The backend process is a comprehensive and automated system that calculates personalized nutritional needs, queries and normalizes food data, adjusts food quantities to meet dietary goals, and optimizes the meal plan to ensure nutritional balance and practicality. This process is exposed via APIs and presented through a user-friendly interface, making it accessible and efficient for generating customized meal plans.

1. User Data Input and Calculation of Nutritional Needs:

- **User Input:** The backend receives personal data such as age, weight, height, gender, and activity level from the user.
- **Basal Metabolic Rate (BMR) Calculation:** The system uses the Mifflin-St Jeor equation to calculate the Basal Metabolic Rate (BMR), which estimates the number of calories the user needs to maintain their current weight at rest.
- **Total Daily Energy Expenditure (TDEE):** The BMR is then multiplied by an activity multiplier (determined by the user's activity level) to calculate the Total Daily Energy Expenditure (TDEE). This represents the total number of calories the user needs to maintain their weight considering their daily activities.

2. Macronutrient Distribution:

- **Standard Ratios:** The calculated TDEE is divided into macronutrient targets—carbohydrates, proteins, and fats—based on standard dietary ratios (e.g., 50% carbs, 30% fats, 20% protein).
- **Meal Allocation:** These macronutrient targets are further distributed across the user's meals (e.g., breakfast, lunch, dinner) according to the user's preferences or standard meal distributions. The user can specify what percentage of their daily calories they want to allocate to each meal.

3. Data Querying and Normalization:

- **Food Database Query:** The backend queries a structured food database using Pandas. This database contains a wide variety of food items, each with detailed nutritional information (calories, protein, carbs, fats, etc.).
- **Normalization of Units:** The nutritional data is normalized to a common unit (e.g., per 100 grams) to ensure consistency and accuracy when scaling food portions. This allows the backend to adjust food quantities while maintaining correct macronutrient calculations.

4. Automation of Food Quantity Adjustment:

- **Initial Selection of Foods:** The backend selects food items from the database that collectively aim to meet the user's macronutrient goals for each meal.
- **Quantity Adjustment:** The system automatically adjusts the quantities of the selected food items to closely match the target macronutrients. This process involves:
- **Scaling:** Increasing or decreasing the portion sizes of each food item based on its nutritional content.

- **Iterative Refinement:** Fine-tuning the quantities to minimize the difference between the meal's nutrient content and the target values, ensuring the closest possible match.

5. Optimization and Balancing:

- **Optimization Techniques:** The backend may employ optimization algorithms to ensure that the meal plans generated not only meet macronutrient targets but also respect practical serving sizes and user preferences.
- **Balancing Across Meals:** The backend ensures that the macronutrient distribution is balanced across all meals in a day, making adjustments as needed to ensure overall daily targets are met without exceeding practical limits.

6. Meal Plan Generation and Output:

- **Final Meal Plan Assembly:** The backend assembles the adjusted food items into a complete meal plan, with each meal containing a list of food items and their respective quantities.
- Output: The generated meal plan is then presented to the user, with an option to view the data in JSON format, visualize it through charts, or save it for later use. The meal plan is tailored to the user's specific dietary needs and preferences, ensuring both nutritional balance and practicality.

7. Integration and Deployment:

- **FastAPI for API Integration:** The backend logic is exposed through a FastAPI-based RESTful API, allowing other applications (such as mobile apps) to interact with the meal planner by sending user data and receiving customized meal plans.
- **Streamlit for User Interface:** The frontend is built using Streamlit, providing an interactive web interface for users to input their data and visualize their meal plans.
- **Deployment Considerations:** The system is designed to be deployed on cloud platforms like Streamlit Cloud, ensuring accessibility and scalability. The file paths are carefully managed to ensure smooth deployment and operation in cloud environments.