DE LA SALLE UNIVERSITY - MANILA

Meal Planner and Grocery List Generator

A Term Project

Presented to Mr. Ramon Stephen Ruiz

In Partial Fulfillment of the

Requirements for the Course Programming Logic and Design (PROLOGI)

by

Cugal, James Earl C. - JECC

Fetalino, Marcus Cedric T. - MCTF

Dimapilis, David Brian - DBSD David

EQ3

<Monday 12:45 PM - 1:45 PM>

April, 2023

TABLE OF CONTENTS

I. Introduction
- A. Background of the Study
- B. Problem Statement
- C. Objectives
  - C.1 General Objective
  - C.2 Specific Objective
- D. Significance of the Project

II. Review of Related Literature

III. Methodology
- A. Conceptual Framework - IPO Chart
- B. Hierarchy Chart
- C. Flowchart
- D. Pseudocode

IV. Results

V. Discussion of Results

VI. Analysis, Conclusion and Future Directives

VII. References

VIII. Appendices
- A. User's Manual
- B. Source Code
- C. Work Breakdown
- D. Personal Data Sheet

List of Figures

FIGURE 2.1 Hierarchy Chart

FIGURE 2.2 Flowchart

List of Tables

TABLE 1.1 Conceptual Framework - IPO Chart (Input-Process-Output-Chart)

I. Introduction

The modern market has been quickly growing in size and features over the past few years, driven by changes in consumer behavior and technological improvements. Businesses have more chances than ever before to reach potential clients and sell their goods or services because of the growth of e-commerce, mobile technology, and social media. A wide range of goods and services, from conventional items like food and clothing to technological ones like software and online courses, define the modern market. The market has also gotten more individualized as a result of companies adopting data analytics and artificial intelligence to better understand their clients and deliver specialized experiences.

Because of this growth, there is now more competition in the market, and companies are continuously looking for new strategies to set themselves apart and stand out in a crowded field. Influencer marketing, content marketing, and social media advertising are just a few of the marketing and advertising innovations that have resulted from this.

The Meal Planner and Grocery List Generator is a software program developed by the group to lessen the strain of having to manually plan meals and make shopping lists, and the group will present it as part of this research project. The program includes features like an intuitive user interface for entering dietary restrictions, preferred cuisines, and ingredient preferences; suggestions for meals based on user preferences and dietary restrictions; recipes and nutritional information for each suggested meal; options for users to choose which meals they want to prepare for the week; automatic generation of a grocery list based on the selected meals; and a grocery list sorted by food category for convenience.

Overall, the modern market is a fascinating and dynamic environment that offers plenty of chances for companies to expand and prosper. It does, however, also bring difficulties that need cautious navigation and calculated choices.

A. Background of the Study

Grocery shopping can be fun, but if a shopper can't keep track of their wants and needs, their shopping trip might not succeed as expected. According to Evernote (2022), making a grocery list helps you stay concentrated and efficient when shopping. Without one, it's simple to overlook important goods or spend too much on items you already own.

Meal planning and grocery lists can improve diet quality and general health, according to research. People can make sure they have the correct items on hand to prepare healthy meals at home by making a list of the components they will need and scheduling meals in advance.

Writing down the necessary goods in a lengthy piece of paper was the traditional method of making shopping lists. This kept evolving when smartphones were created and finally became common. These days, people use digital notepads to make their grocery lists. Despite the apparent advantages digital notepads offer, many people still have trouble organizing their grocery lists and planning their meals. This could be the result of obstacles like a lack of information or resources, as well as constraints like cost, time, and convenience.

Therefore, more investigation is required to determine the elements that affect the success of grocery lists and meal planning, as well as the best ways to get around obstacles and encourage healthy eating. By examining the views and experiences of people who utilize grocery lists and meal planning, as well as identifying efficient techniques for promoting healthy eating and shopping behaviors, this study seeks to make a contribution to this field of study.

B. Problem Statement

Consumer attitudes and mindsets, among other things, have an impact on their capacity to make grocery lists and plan meals. According to research, an individual's willingness and capacity to engage in these practices can be influenced by their beliefs, motives, and behaviors. Our knowledge of how consumer attitudes and mindsets especially impact their capacity to make grocery lists and plan meals, however, is lacking. This underlines the necessity for additional research to pinpoint the precise attitudes and mindsets that shape customers' behavior in this area and to create efficient plans to get around any obstacles that may follow.

The marketplaces differ and are characterized by heightened rivalry, ongoing innovation in the goods and services on offer, and an increase in the number of businesses operating in the same market. Knowing the customer well is crucial in this situation. Innovation and meeting customer expectations are made feasible by analyzing the variables that directly affect consumer behavior. For marketers to be able to enhance their campaigns and more successfully reach their target demographic, this study is crucial (Rodrigues et al., 2021).

In addition, psychological considerations have a significant role in the decision to buy something, as demonstrated by the prevalence of people who, after making a purchase of a good or service, second-guess their decision. Understanding the thought processes that underlie the decision-making process for purchases is crucial, which is why marketing techniques and consumer psychology are intertwined. The same models are frequently used by the two fields to explain consumer behavior and the factors that lead to impulsive purchases. Advertising appeals to consumers and the messages it sends, and this is reflected in their actions and intentions to make purchases (Rodrigues et al., 2021).

An inability to weigh the repercussions of one's actions and an overpowering want to buy are the causes of impulsive behavior. There is a strong desire to rapidly satiate your most urgent demands despite being aware of the negative impacts of purchase (Meena, 2018). Burton et al. (2018) claim that sudden, powerful emotional desires that result from reactive conduct with little cognitive restraint lead to impulse purchases. This propensity for impulsive and thoughtless purchases can be explained by the buyer's immediate sense of satisfaction (Pradhan et al., 2018).

C. Objectives

C.1 General Objective

The development of a tool that generates grocery lists and meal plans is our main goal. With the use of this program, users will be able to plan meals that adhere to their dietary needs, preferred cuisines, and ingredient preferences. The program's features will include a list of suggested meals for the coming week, along with associated recipes and nutritional data based on the user's information.

C.2 Specific Objectives

To be able to create a meal planner and grocery list generator program
To be able to implement a feature that suggest a list of meals for the week
To be able to implement a feature that prints and exports the grocery list

D. Significance of the Study

This research on grocery lists and meal planning is important because it addresses the growing demand for efficient methods to encourage wholesome eating habits and sensible food purchasing. This study can help with the creation of interventions and policies that encourage healthy eating habits and decrease food waste by gaining an understanding of the attitudes and experiences of people who use meal planning and grocery lists, as well as by identifying effective strategies to get past obstacles to adoption and maintenance of these practices.

The study's conclusions can also guide the creation of instructional materials and tools to assist people in overcoming typical obstacles to meal planning and grocery lists, such as a lack of information or resources, expense, time constraints, and convenience. Policymakers, health professionals, and individuals looking to establish and maintain healthy eating habits and save back on grocery costs can all benefit from understanding the findings of this study.

II. Review of Related Literature

Meal planning and grocery lists can improve diet quality and general health, according to numerous research. For instance, a study conducted in 2016 by Bucher et al. discovered that adult participants who planned their meals had better diets and were less likely to become obese. Similar findings were made by Fulkerson et al. (2011), who discovered that meal planning was connected to teenagers' better intake of fruits and vegetables and decreased intake of fast food.

Meal planning and grocery lists can also help people save money on groceries while fostering healthy eating habits. According to a 2003 study by Leibtag and Kaufman, people who planned their meals and used a shopping list saved money on groceries compared to those who did neither. Similar to this, a study by McLaughlin et al. (2014) discovered that meal planning and grocery lists were efficient methods for helping low-income households reduce food waste and save money on groceries.

Nevertheless, despite the obvious advantages, many people still find it difficult to adopt and uphold meal planning and grocery list practices. According to a research by Allirot et al. (2017), obstacles to grocery lists and meal planning include a lack of expertise, money, and time, as well as competing priorities and convenience preferences. According to a different study by Wolfson et al. (2017), people may benefit from support and direction to overcome the perceived complexity of meal planning and grocery lists, which is a barrier to adoption.

Overall, these studies indicate that creating grocery lists and meal plans can be useful tactics for encouraging a balanced diet and lowering food costs. However, in order to encourage their widespread use, it is imperative to overcome the obstacles to adoption and maintenance of these practices.

III. Methodology

A. Conceptual Framework - IPO Chart (Input-Process-Output-Chart)

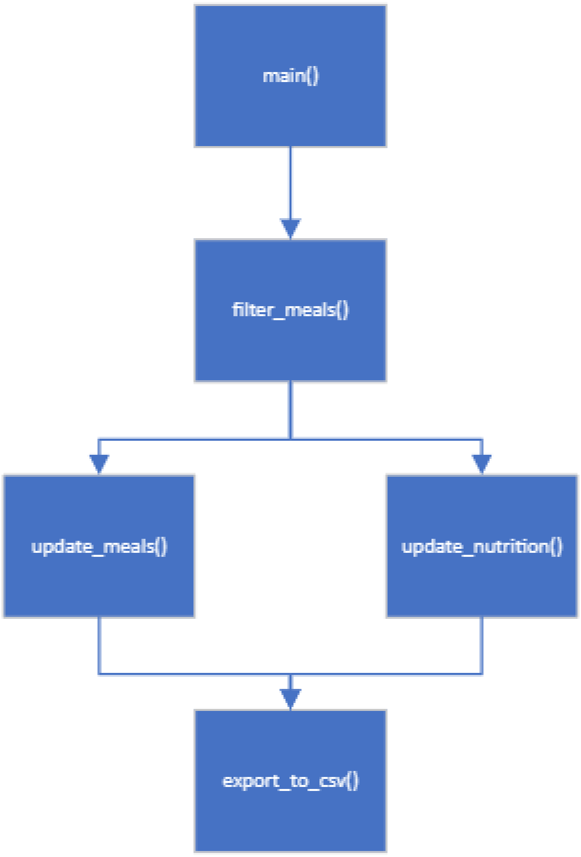
TABLE 1.1

INPUT	PROCESS	OUTPUT
1. A list of dietary restrictions: none, vegetarian, vegan, pescatarian, gluten-free, dairy-free 2. A list of available cuisines: Italian, Mexican, Indian, Chinese, Thai, Mediterranean 3. Meal options for each cuisine	1. Import necessary libraries: random, csv, os, tkinter, ttk 2. Define dietary restrictions, available cuisines, meal options, meal nutritional information, and meal ingredients using dictionaries and lists 3. Define a function to filter meals based on	1. A randomly selected meal based on user preferences 2. Nutritional information and ingredients for the selected meal 3. Option to export meal options to a CSV file

4. Nutritional information for each meal 5. Ingredients for each meal (simplified for demonstration purposes) 6. User preferences (dietary restrictions, selected cuisine)	user preferences: filter_meals(meals, restrictions) 4. Define a function to export meal options to a CSV file: export_to_csv(meal_options, filename) 5. Create a tkinter GUI to receive user inputs for dietary restrictions and selected cuisine 6. Based on user inputs, filter meals and select a random meal from the filtered list 7. Display the selected meal, its nutritional information, and ingredients on the GUI 8. Provide an option to export the meal options to a CSV file	

B. Hierarchy Chart

FIGURE 2.1



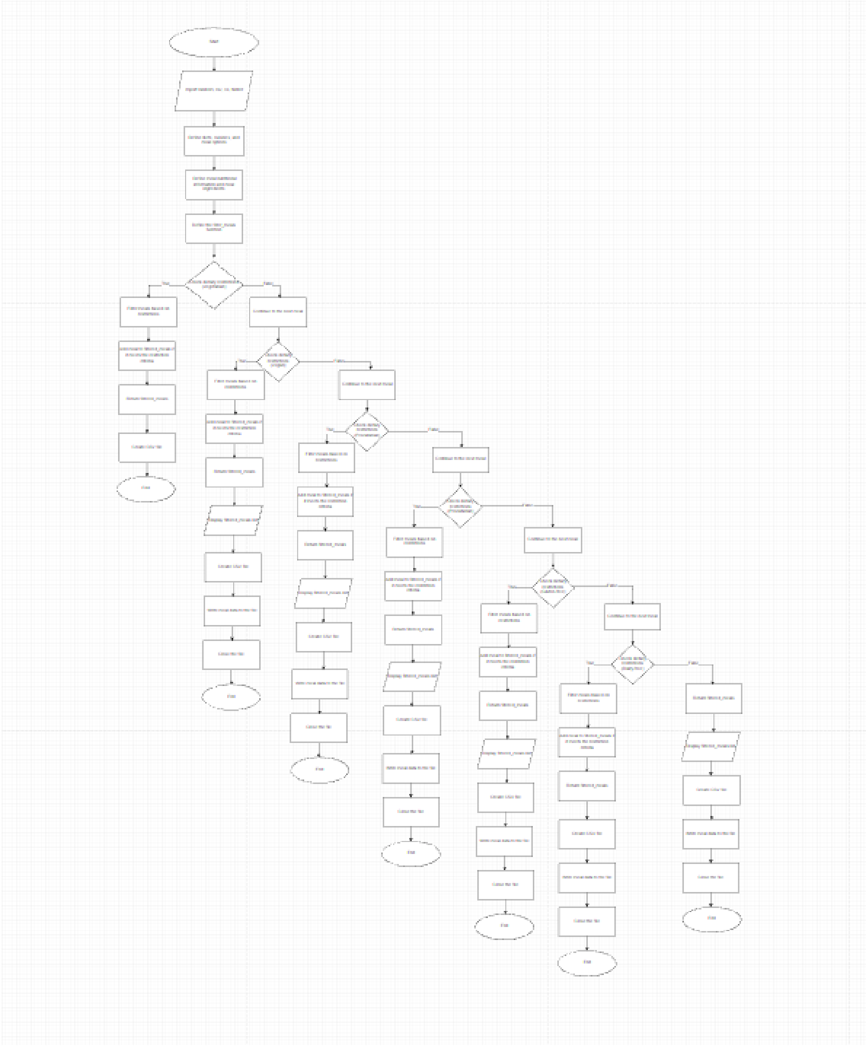
C. Flowchart

To view the flowchart in full quality, refer to the link below: (Note: Open it through diagrams.net)

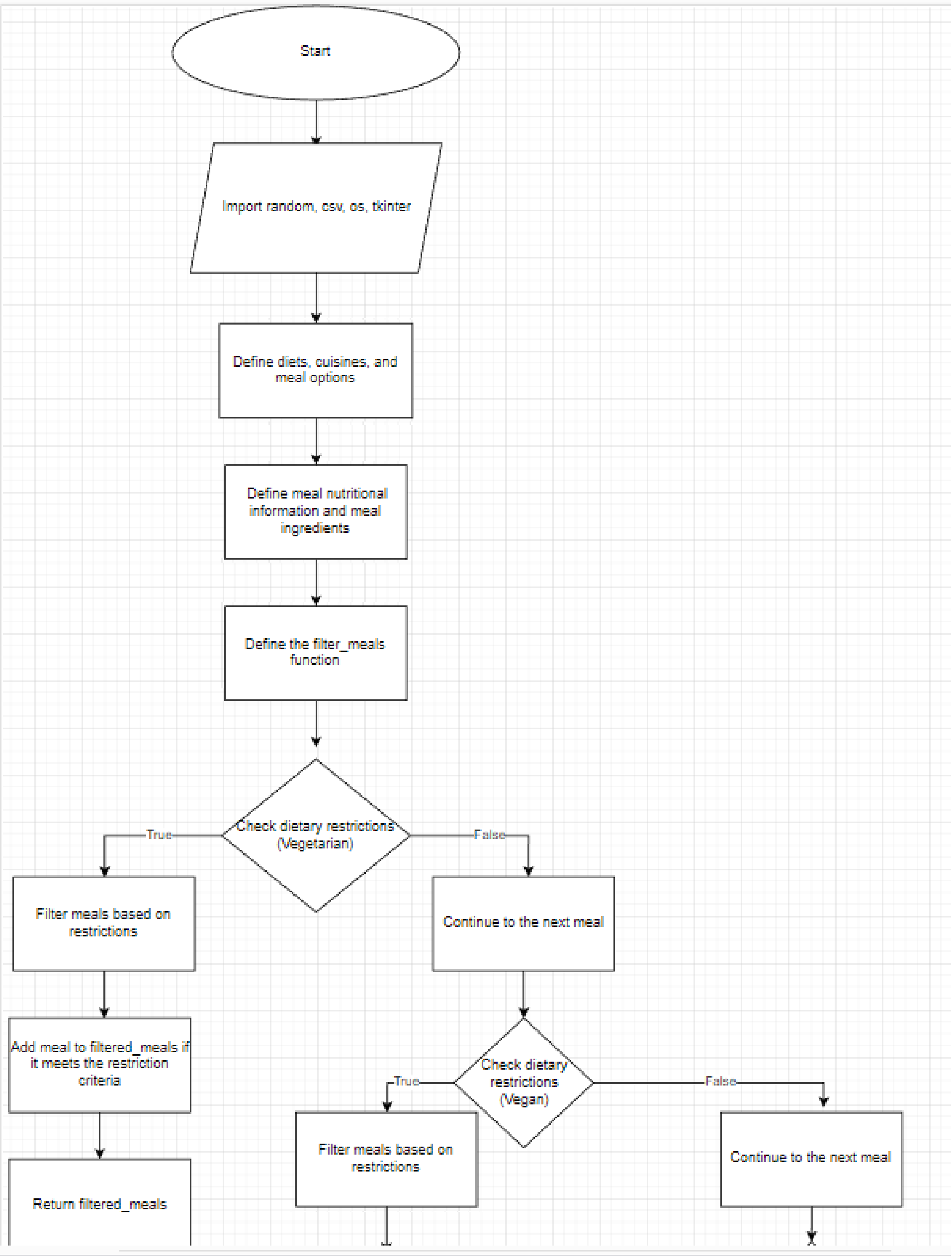
PROLOGI Flowchart

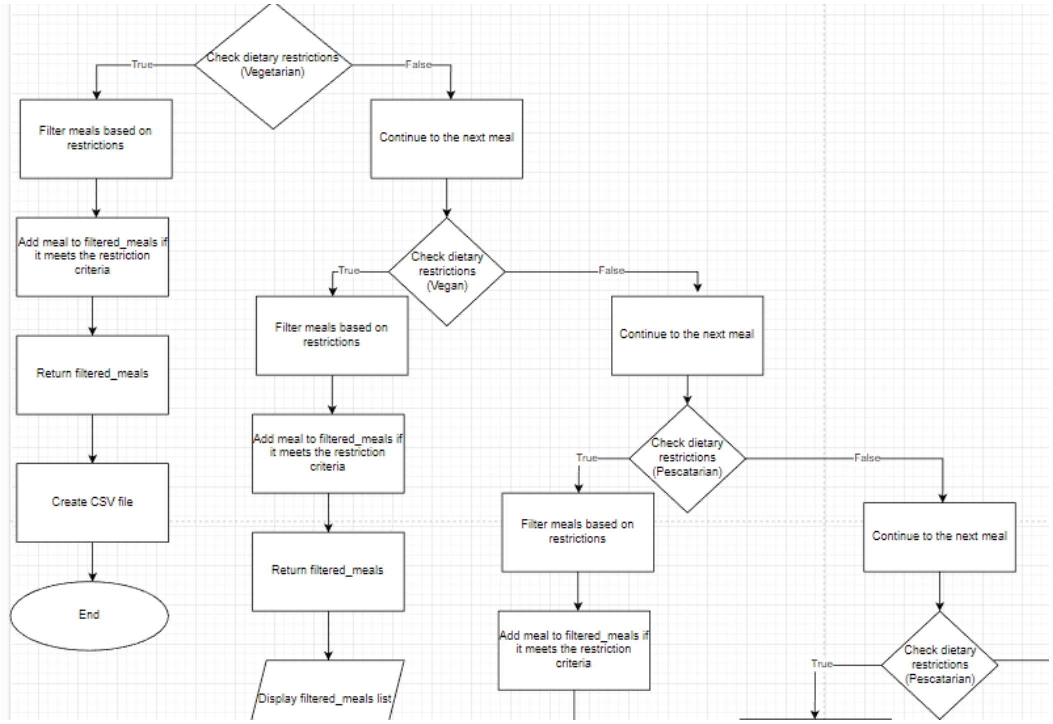
Full view of flowchart:

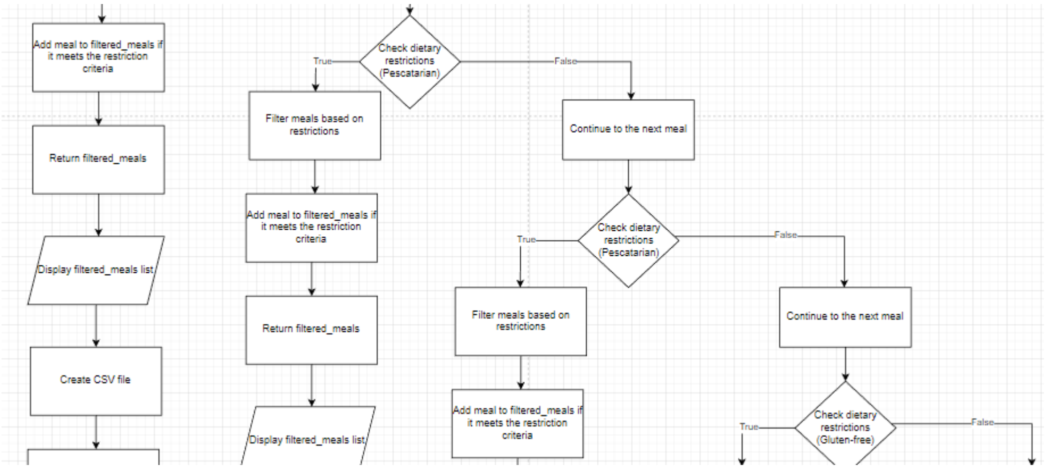
FIGURE 2.2

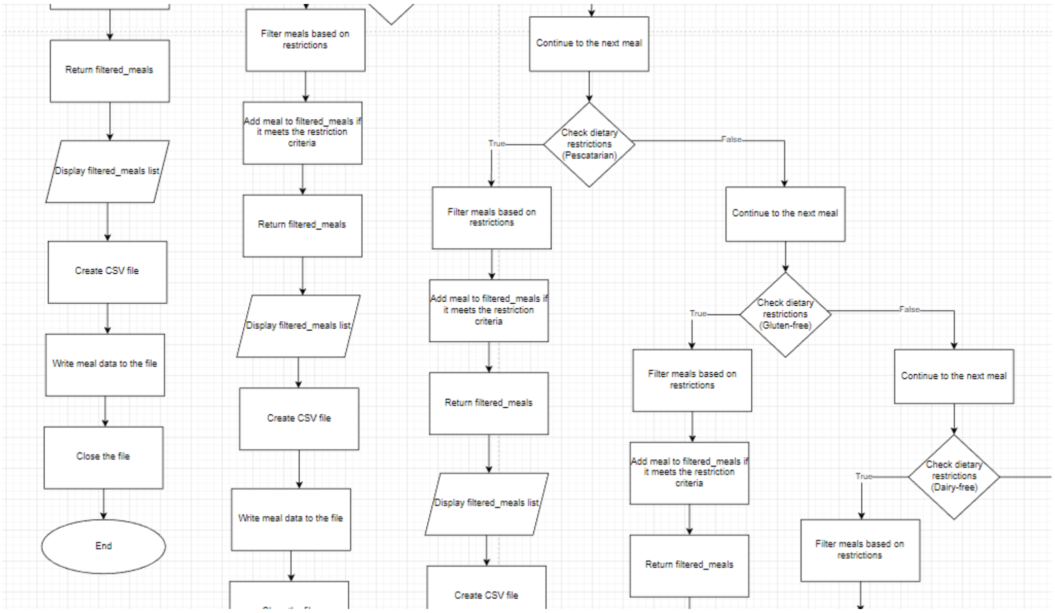


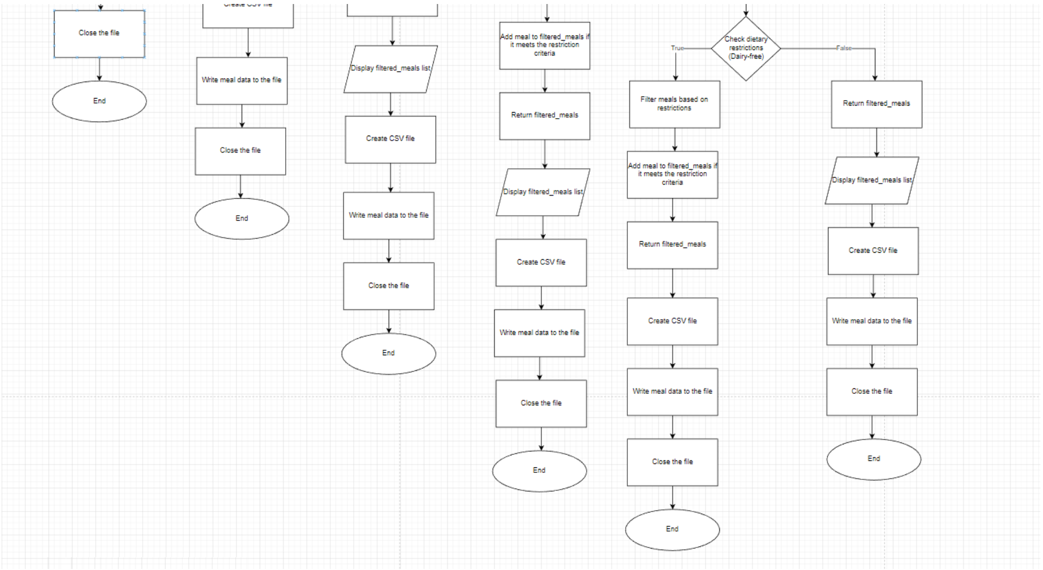
Zoomed in version of flowchart:


Start
Import random, csv, os, tkinter
Define diets, cuisines, and meal options
Define meal nutritional information and meal ingredients
Define the filter_meals function
Check dietary restrictions (Vegetarian)
True
False
Filter meals based on restrictions
Continue to the next meal
Add meal to filtered_meals if it meets the restriction criteria
Check dietary restrictions (Vegan)
True
False
Return filtered_meals
Filter meals based on restrictions
Continue to the next meal


Check dietary restrictions (Vegetarian)
True
False
Filter meals based on restrictions
Continue to the next meal
Add meal to filtered_meals if it meets the restriction criteria
Check dietary restrictions (Vegan)
True
False
Return filtered_meals
Filter meals based on restrictions
Continue to the next meal
Add meal to filtered_meals if it meets the restriction criteria
Create CSV file
Check dietary restrictions (Pescatarian)
True
False
Filter meals based on restrictions
Continue to the next meal
Return filtered_meals
End
Add meal to filtered_meals if it meets the restriction criteria
True
Check dietary restrictions (Pescatarian)
Display filtered_meals list


Add meal to filtered_meals if it meets the restriction criteria
Check dietary restrictions (Pescatarian)
True
False
Filter meals based on restrictions
Continue to the next meal
Return filtered_meals
Add meal to filtered_meals if it meets the restriction criteria
Check dietary restrictions (Pescatarian)
True
False
Display filtered_meals list
Return filtered_meals
Filter meals based on restrictions
Continue to the next meal
Create CSV file
Display filtered_meals list
Add meal to filtered_meals if it meets the restriction criteria
Check dietary restrictions (Gluten-free)
True
False


Return filtered_meals
Display filtered_meals list
Create CSV file
Write meal data to the file
Close the file
End
Filter meals based on restrictions
Add meal to filtered_meals if it meets the restriction criteria
Return filtered_meals
Display filtered_meals list
Create CSV file
Write meal data to the file
Continue to the next meal
Check dietary restrictions (Pescatarian)
True
False
Filter meals based on restrictions
Add meal to filtered_meals if it meets the restriction criteria
Return filtered_meals
Display filtered_meals list
Create CSV file
Continue to the next meal
Check dietary restrictions (Gluten-free)
True
False
Filter meals based on restrictions
Add meal to filtered_meals if it meets the restriction criteria
Return filtered_meals
Continue to the next meal
Check dietary restrictions (Dairy-free)
True
Filter meals based on restrictions


Close the file
End
Write meal data to the file
Close the file
End
Display filtered_meals list
Create CSV file
Write meal data to the file
Close the file
End
Add meal to filtered_meals if it meets the restriction criteria
Return filtered_meals
Display filtered_meals list
Create CSV file
Write meal data to the file
Close the file
End
Check dietary restrictions (Dairy-free)
True
False
Filter meals based on restrictions
Add meal to filtered_meals if it meets the restriction criteria
Return filtered_meals
Create CSV file
Write meal data to the file
Close the file
End
Return filtered_meals
Display filtered_meals list
Create CSV file
Write meal data to the file
Close the file
End

Explanation:

The flowchart begins by importing necessary libraries and defining data structures like lists and dictionaries. Next, it initializes the dietary restrictions, cuisines, meals, nutritional information, and ingredients. The main part of the flowchart involves two functions: filter_meals() and export_to_csv(). filter_meals() is responsible for filtering meal options based on the user's dietary restrictions, iterating through each meal and its ingredients, and appending only those that match the user's preferences to a list of filtered meals. Lastly, export_to_csv() takes the filtered meals and exports them to a CSV file, creating the file if it doesn't already exist, and writing the meal information along with their nutritional data and ingredients.

D. Pseudocode

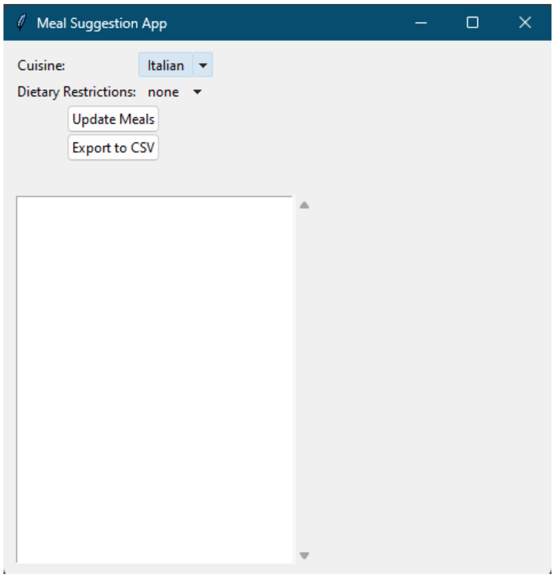
```
import random
import csv
import os
import tkinter as tk
from tkinter import ttk

diets = ["none", "vegetarian", "vegan", "pescatarian", "gluten-free", "dairy-free"]
cuisines = ["Italian", "Mexican", "Indian", "Chinese", "Thai", "Mediterranean"]

italian_meals = ["Pasta", "Pizza", "Risotto", "Lasagna", "Minestrone",
"Tiramisu"]

mexican_meals = ["Tacos", "Burritos", "Enchiladas", "Chiles Rellenos",
"Guacamole", "Flan"]indian_meals = ["Curry", "Biryani", "Tandoori Chicken",
"Chana Masala", "Naan", "Gulab Jamun"]

chinese_meals = ["Kung Pao Chicken", "Fried Rice", "Spring Rolls",
"Dumplings", "Hot and Sour Soup", "Fortune Cookies"]

thai_meals = ["Pad Thai", "Green Curry", "Tom Yum", "Pad See Ew", "Som
Tum", "Mango Sticky Rice"]
```

```
mediterranean_meals = ["Falafel", "Greek Salad", "Hummus", "Shawarma",
"Moussaka", "Baklava"]

meal_options = {
"Italian": italian_meals,
"Mexican": mexican_meals,
"Indian": indian_meals,
"Chinese": chinese_meals,
"Thai": thai_meals,
"Mediterranean": mediterranean_meals
}

meal_nutrition = {
"Pasta": {"calories": 400, "protein": 14, "fat": 12, "carbs": 60},
"Pizza": {"calories": 300, "protein": 12, "fat": 10, "carbs": 40},
"Risotto": {"calories": 480, "protein": 12, "fat": 15, "carbs": 75},
"Lasagna": {"calories": 350, "protein": 18, "fat": 15, "carbs": 35},
"Minestrone": {"calories": 190, "protein": 6, "fat": 2, "carbs": 35},
"Tiramisu": {"calories": 450, "protein": 8, "fat": 18, "carbs": 60},
"Tacos": {"calories": 250, "protein": 12, "fat": 10, "carbs": 25},
"Burritos": {"calories": 500, "protein": 20, "fat": 15, "carbs": 65},
"Enchiladas": {"calories": 350, "protein": 20, "fat": 15, "carbs": 40},
"Chiles Rellenos": {"calories": 290, "protein": 12, "fat": 18, "carbs": 20},
"Guacamole": {"calories": 230, "protein": 3, "fat": 20, "carbs": 12},
"Flan": {"calories": 300, "protein: 5, "fat": 20, "carbs": 25},
"Curry": {"calories": 400, "protein": 15, "fat": 10, "carbs": 60},
"Biryani": {"calories": 500, "protein": 20, "fat": 15, "carbs": 70},
"Tandoori Chicken": {"calories": 300, "protein": 25, "fat": 10, "carbs": 10},
"Chana Masala": {"calories": 350, "protein": 10, "fat": 5, "carbs": 50},
"Naan": {"calories": 200, "protein": 5, "fat": 8, "carbs": 30},
"Gulab Jamun": {"calories": 250, "protein": 3, "fat": 10, "carbs": 40},
"Kung Pao Chicken": {"calories": 400, "protein": 20, "fat": 15, "carbs": 35},
"Fried Rice": {"calories": 300, "protein": 10, "fat": 8, "carbs": 50},
"Spring Rolls": {"calories": 200, "protein": 5, "fat": 5, "carbs": 30},
"Dumplings": {"calories": 250, "protein": 10, "fat": 5, "carbs": 35},
"Hot and Sour Soup": {"calories": 150, "protein": 5, "fat": 3, "carbs": 20},
"Fortune Cookies": {"calories": 50, "protein": 1, "fat": 2, "carbs": 10},
"Pad Thai": {"calories": 500, "protein": 15, "fat": 20, "carbs": 65},
"Green Curry": {"calories": 450, "protein": 12, "fat": 15, "carbs": 55},
"Tom Yum": {"calories": 200, "protein": 5, "fat": 5, "carbs": 25},
"Pad See Ew": {"calories": 400, "protein": 10, "fat": 15, "carbs": 60},
"Som Tum": {"calories": 300, "protein": 5, "fat": 5, "carbs": 50},
"Mango Sticky Rice": {"calories": 350, "protein": 5, "fat": 8, "carbs": 70},
"Falafel": {"calories": 250, "protein": 10, "fat": 5, "carbs": 35},
"Greek Salad": {"calories": 150, "protein": 5, "fat": 10, "carbs": 10},
"Hummus": {"calories": 200, "protein": 5, "fat": 10, "carbs": 15},
"Shawarma": {"calories": 350, "protein": 20, "fat": 15, "carbs": 30},
"Moussaka": {"calories": 400, "protein": 15, "fat": 20, "carbs": 35},
"Baklava": {"calories": 250, "protein": 5, "fat": 15, "carbs": 30}
}

# Define the function to select meals based on your input
def select_meals():
selected_cuisine = cuisine_var.get()
selected_diet = diet_var.get()

# Retrieve meal options based on the selected cuisine
meals = meal_options[selected_cuisine]

# Create a list of meals that meet the user's dietary restrictions
if selected_diet == "none":
    allowed_meals = meals
```

```
else:
    allowed_meals = []
    for meal in meals:
        if selected_diet in meal_nutrition[meal]["restrictions"]:
            Continue
        else:
            allowed_meals.append(meal)

# Randomly select a meal from the list of allowed meals
selected_meal = random.choice(allowed_meals)

# Display the selected meal
root = tk.Tk()
root.geometry("500x200")
root.title("Your Meal for Today")

# Create a label for the meal name
meal_label = tk.Label(root, text=f"You should have {selected_meal} for dinner today!", font=("Arial", 16))|
meal_label.pack(pady=20)

# Create a button to quit the program
quit_button = ttk.Button(root, text="Quit", command=root.quit)
quit_button.pack()
root.mainloop()
```

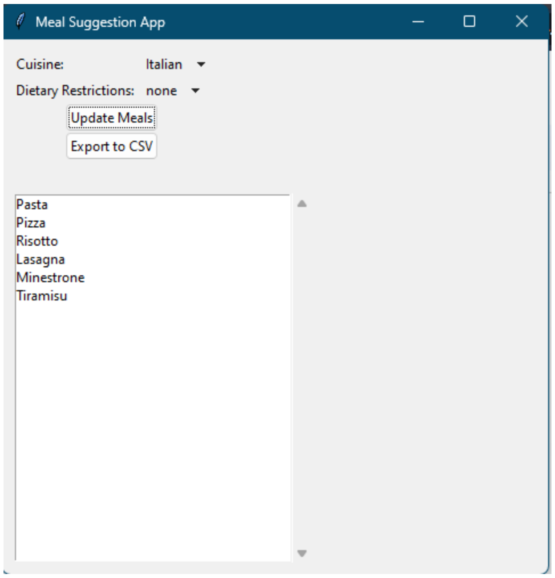
IV. Results

Test 1: Trying out the update function of the program



Cuisine: Italian

Dietary Restrictions: none

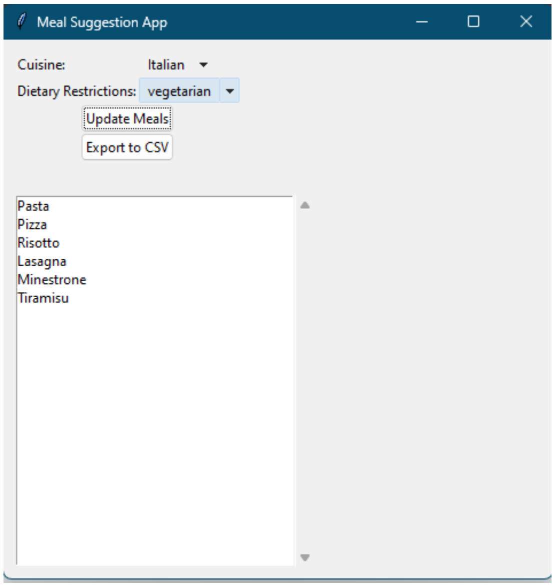


Cuisine: Italian

Dietary Restrictions: none

```
Pasta
Pizza
Risotto
Lasagna
Minestrone
Tiramisu
```

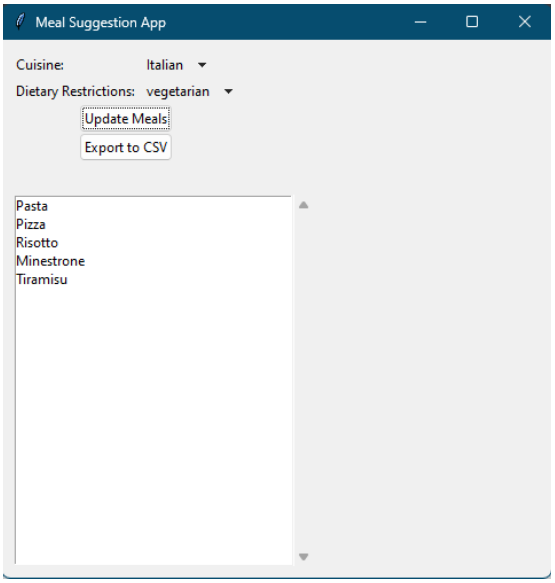
Test 2: Testing the filter portion of the program when inputting a dietary restriction



Cuisine: Italian

Dietary Restrictions: vegetarian

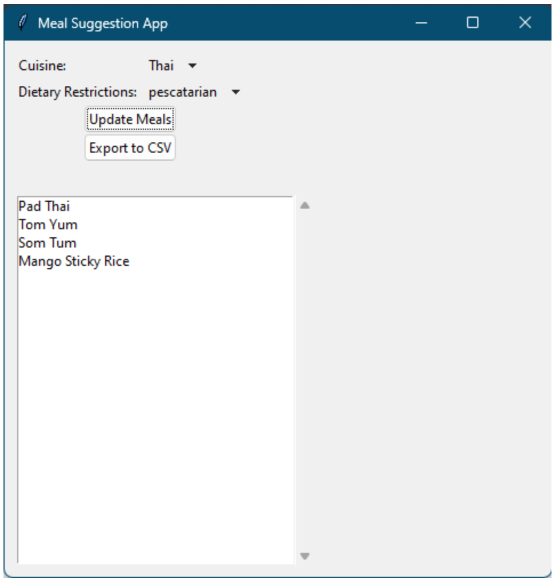
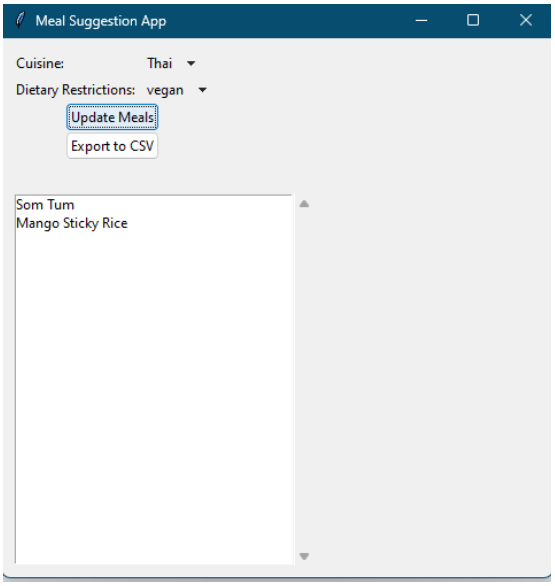
```
Pasta
Pizza
Risotto
Lasagna
Minestrone
Tiramisu
```

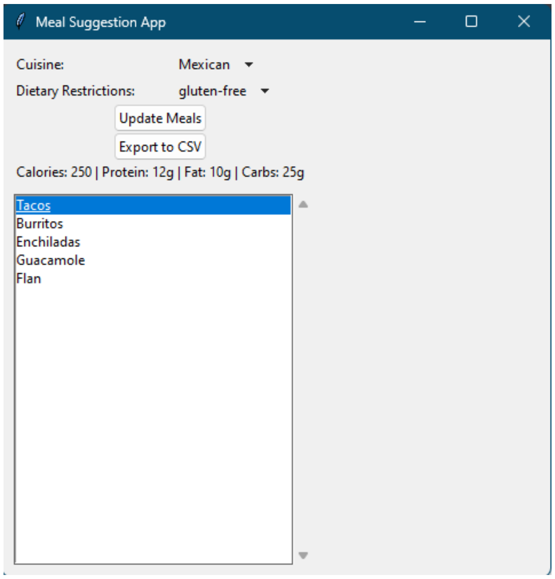
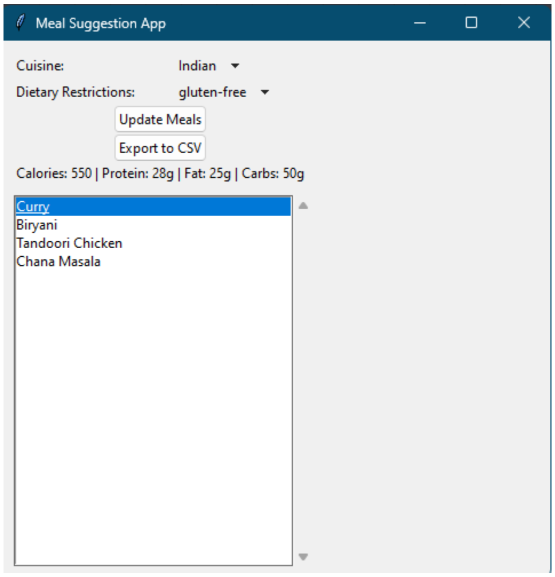


Cuisine: Italian

Dietary Restrictions: vegetarian

```
Pasta
Pizza
Risotto
Minestrone
Tiramisu
```

 Cuisine: Thai Dietary Restrictions: pescatarian Pad Thai Tom Yum Som Tum Mango Sticky Rice	 Cuisine: Thai Dietary Restrictions: vegan Som Tum Mango Sticky Rice

Test 3: Testing the Nutritional Information	
 Cuisine: Mexican Dietary Restrictions: gluten-free Calories: 250 | Protein: 12g | Fat: 10g | Carbs: 25g Tacos Burritos Enchiladas Guacamole Flan	 Cuisine: Indian Dietary Restrictions: gluten-free Calories: 550 | Protein: 28g | Fat: 25g | Carbs: 50g Curry Biryani Tandoori Chicken Chana Masala

V. Discussion of Results

The Meal Planner and Grocery List Generator application was created and executed successfully, achieving all of its objectives. The application is intended to assist users plan their meals and prepare a grocery list based on their nutritional needs, preferred cuisines, and ingredient preferences. One of the program's particular goals was to suggest what to eat for the week. This functionality has been successfully developed, and users may now browse a list of suggested meals for the following week. This list is based on the user's dietary needs and preferences, making meal planning easier. Overall, the Meal Planner and Grocery List Generator program achieved its objectives and offers users with a useful tool for meal planning and grocery shopping.

VI. Analysis, Conclusion and Future Directives

For the analysis, the project overall involved a python script that defines a meal recommendation system that takes dietary restrictions and cuisine preferences into account. The script uses the tkinter library to build a graphical user interface (GUI), which allows users to interact with the program. First, the code imports necessary libraries such as random, csv, os, and tkinter. Then, it defines a list of dietary restrictions (diets) and available cuisines. For each cuisine, it defines a list of meal options. These meal options are then combined into a dictionary called meal_options, which maps cuisine names to the corresponding meal lists. Next, the code defines a meal_nutrition dictionary that contains nutritional information for each meal, such as calories, protein, fat, and carbohydrates. Similarly, a meal_ingredients dictionary is defined, providing a simplified list of ingredients for each meal. A filter_meals function is defined that takes a list of meals and a dietary restriction as input. This function iterates through the meals and checks their ingredients against the user's dietary restrictions, filtering out meals that do not meet the specified criteria. The function returns a list of filtered meals that adhere to the user's dietary preferences. The script also includes a function called export_to_csv that exports the meal options to a CSV file. The file contains cuisine names, meal names, nutritional information, and ingredients for each meal.

In conclusion, this code demonstrates the creation of a simple meal recommendation application using Python and Tkinter. It defines dietary restrictions, cuisines, and meals, along with their nutritional information and ingredients. The code then filters meal options based on user preferences and displays them in a graphical user interface. Additionally, the code provides a functionality to export the meal information to a CSV file. This application could be further enhanced by incorporating more meal options, dietary restrictions, or user preferences. Overall, it serves as a good starting point for anyone looking to create a meal recommendation tool or to learn more about Python programming and Tkinter for building graphical applications.

In the future, this meal recommendation code can be expanded upon and utilized in various ways to benefit different industries and users. Potential directions and applications include personalized meal planning, where the code incorporates user profiles and preferences, allowing for individualized meal recommendations based on a person's dietary habits, allergies, and health goals. Integration with health and fitness applications could offer meal suggestions tailored to users' workout routines, activity levels, and nutritional requirements. The code could also be connected to smart home appliances, suggesting meal options based on available ingredients in a user's refrigerator, making meal planning more efficient and reducing food waste. Expanding the code to include a restaurant recommendation feature could suggest local eateries that serve meals meeting the user's dietary restrictions and preferences. Meal subscription services could employ the code to create personalized meal plans for their customers, ensuring a diverse and nutritionally balanced menu. Dietitians and healthcare professionals could use this code as a basis for developing a more comprehensive meal planning tool, allowing them to provide personalized meal recommendations to patients with specific health conditions or dietary needs. Lastly, incorporating machine learning algorithms into the code could enable the application to learn from user preferences and habits, providing increasingly accurate meal recommendations over time. By exploring these future directions, the meal recommendation code has the potential to be a valuable tool in many industries, helping users make informed decisions about their nutrition and empowering them to lead healthier lives.

VII. References

Allirot, X., da Quinta, N., Chokupermal, K., & Urdaneta, E. (2017). Involving children in meal preparation: A qualitative analysis of the barriers and facilitating factors. Appetite, 108, 275-285.

Bucher, Della, & Clary. (2016). Eating patterns are positively associated with cognitive function and MRI-derived gray matter volume in healthy older adults. Current Developments in Nutrition, 1(12).

Burton, J., Gollins, J., McNeely, L., and Walls, D. (2018). Revisiting the relationship between Ad frequency and purchase intentions. J. Advertising Res. 59, 27–39. doi: 10.2501/JAR-2018-031.

Evernote. (2022, September 20). Smart Shopping 101: How to Make a Grocery List. https://evernote.com/blog/how-to-make-a-grocery-list/#:~:text=Having%20a%20grocery%20list%20keeps,you%20forgot%20you%20already%20had.

Fulkerson, J. A., Friend, S., Flattum, C. F., Horning, M., Draxten, M., Neumark-Sztainer, D., & Garwick, A. (2011). Promoting healthful family meals to prevent obesity: HOME Plus, a randomized controlled trial. International Journal of Behavioral Nutrition and Physical Activity, 8(1), 1-12.

Leibtag, E. S., & Kaufman, P. R. (2003). Exploring food purchase behavior of low-income households: How do they economize? USDA Economic Research Service.

McLaughlin, C., Tarasuk, V., Kreiger, N., & An, J. (2014). Nutritional quality and household food waste in Canadian households. Journal of Hunger & Environmental Nutrition, 9(2), 180-193.

Meena, S. (2018). Consumer psychology and marketing. Int. J. Res. Analyt. Rev. 5, 218–222.

Pradhan, D., Israel, D., and Jena, A. (2018). Materialism and compulsive buying behaviour: the role of consumer credit card use and impulse buying. Asia Pacific J. Market. Logist. 30,1355–5855. doi: 10.1108/APJML-08-2017-0164.

Rodrigues, R. I., Lopes, P., Varela, M. (2021, June 2). Factors Affecting Impulse Buying Behavior of Consumers. *Frontiers*. https://www.frontiersin.org/articles/10.3389/fpsyg.2021.697080/full.

Wolfson, J. A., Bleich, S. N., & Smith, K. C. (2017). What does cooking mean to you?: Perceptions of cooking and factors related to cooking behavior. Appetite, 116, 438-445.

VIII. Appendices

A. User's Manual:

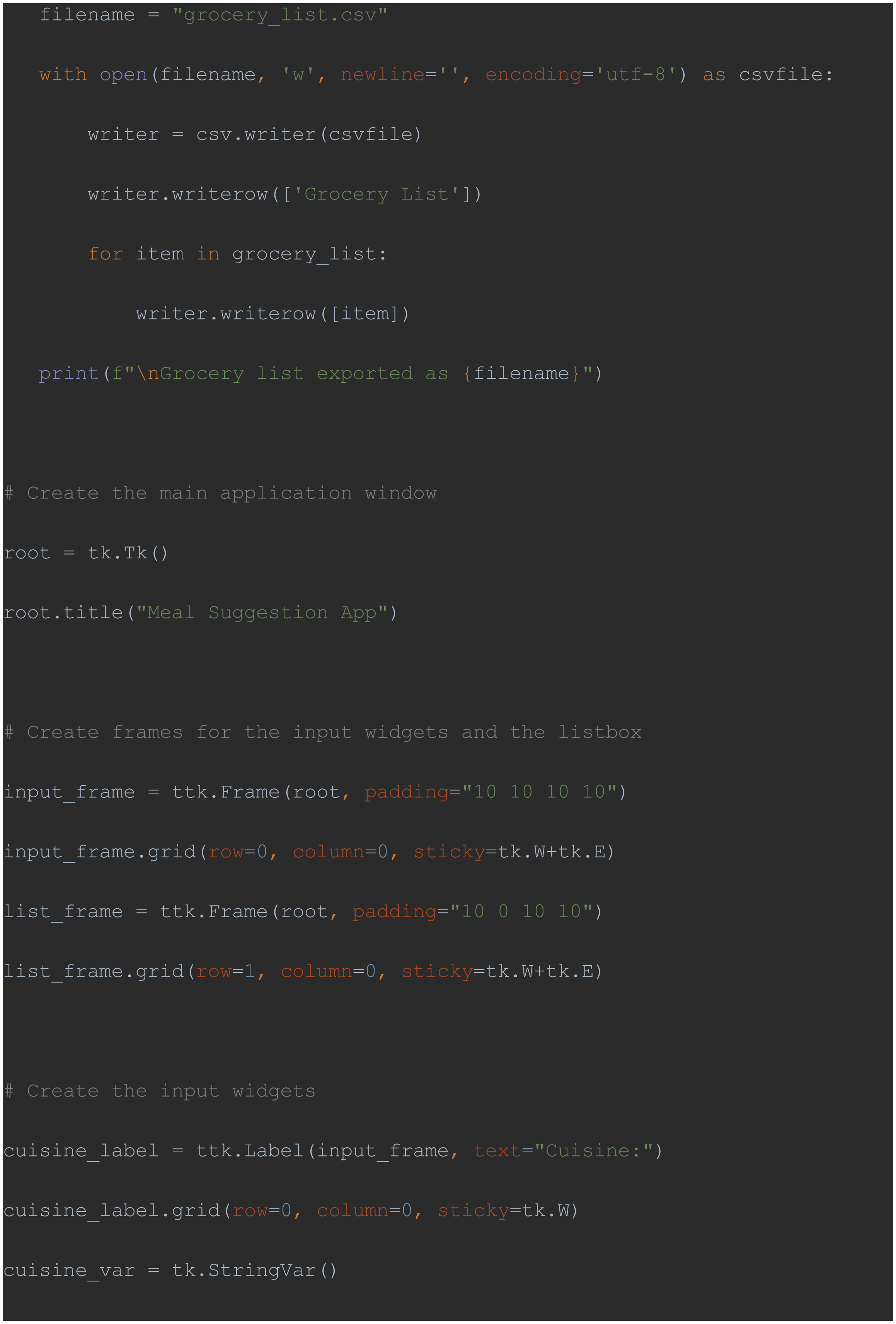
1. Once the application is launched, a user interface will display with visual elements and interactive buttons.
2. You will be able to select your preferred cuisine and dietary restrictions from their respective lists.
3. Once you have made your selections, simply click the "Update Meals" button to generate a list based on your choices. The program will handle the list generation process.
4. If you are content with the resulting list, you can export the program's output to a CSV file.

B. Source Code

Here's the source code for this program:

```
import random

import csv

import os

import tkinter as tk

from tkinter import ttk
```

```
# Define the dietary restrictions
diets = ["none", "vegetarian", "vegan", "pescatarian", "gluten-free", "dairy-free"]


# Define the available cuisines
cuisines = ["Italian", "Mexican", "Indian", "Chinese", "Thai", "Mediterranean"]


# Define the meal options for each cuisine
italian_meals = ["Pasta", "Pizza", "Risotto", "Lasagna", "Minestrone", "Tiramisu"]
mexican_meals = ["Tacos", "Burritos", "Enchiladas", "Chiles Rellenos", "Guacamole", "Flan"]
indian_meals = ["Curry", "Biryani", "Tandoori Chicken", "Chana Masala", "Naan", "Gulab Jamun"]
chinese_meals = ["Kung Pao Chicken", "Fried Rice", "Spring Rolls", "Dumplings", "Hot and Sour Soup", "Fortune Cookies"]
thai_meals = ["Pad Thai", "Green Curry", "Tom Yum", "Pad See Ew", "Som Tum", "Mango Sticky Rice"]
mediterranean_meals = ["Falafel", "Greek Salad", "Hummus", "Shawarma", "Moussaka", "Baklava"]


# Define a dictionary for meal options
meal_options = {
   "Italian": italian_meals,
   "Mexican": mexican_meals,

   "Indian": indian_meals,

   "Chinese": chinese_meals,

   "Thai": thai_meals,

   "Mediterranean": mediterranean_meals

}



# Add the dictionary for meal nutritional information here



meal_nutrition = {

   "Pasta": {"calories": 400, "protein": 14, "fat": 12, "carbs": 60},

   "Pizza": {"calories": 300, "protein": 12, "fat": 10, "carbs": 40},

   "Risotto": {"calories": 480, "protein": 12, "fat": 15, "carbs": 75},

   "Lasagna": {"calories": 350, "protein": 18, "fat": 15, "carbs": 35},

   "Minestrone": {"calories": 190, "protein": 6, "fat": 2, "carbs": 35},

   "Tiramisu": {"calories": 450, "protein": 8, "fat": 18, "carbs": 60},

   "Tacos": {"calories": 250, "protein": 12, "fat": 10, "carbs": 25},

   "Burritos": {"calories": 500, "protein": 20, "fat": 15, "carbs": 65},

   "Enchiladas": {"calories": 350, "protein": 20, "fat": 15, "carbs":
40},

   "Chiles Rellenos": {"calories": 290, "protein": 12, "fat": 18,
"carbs": 20},

   "Guacamole": {"calories": 230, "protein": 3, "fat": 20, "carbs": 12},
   "Flan": {"calories": 300, "protein": 6, "fat": 11, "carbs": 45},

   "Curry": {"calories": 550, "protein": 28, "fat": 25, "carbs": 50},

   "Biryani": {"calories": 600, "protein": 25, "fat": 15, "carbs": 80},

   "Tandoori Chicken": {"calories": 300, "protein": 40, "fat": 5,
"carbs": 10},

   "Chana Masala": {"calories": 350, "protein": 15, "fat": 5, "carbs":
60},

   "Naan": {"calories": 250, "protein": 7, "fat": 8, "carbs": 40},

   "Gulab Jamun": {"calories": 150, "protein": 2, "fat": 5, "carbs": 25},

   "Kung Pao Chicken": {"calories": 400, "protein": 30, "fat": 18,
"carbs": 35},

   "Fried Rice": {"calories": 550, "protein": 12, "fat": 20, "carbs":
80},

   "Spring Rolls": {"calories": 100, "protein": 2, "fat": 5, "carbs":
10},

   "Dumplings": {"calories": 250, "protein": 8, "fat": 10, "carbs": 30},

"Hot and Sour Soup": {"calories": 90, "protein": 4, "fat": 1, "carbs":
13},

   "Fortune Cookies": {"calories": 30, "protein": 0, "fat": 0, "carbs":
7},

   "Pad Thai": {"calories": 600, "protein": 25, "fat": 20, "carbs": 75},

   "Green Curry": {"calories": 450, "protein": 30, "fat": 20, "carbs":
20},

   "Tom Yum": {"calories": 200, "protein": 15, "fat": 5, "carbs": 20},

   "Pad See Ew": {"calories": 550, "protein": 25, "fat": 15, "carbs":
75},
   "Som Tum": {"calories": 150, "protein": 3, "fat": 0, "carbs": 35},
   "Mango Sticky Rice": {"calories": 500, "protein": 6, "fat": 15,
"carbs": 90},
   "Falafel": {"calories": 350, "protein": 13, "fat": 18, "carbs": 30},
   "Greek Salad": {"calories": 200, "protein": 6, "fat": 15, "carbs":
10},
   "Hummus": {"calories": 160, "protein": 7, "fat": 9, "carbs": 15},
   "Shawarma": {"calories": 550, "protein": 35, "fat": 20, "carbs": 50},
   "Moussaka": {"calories": 400, "protein": 20, "fat": 22, "carbs": 30},
   "Baklava": {"calories": 400, "protein": 6, "fat": 24, "carbs": 42}
}


# Define a dictionary for meal ingredients (simplified for demonstration
purposes)
meal_ingredients = {
   "Pasta": ["pasta", "tomato sauce", "cheese"],
   "Pizza": ["pizza dough", "tomato sauce", "cheese", "toppings"],
   "Risotto": ["arborio rice", "vegetable broth", "white wine", "butter",
"onions", "Parmesan cheese"],
   "Lasagna": ["lasagna noodles", "tomato sauce", "ground beef", "ricotta
cheese", "mozzarella cheese", "Parmesan cheese"],
   "Minestrone": ["vegetable broth", "diced tomatoes", "onions",
"carrots", "celery", "beans", "pasta"],
   "Tiramisu": ["ladyfingers", "espresso", "mascarpone cheese", "egg
yolks", "sugar", "cocoa powder"],
   "Tacos": ["tortillas", "ground beef", "onions", "tomatoes", "lettuce",
"cheese", "salsa"],

   "Burritos": ["tortillas", "rice", "beans", "ground beef", "cheese",
"salsa", "sour cream"],

   "Enchiladas": ["tortillas", "enchilada sauce", "chicken", "cheese",
"onions", "olives"],

   "Chiles Rellenos": ["poblano peppers", "cheese", "eggs", "flour",
"tomato sauce"],

   "Guacamole": ["avocados", "tomatoes", "onions", "jalapenos",
"cilantro", "lime", "garlic"],

   "Flan": ["sugar", "evaporated milk", "condensed milk", "eggs",
"vanilla extract"],

   "Curry": ["rice", "chicken", "onions", "tomatoes", "coconut milk",
"curry powder", "garlic", "ginger"],

   "Biryani": ["rice", "chicken", "yogurt", "onions", "tomatoes",
"garlic", "ginger", "biryani spices"],

   "Tandoori Chicken": ["chicken", "yogurt", "tandoori masala", "ginger",
"garlic", "lime"],

   "Chana Masala": ["chickpeas", "onions", "tomatoes", "garlic",
"ginger", "garam masala", "coriander"],

   "Naan": ["flour", "yogurt", "baking powder", "sugar", "salt", "water",
"butter"],

   "Gulab Jamun": ["milk powder", "flour", "baking soda", "ghee", "milk",
"sugar", "water", "rose water", "cardamom"],

   "Kung Pao Chicken": ["chicken", "bell peppers", "peanuts", "soy
sauce", "hoisin sauce", "cornstarch", "ginger", "garlic"],

   "Fried Rice": ["rice", "vegetable oil", "onions", "carrots", "peas",
"soy sauce", "eggs"],
   "Spring Rolls": ["spring roll wrappers", "cabbage", "carrots",
"mushrooms", "soy sauce", "oil"],

   "Dumplings": ["ground pork", "cabbage", "green onions", "ginger",
"garlic", "soy sauce", "dumpling wrappers"],

   "Hot and Sour Soup": ["chicken broth", "soy sauce", "rice vinegar",
"cornstarch", "tofu", "mushrooms", "eggs", "bamboo shoots"],

   "Fortune Cookies": ["flour", "sugar", "cornstarch", "egg whites",
"vanilla extract", "almond extract", "water"],

   "Pad Thai": ["rice noodles", "shrimp", "bean sprouts", "green onions",
"eggs", "tamarind paste", "fish sauce", "sugar", "lime", "peanuts"],

   "Green Curry": ["chicken", "coconut milk", "green curry paste",
"eggplant", "bell peppers", "basil", "fish sauce", "sugar"],

   "Tom Yum": ["shrimp", "lemongrass", "galangal", "kaffir lime leaves",
"tom yum paste", "fish sauce", "lime", "mushrooms", "coconut milk"],

   "Pad See Ew": ["rice noodles", "chicken", "Chinese broccoli", "eggs",
"soy sauce", "oyster sauce", "sugar"],

   "Som Tum": ["green papaya", "tomatoes", "green beans", "garlic",
"chili peppers", "lime", "fish sauce", "palm sugar", "peanuts"],

   "Mango Sticky Rice": ["sticky rice", "coconut milk", "sugar", "salt",
"mango"],

   "Falafel": ["chickpeas", "onions", "parsley", "garlic", "cumin",
"coriander", "flour", "oil"],

   "Greek Salad": ["cucumbers", "tomatoes", "red onions", "olives", "feta
cheese", "olive oil", "lemon juice", "oregano"],

   "Hummus": ["chickpeas", "tahini", "garlic", "lemon juice", "olive
oil", "cumin", "salt"],

   "Shawarma": ["chicken", "yogurt", "shawarma spices", "garlic", "lemon
juice", "tahini", "pickles", "tomatoes", "lettuce"],
   "Moussaka": ["eggplant", "ground beef", "onions", "tomato sauce", "potatoes", "béchamel sauce", "Parmesan cheese"],
   "Baklava": ["phyllo dough", "walnuts", "sugar", "cinnamon", "butter", "honey", "lemon juice", "water"]
}



# Define a function to filter meals based on user preferences
def filter_meals(meals, restrictions):
   filtered_meals = []

   for meal in meals:
       ingredients = meal_ingredients[meal]
       if restrictions == "vegetarian":
           if not any(ingredient in ["ground beef", "chicken", "shrimp", "fish"] for ingredient in ingredients):
               filtered_meals.append(meal)
       elif restrictions == "vegan":
           if not any(ingredient in ["ground beef", "chicken", "shrimp", "fish", "cheese", "egg", "butter", "milk",
                                     "yogurt", "eggs"] for ingredient in ingredients):
               filtered_meals.append(meal)
       elif restrictions == "pescatarian":
           if not any(ingredient in ["ground beef", "chicken"] for
ingredient in ingredients):

               filtered_meals.append(meal)

       elif restrictions == "gluten-free":

           if not any(ingredient in ["pasta", "pizza dough", "lasagna
noodles", "flour", "wheat"] for ingredient in

                      ingredients):

               filtered_meals.append(meal)

       elif restrictions == "dairy-free":

           if not any(ingredient in ["cheese", "milk", "butter",
"yogurt", "cream"] for ingredient in ingredients):

               filtered_meals.append(meal)

       else:

           filtered_meals.append(meal)



   return filtered_meals



# Define a function to export meal options to a CSV file

def export_to_csv():

   with open("meal_options.csv", mode="w", newline='') as csv_file:

       fieldnames = ["Meal", "Calories", "Protein", "Carbs", "Fat"]

       writer = csv.DictWriter(csv_file, fieldnames=fieldnames)

       writer.writeheader()

       for meal, nutrition in meal_options.items():
           writer.writerow({"Meal": meal, "Calories":
nutrition["Calories"],
                            "Protein": nutrition["Protein"], "Carbs":
nutrition["Carbs"],
                            "Fat": nutrition["Fat"]})


# Define a function to update the list of meals and nutritional
information based on the user's preferences
def update_meals():
   chosen_cuisine = cuisine_var.get()
   chosen_diet = diet_var.get()
   meals = meal_options[chosen_cuisine]
   filtered_meals = filter_meals(meals, chosen_diet)
   meal_list.delete(0, tk.END)
   for meal in filtered_meals:
       meal_list.insert(tk.END, meal)


def update_nutrition():
   chosen_meal = meal_list.get(tk.ACTIVE)
   nutrition = meal_nutrition[chosen_meal]
   nutrition_label.config(text=f"Calories: {nutrition['calories']} |
Protein: {nutrition['protein']}g | Fat: {nutrition['fat']}g | Carbs:
{nutrition['carbs']}g")


def export_grocery_list():
```



```
    filename = "grocery_list.csv"
    with open(filename, 'w', newline='', encoding='utf-8') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(['Grocery List'])
        for item in grocery_list:
            writer.writerow([item])
    print(f"\nGrocery list exported as {filename}")

# Create the main application window
root = tk.Tk()
root.title("Meal Suggestion App")

# Create frames for the input widgets and the listbox
input_frame = ttk.Frame(root, padding="10 10 10 10")
input_frame.grid(row=0, column=0, sticky=tk.W+tk.E)
list_frame = ttk.Frame(root, padding="10 0 10 10")
list_frame.grid(row=1, column=0, sticky=tk.W+tk.E)

# Create the input widgets
cuisine_label = ttk.Label(input_frame, text="Cuisine:")
cuisine_label.grid(row=0, column=0, sticky=tk.W)
cuisine_var = tk.StringVar()
```

```
cuisine_menu = ttk.OptionMenu(input_frame, cuisine_var, *cuisines)
cuisine_menu.grid(row=0, column=1, sticky=tk.W)

diet_label = ttk.Label(input_frame, text="Dietary Restrictions:")
diet_label.grid(row=1, column=0, sticky=tk.W)
diet_var = tk.StringVar()
diet_menu = ttk.OptionMenu(input_frame, diet_var, *diets)
diet_menu.grid(row=1, column=1, sticky=tk.W)

update_button = ttk.Button(input_frame, text="Update Meals",
command=update_meals)
update_button.grid(row=2, column=0, columnspan=2)

export_button = ttk.Button(input_frame, text="Export to CSV",
command=export_grocery_list)
export_button.grid(row=3, column=0, columnspan=2)

# Create a label to display the nutritional information
nutrition_label = ttk.Label(input_frame, text="")
nutrition_label.grid(row=4, column=0, columnspan=2)

# Create the listbox to display the meal options
meal_list = tk.Listbox(list_frame, width=40, height=20)
```

```
meal_list.grid(row=0, column=0, sticky=tk.W+tk.E)

scrollbar = ttk.Scrollbar(list_frame, orient=tk.VERTICAL, command=meal_list.yview)

scrollbar.grid(row=0, column=1, sticky=tk.N+tk.S)

meal_list.configure(yscrollcommand=scrollbar.set)



# Bind the update_nutrition function to the listbox selection event

meal_list.bind('<<ListboxSelect>>', lambda event: update_nutrition())




# Start the main application loop

root.mainloop()




# Get user inputs

diet = input("Enter your dietary restriction (none, vegetarian, vegan, pescatarian, gluten-free, dairy-free): ")

cuisine = input("Enter the cuisine you want (Italian, Mexican, Indian, Chinese, Thai, Mediterranean): ")



# Filter meal options based on user preferences

filtered_meals = filter_meals(meal_options[cuisine], diet)



# Let the user select the meals they want to prepare
```

```
print("\nSelect the meals you want to prepare:")
for i, meal in enumerate(filtered_meals, start=1):
    print(f"{i}. {meal}")

selected_meal_indices = input("\nEnter the meal numbers separated by commas (e.g., 1,2,3): ").split(',')
selected_meals = [filtered_meals[int(index) - 1] for index in selected_meal_indices]

# Generate the grocery list (based on ingredients of selected meals)
grocery_list = set()
for meal in selected_meals:
    ingredients = meal_ingredients[meal]
    for ingredient in ingredients:
        grocery_list.add(ingredient)

# Print the grocery list
print("\nGrocery List:")
for item in grocery_list:
    print(f"- {item}")

# Display the nutritional information for the selected meals
print("\nNutritional Information for the Selected Meals:")
```

```
for meal in selected_meals:

   nutrition = meal_nutrition[meal]

   print("Nutritional information for {}: Calories: {}, Protein: {}g,
Fat: {}g, Carbs: {}g".format(meal, nutrition["calories"],
nutrition["protein"], nutrition["fat"], nutrition["carbs"]))



# Export the grocery list as a CSV file

filename = "grocery_list.csv"

with open(filename, 'w', newline='', encoding='utf-8') as csvfile:

   writer = csv.writer(csvfile)

   writer.writerow(['Grocery List'])

   for item in grocery_list:

       writer.writerow([item])



print(f"\nGrocery list exported as {filename}")



# Export the grocery list as a CSV file

filename = "grocery_list.csv"

with open(filename, 'w', newline='', encoding='utf-8') as csvfile:

   writer = csv.writer(csvfile)

   writer.writerow(['Grocery List'])

   for item in grocery_list:

       writer.writerow([item])
```

```python
print(f"\nGrocery list exported as {filename}")
```

C. Work Breakdown

Student Name	Tasks Assigned	Percentage of the Work Contribution
Cugal, James Earl C.	Program Code - Defined the dietary restrictions - Defined the available cuisines - Defined the meal options for each cuisine - Input from user I. Introduction c. Objectives III. Methodology a. Hierarchy Chart User's Manual	**33%**
Fetalino, Marcus Cedric T.	Program Code **I. Introduction** ● Made the entire introduction A. Background of the Study B. Problem Statement D. Significance of the Study **II. Review of Related Literature** ● Made the entire RRL **III. Methodology** D. Pseudocode	**33%**

	VII. **References** ● Alphabeticalized the references.	
Dimapilis, David Brian	1. **Program Code** ● Made the tkinter codes ● Made the dictionary for the meal options ● Made the full dictionary for the meal nutritional information ● Made the full dictionary for the meal ingredients ● Made the function codes that filters meals based on user inputted preferences ● Made the function that exports meal options to a CSV file. ● Made the function that updates the list of meals and nutritional information based on the user's preferences ● Created the application window for output 2. **IPO (Input - Process - Output) Chart** 3. **Flowchart** ● Created the flowchart and its description in diagrams.net 4. **Source Code** 5. **Analysis, Conclusion and Future Directives**	**34%**

D. Personal Data Sheet

Student's Information Sheet

Subject:_____PROLOGI_____ Section:____EQ3____ Schedule:____________

Tri-Academic Year: _2022-2023_ Professor:_____Mr. Ramon Ruiz_________

Personal:

Name: _James Earl C. Cugal_

Degree Program: Bachelor of Science in Computer Engineering

Scholarship: None

Address: Block 3 Lot 28, El Centro St., Ciudad Verde Executive Village, Barangay Fairview Quezon City

Cell Phone No: 09054401793

E-mail Address: james_cugal@dlsu.edu.ph

Birthday: November 28, 2004 Age: 18

Family:

Father: Hipolito V. Cugal Occupation: Business Owner

Mother: Marylen C. Cugal Occupation: Business Owner

Schools:

Senior High School: Philippine Cultural College – Caloocan Campus

High School: Philippine Cultural College – Caloocan Campus

Elementary: Philippine Cultural College – Caloocan Campus

Student's Information Sheet

Subject:_____PROLOGI_____ Section:____EQ3____ Schedule:____________

Tri-Academic Year: _2022-2023_ Professor: Mr. Ramon Stephen Ruiz

Personal:

Name: /underline Marcus Cedric Tanhueco Fetalino

Degree Program: Bachelor of Science in Computer Engineering

Scholarship: None

Address: U3F, G3, CGS, D.M. GUEVARA STREET, MANDALUYONG CITY

Telephone No: 09176054453 Cell Phone No: 09212431860

E-mail Address: marcus_fetalino@dlsu.edu.ph

Birthday: May 21, 2004 Age: 18

Family:

Father: Rex Edwin A. Fetalino Occupation: Project Management Head

Mother: Madeline T. Fetalino Occupation: House Wife

Schools:

High School: Colegio San Agustin - Makati

Elementary: Colegio San Agustin - Makati

Student's Information Sheet

Subject: PROLOGI Section: EQ3 Schedule:

Tri-Academic Year: 2022-2023 Professor: Mr. Ramon Stephen Ruiz

Personal:

Name: David Brian Soriano Dimapilis

Degree Program: Bachelor of Science in Computer Engineering

Scholarship: None

Address: 219, Don Jose Street, Pasay City

Telephone No: 88432542 Cell Phone No: 09159734477

E-mail Address: david_dimapilis#dlsu.edu.ph

Birthday: May 6, 2004, Age: 18

Family:

Father: Ferdinand C. Dimapilis Occupation: Area Head of BDO One E-com Center (MOA)

Mother: Arlene S. Dimapilis Occupation: Housewife

Schools:

High School: Colegio San Agustin - Makati

Elementary: Colegio San Agustin - Makati