# Documentation

## I. Explanation of the purpose of each file in our repository

- bmi.csv - this table is interpreted as a Pandas DataFrame by the code and is used to complete average BMI calculations. It is a Table sourced from Kaggle, an information dataset site that has many different sample data and tables.
- graph - Sheet 1.csv - this sheet is used by the code to produce the graph of a person's weight loss over 15 months. It has a column of 60 weeks, 1 week per line. The code will create another column that has data that the code implements by shaving off a certain amount of calories per person's weight.
- Guidelines.txt - this text file has guidelines that list out maintenance, shred, and bulk, followed by unique corresponding numeric calorie values. This will be read by the Calorie class to calculate customized calories for the user based on the user's attributes and their activity level.
- meals_data.json - This json file contains a dictionary of meal plans like "Meal 1": ["chicken", "rice", "broccoli"]". This will be read by the get_meal_data method and then processed by the get_meal_options method to output a meal option for the user that does not contain any of the user's allergies and contains their food preferences.
- finalproject.py - This is the final project code. The project provides a comprehensive health and nutrition management system for users seeking personalized diet plans and fitness guidance like how to lose, gain, or maintain weight. This code will output bmi, a graph of weightloss/weight gain overtime, and custom meal plans.
- hihi.py - a code that we ran in preparation for this assignment during one of the exercises, in order to test out our GitHub skills.

## II. Instructions on how to run the program from the command line

To run this program from the command line, you must put in the python terminal finalproject.py after the name of the GitHub repository and the file path of where it is stored as a whole.

Example for Mac: python3 finalproject.py
Example for Windows/Linux: python finalproject.py

## III. Instructions on how to use our program and interpret the output of the program

After you run the program, there will be a series of user input values. They go as follows, in the following order: Name, height (in meters), weight (in kg), age,
Sport(high-intensity/moderate-intensity/low-intensity), Daily Activities(lightly active/average/very active), Goal (shred/bulk/maintenance). When inputting sport, activity, and

goal, make sure to copy the exact format that is listed after each of these attributes. If the user puts in something that is not one of the three options for sport, activity, and goal, that will force quit the whole program and will make the user restart the program and input all of their information again from the start.

Example of input:
Name: Elizabeth
Height (in meters): 1.6
Weight (in kg): 50
Age: 20
Sport (high-intensity/moderate-intensity/low-intensity): moderate-intensity
Daily Activities (lightly active/average/very active): average
Goal (shred/bulk/maintenance): bulk

The output is the user's information on the terminal. The output displays the user's information in this order: name, height, weight, age, sport intensity, daily activity level, goal (bulk, shred, maintenance), BMI, and the user's BMI compared to others in their age group.

The code will ask "Do you want detailed dietary advice? (yes/no)." Where the user will either input "yes" or "no". If the user inputs "yes" then the code will output the daily caloric intake and fitness advice for the user's goal (bulk, shred, maintenance). If the user inputs "no" then the code will output the daily caloric intake and will not display the fitness advice.

Then it will ask user input for allergies, which the user can type any allergies they have. When multiple allergies need to be listed, the user will need to comma separate their allergies. If the user doesn't have any allergies put "None".
Example:
Enter your allergies (comma-separated): None
or
Enter your allergies (comma-separated): salmon, almonds
or
Enter your allergies (comma-separated): salmon

The program will ask for food preferences that the user will input. When multiple foods need to be listed, the user will need to comma separate the foods. If the user doesn't have any food preferences, put "None".
Example:
Enter your meal preferences (comma-separated): None
or
Enter your meal preferences (comma-separated): chicken, tofu

or
Enter your meal preferences (comma-separated): chicken

The program will display meals and a graph. The meals listed are recommended meals the user can eat that do not contain the user's allergies. If all of the meals contain the user's allergies, the program will output "Sorry, we couldn't find suitable meal options for your allergies." Disclaimer: the meal preferences do not guarantee that the food preference will be in the meal options. The point of meal preferences is for the user to input their preference and if there is a meal that contains the user's preference, the program will output it along with other meal options. If the meal preferences are not in the meal options provided, the program will output other viable meal options. A graph is displayed that will show change of weight over 60 weeks if the user follows the custom calorie goal set for them daily. The graph measures weight in kg and time is measured in weeks.

## IV. Annotated Bibliography


"Calorie Calculator." Calculator.Net, www.calculator.net/calorie-calculator.html.
This source was used to find the formula in calculating the calories needed per user.

Centers for Disease Control and Prevention. (2024, January 18). Calculating BMI. Centers for Disease Control and Prevention. https://www.cdc.gov/nccdphp/dnpao/growthcharts/training/bmiage/page5_1.html

This source contains the equation used to calculate the body-mass index of an individual person using the metric system. We used this equation in the Calories.bmi_calculation method in order to calculate the user's BMI and print it to the console for the user's viewing.

Hunt, L. (2021, May 29). 28 dinners you only need 3 ingredients to make. BuzzFeed. https://www.buzzfeed.com/lindsayhunt/easy-3-ingredient-dinners-that-are-actually-delicious

The above code has a list of 3-ingredient meals we used to manually create our json file with a list of ingredients and their meals we will be parsing through.

Missonnier, R. (2023, September 1). Age, weight, height, BMI analysis. Kaggle. https://www.kaggle.com/datasets/rukenmissonnier/age-weight-height-bmi-analysis

This source found contains a random sample data set of about 4-5 people per age from the ages of 15-61 and has their calculated BMI values. It is by no means extensive or as representative of a whole population, but offers an example of a way that our code could be used on a larger scale. The dataset is included in our repository as bmi.csv.

**Attribution**

| **Method/function** | **Primary author** | **Required Techniques demonstrated** |
|---|---|---|
| User.get_user_info | Michael Mallonga | f-strings containing expressions **AND** conditional expressions |
| Calories.__init__ | Michael Mallonga | |
| Calories._read_guidelines | Matthew Beltran | with statements |
| Calories.calculate_calories | Matthew Beltran | |
| Calories.calculate_custom_calories | Matthew Beltran | Comprehensions used for calculations |
| Calories.bmi_calculation | Pragya Kumar | groupby() operations on Pandas DataFrames |
| Meals.get_meals_data | Colby Brooks | use of json.dumps(), json.loads(), json.dump(), or json.load() |
| Meals.get_meal_options | Colby Brooks | use of a key function (which can be a lambda expression) with sorted() |
| Meals.graph | Pragya Kumar | visualizing data with pyplot |
| Nutrition.calculate_nutrition_plan | Elizabeth Poon | – |
| Nutrition.display_nutrition_calories | Elizabeth Poon | optional parameters **AND** sequence unpacking |