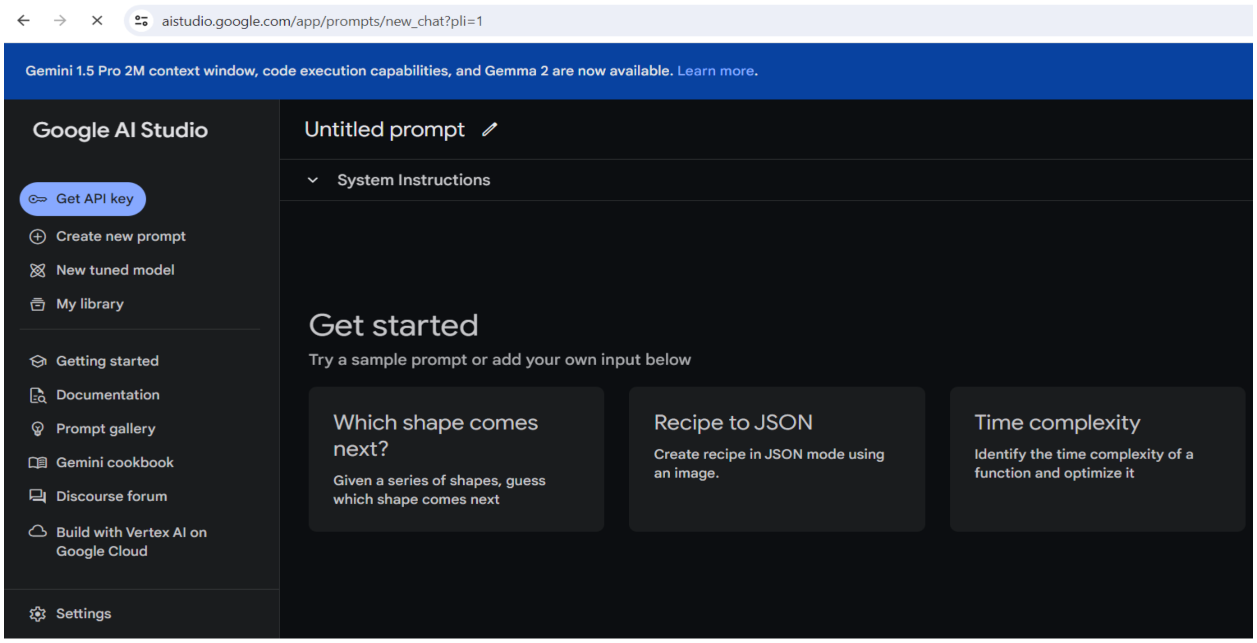
Important: Remember to use your API keys securely. Review Keep your API key secure and then check out the API quickstarts to learn language-specific best practices for securing your API key.

Verify your API key with a curl command

You can use a curl command to verify your setup. You can pass the API key either in the URL:

```
$ API_KEY="YOUR_API_KEY"
$ curl -H 'Content-Type: application/json' \
```

After signing in to your account, navigate to the 'Get an API Key' option. Clicking on this option will redirect you to another webpage as shown below.



Gemini 1.5 Pro 2M context window, code execution capabilities, and Gemma 2 are now available. Learn more.

Get started

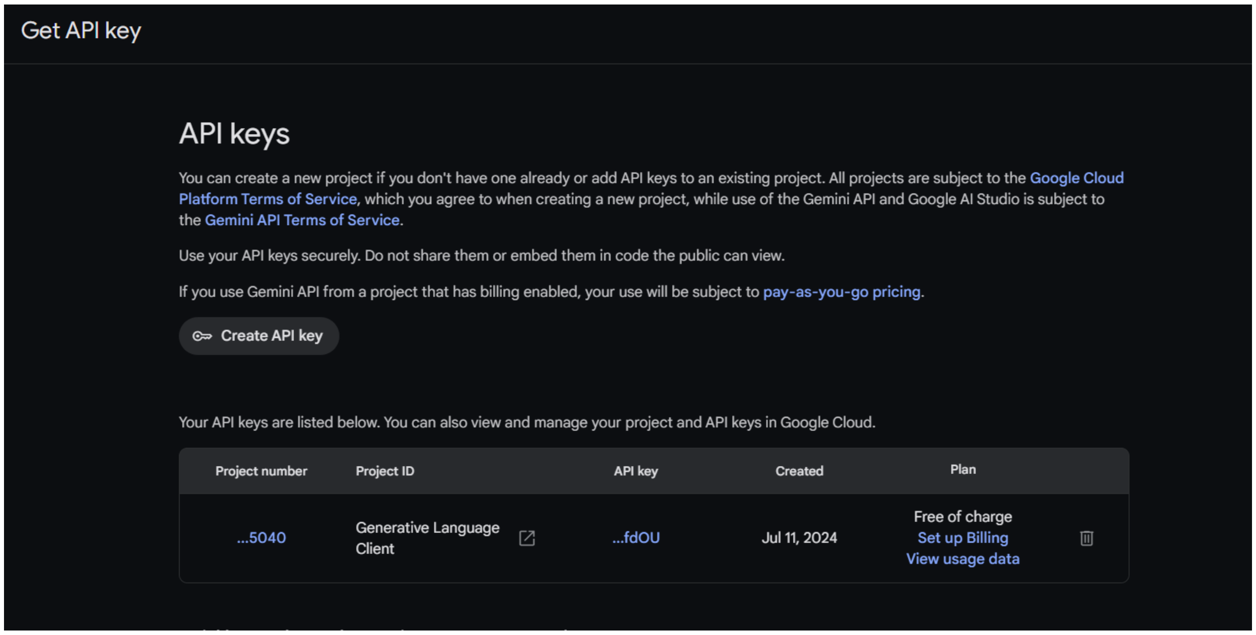
Try a sample prompt or add your own input below

Which shape comes next?
Given a series of shapes, guess which shape comes next

Recipe to JSON
Create recipe in JSON mode using an image.

Time complexity
Identify the time complexity of a function and optimize it

Next, click on 'Create API Key' and choose the generative language client as the project. Then, select 'Create API key in existing project'.


Get API key

API keys

You can create a new project if you don't have one already or add API keys to an existing project. All projects are subject to the Google Cloud Platform Terms of Service, which you agree to when creating a new project, while use of the Gemini API and Google AI Studio is subject to the Gemini API Terms of Service.

Use your API keys securely. Do not share them or embed them in code the public can view.

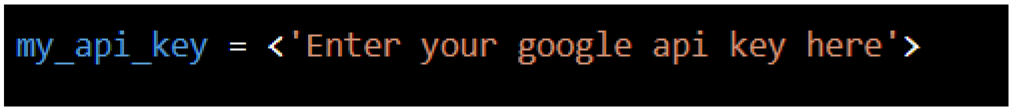
If you use Gemini API from a project that has billing enabled, your use will be subject to pay-as-you-go pricing.

Your API keys are listed below. You can also view and manage your project and API keys in Google Cloud.

Project number	Project ID	API key	Created	Plan
...5040	Generative Language Client	...fdOU	Jul 11, 2024	Free of charge Set up Billing View usage data

Copy the newly generated API key as it is required for loading the Gemini Pro pre-trained model.

Activity 2 : Initialize Google API Key



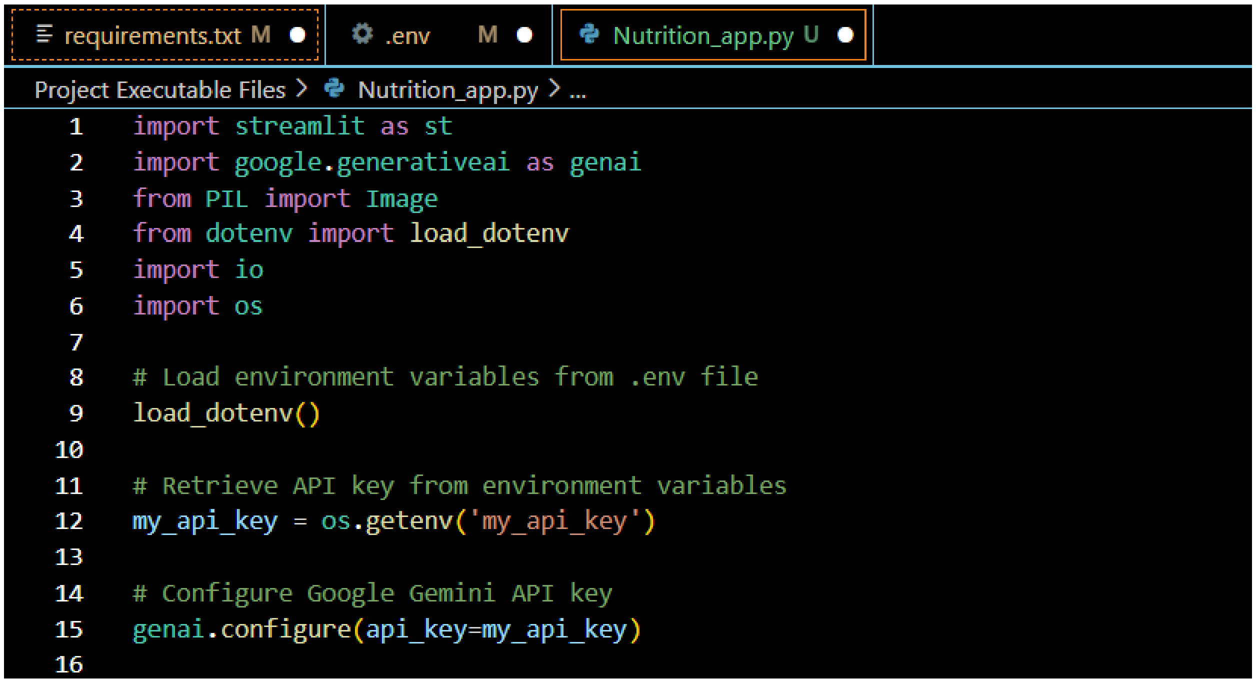
```
my_api_key = <'Enter your google api key here'>
```

- Create a .env file and define a variable named my_api_key.
- Assign the copied Google API key to this variable.
- Paste the API key obtained from the previous steps here.

Milestone 3 : Interfacing With Pre-Trained Model

To interface with the pre-trained model, we'll start by creating an app.py file, which will contain both the model and Streamlit UI code.

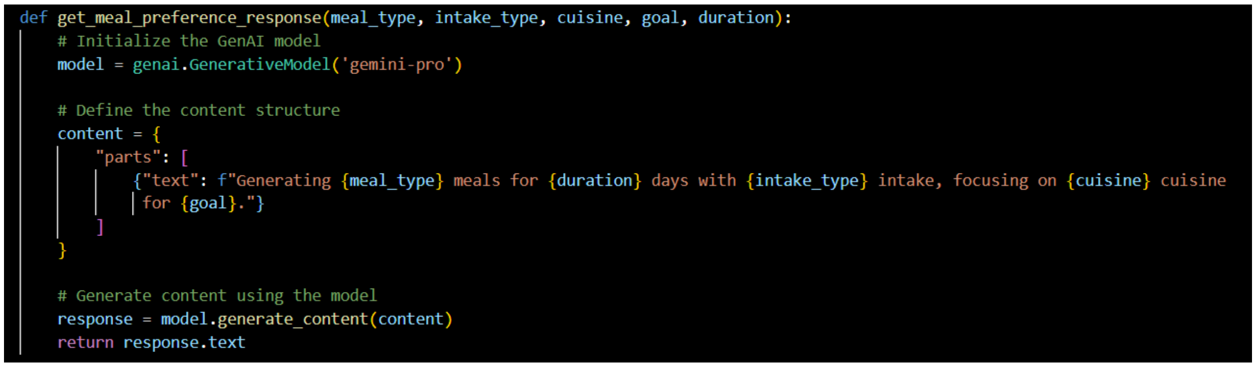
Activity 1 : Load The Gemini Pro API



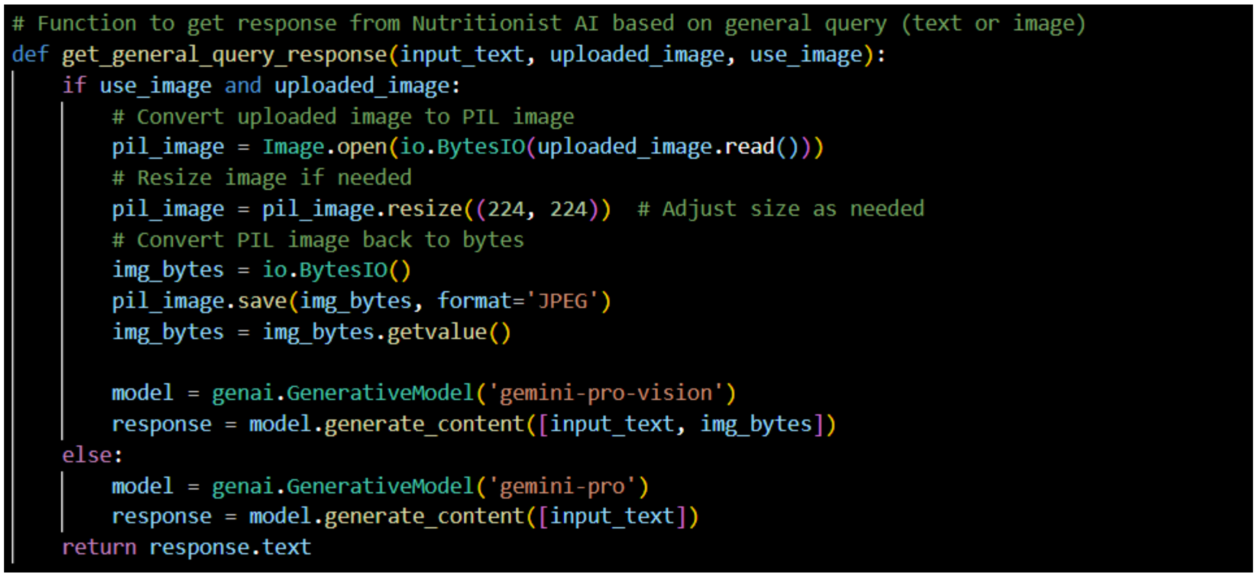
```python
import streamlit as st
import google.generativeai as genai
from PIL import Image
from dotenv import load_dotenv
import io
import os

# Load environment variables from .env file
load_dotenv()

# Retrieve API key from environment variables
my_api_key = os.getenv('my_api_key')

# Configure Google Gemini API key
genai.configure(api_key=my_api_key)

```

This code snippet is for initializing a nutrition app application using Streamlit, an open-source app framework, and Google Generative AI services. The script starts by loading environment variables from a .env file using the load_dotenv() function from the dotenv package. It then imports necessary libraries: streamlit for creating the web app interface, os for accessing environment variables, google.generativeai for utilizing Google's Generative AI capabilities, and PIL.Image for image processing. The genai.configure() function is called to set up the Google Generative AI API with the API key retrieved from the environment variables, ensuring secure and authorized access to the AI services.

Activity 2 : Implement A Function To Get Gemini Response



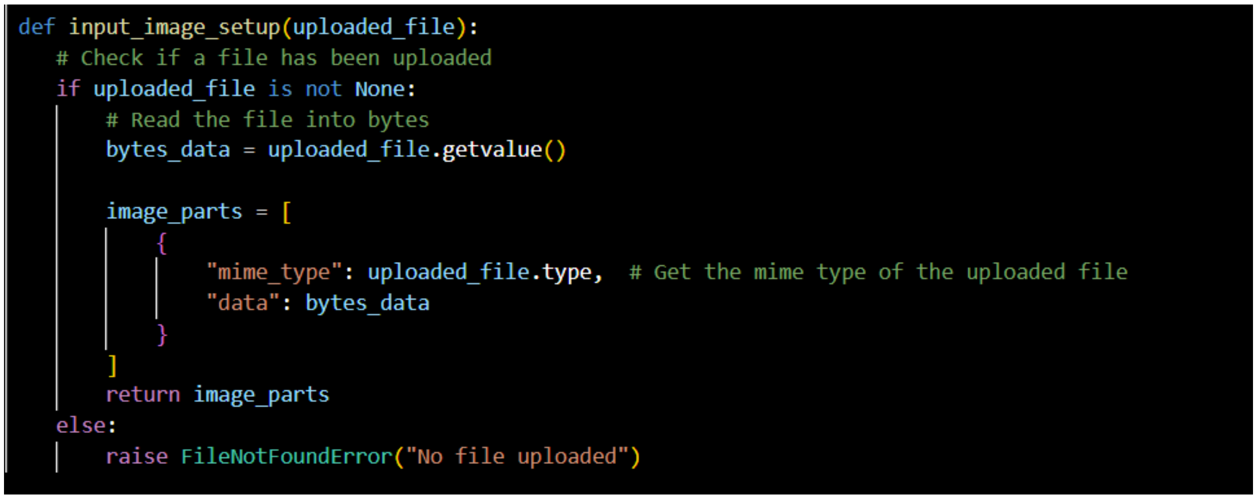
```python
def get_meal_preference_response(meal_type, intake_type, cuisine, goal, duration):
    # Initialize the GenAI model
    model = genai.GenerativeModel('gemini-pro')

    # Define the content structure
    content = {
        "parts": [
            {"text": f"Generating {meal_type} meals for {duration} days with {intake_type} intake, focusing on {cuisine} cuisine
             for {goal}."}
        ]
    }

    # Generate content using the model
    response = model.generate_content(content)
    return response.text
```



```
# Function to get response from Nutritionist AI based on general query (text or image)
def get_general_query_response(input_text, uploaded_image, use_image):
    if use_image and uploaded_image:
        # Convert uploaded image to PIL image
        pil_image = Image.open(io.BytesIO(uploaded_image.read()))
        # Resize image if needed
        pil_image = pil_image.resize((224, 224))  # Adjust size as needed
        # Convert PIL image back to bytes
        img_bytes = io.BytesIO()
        pil_image.save(img_bytes, format='JPEG')
        img_bytes = img_bytes.getvalue()

        model = genai.GenerativeModel('gemini-pro-vision')
        response = model.generate_content([input_text, img_bytes])
    else:
        model = genai.GenerativeModel('gemini-pro')
        response = model.generate_content([input_text])
    return response.text
```

- The function get_meal_preference takes multiple input text as a parameter and sends the response accordingly
- It calls the generate_content method of the model object to generate a response.
- The generated response is returned as text.
- in get_general_query_response there are 2 models based on whether the user wants to give text or image as input criteria based on it gemini pro or gemini pro vision models are loaded

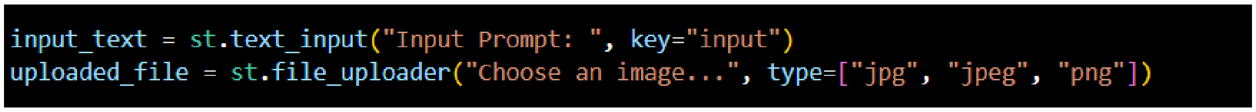
Activity 3 : Implement A Function To Read The Image And Set The Image Format For Gemini Pro Model Input



```
def input_image_setup(uploaded_file):
   # Check if a file has been uploaded
   if uploaded_file is not None:
       # Read the file into bytes
       bytes_data = uploaded_file.getvalue()

       image_parts = [
           {
               "mime_type": uploaded_file.type,  # Get the mime type of the uploaded file
               "data": bytes_data
           }
       ]
       return image_parts
   else:
       raise FileNotFoundError("No file uploaded")
```

The function input_image_setup processes an uploaded image file for a health management application. It first checks if a file has been uploaded. If a file is present, it reads the file's content into bytes and creates a dictionary containing the file's MIME type and its byte data. This dictionary is then stored in a list named image_parts, which is returned by the function. If no file is uploaded, the function raises a FileNotFoundError, indicating that an image file is required but not provided. This setup ensures that the uploaded image is correctly formatted and ready for further processing or analysis in the application.

Activity 4 : Write A Prompt For Gemini Model



```
input_text = st.text_input("Input Prompt: ", key="input")
uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg", "png"])
```

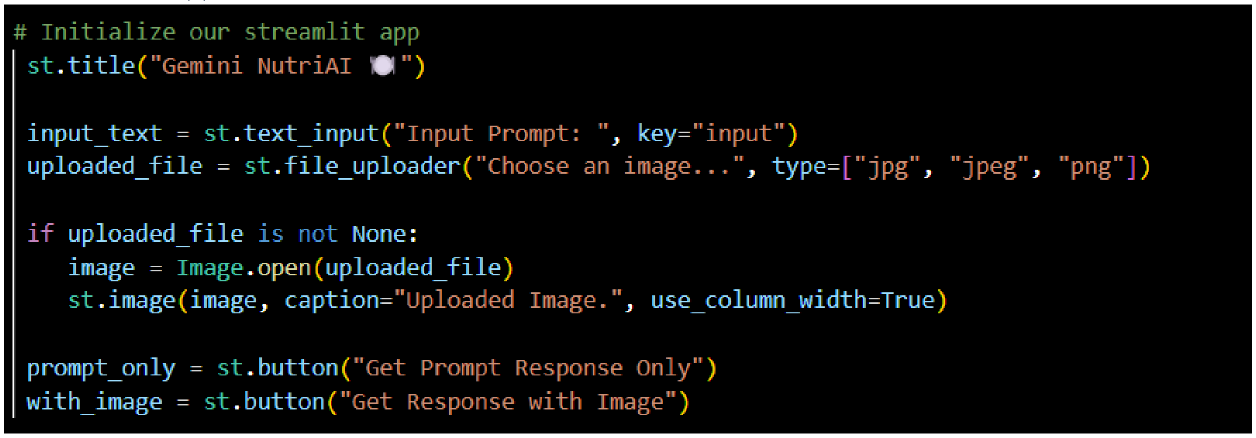
he variable input_prompt is a multi-line string designed as a prompt for a nutritionist AI model. It takes the input from the user for the prompt and accordingly designs the response based on image or text

Milestone 4 : Model Deployment

We deploy our model using the Streamlit framework, a powerful tool for building and sharing data applications quickly and easily. With Streamlit, we can create interactive web applications that allow users to interact with our models in real-time, providing an intuitive and seamless experience.

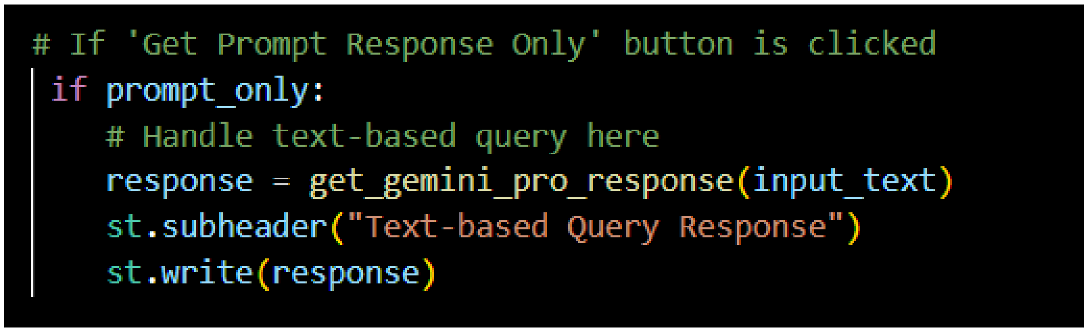
Activity 4 : Integrate With Web Framework

AI Nutritionist Application:



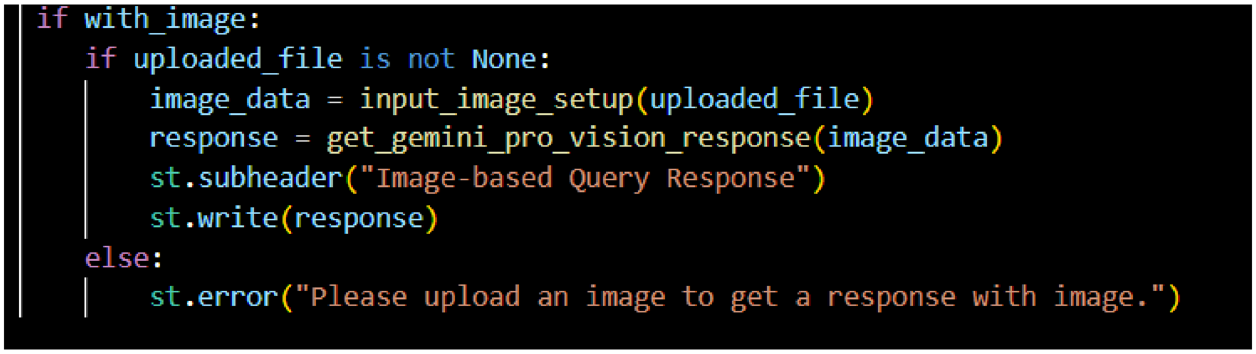
```
# Initialize our streamlit app
st.title("Gemini NutriAI 🍽️")

input_text = st.text_input("Input Prompt: ", key="input")
uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg", "png"])

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

prompt_only = st.button("Get Prompt Response Only")
with_image = st.button("Get Response with Image")
```

If "Get Prompt Response Only is clicked":



```
# If 'Get Prompt Response Only' button is clicked
if prompt_only:
    # Handle text-based query here
    response = get_gemini_pro_response(input_text)
    st.subheader("Text-based Query Response")
    st.write(response)
```

If "Get Response with Image is clicked":



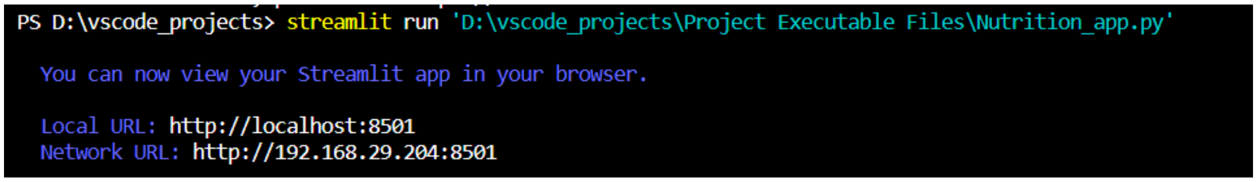
```
if with_image:
    if uploaded_file is not None:
        image_data = input_image_setup(uploaded_file)
        response = get_gemini_pro_vision_response(image_data)
        st.subheader("Image-based Query Response")
        st.write(response)
    else:
        st.error("Please upload an image to get a response with image.")
```

This code initializes a Streamlit application titled "AI Nutritionist App" by setting the page title and creating the app's header. It includes a text input field for users to enter a custom prompt and a file uploader for users to upload an image in JPG, JPEG, or PNG format. If an image is uploaded, it is opened using the PIL library and displayed within the app with a caption. A button labeled "Tell me the total calories" is also provided, which users can click to trigger the application's functionality for analyzing the uploaded image to calculate and display the total calorie content of the food items depicted.

Activity 2 : Host The Application

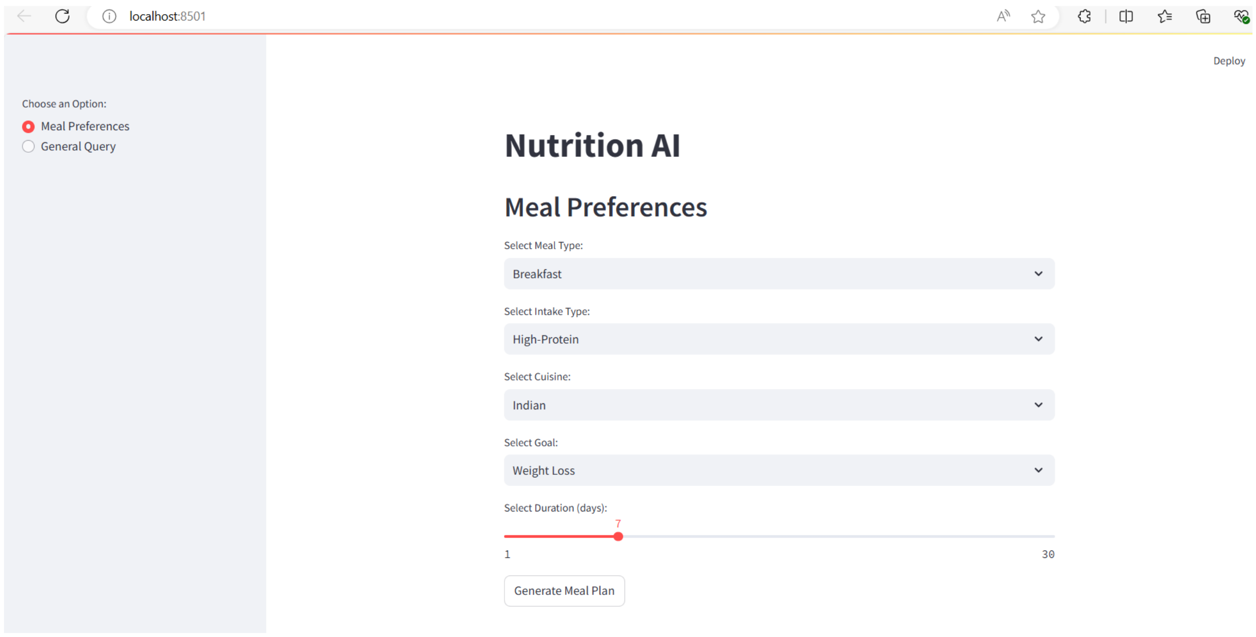
Launching the Application:

- To host the application, go to the terminal, type - streamlit run app.py
- Here app.py refers to a python script.



```
PS D:\vscode_projects> streamlit run 'D:\vscode_projects\Project Executable Files\Nutrition_app.py'

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.29.204:8501
```

on running streamlit the below page is shown where the user is given 2 option to create his own meal plan or to see the calorie count and see various general queries related to his diet



Choose an Option:

[✓ Meal Preferences]

General Query

Nutrition AI

Meal Preferences

Select Meal Type: Breakfast

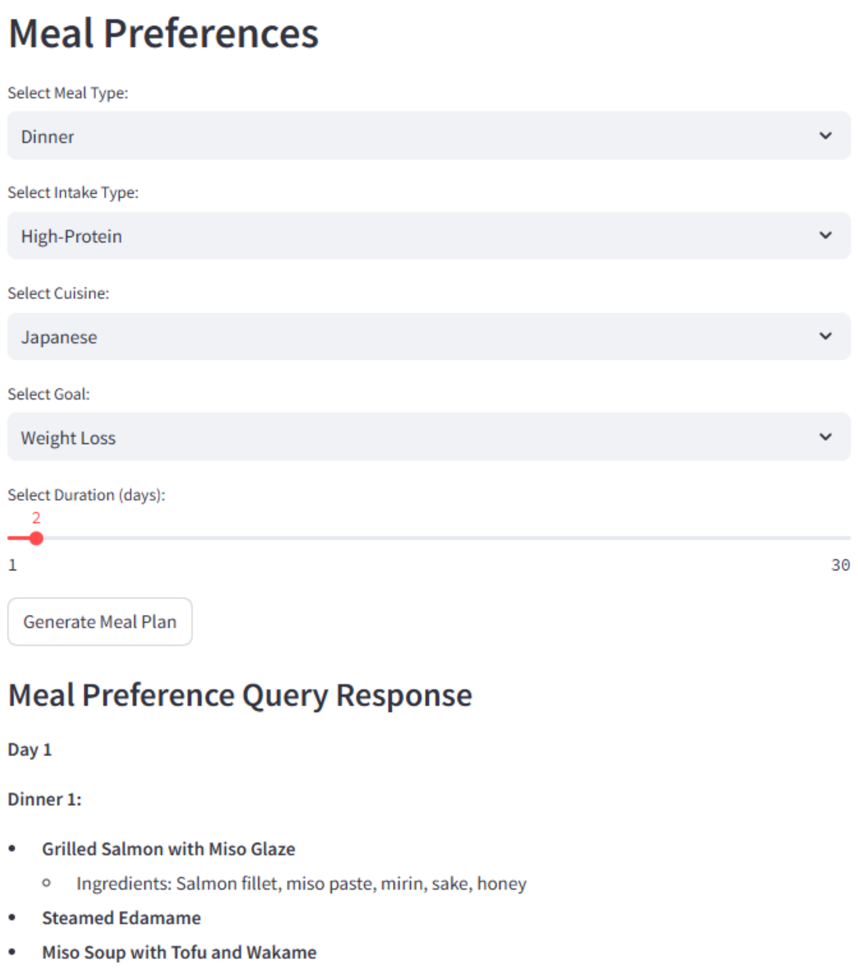
Select Intake Type: High-Protein

Select Cuisine: Indian

Select Goal: Weight Loss

Select Duration (days): 7

here the user has the various option as to how he wants his meal plan from breakfast,lunch,dinner to high protein,low sugar etc to various cuisine and whether he wants to gain or lose weight to how many days he wants to lose it in etc below is a sample output for a random selection.



Meal Preferences

Select Meal Type: Dinner

Select Intake Type: High-Protein

Select Cuisine: Japanese

Select Goal: Weight Loss

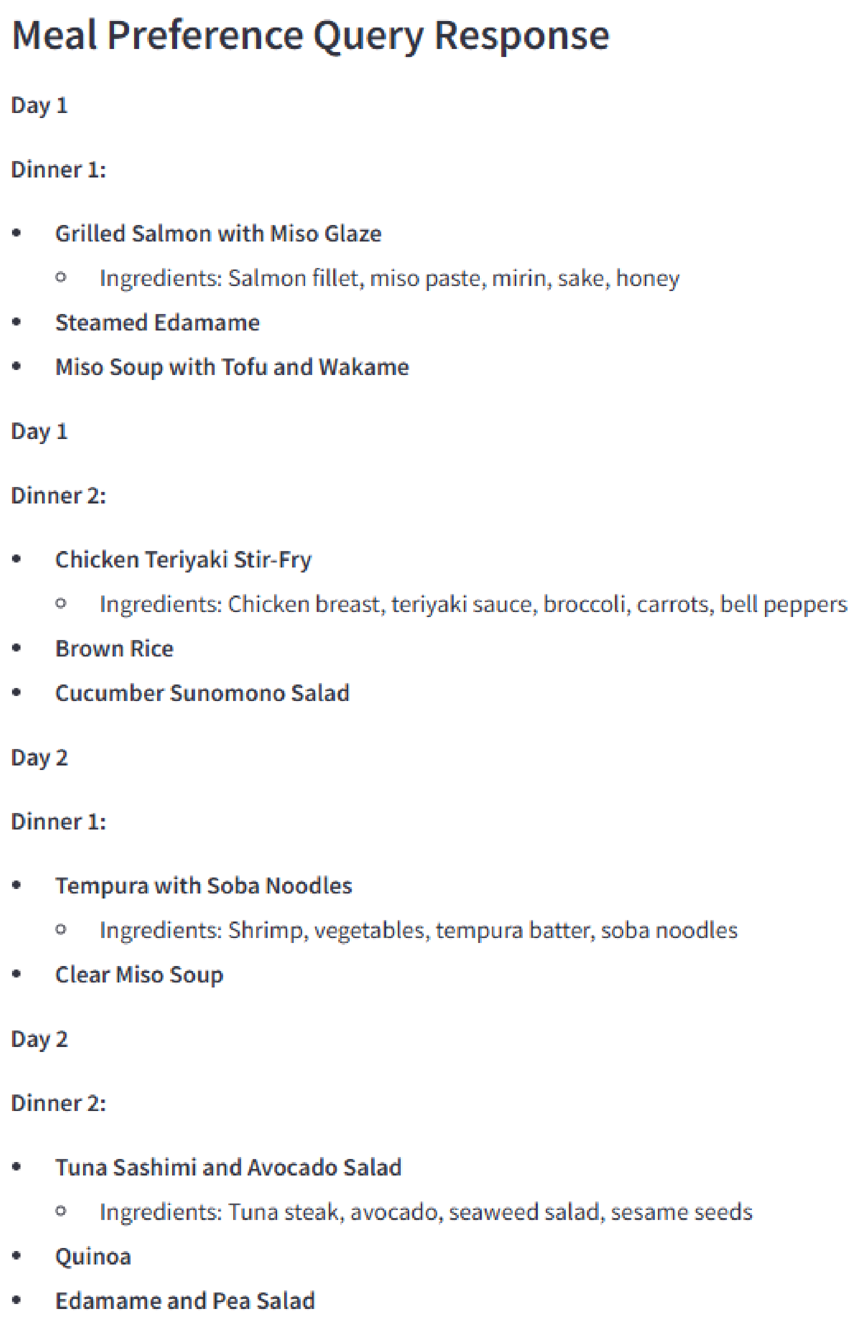
Select Duration (days): 2

Meal Preference Query Response

Day 1

Dinner 1:

- **Grilled Salmon with Miso Glaze**
  - Ingredients: Salmon fillet, miso paste, mirin, sake, honey
- **Steamed Edamame**
- **Miso Soup with Tofu and Wakame**



Meal Preference Query Response

Day 1

Dinner 1:

- **Grilled Salmon with Miso Glaze**
  - Ingredients: Salmon fillet, miso paste, mirin, sake, honey
- **Steamed Edamame**
- **Miso Soup with Tofu and Wakame**

Day 1

Dinner 2:

- **Chicken Teriyaki Stir-Fry**
  - Ingredients: Chicken breast, teriyaki sauce, broccoli, carrots, bell peppers
- **Brown Rice**
- **Cucumber Sunomono Salad**

Day 2

Dinner 1:

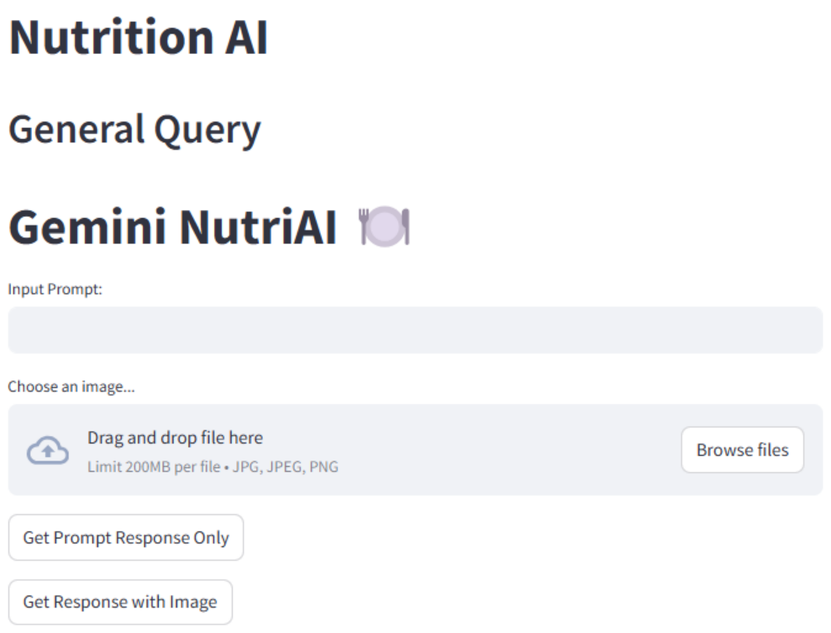
- **Tempura with Soba Noodles**
  - Ingredients: Shrimp, vegetables, tempura batter, soba noodles
- **Clear Miso Soup**

Day 2

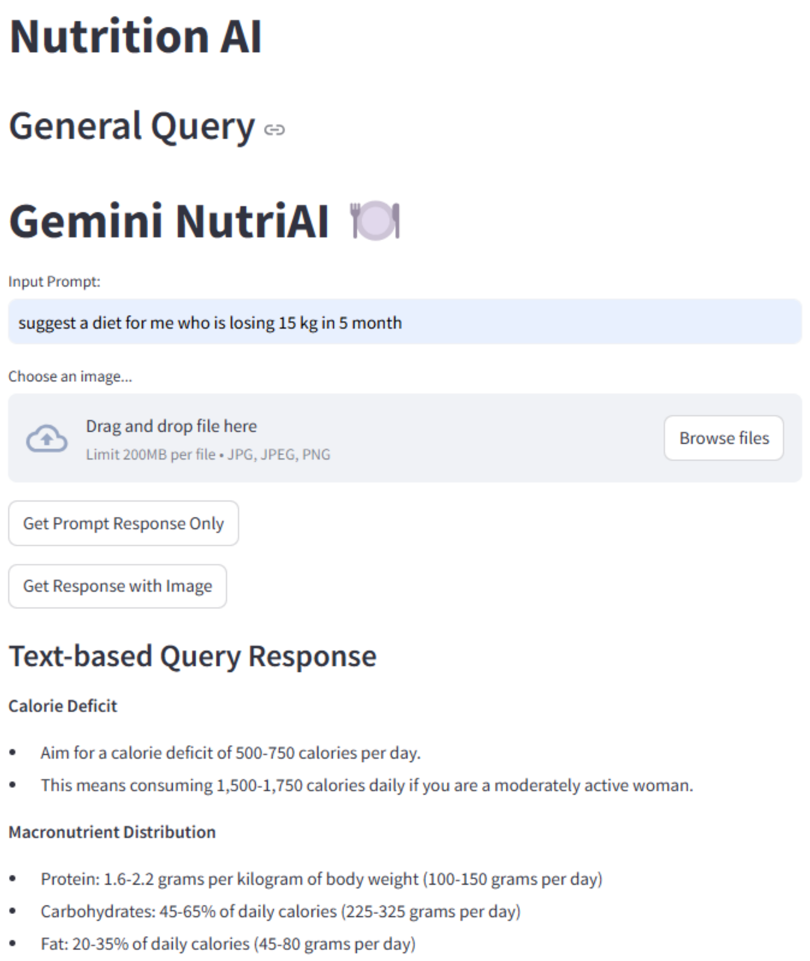
Dinner 2:

- **Tuna Sashimi and Avocado Salad**
  - Ingredients: Tuna steak, avocado, seaweed salad, sesame seeds
- **Quinoa**
- **Edamame and Pea Salad**

on using general query button we get this home page



on giving an input prompt



Nutrition AI

General Query

Gemini NutriAI 🍽️

Input Prompt:

suggest a diet for me who is losing 15 kg in 5 month

Choose an image...

Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG

Text-based Query Response

Calorie Deficit

- Aim for a calorie deficit of 500-750 calories per day.
- This means consuming 1,500-1,750 calories daily if you are a moderately active woman.

Macronutrient Distribution

- Protein: 1.6-2.2 grams per kilogram of body weight (100-150 grams per day)
- Carbohydrates: 45-65% of daily calories (225-325 grams per day)
- Fat: 20-35% of daily calories (45-80 grams per day)

Text-based Query Response

Calorie Deficit

- Aim for a calorie deficit of 500-750 calories per day.
- This means consuming 1,500-1,750 calories daily if you are a moderately active woman.

Macronutrient Distribution

- Protein: 1.6-2.2 grams per kilogram of body weight (100-150 grams per day)
- Carbohydrates: 45-65% of daily calories (225-325 grams per day)
- Fat: 20-35% of daily calories (45-80 grams per day)

Meal Plan

Breakfast

- Oatmeal with fruit and nuts (250-300 calories)
- Greek yogurt with berries and granola (200-250 calories)
- Scrambled eggs with whole-wheat toast (250-300 calories)

Lunch

- Salad with grilled chicken, vegetables, and quinoa (300-350 calories)
- Sandwich on whole-wheat bread with lean protein, veggies, and low-fat cheese (350-400 calories)
- Soup with a side of whole-grain crackers (250-300 calories)

Dinner

- Grilled salmon with roasted vegetables and brown rice (400-450 calories)
- Chicken stir-fry with whole-wheat noodles (450-500 calories)
- Lentil soup with a side of whole-grain bread (350-400 calories)

Snacks

- Apple with peanut butter (200-250 calories)
- Greek yogurt (100-150 calories)
- Cottage cheese with fruit (150-200 calories)

Hydration

- Drink plenty of water throughout the day, at least 8-10 glasses.

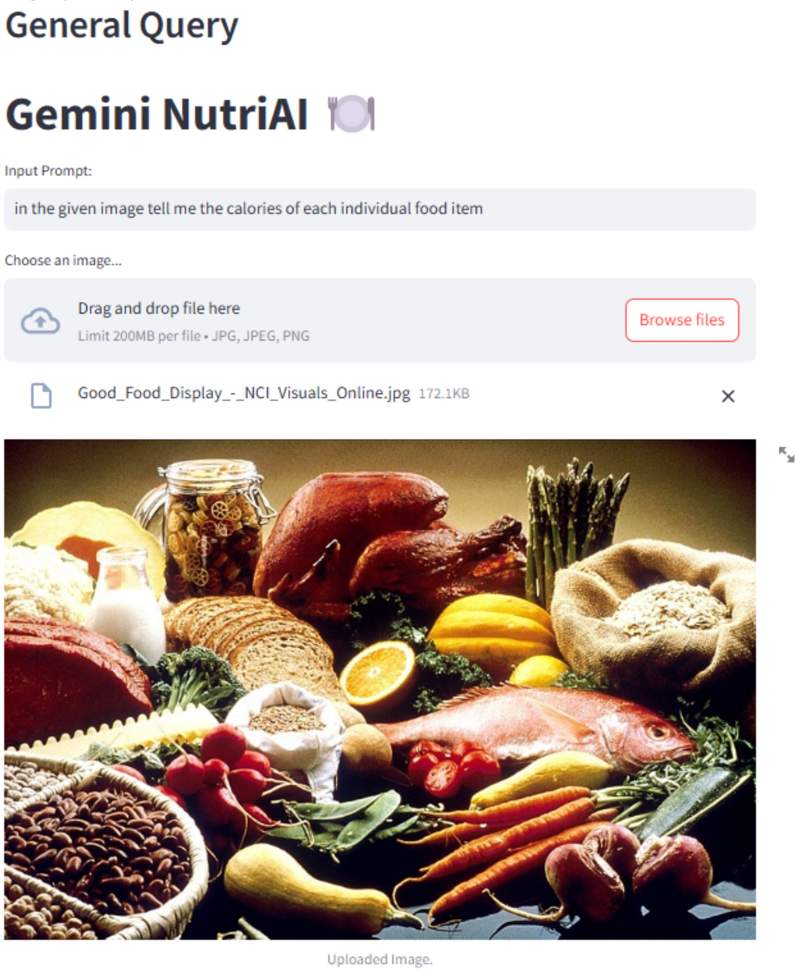
Food Groups to Focus on

- Lean protein: chicken, turkey, fish, beans, tofu
- Fruits and vegetables: aim for a variety of colors
- Whole grains: brown rice, quinoa, oatmeal
- Low-fat dairy: milk, yogurt, cheese

Foods to Limit

- Sugary drinks
- Processed foods
- Red meat
- High-fat foods
- Alcohol

on using the image prompt to count calories



General Query

Gemini NutriAI 🍽️

Input Prompt:

in the given image tell me the calories of each individual food item

Choose an image...

Drag and drop file here

Limit 200MB per file • JPG, JPEG, PNG

Good_Food_Display_-_NCI_Visuals_Online.jpg 172.1KB

Uploaded Image.

Image-based Query Response

A balanced diet is one that provides the body with the nutrients it needs to function properly. These nutrients include carbohydrates, proteins, fats, vitamins, and minerals. A balanced diet also includes plenty of fruits, vegetables, and whole grains.

There are many different ways to achieve a balanced diet. Some people find it helpful to follow a specific diet plan, such as the Mediterranean diet or the DASH diet. Others find it easier to make small changes to their diet, such as adding more fruits and vegetables to their meals.

No matter how you choose to achieve a balanced diet, the important thing is to make sure that you are getting the nutrients your body needs. A balanced diet is essential for good health and well-being.

Here are some tips for achieving a balanced diet:

- Make half of your plate fruits and vegetables.
- Choose whole grains over refined grains.
- Limit unhealthy fats, such as saturated and trans fats.
- Choose lean protein sources, such as fish, chicken, and beans.
- Limit added sugar.
- Drink plenty of water.

By following these tips, you can make sure that you are getting the nutrients your body needs to function properly. A balanced diet is essential for good health and well-being.

Text-based Query Response

Calorie Deficit

- Aim for a calorie deficit of 500-750 calories per day.
- This means consuming 1,500-1,750 calories daily if you are a moderately active woman.

Macronutrient Distribution

- Protein: 1.6-2.2 grams per kilogram of body weight (100-150 grams per day)
- Carbohydrates: 45-65% of daily calories (225-325 grams per day)
- Fat: 20-35% of daily calories (45-80 grams per day)

Meal Plan

Breakfast

- Oatmeal with fruit and nuts (250-300 calories)
- Greek yogurt with berries and granola (200-250 calories)
- Scrambled eggs with whole-wheat toast (250-300 calories)

Lunch

- Salad with grilled chicken, vegetables, and quinoa (300-350 calories)
- Sandwich on whole-wheat bread with lean protein, veggies, and low-fat cheese (350-400 calories)
- Soup with a side of whole-grain crackers (250-300 calories)

Dinner

- Grilled salmon with roasted vegetables and brown rice (400-450 calories)
- Chicken stir-fry with whole-wheat noodles (450-500 calories)
- Lentil soup with a side of whole-grain bread (350-400 calories)

Snacks

- Apple with peanut butter (200-250 calories)
- Greek yogurt (100-150 calories)
- Cottage cheese with fruit (150-200 calories)

Hydration

- Drink plenty of water throughout the day, at least 8-10 glasses.

Food Groups to Focus on

- Lean protein: chicken, turkey, fish, beans, tofu
- Fruits and vegetables: aim for a variety of colors
- Whole grains: brown rice, quinoa, oatmeal
- Low-fat dairy: milk, yogurt, cheese

Foods to Limit

- Sugary drinks
- Processed foods
- Red meat
- High-fat foods
- Alcohol