**Meal Generator and Recommendation Rulebase System: Overview and Documentation**

This document provides an overview of how the **Meal Generator** and **Recommendation Rulebase** systems work. These systems allow users to create a personalized meal plan and receive dietary recommendations based on their nutritional goals. The process involves using a **Genetic Algorithm** for meal plan optimization and a **Rule-based Recommendation Engine** to suggest improvements to the generated plan.

**Background of Meal Generator and Genetic Algorithm**

The **Meal Generator** uses a structured approach to generate a customized meal plan for the user based on their personal inputs such as age, weight, height, activity level, and macronutrient goals (calories, protein, carbohydrates, fats). It also allows users to pick their preferred food items for each meal (Breakfast, Lunch, Dinner).

In the background, a **Genetic Algorithm (GA)** is often employed for optimization tasks where the goal is to find the best combination of items that meet nutritional requirements while adhering to user preferences. The algorithm mimics the process of natural selection by:

1. **Initialization**: Generating random meal plans.
2. **Selection**: Choosing the best meal plans based on their fitness score.
3. **Crossover**: Mixing different meal plans to create new ones.
4. **Mutation**: Introducing slight changes to improve the meal plans.
5. **Termination**: Stopping when the meal plans meet the user’s nutritional targets.

This allows for generating meal plans that closely match the user’s macros and provide a balanced diet.

**Recommendation Rulebase and Its Function**

After the meal plan is generated, the **Recommendation Rulebase Engine** steps in to analyze the meal plan against the user’s target macronutrient goals. The system identifies any nutritional imbalances (e.g., excess fats, insufficient proteins) and suggests **alternative food items** to address these imbalances.

**How the Rulebase Works:**

1. **Input**: The generated meal plan (from the Genetic Algorithm) is fed into the recommendation engine along with the user’s target macros for each meal.
2. **Nutrient Analysis**: The engine calculates the shortfall or excess of calories, proteins, carbs, and fats for each meal in the plan.
3. **Food Alternatives**: Based on this analysis, the engine suggests food alternatives that can help the user meet their nutritional goals more effectively. For example, if a meal is low in protein, it will recommend higher-protein food alternatives.
4. **Optimization**: The system provides recommendations based on the degree of deviation from the user's goals, aiming to optimize the meal plan by improving the nutritional balance.

**Can We Expect Exact Matches in Nutrient Recommendations?**

In some cases, exact matches may not always be possible. This is due to several factors:

* **Other Selected Foods**: The system tries to optimize the meal plan as a whole. Any adjustment to one meal or item will affect the overall distribution of macronutrients across all meals.
* **Optimization of Portions and Food Choices**: The system balances both the portion sizes and the choice of food. If the system cannot find an exact match that meets all the nutritional targets, it will still aim to **improve** the nutritional profile by suggesting alternatives that are closer to the user's goals.
* **Precision and Decimals**: Due to the precision in macronutrient calculations (often down to decimals), it may not be feasible to achieve 100% matching in all cases. However, the recommendations will always provide healthier and more balanced options compared to the initial meal plan.

**API Use Case: Meal Generation and Recommendation System**

This API provides two key functionalities:

1. **Meal Plan Generation**: Generates a personalized meal plan based on user input, including meals and specific food items.
2. **Meal Plan Recommendation**: Provides recommendations to optimize the meal plan based on target macronutrient values (calories, protein, carbs, fats).

**General Flow:**

1. **Generate a meal plan** by submitting user inputs (via API 1).
2. **Get recommendations** for the generated meal plan by submitting it along with target macros (via API 2).

**API 1: Meal Plan Generation**

The Meal Plan Generation API allows users to generate a meal plan tailored to their individual dietary requirements and food preferences. Users input their personal details and meal choices, and the API returns a balanced meal plan.

**Endpoint:**

POST https://zenithapp-e8ce58435ec0.herokuapp.com/generate\_meal\_plan

**Request Example (curl):**

bash

Copy code

curl --location --max-time 5 'https://zenithapp-e8ce58435ec0.herokuapp.com/generate\_meal\_plan' \

--header 'Content-Type: application/json' \

--data '{

"user\_input": {

"name": "John Doe",

"age": 30,

"weight": 75,

"height": 180,

"activity\_level": "moderate",

"calories": 2000,

"protein": 150,

"carbs": 250,

"fats": 60

},

"meal\_selection": {

"meals": {

"Breakfast": 0.3,

"Lunch": 0.4,

"Dinner": 0.3

},

"user\_selected\_items": {

"Breakfast": [

"Crumpet, from white flour, toasted",

"Bean, broad, fresh, boiled, drained",

"Beef, boneless dice or strips, lean, raw"

],

"Lunch": [

"Pikelet, plain, homemade",

"Bean, butter, fresh, boiled, drained",

"Beef, steak, boneless or bone-in, blade, lean, raw"

],

"Dinner": [

"Pancake, plain, homemade",

"Bean, green, fresh, boiled, drained",

"Beef, casserole meat, boneless or bone-in, shin, semi-trimmed, raw"

]

}

}

}'

**Request Parameters:**

* **User Information**:
  + name: The name of the user.
  + age: Age of the user.
  + weight: Weight in kg.
  + height: Height in cm.
  + activity\_level: Activity level (e.g., light, moderate, intense).
  + calories, protein, carbs, fats: User's daily nutritional target.
* **Meal Selection**:
  + meals: Percentage distribution of calories across meals (e.g., 30% breakfast, 40% lunch, 30% dinner).
  + user\_selected\_items: Selected food items for each meal.

**Response Example:**

The response contains the generated meal plan, including food items and their macronutrient breakdown:

{

"Breakfast": {

"items": [

{

"name": "Crumpet, from white flour, toasted",

"quantity": "172.21 g",

"macros": {

"calories": 346.13,

"protein": 9.64,

"carbs": 71.64,

"fats": 1.38

}

},

{

"name": "Bean, broad, fresh, boiled, drained",

"quantity": "295.76 g",

"macros": {

"calories": 174.20,

"protein": 21.89,

"carbs": 7.10,

"fats": 1.48

}

},

{

"name": "Beef, boneless dice or strips, lean, raw",

"quantity": "59.67 g",

"macros": {

"calories": 78.76,

"protein": 16.47,

"carbs": 0.0,

"fats": 1.43

}

}

],

"macros": {

"calories": 599.10,

"protein": 48.00,

"carbs": 78.74,

"fats": 4.29

},

"fitness\_score": 14.86

},

...

}

**API 2: Meal Plan Recommendations**

The Meal Plan Recommendations API analyzes the generated meal plan and provides suggestions to optimize it based on target macronutrient values. This API works by identifying nutrient imbalances in the meal plan and recommending alternative foods.

**Endpoint:**

POST https://zenithapp-e8ce58435ec0.herokuapp.com/generate\_recommendations

**Request Example (curl):**

curl --location --max-time 5 'https://zenithapp-e8ce58435ec0.herokuapp.com/generate\_recommendations' \

--header 'Content-Type: application/json' \

--data '{

"meal\_plan": {

"meals": {

"Breakfast": {

"items": [

{

"name": "Doughnut, jam filled, sugar coated",

"macros": {

"calories": 263.86,

"protein": 4.27,

"carbs": 32.86,

"fats": 12.52

}

},

...

],

"macros": {

"calories": 286.73,

"protein": 4.69,

"carbs": 35.87,

"fats": 13.51

}

},

...

}

},

"target\_macros": {

"Breakfast": {

"calories": 600

"protein": 20,

"carbs": 70,

"fats": 20

},

"Lunch": {

"calories": 800,

"protein": 30,

"carbs": 90,

"fats": 25

},

"Dinner": {

"calories": 600,

"protein": 25,

"carbs": 80,

"fats": 30

}

}

}'

**Request Parameters:**

* **Meal Plan**: This is the output meal plan from **API 1: Meal Plan Generation**. It contains the list of food items selected for each meal, along with their associated macronutrients (calories, protein, carbs, fats).
* **Target Macros**: The target macronutrient values (calories, protein, carbs, fats) for each meal (e.g., Breakfast, Lunch, Dinner) that the user wants to meet.

**Response Example:**

The response contains suggestions for optimizing the meal plan. For each meal, the API identifies the items that deviate from the target macros and suggests alternative foods that can help bring the meal closer to the target.

[

{

"meal": "Breakfast",

"item": "Doughnut, jam filled, sugar coated",

"issue": "Optimize Fats",

"alternatives": [

[

"Pie, savoury, meat, commercial, ready to eat",

-34.30

],

[

"Ice cream, vanilla flavour, regular fat",

-30.40

],

[

"Gelatine, all types",

-141.80

],

[

"Protein powder, whey based, protein >70%, unfortified",

-139.00

],

[

"Musashi 100% Whey Protein Vanilla Milkshake Flavour 900g",

-121.20

]

]

},

{

"meal": "Lunch",

"item": "Bread, from white flour, added iron",

"issue": "Optimize Fats",

"alternatives": [

[

"HEINZ BAKED BEANS THE LIL ONE",

-20.80

],

[

"Tip Top Bakery English Muffins Wholemeal 6 Pack",

-20.50

]

]

}

]

**Explanation:**

* **Meal**: The name of the meal (e.g., Breakfast, Lunch, Dinner).
* **Item**: The food item that is identified as deviating from the target macronutrients.
* **Issue**: The nutrient that needs to be optimized (e.g., fats, carbs, protein).
* **Alternatives**: A list of alternative food items along with their deviation (e.g., reduction in fats).

**Example Workflow**

1. **Generate a Meal Plan**: First, generate a meal plan using the /generate\_meal\_plan API by providing user details such as age, weight, activity level, and selecting food items for each meal.
   * Use the example curl command for generating a meal plan provided in **API 1**.
2. **Generate Recommendations**: Once the meal plan is generated, submit the generated meal plan along with the target macros to the /generate\_recommendations API to receive recommendations for optimizing the nutritional content.
   * Use the example curl command for generating recommendations provided in **API 2**.

**Conclusion**

In this workflow:

1. **Meal Plan Generation**: The system generates a meal plan based on user input and predefined food selections. It uses algorithms to optimize the balance of nutrients across the selected meals.
2. **Meal Plan Recommendation**: Once the meal plan is generated, the system analyzes it against the user's target macros and suggests food alternatives to improve the plan's nutritional content.

These APIs provide users with a personalized meal planning experience, offering flexibility in meal selection while ensuring nutritional balance. The **Meal Plan Generation** step must always precede the **Meal Plan Recommendation** step to ensure that the correct data is passed for optimization.

**For Troubleshooting:**

* Ensure that the generated meal plan from /generate\_meal\_plan is correctly formatted when submitted to /generate\_recommendations.
* Check for any missing or invalid fields in the request.
* Review the error messages for detailed insight into failures. Logs will help identify where the process may be failing (e.g., incomplete meal data, mismatched macro formats).