Process MeNtOR 3.o

Uni-SEP

Process MeNtOR 3

Version:	2
Print Date:	Nov. 20
Release Date:	Nov. 20
Release State:	Released
Approval State:	Pending Approval
Approved by:	Amsal & Long & Judah & Mukul
Prepared by:	Amsal & Long & Judah & Mukul
Reviewed by:	Amsal & Long & Judah & Mukul
Path Name:	https://docs.google.com/document/d/1WDb5QbGgDv1kTAgeFH9m1kbCFJFtR58RuHOZ_17bP4Q/edit
File Name:	Deiverable-2-Report
Document No:	D-002

Github Link:
https://github.com/JudahGoldstein/EECS3311-PROJECT

Document Change Control

Version	Date	Authors	Summary of Changes
V1	Oct. 1	Judah	First meeting note
V2	Oct. 10	Long	Second meeting note
V3	Oct. 14	Long	Third & Fourth meeting notes
V4	Oct. 16	Amsal & Long	Test cases & Activities Plan
V5	Oct. 17	Long	Notes & Architecture diagrams
V6	Oct. 18	Long & Amsal	Introduction Section & Architecture Tables & Sequence Diagram
V7	Oct. 19	Long	Major Design Decisions
V8	Oct. 20	Long	Design Patterns
V9	Nov 10	Amsal	Adding the structured out graph for the nutritional database
V10	Nov 18	Amsal	Fixing documentation errors as indicated in deliverable 1 feedback
V11	Nov 20	Long	Finalizing doc

Document Sign-Off

Name (Position)	Signature	Date
Long Lin	Long Lin	Nov 20
Amsal Ali	Amsal Ali	Nov 20
Judah Goldstein	Judah Goldstein	Nov 20
Mukul Chauhan	Mukul Chauhan	Nov 20

Contents

1 Introduction

1.1 Purpose

This document details the requirements of the system <Nutrifit: Eat, Run, Smile!>.

The Nutrifit application aims to empower users to track and manage their dietary and exercise habits effectively. It offers a comprehensive solution for logging, monitoring, and visualizing nutrient intake, exercise data, and progress towards personal health and fitness goals. In alignment with the principles of the Canada Food Guide, Nutrifit provides a user-friendly interface to promote a healthier lifestyle.

This document delves into the core requirements of Nutrifit, detailing essential features and functionalities that will be implemented to ensure a seamless user experience. It serves as a blueprint for the system's development, offering clear guidance for the project team, stakeholders, and end-users.

The following sections will provide an in-depth exploration of the specific requirements, use cases, system components, and design considerations, tailored to the Nutrifit application.

1.2 Overview

The primary objective of Nutrifit is to offer a user-friendly platform for individuals to log their meals and exercise activities, enabling the tracking of calorie intake, nutrient composition, and energy expenditure. By visualizing this data, users can gain valuable insights into their dietary habits and fitness progress.

The system aims to:

- *Assist users in making informed dietary choices aligned with the Canada Food Guide.*
- *Calculate the Basal Metabolic Rate (BMR) and estimate calorie expenditure based on exercise levels.*
- *Predict potential weight loss outcomes based on diet and exercise patterns.*
- *Provide intuitive data visualization for better understanding.*

Document Structure:

This document is structured to provide a comprehensive understanding of Nutrifit's requirements, design, and functionality. It encompasses a range of sections, including use cases, system components, and diagrams, each contributing to a holistic view of the system. As you navigate through the document, you will find detailed insights into how Nutrifit addresses the challenges of dietary and exercise management while promoting healthier lifestyles.

The subsequent sections will elaborate on specific project components and functional requirements, ultimately guiding the development and realization of the Nutrifit application.

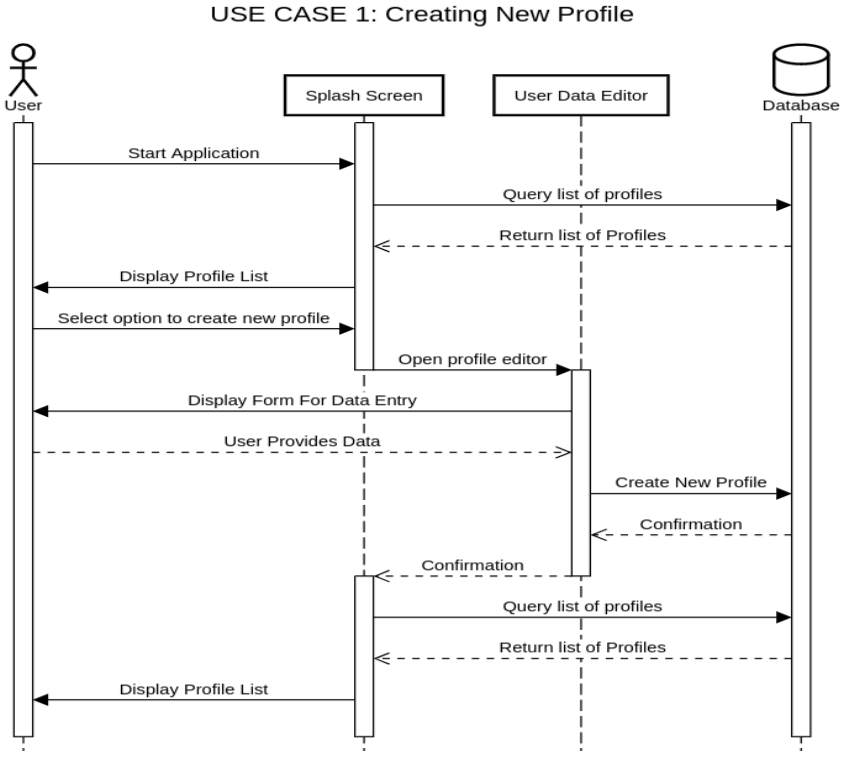
1.3 Resources - References

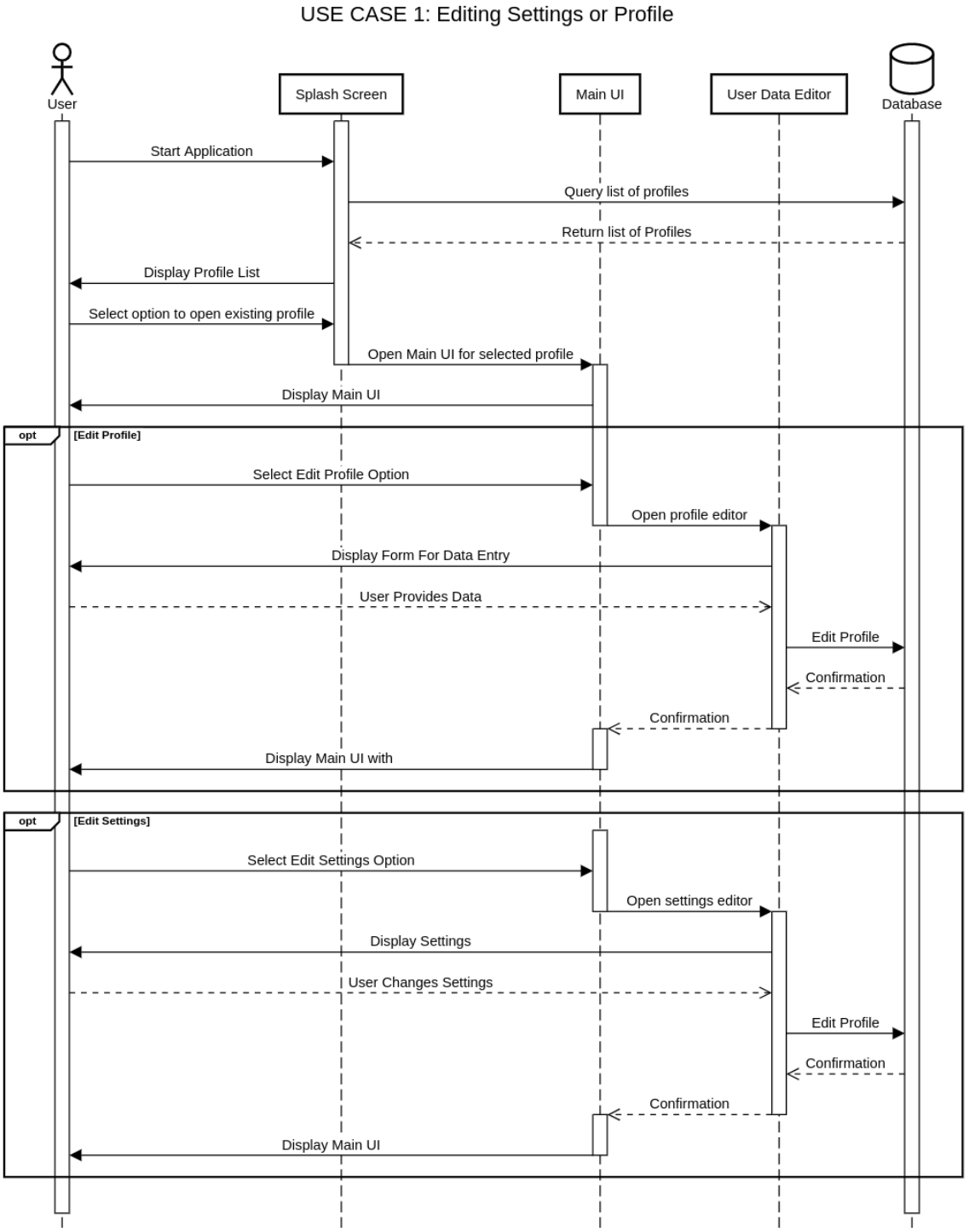
Verdier, P. (1983). The canadian nutrient file. Canadian Institute of Food Science and Technology Journal, 16(3), xviii. https://doi.org/10.1016/s0315-5463(83)72167-x

2 Sequence Diagrams

Use Case 1

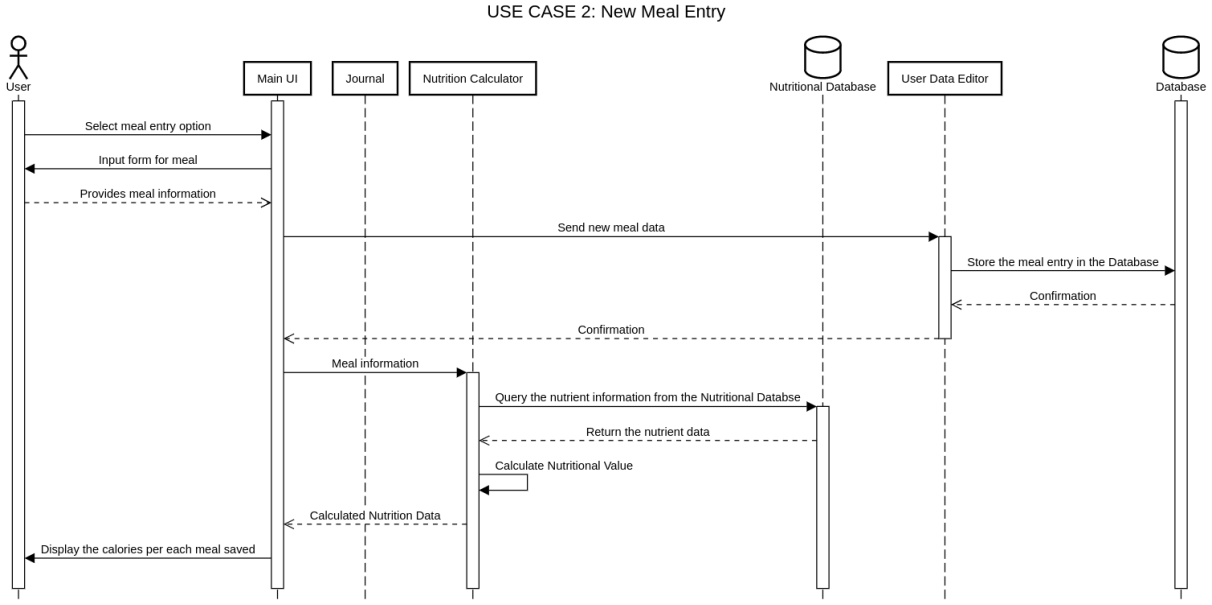
Summary: This use case primarily focuses on the creation and editing of the user's basic data. Data flows from the user into the UI, then the UI to the Database that stores the user data, then based on the new data the UI for the user is updated.

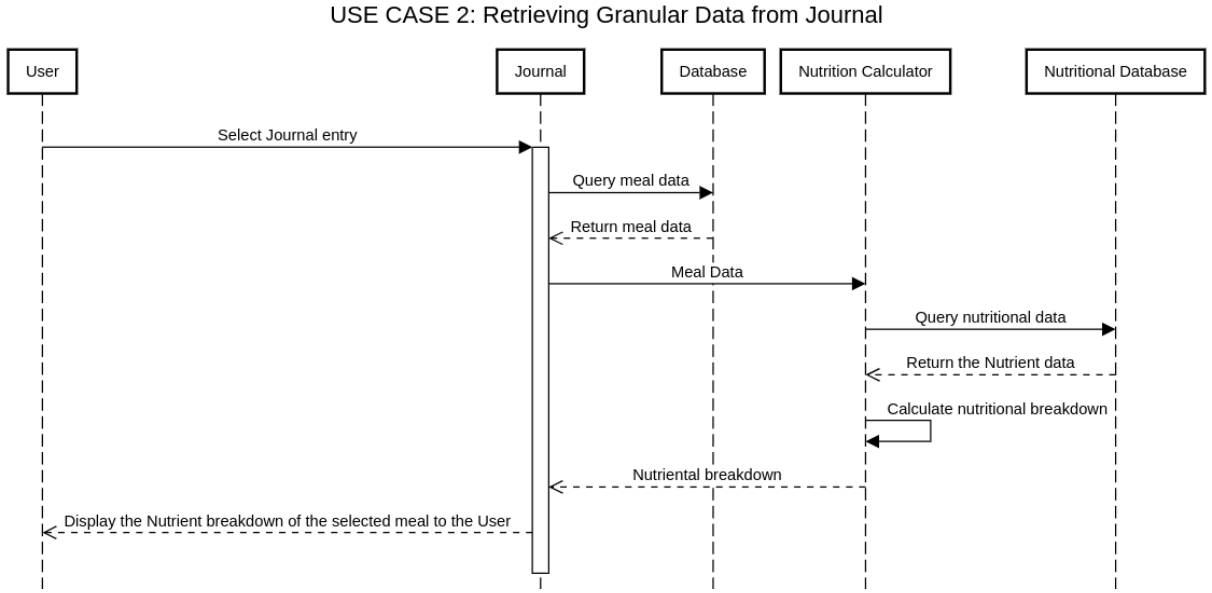




Use Case 2

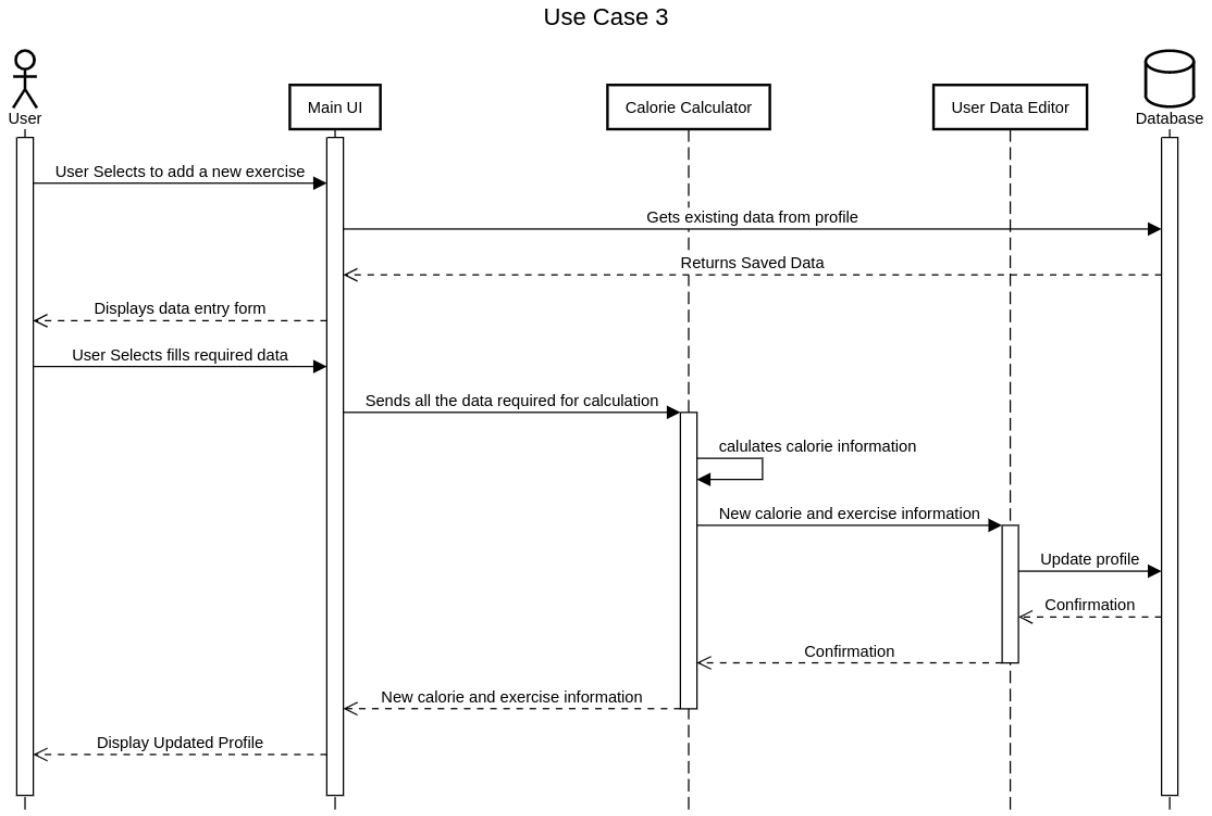
Summary: This use case requires two Databases and two sets of UI participants. the "Nutritional Database" provides Nutrition information (provided), and the "Database" stores user data. For adding a meal the data should flow from the user, to the database, then back to the user passing through the Nutrition calculator which takes its information from the Nutritional Database to derive the nutritional information of the meal. To get the detailed breakdown of the meal in the Journal view The same data flow is used, but instead of the data coming from the user it comes from the saved data already in the Database.





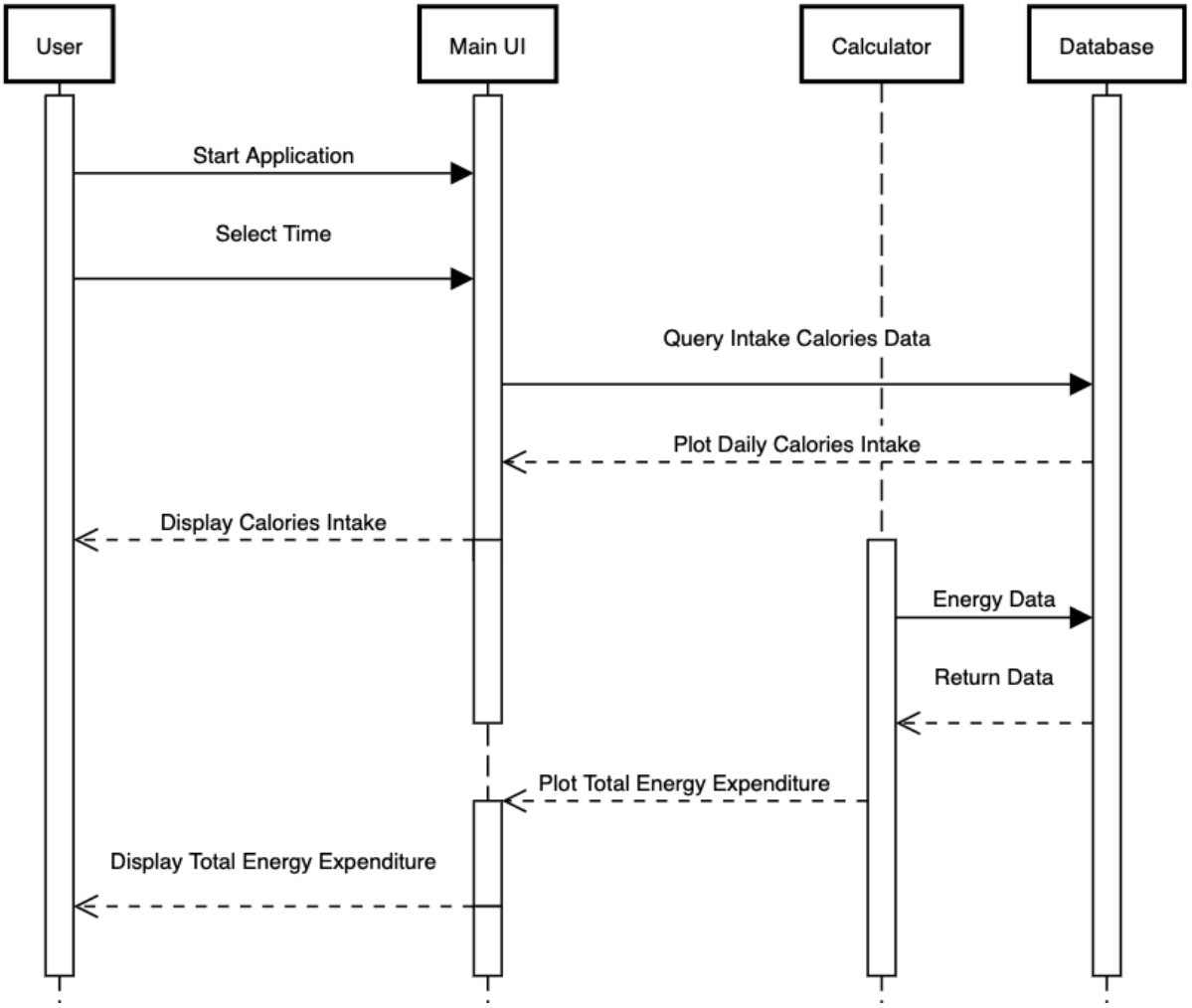
Use Case 3

Summary: This Use case concerns entering exercise information into the program. First information already stored in the database is pre-filled into the data-entry form presented to the user, then the user inputs the exercise information. The exercise information is then passed on to the calculator which derives the calorie information which is then passed back to the UI for the user.



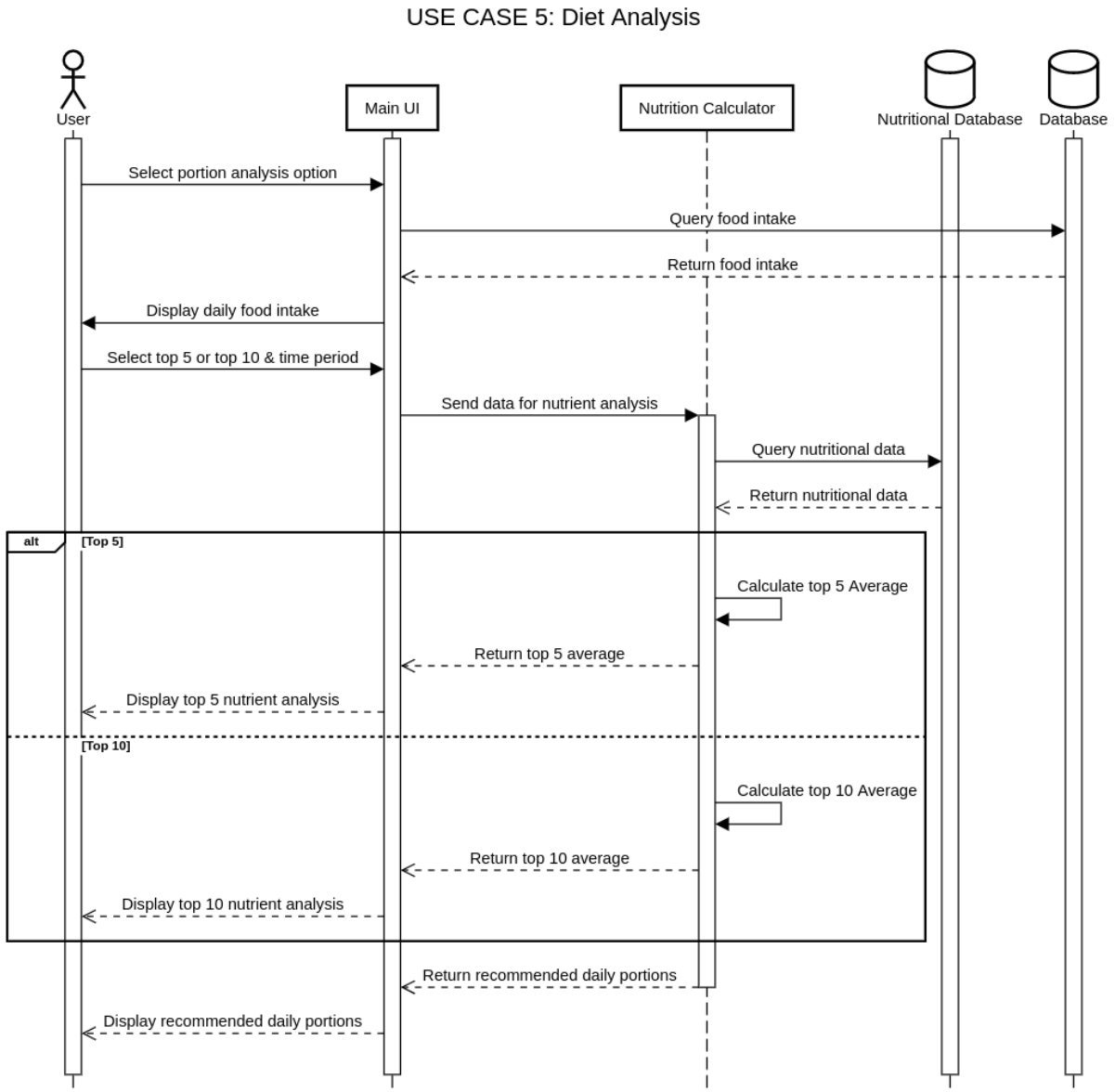
Use Case 4

Summary: This use case enables users to visually track their calorie intake and exercise over a specific time period by selecting a start and end date. The application generates a graphical representation of daily calorie intake and total daily energy expenditure, providing users with insights into their energy balance and helping them make informed decisions about their diet and exercise routines based on historical data.



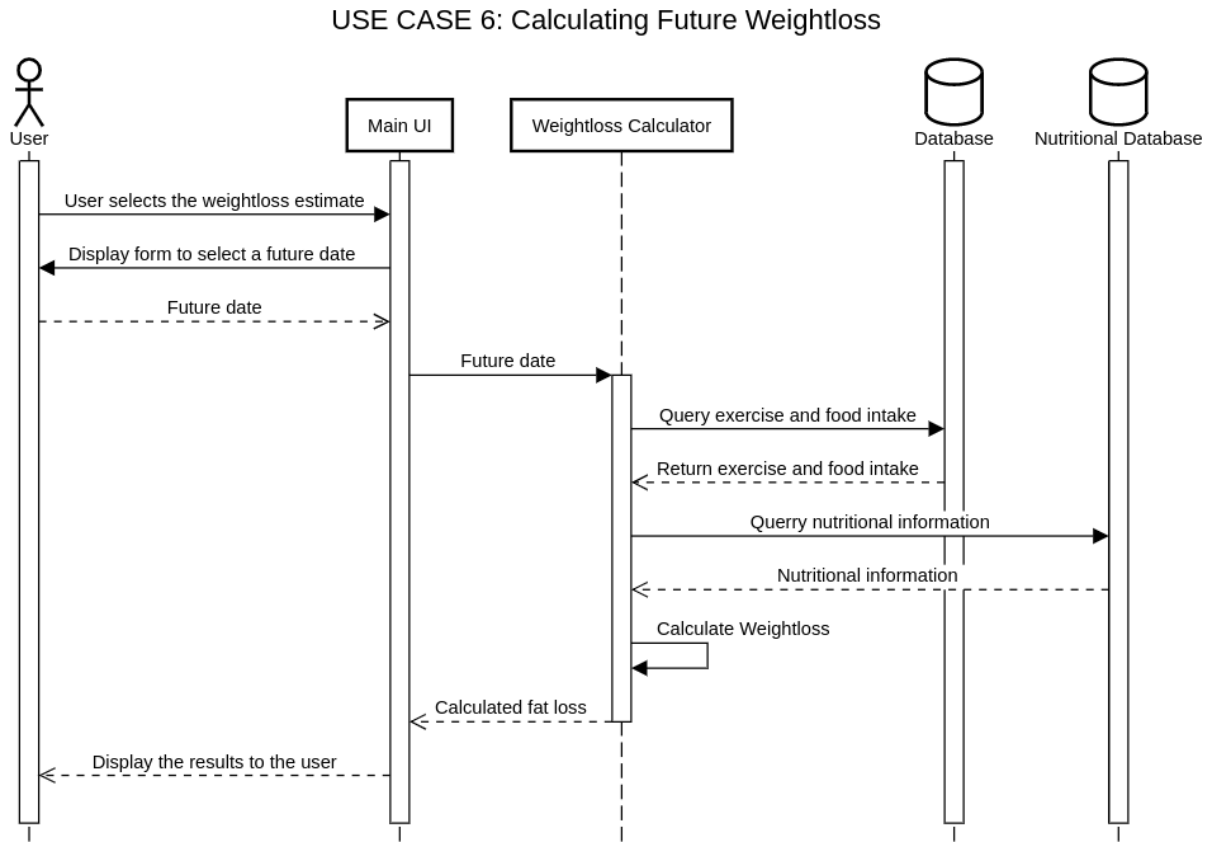
Use Case 5

Summary: This use case centers around analyzing diet information already stored in the user data "Database" upon request that data flows to a calculator that provides the requested analytics to the user. The calculator also takes in data from the Nutritional Database (provided) for its calculation.



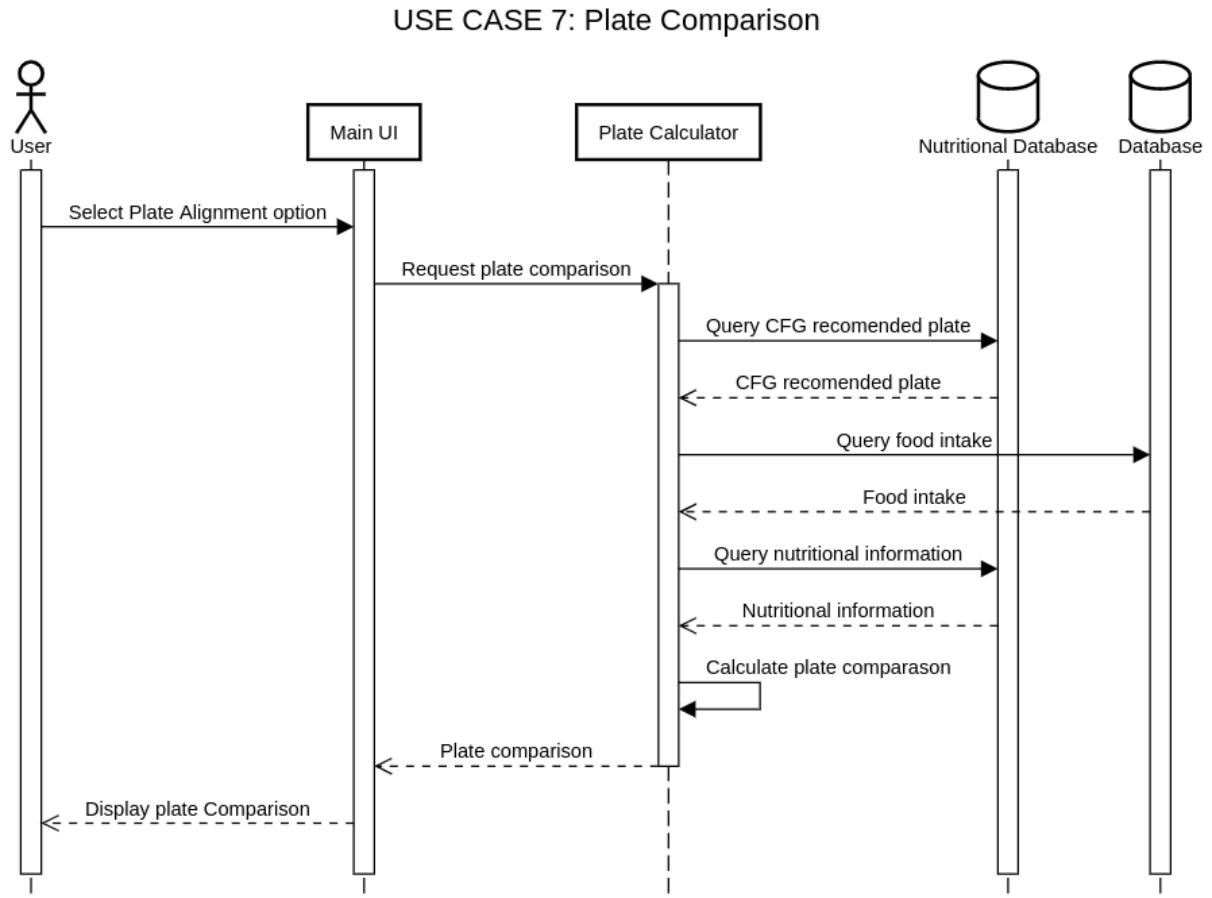
Use Case 6

Summary: This use case centers around calculating the weight loss the user can expect by a certain date given their current diet and exercise trends. The calculator takes in the future date from the user, the diet and exercise data from the user data d]Database, and the nutritional data corresponding to that diet information from the Nutritional Database (provided). Once the future fat loss is calculated it is passed back to be displayed to the user.



Use Case 7

Summary: This use case focuses on the Plate Alignment feature of the application. the user does not need to provide any information for this, the calculator in this case collects the CFG suggested plate alignment from the Nutritional Database (provided), personal diet information about the user from the user data Database, and the nutritional data about that diet from the Nutritional Database. Then once all the data needed is collected it calculates a comparison between the user's average plate and the suggested plate and passes that information back to the UI to be displayed for the user.



3 Major Design Decisions

Design Choices:

Microservices Architecture: The project's design choice includes adopting a microservices architecture to enhance scalability and maintainability. Each functional component is encapsulated within a separate microservice, enabling independent development, testing, and deployment. This design choice aligns with the modularization criteria by promoting high cohesion within each microservice.

Single Page Application (SPA): The user interface is designed as a SPA using a modern JavaScript framework. This decision improves the user experience and responsiveness. The modularization criteria are addressed by organizing the codebase into reusable components, which results in high cohesion within each component.

Solid Principles we included:

Single Responsibility Principle: Each module within the Nutrifit application adheres to the SRP. For instance, the "User Profile" microservice is responsible for managing user profiles without taking on unrelated functionalities.

Dependency Inversion Principle: High-level modules are not directly dependent on low-level modules. Abstractions are used to decouple components, promoting a more maintainable and adaptable architecture.

GRASP Principles we included:

High Cohesion: To achieve high cohesion, the software design promotes grouping related functions and data within individual modules. For instance, within each microservice, business logic, data storage, and API endpoints are grouped to address a specific functional area. This ensures that each module has a clear, well-defined purpose.

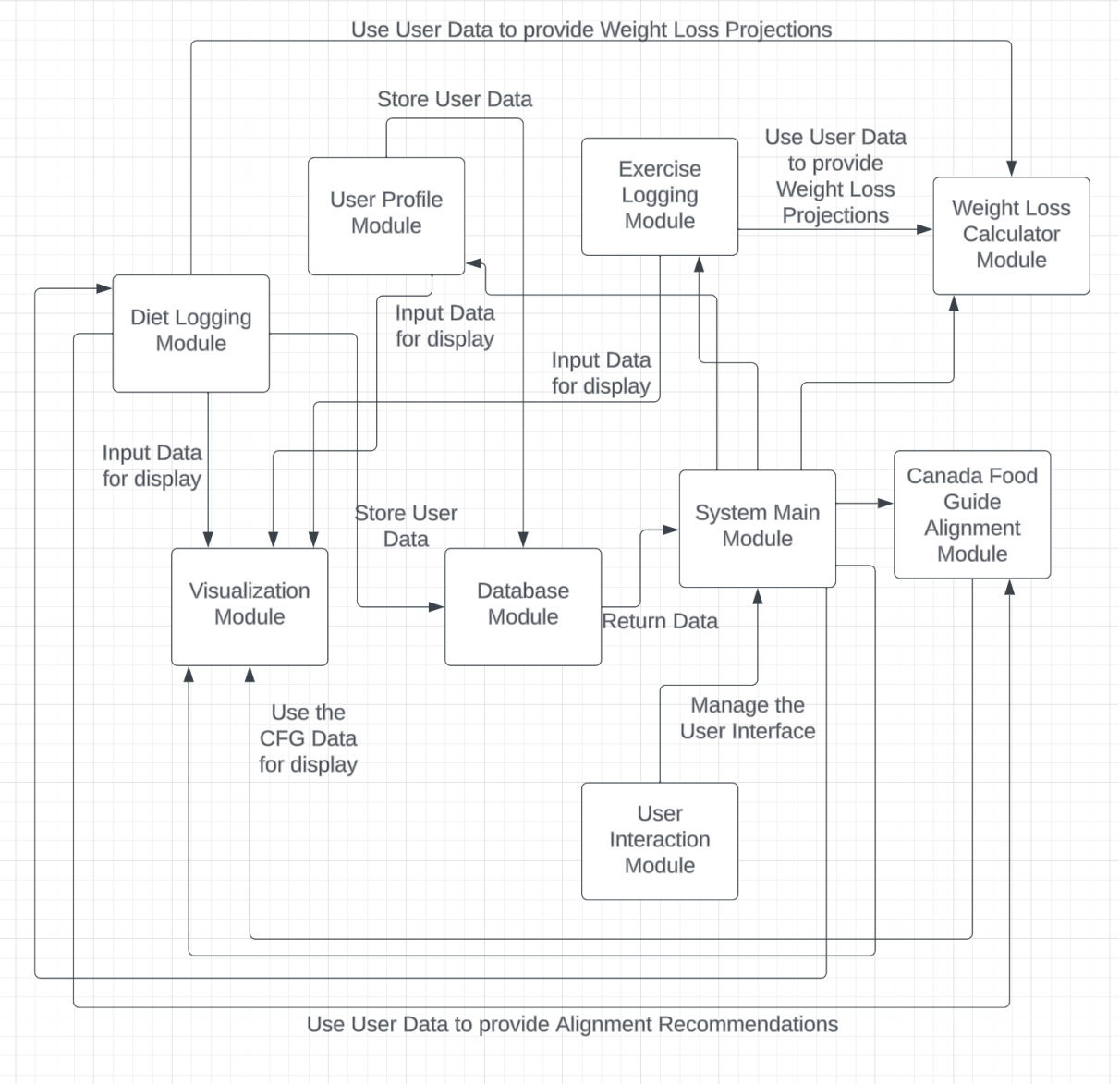
Low Coupling: Low coupling is achieved by minimizing dependencies between modules. Each microservice and UI component is designed to communicate through well-defined, independent interfaces (e.g., RESTful APIs). This reduces the impact of changes in one module on others, enhancing system maintainability.

Some of the quality considerations:

Low Complexity: The system is designed with a focus on low complexity, ensuring that each module and microservice has a clear and simple purpose. This not only enhances maintainability but also facilitates ease of collaboration among team members. Code readability is emphasized, with meaningful variable and method names, aiding in the comprehension of the system's logic and functionality.

Separation of Concerns: The project follows the principle of separation of concerns to keep distinct aspects of the software separate. This includes separating the user interface (presentation layer) from business logic (application layer) and data access (persistence layer). Each layer is encapsulated within specific modules to ensure that each is responsible for its unique concerns.

4 Architecture



Modules			
Module Name	**Description**	**Exposed Interface Names**	**Interface Description**
M1	"User Profile"	M1:I1 M1:I2	M1:I1 - User Profile Operations M1:I2 - Settings and Preferences
M2	"Diet Logging"	M2:I3	M2:I3 - Diet Entry
M3	"Exercise Logging"	M3:I4	M3:I4 - Exercise Entry
M4	"Data Visualization"	M4:I5	M4:I5 - Data Presentation

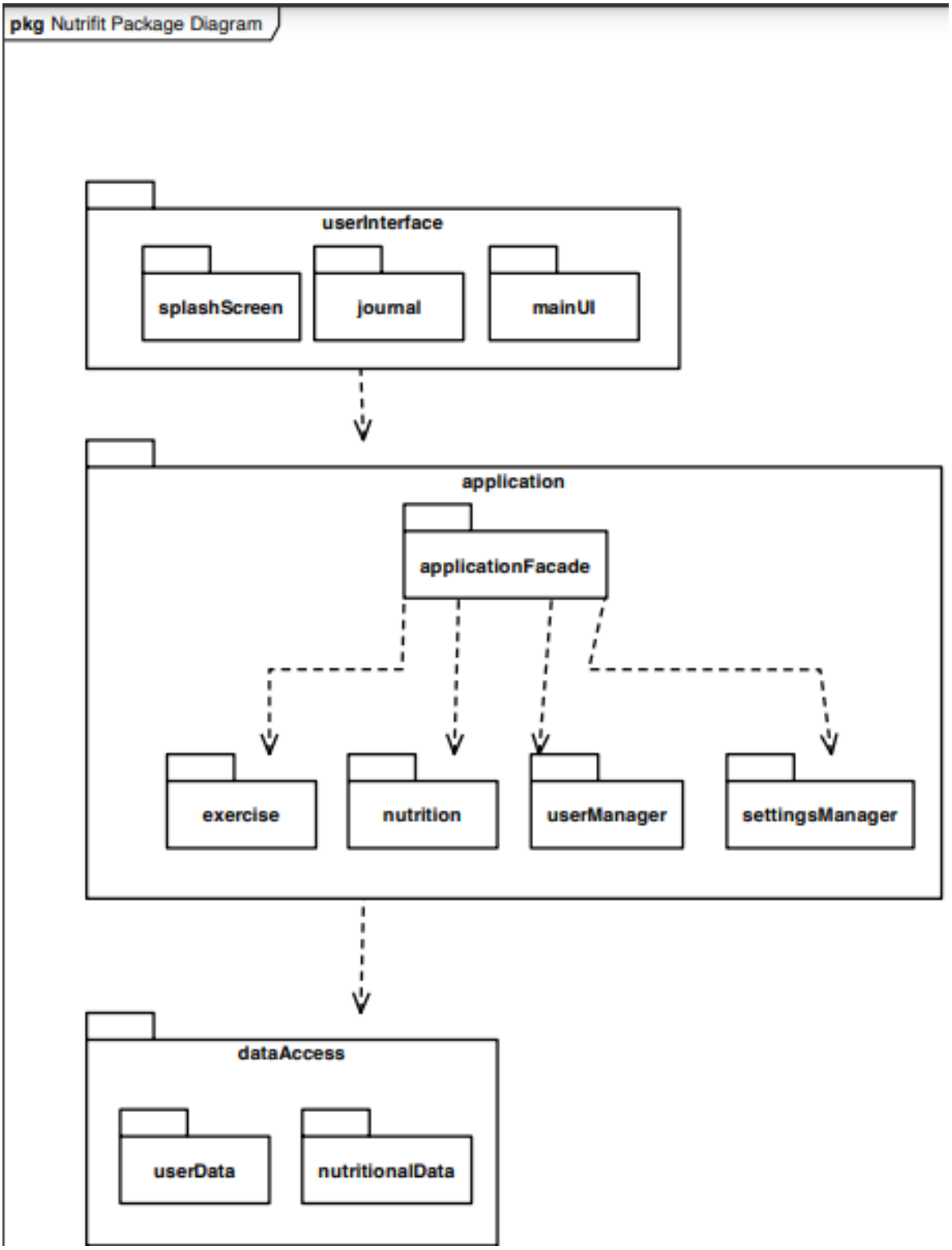
M5	"Weight Loss Calculator"	M5:I6	M5:I6 - Weight Loss Calculator
M6	"Plate Alignment"	M6:I7	M6:I7 - CFG Compliance
M7	"User Interaction Features"	M7:I8	M7:I8 - User Interaction Features
M8	"Database"	M8:I9	M8:I9 - Database Access
M9	"Documentation & Testing"	M9:I10	M9:I10 - Documentation and Testing

Interfaces		
Interface Name	**Operations**	**Operation Descriptions**
M1:I1	<int> I1:Op1() <String> I1:Op2(int x)	M1:I1:Op1(): “Calculate and return total calories.” M1:I1:Op2(int x): “Retrieve and return user's profile name.”
M2:I2	<String> M1:I2:Op3()	M2:I2:Op3() “Retrieve User Settings”
M3:I3	<int> M2:I1:Op1()	M3:I3:Op4() “Log Diet Entry”
M4:I4	<int> M3:I1:Op1()	M4:I4:Op5() “Log Exercise Entry”
M5:I5	List<WeightLossPrediction> M4:I1:Op1()	M5:I5:Op6() “Calculate Weight Loss Prediction”
M6:I6	List<CFGComplianceData> M6:I1:Op1()	M6:I6:Op7() “Ensure CFG Compliance”

5 Detailed Class Diagrams

5.1 UML Class Diagrams

Main Diagram with the different packages



Application Package

setDurationMinutes method : sets the minutes of exercise done

getCaloriesBurnerd method: gets the calories burned during the exercise

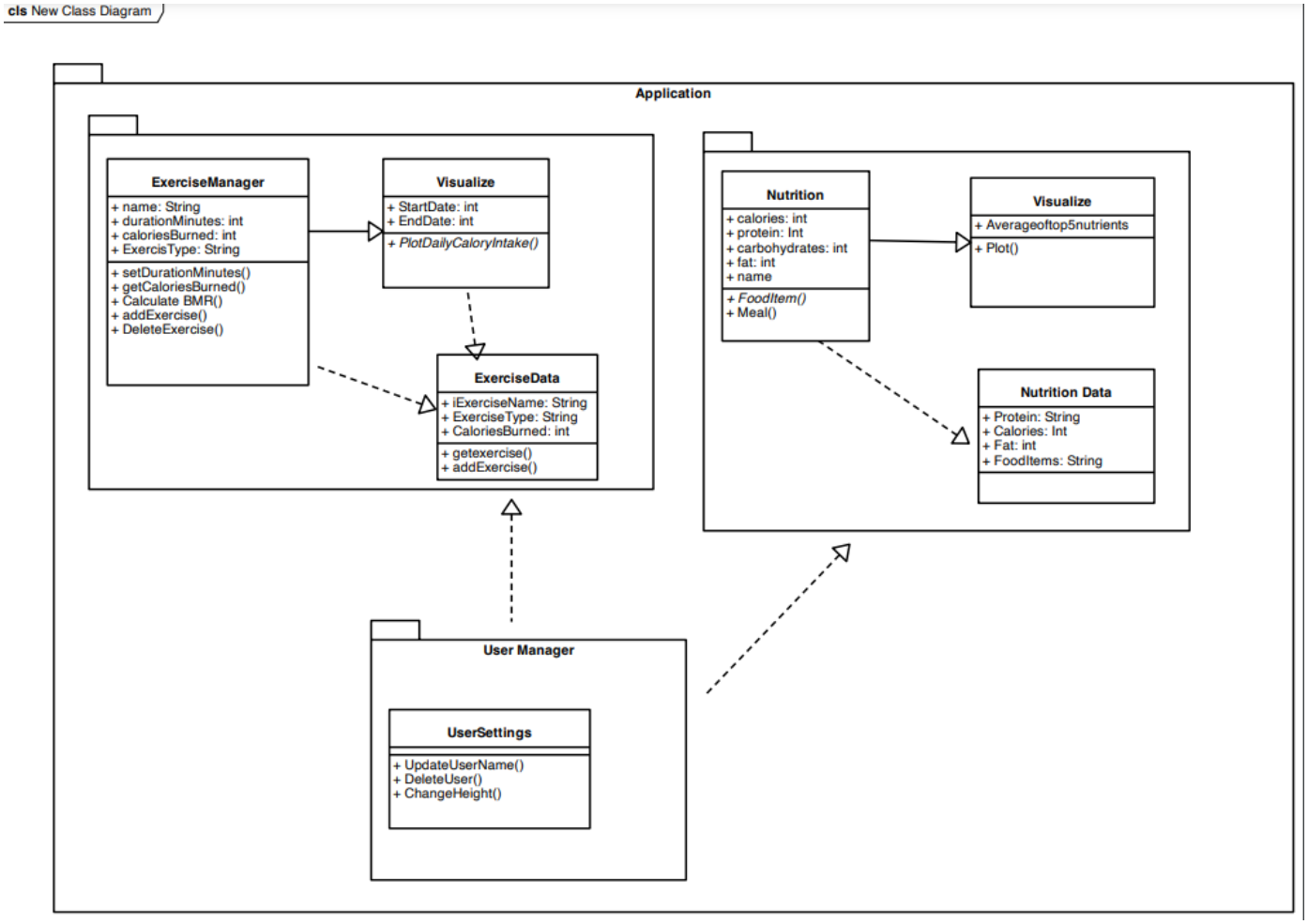
calculateBMR method: calculates the BMR

addExercise method; adds the different types of exercise

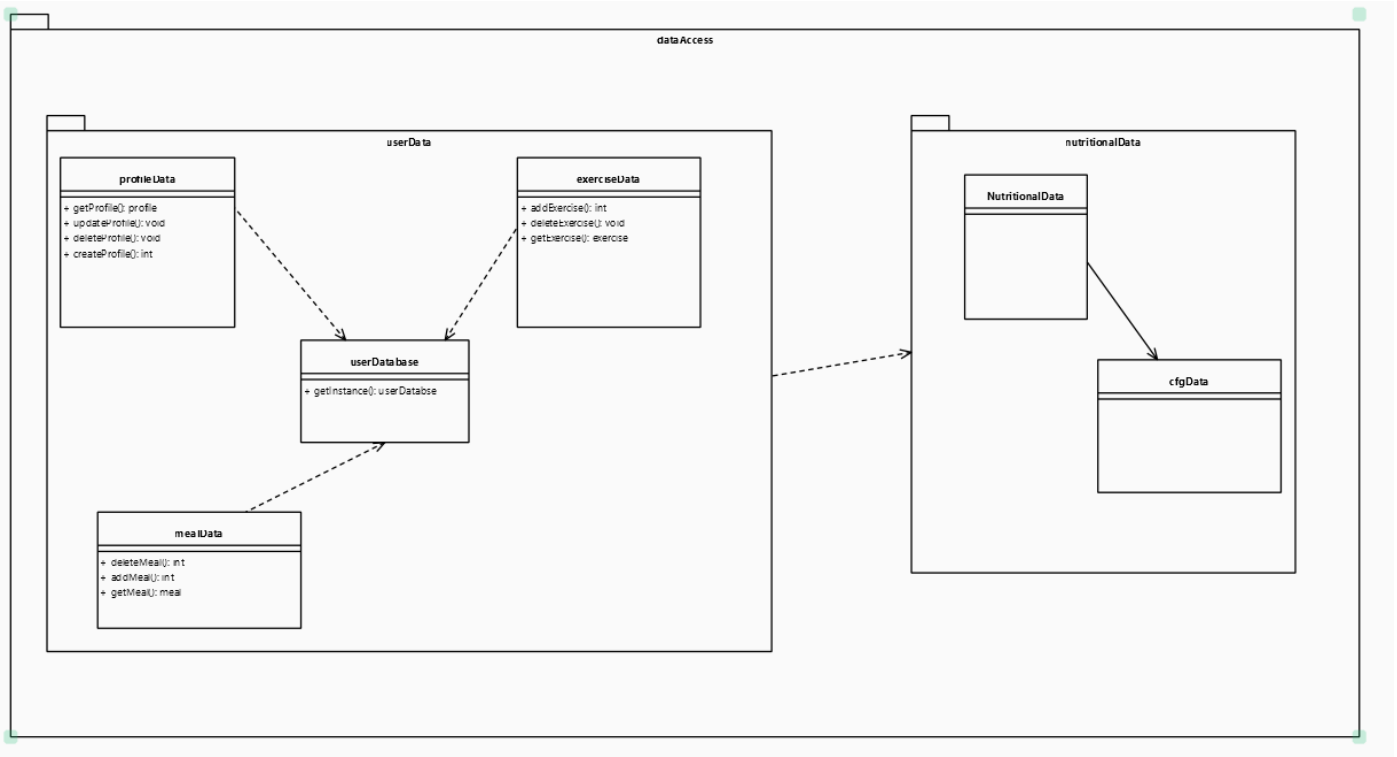
deleteExercise method: deletes the exercise a user does not want to do

getExercise method: gets the exercise done

deleteUser: deletes the user account when user wishes to no longer use the account



dataAccess Package: the database package



getProfile method: gets the user profile and stores it in the database

updateProfile method: any updates a user makes to the profile

deleteProfile method: deletes the profile and removes from the database

createProfile method: creates the user profile and stores it in the database

addExercise method: adds the exercise a user does and stores it in the users database

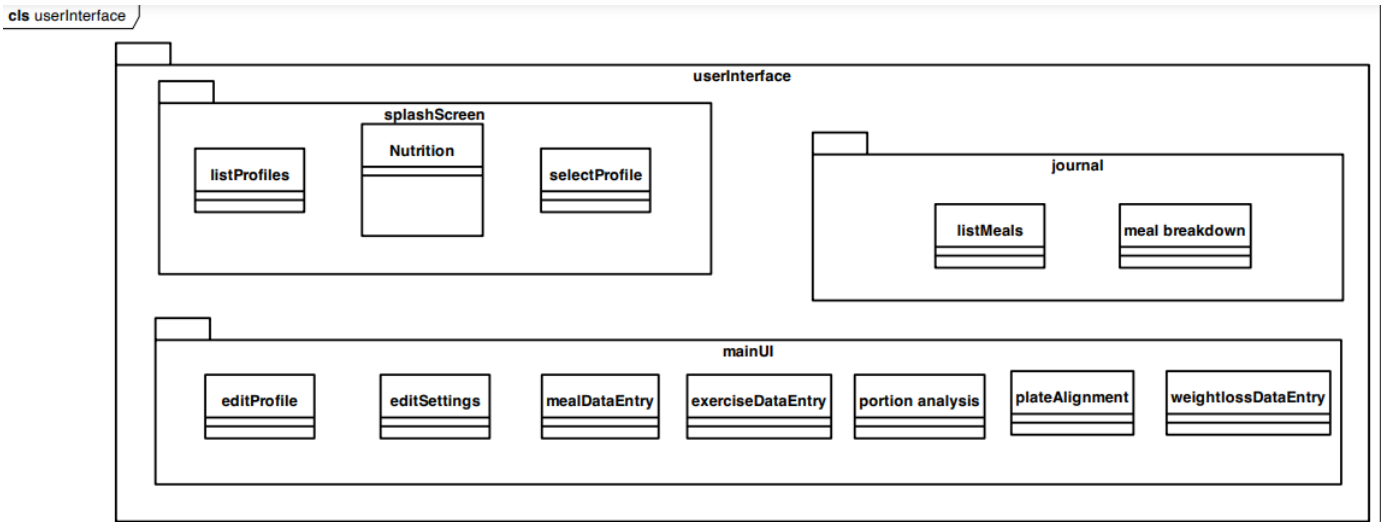
deleteExercise method: deletes exercise a user does and removes it in the users database

getMeal method: gets the meal a user takes and stores it in the database

addMeal method: adds the meal the user takes and stores it in the database

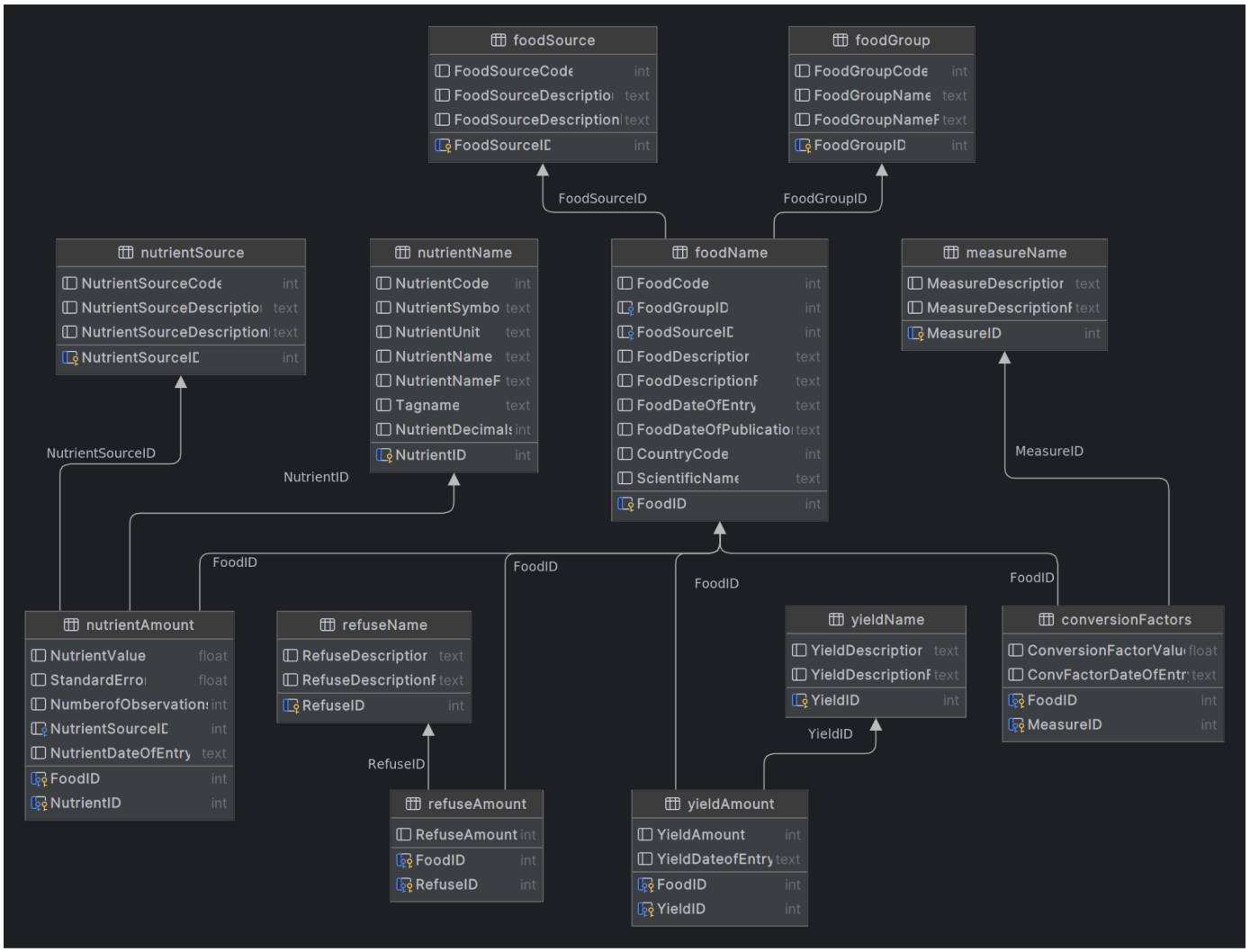
deleteMeal method: deletes the meal a user takes and removes it in the users database

UserInterface Package:



Canada Nutritional Database structure Layout

The Canada Nutritional Database is structured to efficiently organize and manage nutritional information required by the Nutrifit application. The database comprises several interconnected classes, each serving a specific purpose in storing and retrieving nutritional data.



6 Use of Design Patterns

Some of the design patterns we plan on incorporating in the deliverables are:

Factory Method Pattern:

Consider using the Factory Method pattern when creating different types of objects in your application. For instance, when handling different types of exercises, foods, or nutrient data, a factory method can simplify object creation by providing an interface for creating these objects.

Observer Pattern:

Implement the Observer pattern to update the user interface when data changes. For example, when the user logs a new meal or exercise, the Observer pattern can notify the UI to update the displayed information.

Singleton Pattern:

We use the Singleton pattern to ensure that certain classes, such as a database manager or application configuration manager, have only one instance throughout the application's lifecycle.

Strategy Pattern:

The Strategy pattern can be employed to allow users to choose different calculation strategies, such as BMR calculations or weight loss predictions. By encapsulating these algorithms, you can easily switch between them based on user preferences.

Adapter Pattern:

If the application interacts with external data sources or APIs (e.g., the Canadian Nutrient File), the Adapter pattern can be used to make these external interfaces compatible with the application's internal data structures.

State Pattern:

In the context of user profiles and settings, the State pattern can be used to manage different states of user profiles, such as user preferences, measurement units, and dietary restrictions.

Command Pattern:

Implement the Command pattern to encapsulate user interactions and actions. For instance, when users log a meal or exercise, the Command pattern can encapsulate these actions, allowing for easy undo/redo functionality.

7 Activities Plan

7.1 Project Backlog and Sprint Backlog

Product Backlog:

- User Profile Management
- Diet Logging
- Exercise Logging
- Data Visualization
- Weight Loss Calculator
- Canada Food Guide Alignment
- User Interaction Features
- Database Setup
- Documentation & Testing
- User Support & Feedback

Sprint Backlog for Sprint 1 (Weeks 1-4):

Sprint Goal: Basic Application Setup and User Profiles

- ID 2: Database Setup
- ID 3: Splash Screen & Profile Setup (part of User Profile Management)
- ID 8: User Interaction Development (Initial Setup)
- ID 9: Documentation & Testing (Complete Documentation)

Sprint Backlog for Sprint 2 (Weeks 5-8):

Sprint Goal: Diet and Exercise Logging, Data Visualization, Weight Loss Calculator, Canada Food Guide Alignment, User Interaction Features

- ID 1: Project preparation & Initial Implementation (Continued Refinement)
- ID 4: Diet and Exercise Logging
- ID 5: Data Visualization
- ID 6: Weight Loss Calculator
- ID 7: Canada Food Guide Alignment
- ID 8: User Interaction Development (Additional Features)
- ID 9: Documentation & Testing (Continuous)
- ID 10: User Support & Feedback (Ongoing Integration)

Additional Week (Week 9):

Sprint Goal: Feedback and Wrap-up

- ID 10: Review the progress, address any feedback, make final adjustments, and wrap up

EECS3311

Project Implementation Plan & Schedule

				Oct				Nov				Dec			
Deliverables		**Predecessor**	**Duration**	W1	W2	W3	W4	W1	W2	W3	W4	W1	End		
ID	**Task Name**														
1	Project preparation & Initial Implementation		3 w												
2	Database Setup		1 w												
3	Splash Screen & Profile Setup		2 w												
4	Diet and Exercise Logging	2	1 w												
5	Data Visualization	3	1 w												
6	Weight Loss Calculator	2	1 w												
7	Canada Food Guide Alignment	2,3	2 w												
8	User Interaction Development		2 w												
9	Documentation & Testing		ongoing												
10	User Support & Feedback		ongoing												

7.2 Group Meeting Logs

In this Section you write minutes of each meeting, listing the attendance, what the topics of discussion in the meeting were, any decisions that were made, and which team members were assigned which tasks. These minutes must be submitted with the project report in each deliverable and will provide input to be used for the overall assessment of the project.

Present Group Members	Meeting Date	Issues Discussed / Resolved
All	Oct. 1	Getting the repository set up, ticket system started, and planning concluded
Long & Amsal & Judah	Oct. 10	Go over and complete Use Case Sequence Diagrams
Long & Amsal	Oct. 13	Final check on completed tasks and discuss about other tasks
Long & Mukul	Oct. 14	Discuss the next working plan and assign tasks
Amsal & Long & Judah	Oct 16	Finishing the final edits on Sequence Diagrams. Discussing and creating test cases.
All	Oct 18	Finalized Test Cases, beginning the initial implementation, finalizing parts of documents
All	Oct 20	Finishing up the final details of the document and implementation
Amsal & Long & Judah	Nov 8	Discussing the deliverable 2 and coming up with ideas to complete the use cases
Long & Judah & Mukul	Nov 20	Finalized the project

8 Test Driven Development

Test cases will be provided in the form of a table as follows:

Test ID	TC01
Category	Profile Creation and Edit
Requirements Coverage	UC1-Profile-Creation
Initial Condition	The system is initiated and runs
Procedure	1. User selects "Create Profile" on the screen. 2. User enters basic information and clicks "Create."
Expected Outcome	User profile is created successfully.
Notes	User data should be stored correctly in the database.

Test ID	TC02
Category	User Authentication
Requirements Coverage	UC2-User-Login
Initial Condition	User account exists, system running
Procedure	1. Open the application and navigate to the login screen 2. Enter valid username and password 3. Click the "Login" button
Expected Outcome	Login successfully message shown, and the User Main UI is displayed with the user's name.
Notes	Verify that the user info is shown correctly.

Test ID	TC03
Category	Diet Logging
Requirements Coverage	UC3-Log-Diet-Data
Initial Condition	User has an existing profile
Procedure	1. User selects "Log Diet" from the main UI. 2. User enters meal details and clicks "Log Meal."
Expected Outcome	Nutritional information for the meal is calculated and displayed in the journal.
Notes	Nutrient values should be accurately calculated

Test ID	TC04
Category	Exercise Logging
Requirements Coverage	UC4-Log-Exercise-Data
Initial Condition	User has an existing profile
Procedure	1. User selects "Log Exercise" from the main UI. 2. User logs an exercise with type, duration, and intensity.
Expected Outcome	Calories burned for the exercise are calculated and displayed in the journal.
Notes	The calorie calculations done should be accurate

Test ID	TC05
Category	Daily Calorie Intake Visualization
Requirements Coverage	UC5-Visualize-Calorie-Intake
Initial Condition	Both the Diet and exercise data are logged
Procedure	1. User selects "Visualize" -> "Calorie Intake" from the main UI. 2. User chooses a time period.
Expected Outcome	A visual representation of daily calorie intake is displayed (e.g., line chart or bar graph).
Notes	The chart should accurately represent calorie intake.

Test ID	TC06
Category	Daily Nutrient Intake Visualization
Requirements Coverage	UC6-Visualize-Nutrient-Intake
Initial Condition	Diet data is logged
Procedure	1. User selects "Visualize" -> "Nutrient Intake" from the main UI. 2. User chooses a time period.
Expected Outcome	A visual representation of daily nutrient intake is displayed, including top nutrients and a notification.
Notes	The chart should accurately represent nutrient intake.

Test ID	TC07
Category	Weight Loss Calculator
Requirements Coverage	UC7-Weight-Loss-Calculator
Initial Condition	Diet and exercise data logged
Procedure	1. User selects "Weight Loss Calculator" from the main UI. 2. User enters a future date.
Expected Outcome	The app predicts the amount of fat the user can lose by the specified date based on logged data.
Notes	The prediction should be accurate

Test ID	TC08
Category	Profile and Settings Update
Requirements Coverage	UC8-Profile-Settings-Update
Initial Condition	User has an existing profile
Procedure	1. User selects "Edit Profile" or "Edit Settings" from the main UI. 2. User makes changes and saves.
Expected Outcome	Profile or settings are updated accordingly in the database and displayed in the main UI.
Notes	Data updates should be reflected in the user interface.

Test ID	TC09
Category	Database Initialization
Requirements Coverage	UC9 - Succesfull-Database-Connection
Initial Condition	The system is initiated
Procedure	1. Start the application for the first time.
Expected Outcome	The application initializes and connects to the database successfully. The user can create profiles, log diet and exercise data.
Notes	Ensure the database is created and accessible.

Test ID	TC10
Category	Date Selection in Visualization
Requirements Coverage	UC10-Visualize-Calorie-Intake & Nutrient-Intake
Initial Condition	Data is available for the selected date range
Procedure	4. User selects a date range. 5. Click the show button for visualization.
Expected Outcome	The visualized data should be limited to the chosen date range.
Notes	Verify that the date filter functions correctly.