Assignment – 1 Daily

Calorie Tracker CLI

Name – Jagrit Jaggi

Roll no. – 2501730191

Course – B-Tech CSE AI/ML

Section - A

Key Features Implemented

- **Task 1:** Welcome screen with descriptive banner using string repetition and formatted output
- **Task 2:** Dynamic meal input loop storing data in parallel lists `meals`and `calories`
- **Task 3:** Total sum calculation via `sum()`, average computation, and user-defined daily limit input
- **Task 4:** Conditional warning system using `if-else`statements with comparison operators
- **Task 5:** Formatted summary table with left-aligned meal names and precise calorie display using f-strings
- **Task 6 (Bonus):** Timestamped file export using `datetime`module and `open()`in write mode

Technical Implementation

Data Structures

The program utilizes parallel lists to store meal names and calorie values, enabling synchronized iteration through meal-calorie pairs using the `zip()`function. This approach is simple yet effective for the assignment scope.

Control Flow

The application implements a linear flow with a single loop for meal input and conditional branching for limit comparison and file saving options. The logic is straightforward and easy to follow.

String Formatting

F-strings are extensively used for dynamic output formatting, including left-alignment of text (`:<20`) and decimal precision (`.2f`) for numerical display. Escape sequences like `\n`manage line breaks and spacing effectively.

File I/O Operations

The bonus task employs context managers (`with`statement) for safe file handling, ensuring proper file closure after write operations. Timestamps are generated using `datetime.datetime.now()`for unique file naming.

Learning Outcomes

By completing this assignment, the student has demonstrated proficiency in:

- Input/output operations using `input()`and `print()`
- Type conversion between strings, integers, and floats
- Collection management with Python lists

- Loop constructs (`for`loops with `range()`)
- Conditional statements (`if-else`)
- Mathematical operations (`sum()`, arithmetic calculations)
- String manipulation and formatting with f-strings
- File operations with read/write access
- Code organization and commenting practices

Assignment Compliance

The project successfully fulfills all assignment requirements from Lab-Assignment-1:

1. Project setup with proper script header and welcome message
2. Data collection with loop-based input handling
3. Calorie calculations including totals and averages
4. Warning system based on daily limit comparison
5. Neatly formatted summary output
6. Bonus file saving functionality with timestamps

All code is original, properly commented, and demonstrates clean programming practices suitable for submission via GitHub repository as specified in the assignment rubric.

Code

```
import datetime

#Task 1:

Introduction

print("="*60)
print(" Welcome to the Daily Calorie Tracker CLI ")
print("="*60)
print("This tool helps you log your meals and track total calorie
intake.") print("You can compare it against your daily limit and even
save a report.")

#Task 2: Data Collection

meals = []
calories = []
num_meals = int(input("How many meals did you have
today?"))
```

```
for i in range(num_meals):
meal_name = input(f"\nEnter name of meal {i+1}: ")
calorie_value = float(input(f"Enter calories for
{meal_name}: ")) meals.append(meal_name)
calories.append(calorie_value)

#Task 3: Calorie Calculations

total_calories = sum(calories)
average_calories = total_calories / len(calories)
daily_limit = float(input("\nEnter your daily calorie limit:

")) #Task 4: Warning System

if total_calories > daily_limit:
status_message = " You have exceeded your daily calorie
limit! " else:
status_message = " Great! You are within your daily calorie limit.

" #Task 5: Formatted Output
```

```
print(" Daily Summary ")
print(f"{'Meal
Name':<20}{'Calories'}") print("-" *
35)
for meal, cal in zip(meals,
calories):
print(f"{meal:<20}{cal}")
print("-" * 35)
print(f"{'Total:':<20}{total_calories}")
print(f"{'Average:':<20}{average_calories:.2
f}") print("-" * 35)
print(status_message)

#Task 6 (Bonus): Save to

File

save = input("Would you like to save this report? (yes/no):
").strip().lower() if save == "yes":
timestamp = datetime.datetime.now().strftime("%Y-%m-%d_%H-
%M-%S") filename = f"calorie_log_{timestamp}.txt"
with open(filename, "w") as f:
f.write(" DAILY CALORIE TRACKER REPORT ")
f.write(f"Date & Time: {datetime.datetime.now()}\n\n")
f.write(f"{'Meal Name':<20}{'Calories'}\n")
f.write("-" * 35 + "\n")
for meal, cal in zip(meals,
calories):
f.write(f"{meal:<20}{cal}\n")
f.write("-" * 35 + "\n")
f.write(f"{'Total:':<20}{total_calories}\n")
f.write(f"{'Average:':<20}{average_calories:.2f
}\n") f.write("-" * 35 + "\n")
f.write(f"Status: {status_message}\n")
print(f"\n Report saved successfully
as'{filename}'!\n") else:
print("\nReport not saved. Thank you for using the

tracker! ") #End of Program
```