Workout Tracker

VIU CSCI 370 Database Systems

Final Project

By: Saad Nisar Hussain

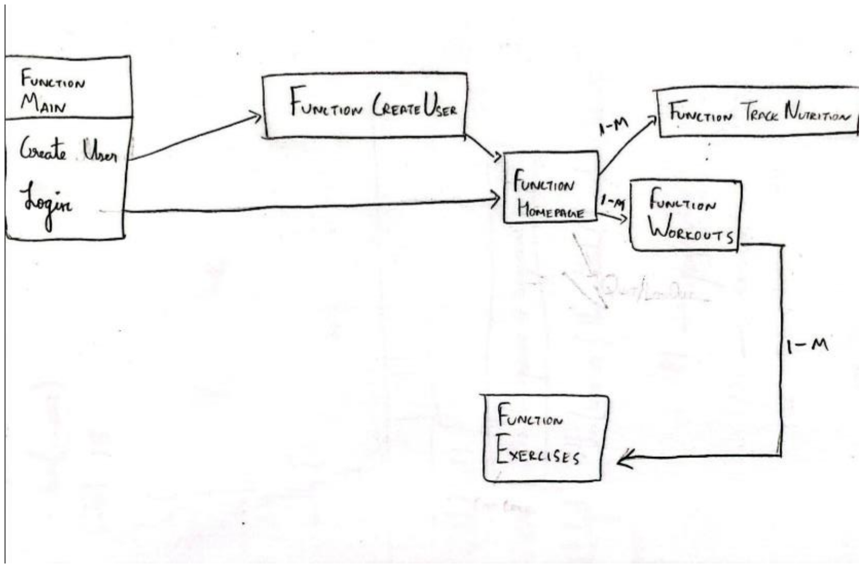
Contents

Introduction

Workout Tracker is an interface-based application that allows users to keep track of their workouts, exercises, and nutrition to achieve their desired fitness goals without any need for complex calculations. This application is based on C++ programming language, with Oracle as a backend engine.

Data Description

The function-level relationships are shown below



The above is *not* an ER diagram, It's just to show how one function is connected to another. **Main** will call either **CreateUser** or **Login** and depending on that, it will take the user to the **homepage** where the important functionalities are present. These functionalities are divided into 2, namely **TrackNutrition** and **Workouts**. When the user selects the option to track nutrition, it will take the user to the **TrackNutrition** function where the user will be prompted to enter calorie intake, based on that the data is calculated and shown and also stored in the database. If the user selects **Workouts,** he will be given modification and add/delete options and from there the user can view **Workouts** by entering the **Exercise** function. The **Exercise** function displays the exercises of a particular workout with its details, where again, the user can perform modifications. Some functions produce primary keys but are not included in the above diagram

Application Requirements

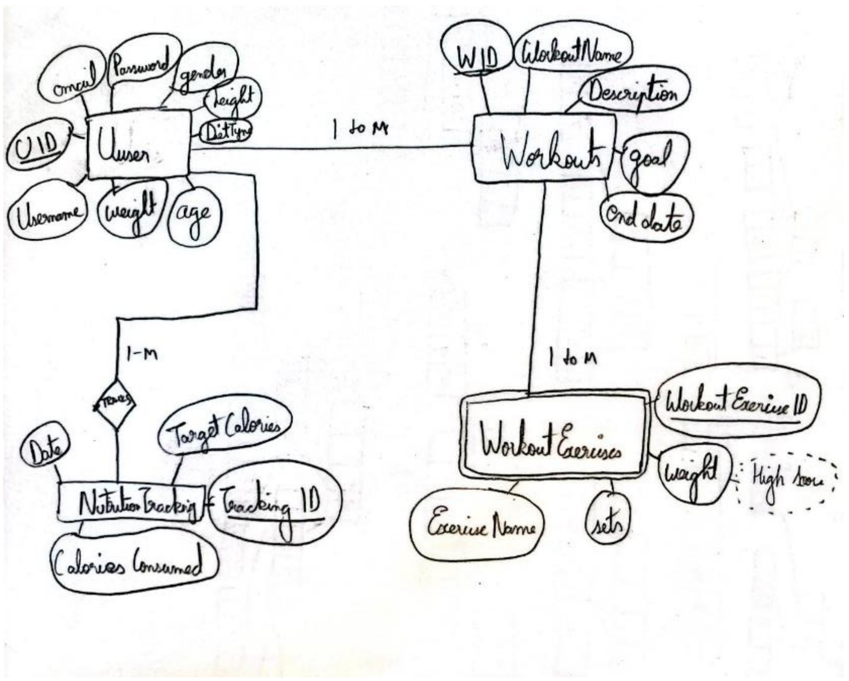
The 3 most important functionalities are

1. creating/deleting/editing workouts and exercises
2. getting information on physical health based on BMI and calorie intake
3. tracking fitness progress

These are primary features, secondary features include checking constraints, deleting exercises before workouts, making unique primary keys, and giving better user interaction.

Database Design

ER Diagram



Explanation:

In terms of tracking, A user can enter multiple entries under his/her name and can keep track of them. So, it's 1-M relationship. Similarly, for Workouts, 1 user can have many workouts(such as "upper body", "lower body", "push day", "pull day", etc). 1 workout can have multiple exercises(such as workout "Leg Day" can have exercises such as "Squats" , "Lunges", etc). Note that Workout Exercises table is a weak entity as it cannot exist without its workout and also Workout Exercises has a discriminator called "High Score". Based on the rules of a discriminator, it can give meaning to a particular exercise and can distinguish itself from weight, whereas in larger picture it is meaning less as it does not give meaning.

SQL Statements For Creating Table

```
CREATE TABLE UUser (

  UserID INT PRIMARY KEY,

  Username VARCHAR(20),

  Email VARCHAR(50),

  Password VARCHAR(30),

  Age INT,

  Gender VARCHAR(6),

  Height INT,

  Weight INT,

  DietType VARCHAR(20)

);

CREATE TABLE Workouts (

  WorkoutID INT PRIMARY KEY,

  UserID INT,

  WorkoutName VARCHAR(30),

  Description VARCHAR(50),

  Goal INT,

  EndDate DATE,

  FOREIGN KEY (UserID) REFERENCES Uuser

);

CREATE TABLE WorkoutExercises (

  WorkoutExerciseID INT PRIMARY KEY,

  WorkoutID INT,

  ExerciseName VARCHAR(100),

  Sets INT,
  Reps INT,
  Weight INT,
  HighScore INT,
  FOREIGN KEY (WorkoutID) REFERENCES Workouts,
  CHECK (HighScore > Weight) -- Constraint for HighScore must be greater than Weight
);

CREATE TABLE NutritionTracking (
  TrackingID INT PRIMARY KEY,
  UserID INT,
  CaloriesConsumed INT,
  CaloriesTarget INT,
  DateRecorded DATE DEFAULT SYSDATE, -- Set DateRecorded to the current date using SYSDATE
  FOREIGN KEY (UserID) REFERENCES Uuser
);
```

NOTE:

- There are more constraints and automation when interacting with tables through C++ functions and not with the actual base, this helps with the program not to break when running.
- I kept the User table as UUser because I already have a table with that name and was having conflicts when creating it.

Implementation

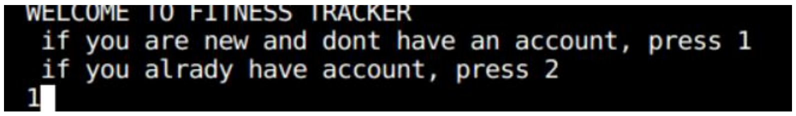
The implementation is divided into 2.

NOTE: To navigate between the options, use the numbers corresponding to it.

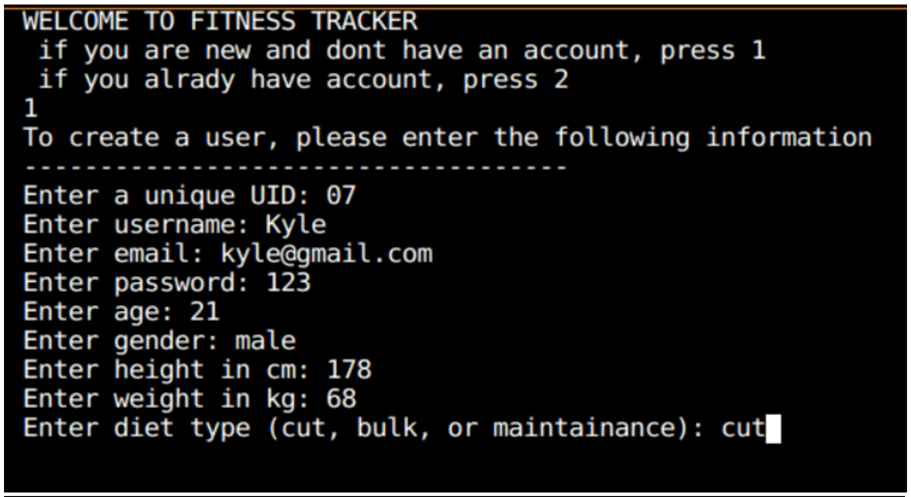
1. Creating a user and making a workout list
2. Logging in as an existing user and tracking nutrition.

Creating a user and making a workout list with exercises

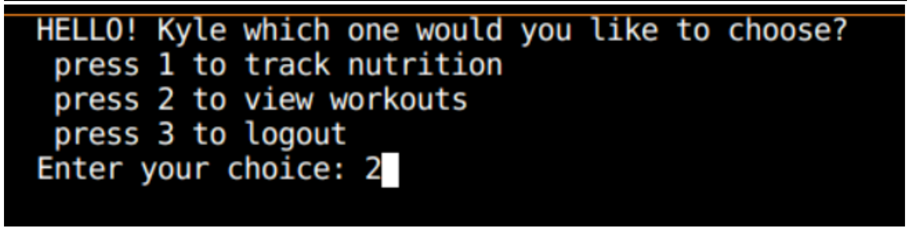
1.1. Once you login with CSCI lab username and password you will get the screen shown below.



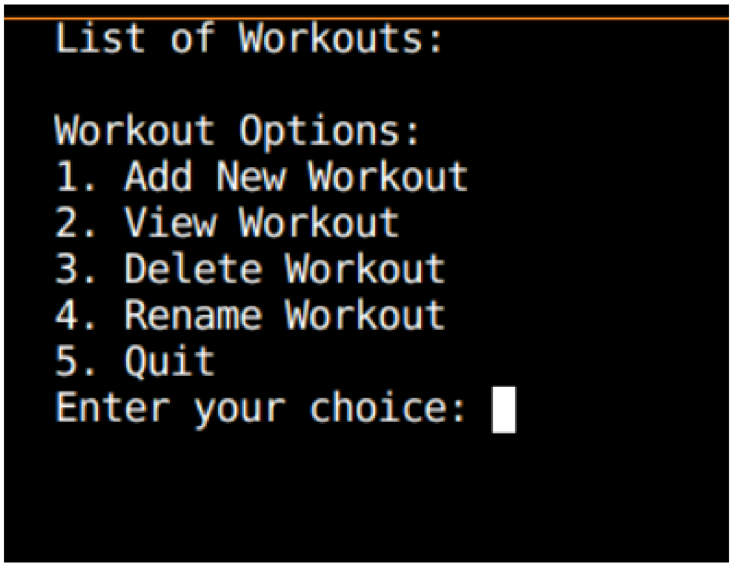
```
WELCOME TO FITNESS TRACKER
 if you are new and dont have an account, press 1
 if you alrady have account, press 2
1
```



```
WELCOME TO FITNESS TRACKER
 if you are new and dont have an account, press 1
 if you alrady have account, press 2
1
To create a user, please enter the following information
-----------------------------------
Enter a unique UID: 07
Enter username: Kyle
Enter email: kyle@gmail.com
Enter password: 123
Enter age: 21
Enter gender: male
Enter height in cm: 178
Enter weight in kg: 68
Enter diet type (cut, bulk, or maintainance): cut
```

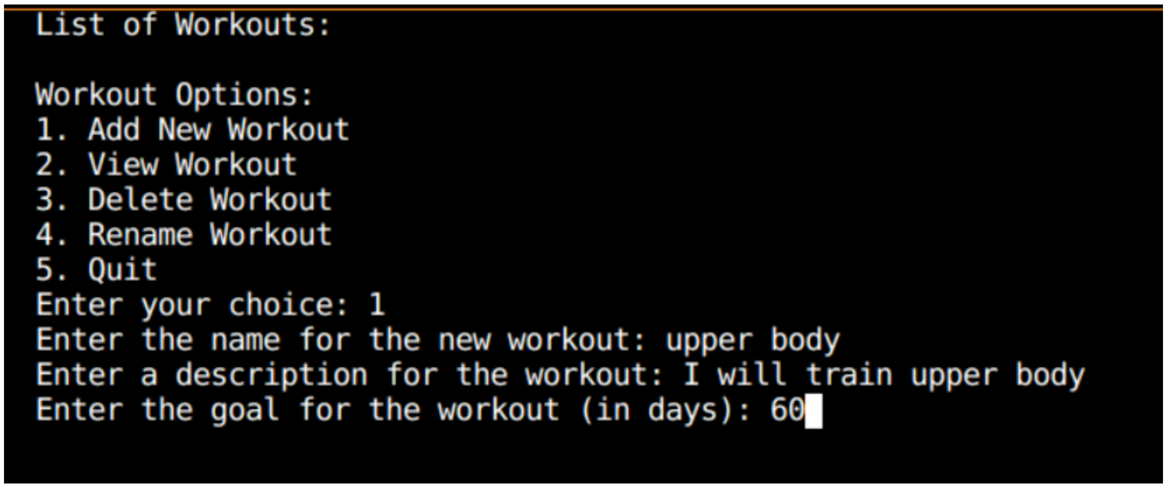


```
HELLO! Kyle which one would you like to choose?
  press 1 to track nutrition
  press 2 to view workouts
  press 3 to logout
Enter your choice: 2
```



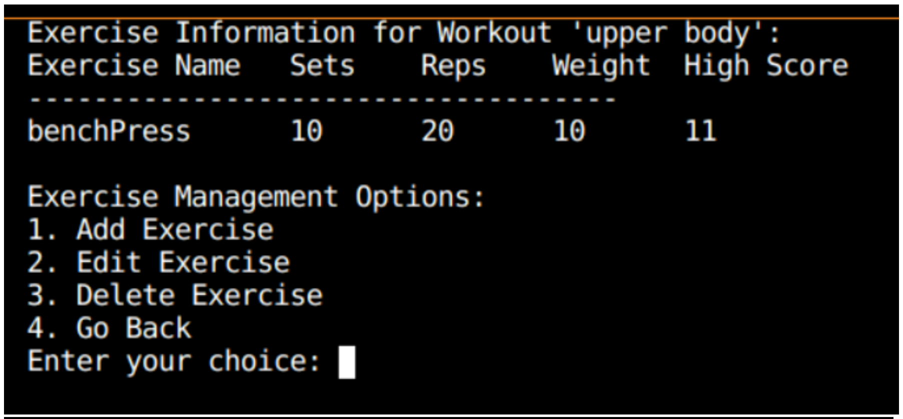
```
List of Workouts:

Workout Options:
1. Add New Workout
2. View Workout
3. Delete Workout
4. Rename Workout
5. Quit
Enter your choice: 
```



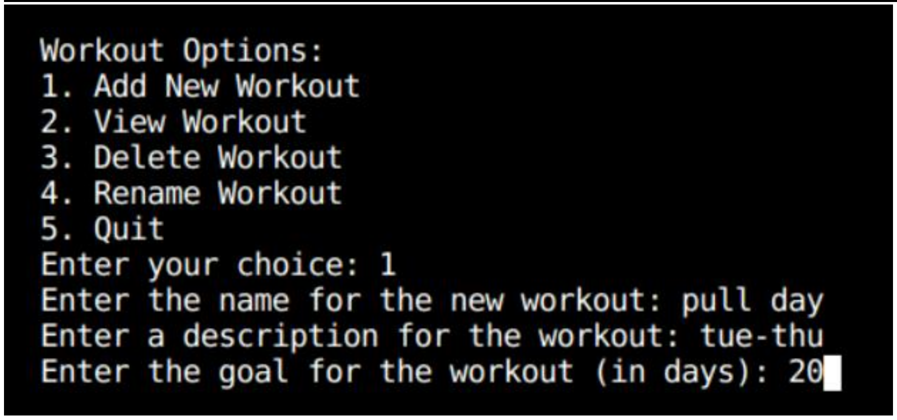
```
List of Workouts:

Workout Options:
1. Add New Workout
2. View Workout
3. Delete Workout
4. Rename Workout
5. Quit
Enter your choice: 1
Enter the name for the new workout: upper body
Enter a description for the workout: I will train upper body
Enter the goal for the workout (in days): 60
```



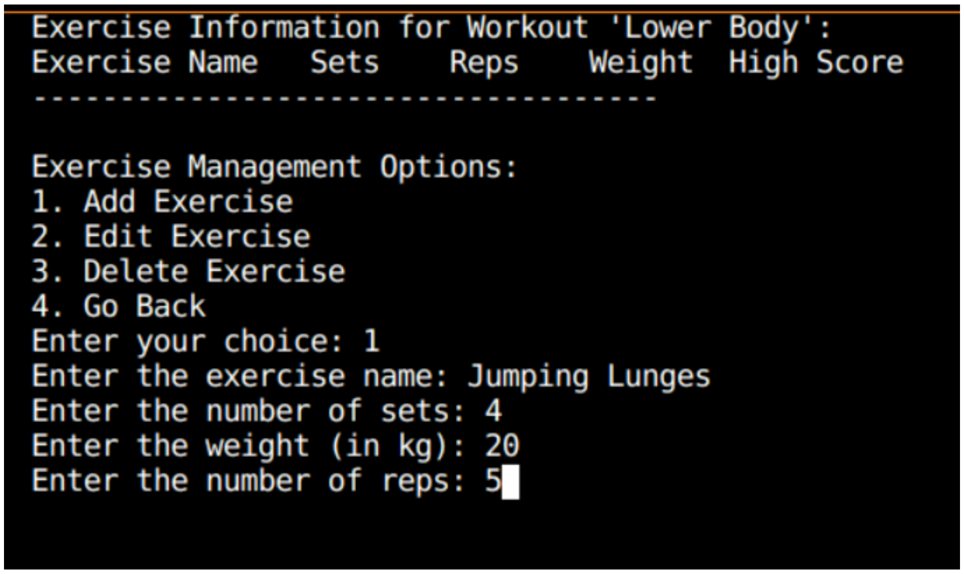
```
Exercise Information for Workout 'upper body':
Exercise Name   Sets    Reps    Weight  High Score
-----------------------------------
benchPress      10      20      10      11

Exercise Management Options:
1. Add Exercise
2. Edit Exercise
3. Delete Exercise
4. Go Back
Enter your choice: 
```



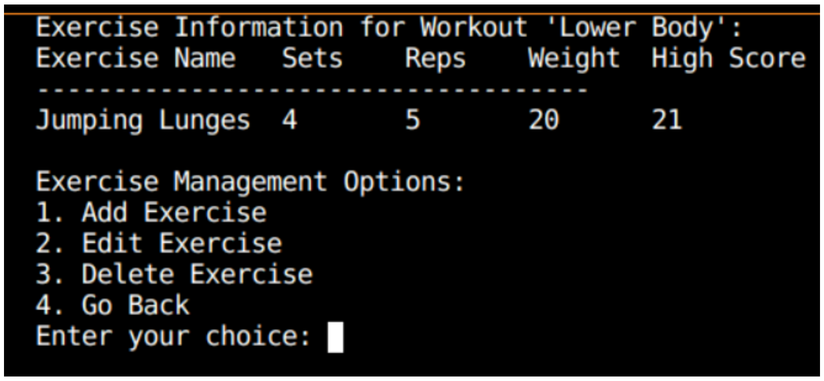
```
Workout Options:
1. Add New Workout
2. View Workout
3. Delete Workout
4. Rename Workout
5. Quit
Enter your choice: 1
Enter the name for the new workout: pull day
Enter a description for the workout: tue-thu
Enter the goal for the workout (in days): 20
```

Inside Lower Body workout (press 2 from workouts section and enter workout name to view it)



```
Exercise Information for Workout 'Lower Body':
Exercise Name   Sets    Reps    Weight  High Score
-----------------------------------

Exercise Management Options:
1. Add Exercise
2. Edit Exercise
3. Delete Exercise
4. Go Back
Enter your choice: 1
Enter the exercise name: Jumping Lunges
Enter the number of sets: 4
Enter the weight (in kg): 20
Enter the number of reps: 5
```



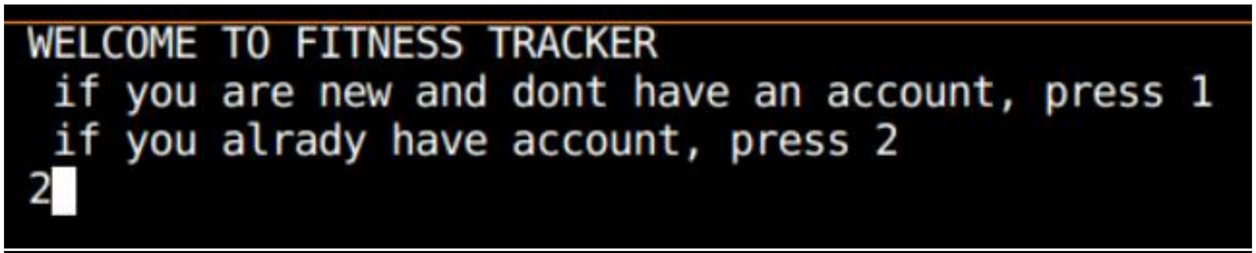
```
Exercise Information for Workout 'Lower Body':
Exercise Name   Sets    Reps    Weight  High Score
----------------------------------
Jumping Lunges  4       5       20      21

Exercise Management Options:
1. Add Exercise
2. Edit Exercise
3. Delete Exercise
4. Go Back
Enter your choice:
```

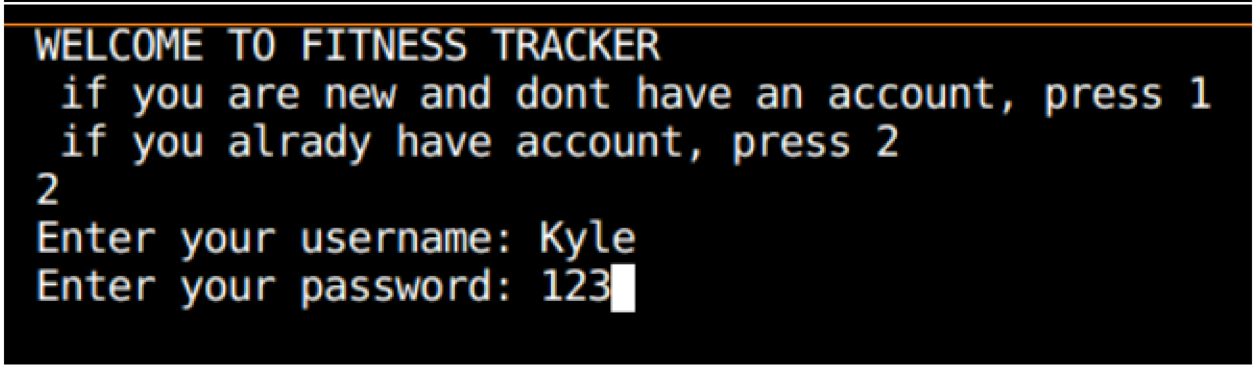
Then 4->5 to quit the program

Logging in as an existing user and checking the nutrition information

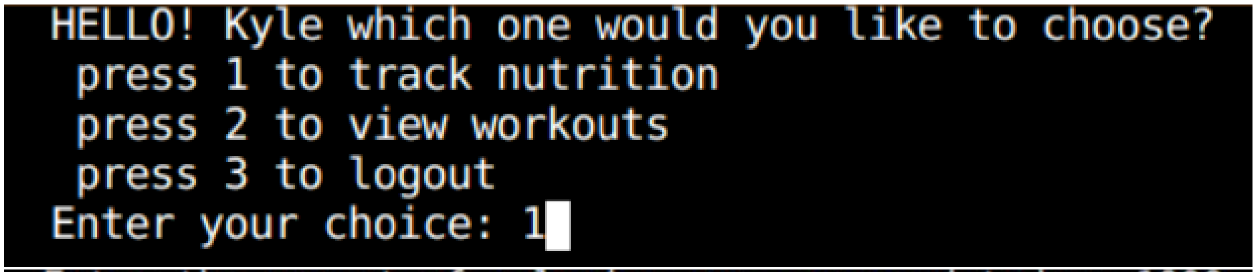
(Note that the program will terminate after the completion of the nutritionTracking function)



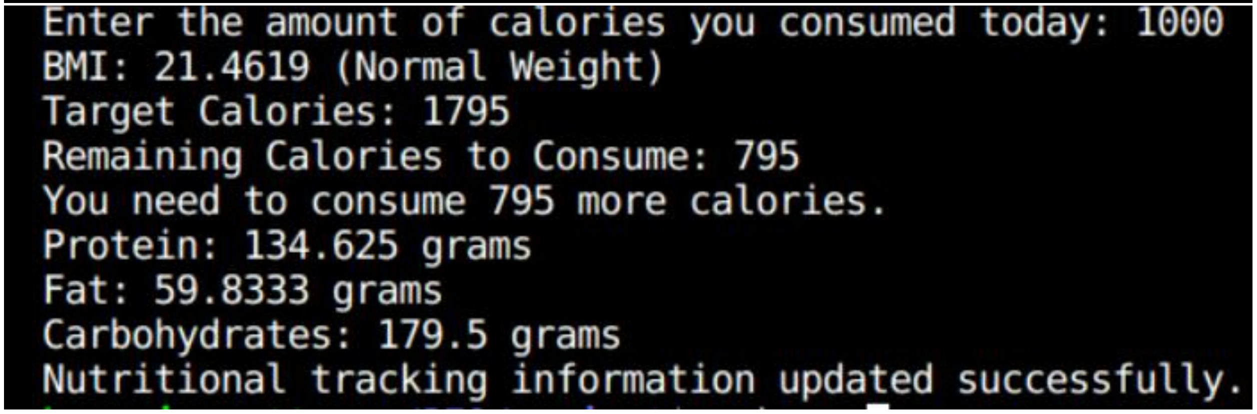
```
WELCOME TO FITNESS TRACKER
 if you are new and dont have an account, press 1
 if you alrady have account, press 2
2
```



```
WELCOME TO FITNESS TRACKER
 if you are new and dont have an account, press 1
 if you alrady have account, press 2
2
Enter your username: Kyle
Enter your password: 123
```



```
HELLO! Kyle which one would you like to choose?
 press 1 to track nutrition
 press 2 to view workouts
 press 3 to logout
Enter your choice: 1
```



```
Enter the amount of calories you consumed today: 1000
BMI: 21.4619 (Normal Weight)
Target Calories: 1795
Remaining Calories to Consume: 795
You need to consume 795 more calories.
Protein: 134.625 grams
Fat: 59.8333 grams
Carbohydrates: 179.5 grams
Nutritional tracking information updated successfully.
```

Assumptions

Assumptions are fairly realistic except for the first one.

- User doesn’t make any spelling mistakes when entering information
- Highscore initially will be weight+1 (the user can update it later and can set it higher but not lower)
- User enters nutrition data before dinner to see the information, this is more inclined toward logic as the resulting amount of calories will indicate how much to eat or not to eat at all based on the goal.
- This app is only intended for weight training exercises.

Testing

This section contains sample data to be inserted and tested aswell as the actual test cases with expected results.

My suggestion:

1. is to run the program using this “Kyle” user create exercises and perform modifications on it to test the program(this is less time-consuming).
2. For creating a user, just create one :)

Sample Data to insert in oracle

NOTE: My suggestion is to just create an account and play around, it inserts some values automatically such as generating PKs, filling dates etc. I have provided this tested insert statements just in case if you want to insert and test

UUSER

INSERT INTO Uuser (UserID, Username, Email, Password, Age, Gender, Height, DietType, Weight)

VALUES (1, 'Kyle', 'kyle@gmail.com', '123', 21, 'male', 178, 'cut', 68);

Workouts

INSERT INTO Workouts (WorkoutID, UserID, WorkoutName, Description, Goal, EndDate)

VALUES (42, 1, 'push day', 'mon-tue-wed', 70, TO_DATE('2024-06-30', 'YYYY-MM-DD'));

INSERT INTO Workouts (WorkoutID, UserID, WorkoutName, Description, Goal, EndDate)

VALUES (65, 1, 'upper body', 'I will train upper body', 60, TO_DATE('2024-07-05', 'YYYY-MM-DD'));

INSERT INTO Workouts (WorkoutID, UserID, WorkoutName, Description, Goal, EndDate)

VALUES (26, 1, 'Lower Body', 'Will train lower body', 70, TO_DATE('2024-06-15', 'YYYY-MM-DD'));

INSERT INTO Workouts (WorkoutID, UserID, WorkoutName, Description, Goal, EndDate)

VALUES (56, 1, 'pull day', 'tue-thu', 20, TO_DATE('2024-05-10', 'YYYY-MM-DD'));

Workout exercises

INSERT INTO WorkoutExercises (WorkoutExerciseID, WorkoutID, ExerciseName, Sets, Reps, Weight, HighScore)

VALUES (1, 65, 'Bench Press', 4, 10, 100, 120);

INSERT INTO WorkoutExercises (WorkoutExerciseID, WorkoutID, ExerciseName, Sets, Reps, Weight, HighScore)

VALUES (2, 65,’Lat Pull’, 4, 10, 70, 80);

INSERT INTO WorkoutExercises (WorkoutExerciseID, WorkoutID, ExerciseName, Sets, Reps, Weight, HighScore)

VALUES (3, 65, 'Barbell Curl', 4, 10, 25, 35);

Test Cases

MAIN FUNCTION TESTING

TEST NAME	**INPUT**	**EXPECTED OUTPUT**
Valid Login Test	Correct CSCI username and password for an existing account.	Successfully logs in and proceeds to the main menu of the fitness tracker.
New User Creation Test:	Choosing the option to create a new account.	Prompts for creating a new user with necessary details such as username, password, height, weight, etc., and successfully creating the user account.
Invalid Choice Test:	Entering an invalid choice (other than 1, 2, or 3) at the main menu.	Displays an error message indicating an invalid choice and prompts for re-entering a valid choice.
Program Termination Test:	Choosing the option to terminate the program.	Displays a termination message and exits the program.
Environment Creation Test:	Provide correct CSCI username and password to create the environment and connection.	Successfully creates the environment and connection to the database
Environment Termination Test:	Choosing to terminate the program after using the fitness tracker.	Properly terminates the connection and environment before exiting the program.

CREATE USER FUNCTION TESTING

Valid User Creation Test:	Provide unique values for UID, username, email, password, age, gender, height, weight, and diet type (e.g., "cut").	Successfully creates the user account in the UUser database table and displays "Insertion Successful".
Existing UID Test:	Provide a UID that already exists in the database.	Display an error message indicating that the UID already exists and prompt for entering a different UID.

SQL Exception Test:	Simulate an SQL exception during the UID check or user insertion process.	Simulate an SQL exception during the UID check or user insertion process.
Redirect to Homepage Test:	Successfully create a user account and check if the function redirects to the homepage.	After successful user creation, the function should clear the screen and call the homepage function with appropriate parameters.

HOMEPAGE FUNCTION TESTING

Track Nutrition Test:	Choose option 1 to track nutrition.	Calls the nutritionTracking function with appropriate parameters and tracks nutrition data for the user.
View Workouts Test:	Choose option 2 to view workouts.	Calls the workouts function with appropriate parameters and displays workout information for the user.
Logout Test:	Choose option 3 to logout.	Logs out the user and exits the homepage function.
Invalid Choice Test:	Enter an invalid choice (other than 1, 2, or 3).	Displays an error message indicating an invalid choice and prompts for re-entering a valid choice.
Nutrition Tracking Call Test:	Check if choosing option 1 correctly calls the nutritionTracking function with the user's information.	Upon choosing option 1, the nutritionTracking function should be called with the user's UID, height, weight, and diet type.
Workouts View Call Test:	Check if choosing option 2 correctly calls the workouts function with the user's information.	Upon choosing option 2, the workouts function should be called with the user's name and UID.
Logout Functionality Test:	Check if choosing option 3 successfully logs out the user and terminating the program	Upon choosing option 3, the function should log out the user and exit the homepage function and terminates the program.

<u>LOGIN FUNCTION TESTING</u>

Valid Login Test:	Enter username "Kyle" and password "123".	Logs in successfully and calls the homepage function with the correct user information (username, UID, diet type, height, weight).
Invalid Username/Password Test:	Enter a non-existent username "NonExistingUser" and password "123".	Displays "Incorrect information, credentials wrong, or the user doesn't exist" message.

Add New Workout Test:	Choose option 1 to add a new workout, Workout Name: "Leg Day", Description: "Leg strength training", Goal: 30 days	Successfully adds the new workout "Leg Day" for the user "test" and the workout name can be seen in the top.
View Existing Workout Test:	Choose option 2 to view a workout. Workout Name: "Leg Day"	Displays the details of the "Leg Day" workout by prompting to workoutExercises function.
Delete Existing Workout Test:	Choose option 3 to delete a workout. Workout Name to Delete: "Leg Day", Confirm Deletion: Y (Yes).	Deletes the "Leg Day" workout from the database.
Rename Existing Workout Test:	Choose option 4 to rename a workout, Old Workout Name: "Leg Day", New Workout Name: "Upper Body".	Changes can be seen in the table and screen
Quit Test:	Choose option 5 to quit the workout management.	Exits the workout management loop and terminates the program.
Invalid Choice Test:	Choose an invalid option (e.g., 6).	Displays “Invalid choice” and continues the loop

EXERCISE FUNCTION TESTING

Add Exercise Test:	Choose option 1 to add a new exercise.	Successfully adds the exercise "Squats" to the "Leg Day" workout.
Edit Exercise Test:	Choose option 2 to edit an exercise. Exercise Name: "Squats", Column Name: "Reps",New Value: 15	Updates the reps of the "Squats" exercise to 15 if it exists, otherwise prompts that the exercise doesn't exist.
Delete Exercise Test:	Choose option 3 to delete an exercise. Exercise Name to Delete: "Squats"	Deletes the "Squats" exercise if it exists, otherwise prompts that the exercise doesn't exist.
Go Back Test:	Choose option 4 to go back.	Exits the exercise management loop and returns to the workout management options.
Invalid Choice Test:	Choose an invalid option (e.g., 5).	Displays "Invalid choice" and continues the loop.

GENERATE NUTRITIONID FUNCTION TESTING

Generate Unique Nutrition ID Test:	Call the generateNutritionID function multiple times.	Each call should return a unique string that does not already exist in the NutritionTracking table.

GENERATE RANDOM NUMBER FUNCTION TESTING

Generate Unique Random Number Test:	Call the generateRandomNumber function multiple times.	Each call should return a unique string that does not already exist in the Workouts table.

CALCULATE END DATE FUNCTION TESTING

Calculate End Date Test:	Goal days = 30 (for example).	The function should calculate the end date correctly based on the current date + 30 days.

NUTRITION TRACKING FUNCTION TESTING

Valid Input Test:	Provide valid input for username, weight, height, and diet type.	The function should calculate and display BMI, target calories, remaining calories, and macronutrient recommendations correctly. It should also update the NutritionTracking table with the provided information.
Database Update Test:	Test the database update functionality by checking if the NutritionTracking table is updated correctly with the provided information.	Displays “insertion successful”

GENERATE RANDOM EXERCISE ID FUNCTION TESTING

Generate Unique Exercise ID Test:	Call the generateRandomExerciseID function multiple times.	Each call should return a unique string that does not already exist in the WorkoutExercises table.

Things I could have done better

- Better constraints.
- Better and more Test cases.
- More iterations to make things better.

Conclusion And Future Work

This is a really fascinating project. I made this project in such a way that it wont restrict me to the CSCI 370 course but will give me a good amount of opportunities to improve in the future(long term-based project)even though I was familiar with the domain, though I have successfully implemented most of the tasks proposed in the proposal document, I still feel like I could have done a better job at implementing it. Some of the attributes are redundant in the current application but are useful in future implementation as they will be used in more complex calculations. I thoroughly loved the process. I'll be putting in more effort on the project in the future. I wrap up by stating that to improve my work and make it even better, I will keep working on it and apply the knowledge I've gained from other courses as well and make it more fascinating.