Project Report

Function: recommend_recipes
The recipe recommendation algorithm takes available ingredients and user dietary preferences as inputs. It extracts ingredient names into a set for efficient lookup and iterates over each recipe in the database. A recipe is considered if at least half of its required ingredients are available and match the user's dietary preference (except for "regular", as regular can eat from all diets). The algorithm returns recipes that can be made with minimal additional ingredients, promoting convenience and adherence to dietary goals.

Function: create_shopping_list
This function generates a shopping list based on recommended recipes and available ingredients. It extracts ingredient names into a set, initializes an empty dictionary to store the shopping list, and iterates over each recommended recipe. For each recipe, it checks the required ingredients against the available ones. Missing or insufficient ingredients are added to the shopping list with their required quantities. The algorithm converts the shopping list dictionary into a structured list of dictionaries representing items needed to purchase.

Function: create_meal_plan
This function generates a meal plan based on the user's dietary preferences and matched recipes. It makes sure that we meet all daily goals for nutrients precisely, with only a 200-calorie margin of error for the daily calories whether that be higher or lower. It mainly uses the random.choice library function to do picking of specific recipes, and a daily nutrient audit after adding each recipe to decide if it should continue adding recipes for the current day or not.

Function: write_output
This function writes the output from the meal plan and shopping list into a text file for the user. It writes the respective day, the total daily nutrient counts and then all of the day's meals. It does this for all of the days of the week. It also writes the ingredients that should be purchased at the bottom in the form of a shopping list.

Function: visualize_nutrition
The visualize_nutrition function generates a bar chart to display daily nutritional data, helping users understand their dietary patterns at a glance. It accepts daily_nutrition, a list of tuples where each tuple contains a day's name and its associated nutrition values (calories, protein, carbs, fat, and fiber). The function extracts individual days and nutrient values into separate lists for plotting. Using Matplotlib, it creates grouped bar charts, where each group corresponds to a day, and each bar within the group represents a specific nutrient. The function ensures clear labeling of axes, a legend for nutrient identification, and proper alignment of tick marks to improve readability.

Function: read_preferences
The read_preferences function loads and processes user dietary preferences from a JSON input file. It reads the file's contents, parses the JSON data, and extracts the user_preferences

field into a list. This conversion allows for streamlined handling of dietary preference data, which can then be used by other components of the system. By leveraging JSON's flexibility, the function accommodates a variety of user-defined preferences, ensuring compatibility with diverse use cases.

Function: read_ingredients

The read_ingredients function parses a CSV file to generate a structured list of ingredient dictionaries. Each ingredient entry includes its name, quantity, and unit, extracted from the CSV file's rows. The function skips the header line containing column names and splits each subsequent line to populate dictionary fields. These dictionaries are appended to a list, creating a comprehensive dataset of ingredients. This structured approach simplifies further processing, such as matching available ingredients with recipes or generating shopping lists.