Project On Fitness Tracker App

Group member :-

Name	Metric No
ABDIKARIN OSMAN MOHAMED	BI220042
FAHAD ABDIHAKIM MOHAMED	AI220350
MARIAN ABDIRISAK MOHAMED	AI200337
ABDULLAHI MOHAMED ABDULLAHI	BI220050

Introduction:

In the fast-paced world we live in today, health and fitness have become paramount for individuals seeking a balanced and active lifestyle. The Fitness Tracker app is a comprehensive solution designed to assist users in monitoring and managing their fitness goals seamlessly. Developed using object-oriented programming (OOP) principles and implemented in C++, this application offers a user-friendly interface for both administrators and users.

Objective

The primary objective of the Fitness Tracker app is to provide users with a tool that not only tracks their physical activities but also calculates essential health metrics such as Basal Metabolic Rate (BMR) and calories burned. This app caters to two distinct user types: administrators responsible for managing user data, and individual users looking to maintain a healthy lifestyle.

Features

User Registration and Management:

Users can create individual accounts with unique usernames.

The system generates a personalized User ID for each registered user.

Administrators have the ability to add, update, and delete user accounts.

Health Metrics Calculation:

The app calculates the Basal Metabolic Rate (BMR) based on user-specific factors such as age, weight, height, and gender.

Users receive insights into their daily calorie expenditure based on their chosen activity level.

Activity Classification:

Users can choose their activity level from a range of options, including sedentary, lightly active, moderately active, very active, and extra active.

Each activity level is associated with a unique identifier, aiding in personalized data management.

User-Friendly Interface:

The application features an intuitive and easy-to-navigate interface for a seamless user experience.

Administrators can efficiently manage user data through a dedicated set of functionalities.

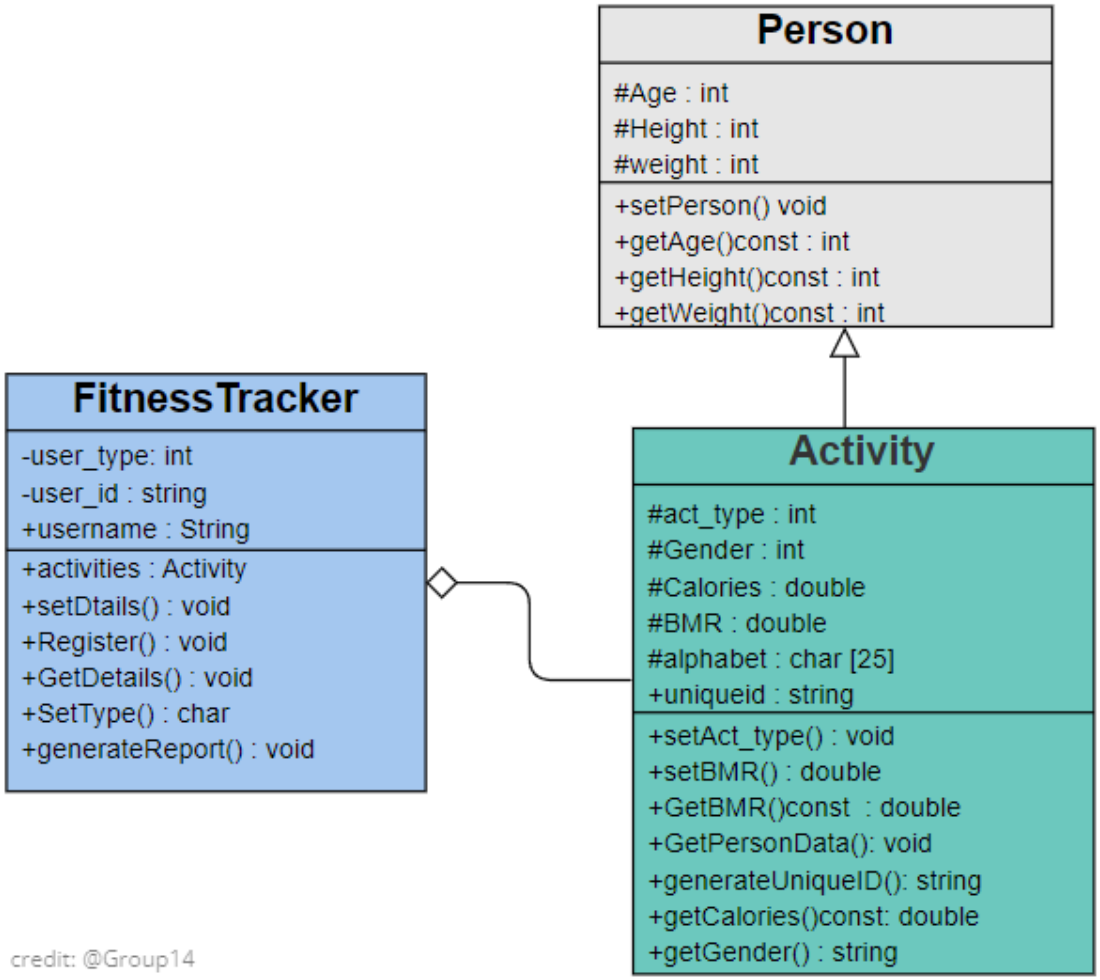
Implementation:

The Fitness Tracker app leverages the power of C++ and OOP concepts to create a modular and extensible system. It utilizes linked lists to organize user data efficiently. The inclusion of dynamic data generation, such as unique user IDs, adds an element of randomness and uniqueness to each user's profile.

This project report provides an in-depth analysis of the development process, challenges faced, and the overall architecture of the Fitness Tracker app. As we delve into the details of each feature and functionality,

it is our hope that this application serves as a valuable tool for individuals striving to achieve their fitness goals.

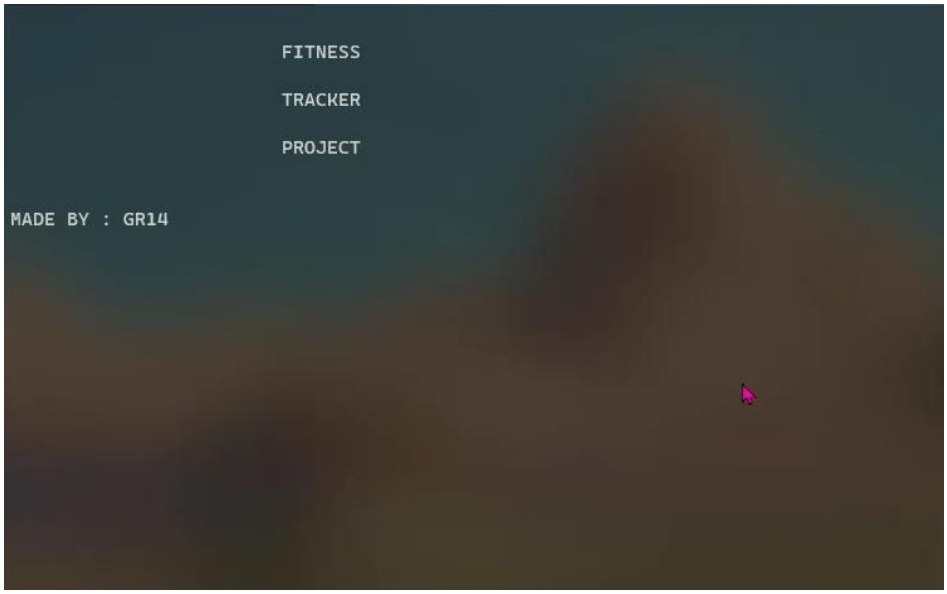
- Class Diagram:

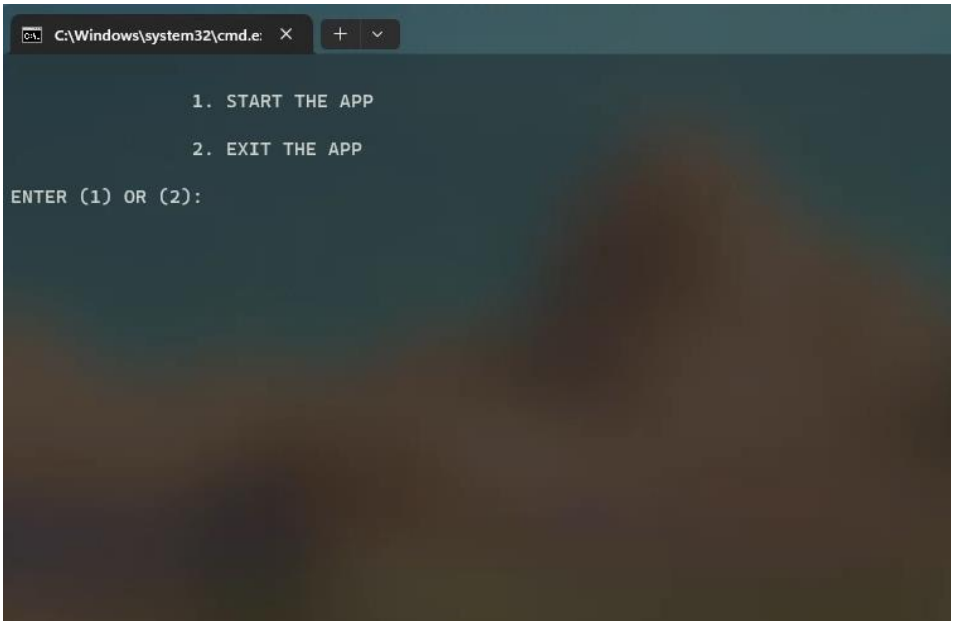


Explanation:

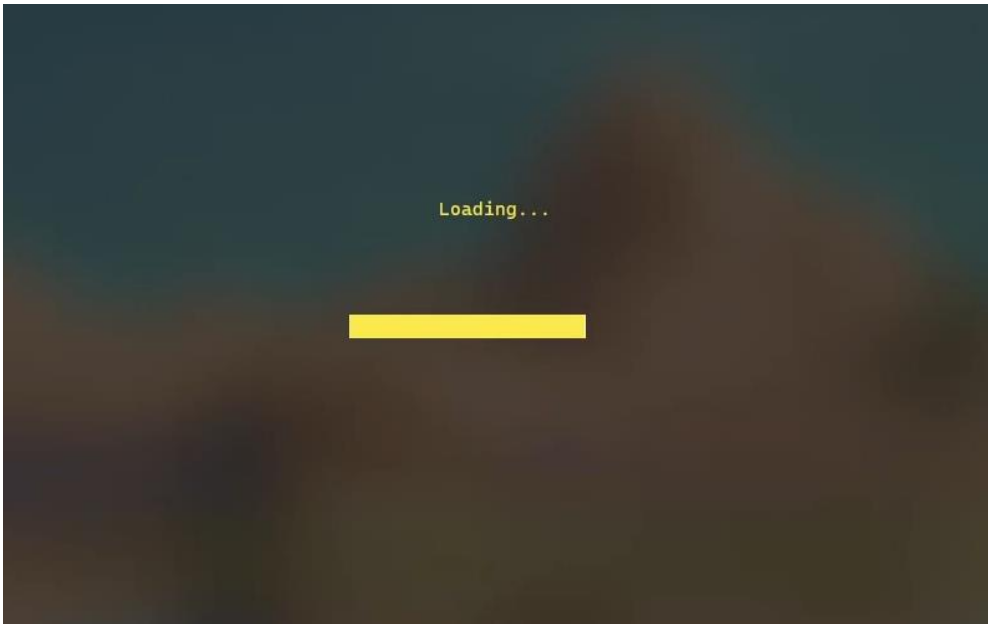
Methods	Explanation
Person::Setperson()	Set User's Age, Weight, And Height.
Person::Getage()	Get user's age.
Person::Getheight()	Get user's height.
Person::Getweight()	Get user's weight.
Activity::Setact_Type()	Set the type of activity.
Activity::Setbmr()	Calculate and set user's BMR.
Activity::Getbmr()	Get user's BMR.
Activity::Getpersondata()	Display user's age, height, and weight.
Activity::Generateuniqueid()	Generate a unique ID for the user.
Activity::Getcalories()	Get calories burned based on activity.
Activity::Getgender()	Get user's gender as a string.
Fitnesstracker::Settype()	Set user type as admin or user.
Fitnesstracker::Setdtails()	Set details such as BMR for the user.
Fitnesstracker::Register()	Register a user with a username.
Fitnesstracker::Getdetails()	Display user details.
Fitnesstracker::Generatereport()	generate a report and save to "Report.txt".

Code Output:





(1) entered: (starting the App):



admin account Type has abilities to add delete search update accounts and generate reports.

```
..............Login Type.............
1.administrator Account Type
2.User Account Type
Choose (1) Or (2) : 1


.............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.........................................................................
Enter (1-7):
```

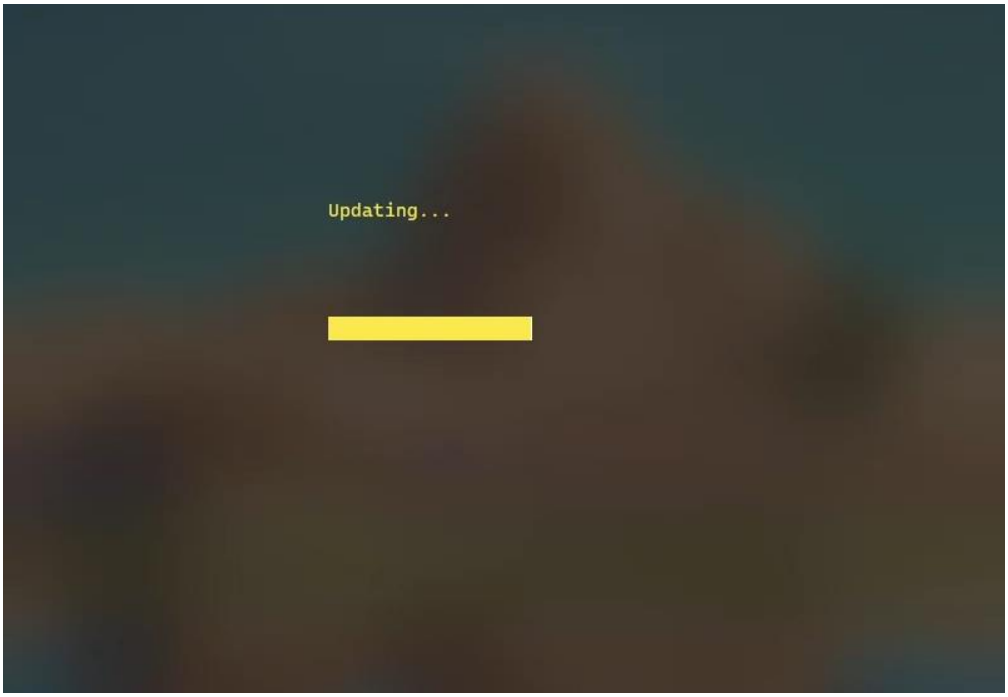
Some input Validation ⬇

```
..............Login Type.............
1.administrator Account Type
2.User Account Type
Choose (1) Or (2) : 1


.............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.........................................................................
Enter (1-7): d
Invalid choice. Enter (1-7): 2
Enter The UserName To delete: b
There Is No Data Yet..
User b not found.


.............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.........................................................................
Enter (1-7):
```

```
Enter (1-7): 1
..............User Account Creation.............
Enter UserName : mahir
..Choose Your Gender..
1. Male          2. Female
Enter Your Choice : 1
.........BMR Calculation.........
Enter Your Age : 0
Invalid input!
Enter Your Age : 20
Enter Your Weight : 1
Not a Normal Weight Try Again
Enter Your Weight (kg) : 900
Not a Normal Weight Try Again
Enter Your Weight (kg) : 65
Enter Your Hight : 1
1 Is Invalid Height Try Again
Enter Your Hight(cm) : 500
500 Is Invalid Height Try Again
Enter Your Hight(cm) : 175

.............................which Activities You Do ......................
1.sedentary (little or no exercise)

2.lightly active (light exercise/sports 1-3 days/week)
```

Update the whole information for certain user.

```
Your Auto Generted ID : FO10


.................................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
..........................................................................
Enter (1-7): 3
Enter The UserName To Update: mahir
Displaying Existing Data For mahir :
...............User Data..............
|Username        : @mahir
|User Id         : FO10
|User Gender     : Male
|Age             : 20
|Height          : 175
|Weight          : 65
|User BMR        : 1648.75
|Calories Burned : 2555.56
|_____________________________________________
.....................Update Field.....................
..Choose Your Gender..
1. Male         2. Female
Enter Your Choice : 1
..........BMR Calculation..........
Enter Your Age : 30
Enter Your Weight : 70
Enter Your Hight : 1
```




```
User mahir updated successfully.


.................................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
..........................................................................
Enter (1-7): 5
...............User Data..............
|Username        : @mahir
|User Id         : K11
|User Gender     : Male
|Age             : 30
|Height          : 178
|Weight          : 70
|User BMR        : 1667.5
|Calories Burned : 2876.44
|_____________________________________________
```

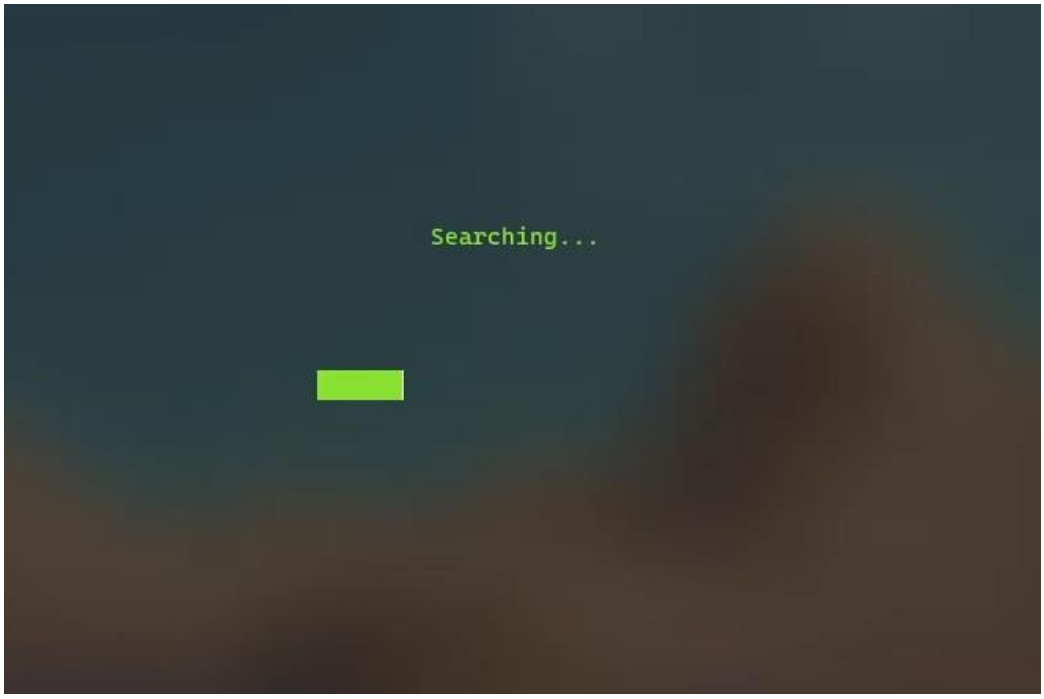
```
.............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.....................................................................
Enter (1-7): 1
.............User Account Creation.............
Enter UserName : ali
..Choose Your Gender..
1. Male         2. Female
Enter Your Choice : 1
.........BMR Calculation.........
Enter Your Age : 40
Enter Your Weight : 79
Enter Your Hight : 180

.............................which Activities You Do ..............................
1.sedentary (little or no exercise)

2.lightly active (light exercise/sports 1-3 days/week)

3.moderately active (moderate exercise/sports 3-5 days/week)

4.very active (hard exercise/sports 6-7 days a week)

5.extra active (very hard exercise/sports & physical job or 2x training)

.....................................................................................
Choose (1-5) :
```

Automatically Sorting user Names in alphabetical order.

```
Enter (1-7): 5
.............User Data.............
|Username        : @ali
|User Id         : C12
|User Gender     : Male
|Age             : 40
|Height          : 180
|Weight          : 79
|User BMR        : 1720
|Calories Burned : 2365
|_____________________________________________
.............User Data.............
|Username        : @mahir
|User Id         : K11
|User Gender     : Male
|Age             : 30
|Height          : 178
|Weight          : 70
|User BMR        : 1667.5
|Calories Burned : 2876.44
|_____________________________________________


.............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.....................................................................
Enter (1-7):
```

Ability to search existing username.

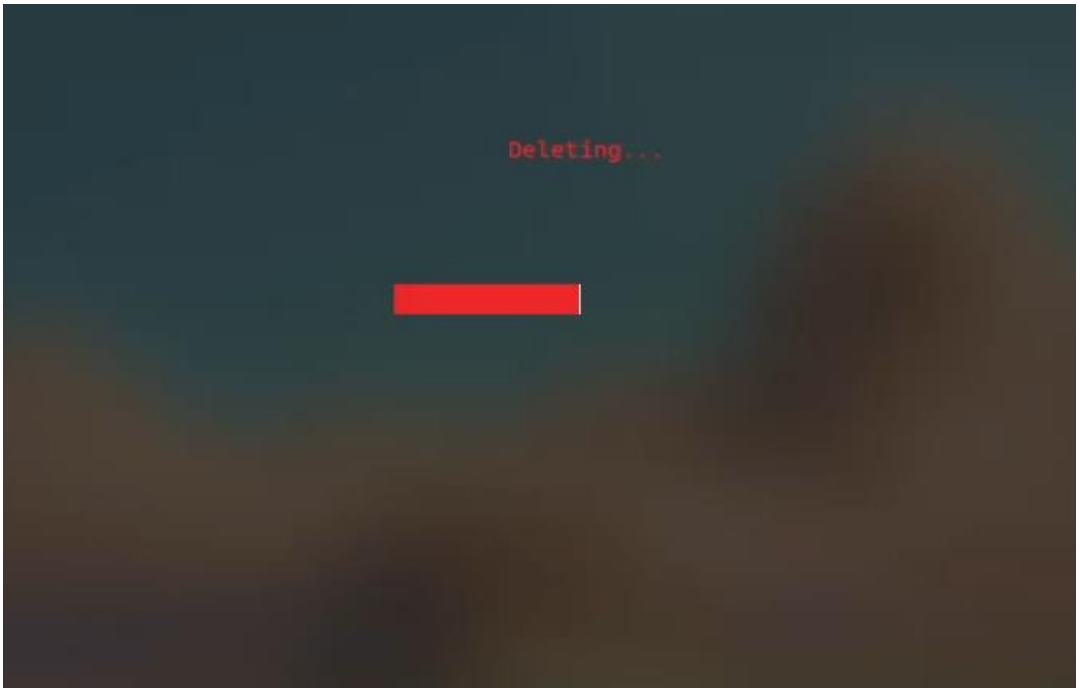
```
.............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.....................................................................
Enter (1-7): 6
Enter The UserName To Search: ali
```




```
------------- UserName :alifound successfully-------------
..............User Data..............
|Username        : @ali
|User Id         : C12
|User Gender     : Male
|Age             : 40
|Height          : 180
|Weight          : 79
|User BMR        : 1720
|Calories Burned : 2365
|-----------------------------------------
.....................................................................


..............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.....................................................................
Enter (1-7):
```

Ability to Delete existing username.

```
..............................Menu.....................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.....................................................................
Enter (1-7): 2
Enter The UserName To delete: ali
```




```
User ali deleted successfully..


.................................Menu.........................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
..............................................................................
Enter (1-7): 5
..............User Data..............
|Username        : @mahir
|User Id         : K11
|User Gender     : Male
|Age             : 30
|Height          : 178
|Weight          : 70
|User BMR        : 1667.5
|Calories Burned : 2876.44
|_____________________________________________
```

Second account type user account;

```
..............Login Type..............
1.administrator Account Type
2.User Account Type
Choose (1) Or (2) : 2
```

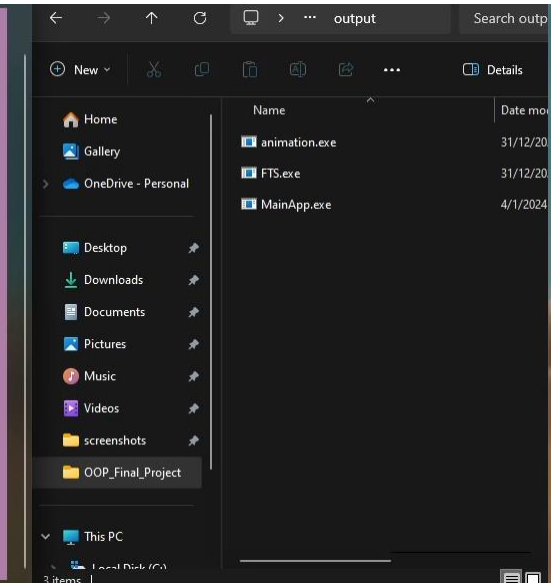
Ability to directly calculate essential health metrics such as Basal Metabolic Rate (BMR) and calories burned.

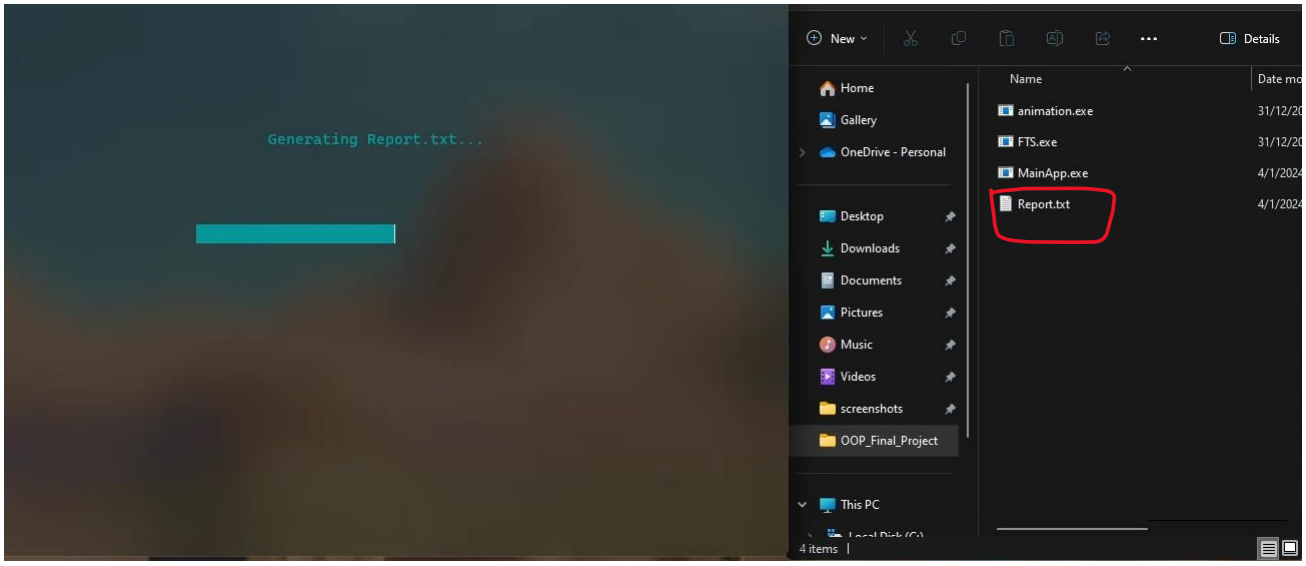
```
2.User Account Type
Choose (1) Or (2) : 2



.....................................................................................
                1. BMR & Calories Calculation           2.Exit

choose (1)or(2) : 1
..Choose Your Gender..
1. Male          2. Female
Enter Your Choice : 1
..........BMR Calculation..........
Enter Your Age : 30
Enter Your Weight : 76
Enter Your Hight : 187

.................................which Activities You Do .......................
1.sedentary (little or no exercise)

2.lightly active (light exercise/sports 1-3 days/week)

3.moderately active (moderate exercise/sports 3-5 days/week)

4.very active (hard exercise/sports 6-7 days a week)

5.extra active (very hard exercise/sports & physical job or 2x training)

.....................................................................................
Choose (1-5) :
```

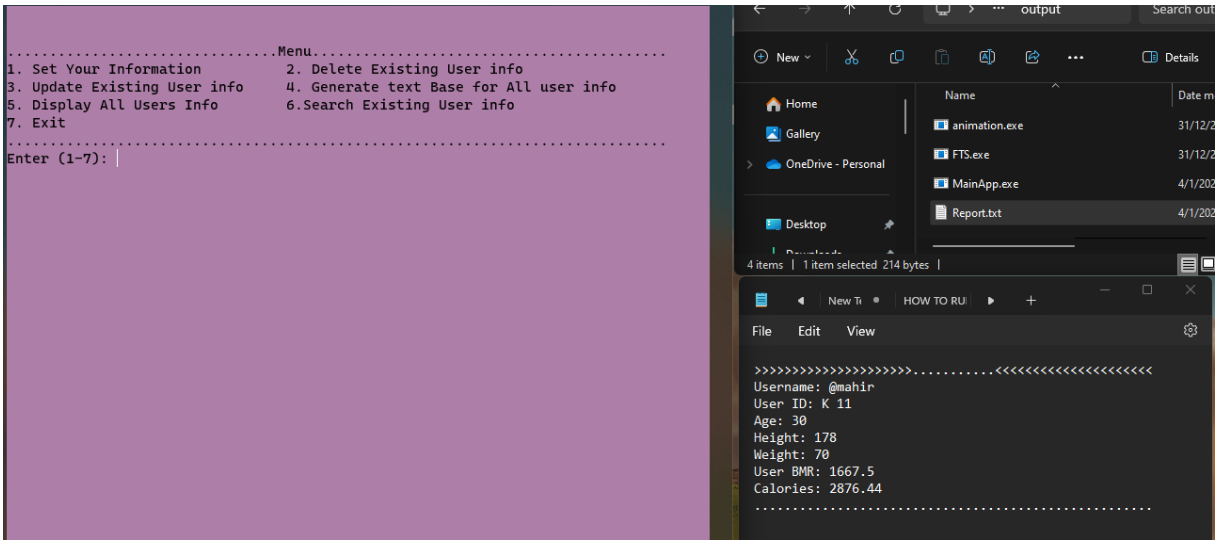
```
.....................................................................................
Choose (1-5) : 4
.....................................Result.........................................

Your BMR : 1783.75
Your Calories : 3076.97
.....................................................................................



.....................................................................................
                1. BMR & Calories Calculation           2.Exit
```

ability to generate user Info as Text format for Reports.

```
2.User Account Type
Choose (1) Or (2) : 1



...............................Menu.........................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.....................................................................................
Enter (1-7): 5
..............User Data..............
|Username        : @mahir
|User Id         : K11
|User Gender     : Male
|Age             : 30
|Height          : 178
|Weight          : 70
|User BMR        : 1667.5
|Calories Burned : 2876.44
|_____________________________________________


...............................Menu.........................................
1. Set Your Information          2. Delete Existing User info
3. Update Existing User info     4. Generate text Base for All user info
5. Display All Users Info        6.Search Existing User info
7. Exit
.....................................................................................
Enter (1-7): 4
```







Class Source Code:

```
#include <iostream>

#include <Windows.h>
#include <fstream>
#include <string>
#include <limits>
#include<cstdlib>
#include<bits/stdc++.h>//for lowecse the usename so that we can sort them
using namespace std;
#define SZ 40
class Person
{
    protected:
        int Age;
        int hight;
        int weight;
    public:
            /*
            ****************************************************************
                        Set User's Age, Weight, And Height.                 *
            ****************************************************************
```

```
            */
        void SetPerson()
        {
            cout<<"Enter Your Age : ";
            cin>>Age;
            while (cin.fail()|| Age <=0)//cin.fail() is to insure misbehaviour
            {
                cin.clear();
                cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
                cout<<"Invalid input!"<<endl;
                cout<<"Enter Your Age : ";
                cin>>Age;
            }//looping until user inputs a correct or valid number✨

            cout<<"Enter Your Weight : ";
            cin>>weight;
            while (cin.fail()||weight <= 2 || weight >= 220)
            {
                cin.clear();
                cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
                cout<<"Not a Normal Weight Try Again"<<endl;
                cout<<"Enter Your Weight (kg) : ";
                cin>>weight;
            }//looping until user inputs a correct or valid number✨

            cout<<"Enter Your Hight : ";
            cin>>hight;
            while(cin.fail()||hight <=43 || hight >= 251)
            {
                cin.clear();
                cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
                cout<<hight<<" Is Invalid Height Try Again"<<endl;
                cout<<"Enter Your Hight(cm) : ";
                cin>>hight;
            }//looping until user inputs a correct or valid number✨
        }
        //getterts :)
        int getAge()const
        {
            return Age;
        }
        int getHeight()const
        {
            return hight;
        }
        int getWeight()const
        {
            return weight;
        }
};
//_____________________________________________________________________________
_____________________
class Activity:public Person //Now class is a child of class Person🔨
{
```

```
    protected:
        int act_type;
        double Calories;
        int Gender; //1 for men 👨 2 for women 👩 |
        double BMR;
        char alphabet[SZ]={"ABCDEFGHIJKLMNO"};//here a Random alphptic we'll use it for
generaing ids

    public:
        string Uniqueid;
        void setAct_type();
        double setBMR();
        double GetBMR()const;
        void GetPersonData();
        string generateUniqueID();
        double getCalories()const;
        string getGender();
};
/*
            ****************************************************************
                        Get user's gender as a string.      .                *
            ****************************************************************
*/
string Activity::getGender()
{
    switch(Gender)
    {
        case 1:
            return "Male";
            break;
        case 2:
            return "Female";
            break;
        default:
            return "Unknown";
    }
};
/*
            ****************************************************************
                        Display user's age, height, and weight.              *
            ****************************************************************
*/
void Activity::GetPersonData()
{
    cout<<"|Age             : "<<Age<<endl;
    cout<<"|Height          : "<<hight<<endl;
    cout<<"|Weight          : "<<weight<<endl;
}
/*
            ****************************************************************
                        Generate a unique ID for the user                    *
            ****************************************************************
*/
string Activity::generateUniqueID()//this is for Random creation Number
```

```
{
    static int counter = 10;//Id for first Registerer will start from 10
    char randomAlphabet = alphabet[rand() % (sizeof(alphabet) - 1)];// randomAlphabet
could now hold any Uppercase letter from A to O in char alphabet[SZ]={"ABCDEFGHIJKLMNO"};
    int randomNumber = counter++;
    return string(1, randomAlphabet) + to_string(randomNumber);
}
/*
            ****************************************************************
                        Set most of user info & set program logic.           *
            ****************************************************************
*/
double Activity::setBMR()
{
   while (true) {

        std::cout << "..Choose Your Gender.." << std::endl;
        std::cout << "1. Male\t\t2. Female" << std::endl;
        std::cout << "Enter Your Choice : ";

        std::cin.clear();  // to clear existing lines
        std::cin >> Gender;

        if (std::cin.fail() || (Gender != 1 && Gender != 2)) {
            std::cin.clear();  // clear the input buffer
            std::cin.ignore(INT_MAX, '\n');  // ignore any remaining characters in the
buffer
            std::cout << "Invalid input. Please enter 1 for Male or 2 for Female." <<
std::endl;
        } else {
            break;  // exit the loop if input is valid
        }
    }
    switch(Gender)
    {
        case 1:
            cout<<".........BMR Calculation........."<<endl;
            SetPerson();
            BMR = 10*weight+(6.25*hight)-(5*Age)+5;//for men
            break;
        case 2:
            cout<<".........BMR Calculation........."<<endl;
            SetPerson();
            BMR = 10*weight+(6.25*hight)-(5*Age)-161; //for women
            break;
        default:
            cout<<"Something Went Wrong):"<<endl;
            break;
    }
    cout<<endl;
    setAct_type();
    switch (act_type)
    {
    case 1:
```

```
        cin.clear();
        Activity::Uniqueid = string(1, alphabet[0]) + generateUniqueID();//if user chosed
act1 his ID will start from A
        return Calories = BMR*1.2; //act1
        break;
    case 2:
        cin.clear();
        Activity::Uniqueid = string(1, alphabet[2]) + generateUniqueID();//if user chosed
act2 his ID will start from C
        return Calories = BMR*1.375;//act2
        break;
    case 3:
        cin.clear();
        Activity::Uniqueid = string(1, alphabet[5]) + generateUniqueID();//if user chosed
act3 his ID will start from F
        return Calories = BMR*1.55;//act3
        break;
    case 4:
        cin.clear();
        Activity::Uniqueid = string(1, alphabet[10]) + generateUniqueID();//if user chosed
act4 his ID will start from K
        return Calories = BMR*1.725;//act4
        break;
    case 5:
        cin.clear();

        Activity::Uniqueid = string(1, alphabet[3]) + generateUniqueID();//same as other
        return Calories = BMR*1.9;//act5
        break;
    default:
        cout<<"InValid Choice Number to enter is b/w 1-6 "<<endl;
        return 0.0;
        break;
    }
};
/*
            ****************************************************************
                        Set the type of activity.                           *
            ****************************************************************
*/
void Activity::setAct_type()
{
    cout<<".............................which Activities You Do
............................."<<endl;
    puts("1.sedentary (little or no exercise)\n");
    Sleep(50);
    puts("2.lightly active (light exercise/sports 1-3 days/week)\n");
    Sleep(50);
    puts("3.moderately active (moderate exercise/sports 3-5 days/week)\n");
    Sleep(50);
    puts("4.very active (hard exercise/sports 6-7 days a week)\n");
    Sleep(50);
    puts("5.extra active (very hard exercise/sports & physical job or 2x training)\n");
    Sleep(50);
```

```
    puts("..........................................................................
.......");
    cout<<"Choose (1-5) : ";
    cin>>act_type;
    while(cin.fail()|| act_type <1 || act_type >5)
    {
        cin.clear();
        cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
        cout<<"Invalid choice!! . Choose (1-5) : ";
        cin>>act_type;
    }
};
double Activity::GetBMR()const
{
    return BMR;
};
double Activity::getCalories()const
{
    return Calories;
}
//_____________________________________________________________________________
_____________________
class FitnessTracker
{
    private:
        int user_type;
        string user_id;

    public:
        Activity activities;
        string Username;
        void setDtails();
        void Register();
        void GetDetails();
        char SetType();
        void generateReport();// :)
};
/*
            ***************************************************************
                        Set user type as admin or user.                    *
            ***************************************************************
*/
char FitnessTracker::SetType()
{
    cout<<".............Login Type............."<<endl;
    puts("1.administrator Account Type");
    puts("2.User Account Type");
    cout<<"Choose (1) Or (2) : ";
    cin.clear();
    cin>>user_type;
    while(user_type != 1 && user_type != 2&&cin.fail())//cin.fail() used  again misbehaive
like if you enter string where to enter a int✨
    {
        cin.clear();
```

```
        cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
        cout<<"     Error!!   "<<endl;
        cout<<"("<<user_type<<")"<<"Not in the List "<<endl;
        puts("1.administrator Account Type");
        puts("2.User Account Type");
        cout<<"Choose (1) Or (2) : ";
        cin>>user_type;
    }
    switch (user_type)//here is when we deciding b/w user or admin
    {
    case 1:
        return 'A';//A stands for admin
        break;
    case 2:
        return 'U';//U for user
        break;
    default:
        cout<<"Something Went Wrong!!"<<endl;
        return 'E';
        break;
    }
};
/*
            ***************************************************************
                        Set details such as BMR for the user.               *
            ***************************************************************
*/
void FitnessTracker::setDtails()
{
    activities.setBMR();
};

/*
            ***************************************************************
                      Register a user with a username.                      *
            ***************************************************************
*/
void FitnessTracker::Register()
{
    cout<<".............User Account Creation............."<<endl;
    cout<<"Enter UserName : ";
    cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
    getline(cin,Username);

    transform(Username.begin(), Username.end(), Username.begin(), ::tolower); //ref.
https://en.cppreference.com/w/cpp/string/wide/towlower
    //?sorting is alphapetic sensitive so we convert username to lowercase
};

/*
            ***************************************************************
                        Display user details.                               *
            ***************************************************************
*/
```

```
void FitnessTracker::GetDetails()
{
    cout<<".............User Data.............. "<<endl;
    cout<<"|Username        : @"<<Username<<""<<endl;
    cout<<"|User Id          : "<<activities.Uniqueid<<""<<endl;
    cout<<"|User Gender      : "<<activities.getGender()<<""<<endl;
    activities.GetPersonData();
    cout<<"|User BMR         : "<<activities.GetBMR()<<""<<endl;
    cout<<"|Calories Burned : "<<activities.getCalories()<<""<<endl;
    cout<<"|_____________________________________________"<<endl;
};
/*
            ***************************************************************
                        generate a report and save to "Report.txt".          *
            ***************************************************************
*/
void FitnessTracker::generateReport()
{
    fstream myFile;
    myFile.open("Report.txt", ios::app);
    if(myFile.is_open())
    {
        myFile <<">>>>>>>>>>>>>>>>>>>>>>.....<<<<<<<<<<<<<<<<<<<<<<\n";
        myFile << "Username: @" << Username<<"\n";
        myFile << "User ID: " <<  activities.Uniqueid<<"\n";
        myFile << "Age: " <<activities.getAge()<<"\n";
        myFile << "Height: " <<activities.getHeight()<<"\n";
        myFile << "Weight: " <<activities.getWeight()<<"\n";
        myFile << "User BMR: "<<activities.GetBMR()<<"\n";
        myFile << "Calories: "<<activities.getCalories()<<"\n";
        myFile <<".................................................\n";
    }
}
```

Link list Source Code:

```
#include "FTS.hpp"//myclases
#include <Windows.h>
#include  <list>
#include <algorithm>
#include "animation.hpp"
/*
    *methods needed in this **
    //Search()
    //Display()
    //Update()
    //Delete()
    //Add()
```

```
    generateReport()
*/
class FtsList
{
    private:
        struct FtsRecord
        {
            FitnessTracker trckerapp;
            FtsRecord *Nextptr;
        };
        FtsRecord *head;
    public:
        void insert(FitnessTracker newtrckerapp);//LIFO Type
        void Delete(string Targetusr);
        void Search(string usr);
        void Update(string TrgtUsr);
        void sort();
        void Display();
        void generateReport();
        int Isempty();
        void updtebar();
FtsList()
{
    head = nullptr; //instrcutor first runs when the main app run so it automticly first
head to Null:)
}
};
int FtsList::Isempty()
{
    if(head==nullptr)
    {
        return 0;
    }else
    {
        return 1;
    }
};
void FtsList::sort()//reff= https://youtu.be/IgLc2z_6qPg also we  used sort and algorithum
librr ✨
{
    if(Isempty()==0)
    {
        cout<<"There Is No Data Yet.."<<endl;
        return;//nothing.. 🗐
    }
    list<FitnessTracker> FTRecords; //so It's a container that can hold instances of
FitnessTracker
    FtsRecord *Temp = head;
    while (Temp!=nullptr)
    {
        FTRecords.push_back(Temp->trckerapp);//it Copies each FitnessTracker object from
the linked list nodes to the FTRecords list
        Temp = Temp->Nextptr;
    }
```

```
  FTRecords.sort([](const FitnessTracker &a, const FitnessTracker &b) {
        return a.Username < b.Username;
    });
    for (auto& record : FTRecords)//using a range-based for loop
    {
        record.GetDetails();
    }
};
void FtsList::insert(FitnessTracker newtrckerapp)
{
    FtsRecord *temp = head;
    //!Here we are checking if username already in the nodes
    while (temp != nullptr)
    {
        if (temp->trckerapp.Username==newtrckerapp.Username)
        {
            cout << "Username '" << newtrckerapp.Username << "' already exists. Cannot add
duplicate user." << endl;
            return;
        }
        temp = temp->Nextptr;
    }
    FtsRecord *newNodeptr = new FtsRecord;
    if (newNodeptr ==nullptr)
    {
        cout<<"Error No More Memory Avaible in The Computer ):"<<endl;
    }else
    {
        newNodeptr->trckerapp = newtrckerapp;
        newNodeptr->Nextptr = head;
        head = newNodeptr;
    }

}
void FtsList::Delete(string Targetusr)
{
    if(Isempty()==0)
    {
        cout<<"There Is No Data Yet.."<<endl;

    }
    FtsRecord *currnt_record = head;
    FtsRecord *previous_record = nullptr;
    // so here is Traverse the list to find the node to delete
    while (currnt_record != nullptr && currnt_record->trckerapp.Username != Targetusr)
    {
        previous_record = currnt_record;
        currnt_record = currnt_record->Nextptr;
    }
    if(currnt_record != nullptr)
    {
        if(previous_record==nullptr)
        {
            head = currnt_record->Nextptr;
```

```
        }else
        {
            previous_record->Nextptr = currnt_record->Nextptr;
        }
        delete currnt_record;
        deleteBar();
        cout<<"User "<< Targetusr<<" deleted successfully.."<<endl;

    }else
    {
        cout<<"User "<<Targetusr<<" not found."<<endl;
    }
};
 void FtsList::Update(string TrgtUsr)
 {
    if(Isempty()==10)
    {
        cout<<"Nothing To Update Here"<<endl;

    }
    FtsRecord *Curr = head;
    while(Curr != nullptr&&Curr->trckerapp.Username!=TrgtUsr)
    {
        Curr= Curr->Nextptr;

    }
    if(Curr!=nullptr)
    {
        cout<<"Displaying Existing Data For "<<TrgtUsr<<" :"<<endl;
        Curr->trckerapp.GetDetails();

        //time to update
        cout<<".....................Update Field....................."<<endl;
        Curr->trckerapp.setDtails();
        updtebar();
        cout << "User " << TrgtUsr << " updated successfully." << endl;

    }else
    {
        cout<<"User with Name "<<TrgtUsr<< " Didn't Found.."<<endl;
    }
 };
void FtsList::generateReport()
{
    FtsRecord *temp = head;
    if (temp==nullptr)
    {
        cout<<"Nothing To Generate Here"<<endl;

    }else
    {
        while(temp!=nullptr)
        {
            temp->trckerapp.generateReport();
```

```
            temp = temp->Nextptr;
        }
         ReportgenBar();
    }
};
void FtsList::Display()
{
    FtsRecord *Temp = head;
    if(Temp==nullptr)
    {
        cout<<"Nothing To Generate Here"<<endl;
    }else
    {
        while (Temp != nullptr)
        {
            Temp->trckerapp.GetDetails();
            Temp=Temp->Nextptr;
        }
    }
}
void FtsList::Search(string usr)
{
    FtsRecord *Temp = head;
    if(Temp==nullptr)
    {
        cout<<"No Data Entered Yet!"<<endl;
    }else
    {
        while (Temp!=nullptr&&Temp->trckerapp.Username != usr)
        {
            Temp = Temp->Nextptr;
        }
        if(Temp != nullptr)
        {
            searchbar();
            cout<<"------------- UserName :"<<usr <<"found successfully-------------
"<<endl;
            Temp->trckerapp.GetDetails();
            cout<<"................................................................."<
<endl;
        }else
        {
            cout<<"\t\t!"<<endl;
            cout<<"UseName "<<usr<<" Not Found"<<endl;
        }
    }
};
void FtsList::updtebar()
{
    system("COLOR 0e");
    system("cls");
    SetConsoleCP(437);
    SetConsoleOutputCP(437);
    int bar1, bar2;
```

```
    bar1 = 177;
    bar2 = 219;
    cout << "\n\n\n\n\n\n\n\n\t\t\t\t\tUpdating...";
    cout << "\n\n\n\n\n\t\t\t\t\t";
    for (int i = 0; i < 25; i++)
        cout << (char)bar1;
    cout << "\r";
    cout << "\t\t\t\t\t";
    for (int i = 0; i < 25; i++) {
        cout << (char)bar2;
        Sleep(50);
    }
    system("cls");
};
```

Main Driver Source Code:

```
#include <fstream>
#include <Windows.h>
//#include "animation.hpp"
#include "Linklist(FTS).hpp"
#include <ctime>
using namespace std;
int main()
{
    srand(static_cast<unsigned int>(time(nullptr)));//to use window time to generte
randomatic number/Alphapets
    FtsList FitnesList;
    FitnessTracker FTStracker;
 int ch;
 string Targetuser;
    puts("\n\t\t\tFITNESS\n\n\t\t\tTRACKER\n\n\t\t\tPROJECT\n");
    puts("\nMADE BY : GR14");
    Sleep(1900);
    system("CLS");
    cout << "\n\t\t1. START THE APP\n\n\t\t2. EXIT THE APP\n" << endl;
    cout << "ENTER (1) OR (2): ";
    cin.clear();
    cin >> ch;
    while(cin.fail()||ch<1||ch > 2)
    {
        cin.clear();
        cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
        cout << "Invalid input! Enter (1) or (2): ";
        cin >> ch;
    }
    progressbar();
    while (ch == 1) {

        switch (FTStracker.SetType()) {
            case 'A':
                int adminChoice;
```

```
                do {
                    cout<<"\n"<<endl;
                    system("COLOR d0");
                    cout<<".............................Menu.....................
..............."<<endl;
                    cout << "1. Set Your Information \t 2. Delete Existing User info" <<
endl;
                    cout << "3. Update Existing User info \t 4. Generate text Base for All
user info" << endl;
                    cout << "5. Display All Users Info \t 6.Search Existing User info" <<
endl;
                    cout << "7. Exit"<<endl;
                    puts("...................................................................
..............");
                    cout << "Enter (1-7): ";
                    cin.clear();
                    cin >> adminChoice;

                    while(cin.fail()||adminChoice <1 || adminChoice > 7)
                    {
                        cin.clear();
                        cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
                        cout << "Invalid choice. Enter (1-7): ";
                        cin >> adminChoice;
                    }

                    switch (adminChoice)
                    {
                        case 1:
                            FTStracker.Register();
                            FTStracker.setDtails();
                            cin.clear();
                            cout<<"Your Auto Generted ID :
"<<FTStracker.activities.Uniqueid<<endl;
                            FitnesList.insert(FTStracker);
                            break;
                        case 2:
                            cout<<"Enter The UserName To delete: ";
                            std::cin.ignore(std::numeric_limits<std::streamsize>::max(),
'\n');
                            getline(cin,Targetuser);
                            FitnesList.Delete(Targetuser);
                            break;
                        case 3:
                            cout<<"Enter The UserName To Update: ";
                            std::cin.ignore(std::numeric_limits<std::streamsize>::max(),
'\n');
                            getline(cin,Targetuser);
                            FitnesList.Update(Targetuser);
                            break;
                        case 4:
                            FitnesList.generateReport();
                            break;
                        case 5:
```

```
                        FitnesList.sort();
                        break;
                    case 6:
                        cout<<"Enter The UserName To Search: ";
                        std::cin.ignore(std::numeric_limits<std::streamsize>::max(),
'\n');
                        getline(cin,Targetuser);
                        FitnesList.Search(Targetuser);
                        break;
                    case 7:
                        system("cls");
                        break;
                    default:
                        cout << "Invalid choice. Try again." << endl;
                        break;
                }
            } while (adminChoice != 7);
            break;
        case 'U':
            int user_Choice;

            do
            {
                system("Color b0");
                puts("\n\n");
                puts("..........................................................
.............");
                cout << "\t\t1. BMR & Calories Calculation \t\t2.Exit" << endl;
                cout << "\nchoose (1)or(2) : ";
                cin.clear();
                cin>>user_Choice;
                switch (user_Choice)
                {
                case 1:
                    FTStracker.setDtails();
                    puts(".................................Result.................
.................");
                    puts("");
                    cout<<"Your BMR : "<<FTStracker.activities.GetBMR()<<endl;
                    cout<<"Your Calories :
"<<FTStracker.activities.getCalories()<<endl;
                    puts("..........................................................
.................");
                    break;
                case 2:
                    system("cls");
                    break;
                default:
                    break;
                }
            } while (user_Choice != 2);

        default:
            break;
```

```
        }

        cout << "\n\t\t1. START THE APP\n\n\t\t2. EXIT THE APP\n" << endl;
        cout << "ENTER (1) OR (2): ";
        cin >> ch;
        while(cin.fail()||ch<1||ch > 2)
        {
            cin.clear();
            cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
            cout << "Invalid input! Enter (1) or (2): ";
            cin >> ch;

        }
        progressbar();
    }
    return 0;
}
```