# SRM GYM MANAGEMENT SYSTEM

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

**Bachelor of Technology**

In

**Computer Science and Engineering**

**School of Engineering and Sciences**

Submitted by

**Sharini Deepthi-AP21110011425**

**Mohammad Naaguru-AP21110011452**

**Narasimha-AP21110011478**

**Srinath-AP21110011437**

**Ahammad-AP21110011465**

****

Under the Guidance of

**Dr.Baskhar Santhosh**

**SRM University–AP**

**Neerukonda, Mangalagiri, Guntur**

**Andhra Pradesh – 522 240 [December, 2022]**

**Department of Computer Science and Engineering**

**SRM Unuversity,AP**

****

**CERTIFICATE**

This is to certify that the work present in this Project entitled “**SRM GYM Management System**” has been carried out by **Srinath, Narasimha, Ahammad, Sharini Deepthi, Mohammad Naaguru** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology (Computer Science and Engineering) in **School of Engineering and Sciences**.

**Signature of Supervisor Signature of Head of the Department**

Dr.Baskhar Santhosh Dr. Jatindra Kumar Dash

**Acknowledgments**

The pleasure that follows the successful completion of this project would remain incomplete without a word of gratitude for the people and without whose cooperation this project would remain incomplete. It is not a mere formality to place a record of the tireless efforts, ceaseless cooperation, constant guidance, and encouragement of the people closely associated with the assignment but a distinct necessity for the ethnicity and readability of the project.

We are so thankful to our instructor Mr.Baskhar Santhosh, for his scholarly guidance, advice, and encouragement, and also for providing the necessary facilities to carry out the dissertion in the prescribed period.

# Table of Contents

# 1.Abstract----------------------------------------------

# 2. Introduction-----------------------------------------

# 3. Methodology----------------------------------------

# 3.1-Main concept of the program--------

# 3.2-Outputs--------------------------------------4.Conclusion-------------------------------------------5.References--------------------------------------------Abstract

SRM GYM Management System is designed to facilitate a gym and fitness centre to automate its operations of keeping records and store gym management system project in form of a large and user-friendly database for gym management system project facilitating easy access to gym management system project personnel. This System is an online service that can be setup for your gym to help manage classes, memberships, keep track with detailed statistics, and customer management system.

This system is made in order to effectively and efficiently cater to requirements of gym management system centre. A person who generally holds gym management system tasks to manage gym centre needs to keep records of all transactions as well as data manually. Generally, in order to structure gym management system tasks, separate registers are maintained. This whole process thus becomes quite cumbersome for the gym management system to be controlled manually. Moreover, any wrong data entered mistakenly can bring serious results.

Gym management system project software is capable enough to allow the gym management system project concerned person to store and retrieve any type of record with just a single click of the mouse.

# Introduction

SRM Gym Management System project’s key is solving gym management system problems of information management. In general, a server must reliably manage a large amount of data in a multi-user environment so that many users can concurrently access gym management system data. A database server must also prevent unauthorized access and provide efficient solutions for failure recovery.

Generally, in order to structure gym management system tasks, separate registers are maintained. This whole process thus becomes quite cumbersome for the gym management system to be controlled manually. Moreover, any wrong data entered mistakenly can bring serious results. This manually managed gym management system is also heavily prone to data loss due to certain causes of misplacement of registers, destruction of registers, unauthorized access of registers etc which may lead to dangerous consequences. Cost of maintaining of data and records is also very high. Data redundancy is also a great issue in this system.

* Gym management system project software is capable enough to allow the gym management system project concerned person to store and retrieve any type of record with just a single click of the mouse.
* With the help of object-oriented programming, we are able to hide the data from normal users and made it accessible only to the admin, thus preventing unauthorized access to data.
* The Password system made it even more secure and safe.
* No need to manage bulky registers now, as the data is stored safely.
* Reduces the dependence on Man-Power.
* Data redundancy is now not a problem as the data modified from one data entry form will reflect on other forms too.

# Methodology

* We have created separate logins for the users, trainers and the gym managers i.e admin, in which the gym manager is password protected.
* In this project, the gym manager can add, update, delete and create gym memberships and can assign them to a particular user.
* The user can quit the gym anytime, whenever he or she wanted.
* The software gives facility for the users to edit their profiles.
* Only the admin can delete and edit the membership.
* During the purchase of membership, the trainer is allotted automatically.
* This project uses the concept of file handling.

## ………….The main concept of thisprogram…………

* Login system
* Registration activities of the member
  + Join a Gym
  + Quit Gym
  + Edit User Profile
  + Create Membership
  + Display all records
  + Search particular Record
  + Edit Membership Details
  + Delete Membership
* When you run the project from compiler the menu of modes is displayed. If you’re a user press 1 , gym manager or admin press 2 or trainer press 3.
* For User three options will be displayed- 1. Join Gym 2. Quit Gym 3. Edit your profile.
  + By choosing 1 the user will be able to join the gym after providing some valid details.
  + By choosing 2 the user will be able to quit the gym upon providing the member number.
  + By choosing 3 the user will be able to modify the details about the existing member.
* The admin will require to sign in using the admincode: “admin” and the password “fitgym”. After giving the correct code and password a menu will be displayed.
  + By choosing 1, the admin is able to create a member.
  + By choosing 2, all the information will be displayed.
  + By choosing 3, a particular record can be searched.
  + By choosing 4, the details of the members can be modified.
  + By choosing 5, the admin is able to delete a member.
  + By choosing 6, go back to the main menu.
* For Trainer won’t require any login credentials, it just needed the trainer number and it will display the class which they are allotted and Names of members to whose he will going to give training.

## CODE

**Main function:**

int main()

{

    int i, k;

    string name, code;

    cout << " \*\*\*\*\*\*\*\*\*\*\*\* SRM GYM MANAGEMENT SYSTEM\*\*\*\*\*\*\*\*\*\*\*\* " << endl;

    cout<<"\n";

    cout << endl

         << "SELECT MODE" << endl

         << "1.User" << endl

         << "2.Admin" << endl

         << "3.Trainer" << endl;

    cin >> i;

    if (i == 1)

    {

        cout << endl

             << "you're in user mode" << endl;

        fnuser();

    }

    if (i == 2)

    {

        cout << endl

             << "you're in admin mode" << endl;

        alogin ad;

        k = ad.adlogin();

        if (k == 1)

        {

            fnmanage();

        }

        else

        {

            cout << "you cannot access manageral details!";

        }

    }

    if (i == 3)

    {

        cout << endl

             << "you're in trainers mode" << endl;

        fntrainer();

    }

    return 0;

}

**User:**

**Creating member class for user details:**

**User can select LOGIN, QUIT, SIGNUP:**

void fnuser()

{

    for (;;)

    {

        int m;

        cout << "1.SIGN UP(join gym)\n2.QUIT GYM\n3.LOG IN(edit profile)\n";

        cout << endl

             << "enter your choice" << endl;

        cin >> m;

        switch (m)

        {

        case 1:

            system("cls");

            save\_member();

            break;

        case 2:

            delete\_member();

            break;

        case 3:

            edit\_member();

            break;

        }

    }

}

**Function for creating new user:**

void create\_mem()

    {

        int k, l, j;

        cout << endl

             << "Please Enter The member Number: ";

        cin >> member\_number;

        cout << endl

             << "Please Enter The Name of The member: ";

        getchar();

        cin.getline(mem\_name, 50);

        cout << endl

             << "Please Enter The contact number: ";

        cin >> contact;

        cout << "1.gold class\n2.silver class\n";

        cout << "enter the choice\n";

        cin >> k;

        ofstream t1;

        if(k==1)

        {

        t1.open("trainer1.txt",ios::app);

        t1<<mem\_name<<endl;

        }

        ofstream t2;

        if (k==2){

            t2.open("trainer2.txt",ios::app);

            t2<<mem\_name<<endl;

        }

        switch (k)

        {

        case 1:

        {

            strcpy(classs, "gold");

            fee = 2500;

            cout << "your monthly fee would be: " << fee << endl;

            l = trainee\_allotment(classs);

        }

        break;

        case 2:

        {

            strcpy(classs, "silver");

            fee = 2000;

            cout << "your monthly fee would be: " << fee << endl;

            l = trainee\_allotment(classs);

        }

        break;

        }

        time\_slots();

    }

**Creating member class:**

**Functions to display and save the information of the members:**

class member

{

    int member\_number, j;

    char mem\_name[50], classs[50], timings[50], mem\_phonenum[11];

    float fee;

    long long int contact;

public:

    int time\_slots()

    {

        int k;

        cout << "please select your preferred timings\n press 1 for: morining 6-7\n press 2 for: morning 7-8\npress 3 for:morning 8-9\n";

        cout << " press 4 for: evening 4-5\npress 5 for:evening 5-6\npress 6 for:evening 6-7\n";

        cin >> k;

        switch (k)

        {

        case 1:

            strcpy(timings, "6AM-7AM");

            break;

        case 2:

            strcpy(timings, "7AM-8AM");

            break;

        case 4:

            strcpy(timings, "4PM-5PM");

            break;

        case 5:

            strcpy(timings, "5PM-6PM");

            break;

        case 6:

            strcpy(timings, "6PM-7PM");

            break;

        }

        return 0;

    }

    void create\_mem()

    {

        int k, l, j;

        cout << endl

             << "Please Enter The member Number: ";

        cin >> member\_number;

        cout << endl

             << "Please Enter The Name of The member: ";

        getchar();

        cin.getline(mem\_name, 50);

        cout << endl

             << "Please Enter The contact number: ";

        cin >> contact;

        cout << "1.gold class\n 2.silver class\n";

        cout << "enter the choice\n";

        cin >> k;

        ofstream t1;

        if(k==1)

        {

        t1.open("trainer1.txt",ios::app);

        t1<<mem\_name<<endl;

        }

        ofstream t2;

        if (k==2){

            t2.open("trainer2.txt",ios::app);

            t2<<mem\_name<<endl;

        }

        switch (k)

        {

        case 1:

        {

            strcpy(classs, "gold");

            fee = 2500;

            cout << "your monthly fee would be: " << fee << endl;

            l = trainee\_allotment(classs);

        }

        break;

        case 2:

        {

            strcpy(classs, "silver");

            fee = 2000;

            cout << "your monthly fee would be: " << fee << endl;

            l = trainee\_allotment(classs);

        }

        break;

        }

        time\_slots();

    }

    void show\_mem()

    {

        cout << endl

             << "member code: " << member\_number;

        cout << endl

             << "Name: " << mem\_name;

        cout << endl

             << "category: " << classs;

        cout << endl

             << "fee: " << fee;

        cout << endl

             << "contact: " << contact;

        cout << endl

             << "timings: " << timings << endl;

    }

    int getmem()

    {

        return member\_number;

    }

    float getfee()

    {

        return fee;

    }

    char \*getName()

    {

        return mem\_name;

    }

    long long int getcontact()

    {

        return contact;

    }

};

**Allotment of membership:**

void allotment\_gold(int gc, int sc)

{

    if (gc != 0)

    {

        cout << "trainer" << gc << " alloted\n";

    }

    else

    {

        cout << "\nno trainer available\n";

    }

}

void allotment\_silver(int gc, int sc)

{

    if (sc != 0)

    {

        cout << "trainer" << sc << " alloted\n";

    }

    else

    {

        cout << "\nno trainer available\n";

        if (gc != 0)

        {

            cout << "enter  another class\n";

            allotment\_gold(gc, sc);

        }

    }

}

**Allotment of Trainer:**

int trainee\_allotment(char \*c)

{

    char class\_type[10];

    int i, gold\_class = 2, silver\_class = 3;

    strcpy(class\_type, c);

    if ((gold\_class != 0) || (silver\_class) != 0)

    {

        if (strcmp(class\_type, "gold") == 0)

        {

            gold\_class--;

            allotment\_gold(gold\_class, silver\_class);

        }

        else if (strcmp(class\_type, "silver") == 0)

        {

            silver\_class--;

            allotment\_silver(gold\_class, silver\_class);

        }

    }

    else

    {

        cout << "no trainer availbale\n";

        return 1;

    }

    return 0;

}

**Admin:**

**ADMIN login with password:**

class alogin

{

    string admin;

public:

    int adlogin()

    {

        cout << "enter the admin code" << endl;

        cin >> admin;

        if (admin == "admin")

        {

            int p = getadpassword();

            if (p == 1)

            {

                cout << "LOGIN Succesful" << endl;

                return 1;

            }

        }

        else

        {

            cout << "user name not found" << endl;

            adlogin();

            cout << "LOGIN Failed" << endl;

            return 0;

        }

        return 0;

    }

    int getadpassword();

};

int alogin::getadpassword()

{

    char pin[100];

    int k=0;

    cout << "enter the password" << endl;

    while(pin[k-1]!='\r') {

        pin[k]=getch();

        if(pin[k-1]!='\r') {

            cout<<"\*";

        }

        k++;

    }

    pin[k-1]='\0';

    string s=pin;

    if (s =="fitgym")

    {

        cout<<"\n";

        cout<<endl<< "WELCOME!!" <<endl;

        return 1;

    }

    else

    {

        cout << "entered password is incorrect" << endl;

        getadpassword();

        return 0;

    }

}

**Gym managers can select add, update, delete, search for particular record, display all records**.

void fnmanage()

{

    for (;;)

    {

        system("cls");

        int option;

        cout << "\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

        cout << "\n\tPress 1 to CREATE MEMBER";

        cout << "\n\tPress 2 to DISPLAY ALL RECORDS";

        cout << "\n\tPress 3 to SEARCH FOR A PARTICULAR RECORD ";

        cout << "\n\tPress 4 to EDIT MEMBER DETAILS";

        cout << "\n\tPress 5 to DELETE MEMBER";

        cout << "\n\tPress 6 to GO BACK TO MAIN MENU";

        cout << "\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

        cout << "\n\n\tOption: ";

        cin >> option;

        switch (option)

        {

        case 1:

            system("cls");

            save\_member();

            break;

        case 2:

            show\_all();

            break;

        case 3:

            int num;

            system("cls");

            cout << "\n\n\tPlease Enter The member Number: ";

            cin >> num;

            display\_record(num);

            break;

        case 4:

            edit\_member();

            break;

        case 5:

            delete\_member();

            break;

        case 6:

            system("cls");

            break;

        default:

            fnmanage();

        }

    }

}

**Create member:**

fstream fp;

member m1;

void save\_member()

{

    fp.open("list\_of\_members.txt", ios::app);

    m1.create\_mem();

    fp.write((char \*)&m1, sizeof(m1));

    fp.close();

    cout << endl

         << endl

         << "the member has been succesfully added ";

    getchar();

}

**Display all records:**

void show\_all()

{

    system("cls");

    cout << endl

         << "\t\tRECORDS...";

    fp.open("list\_of\_members.txt", ios::in);

    while (fp.read((char \*)&m1, sizeof(m1)))

    {

        m1.show\_mem();

        getchar();

    }

    fp.close();

}

**Search for particular records:**

void display\_record(int num)

{

    bool found = false;

    fp.open("list\_of\_members.txt", ios::in);

    while (fp.read((char \*)&m1, sizeof(m1)))

    {

        if (m1.getmem() == num)

        {

            system("cls");

            m1.show\_mem();

            found = true;

        }

    }

    fp.close();

    if (found == true)

        cout << "\n\nNo record found";

    getchar();

}

**Edit member details:**

void edit\_member()

{

    int num;

    bool found = false;

    // system("cls");

    cout << endl

         << endl

         << "\tPlease Enter The member number: ";

    cin >> num;

    fp.open("list\_of\_members.txt", ios::in | ios::out);

    while (fp.read((char \*)&m1, sizeof(m1)) && found == false)

    {

        if (m1.getmem() == num)

        {

            m1.show\_mem();

            cout << "\nPlease Enter The New details of the member: " << endl;

            m1.create\_mem();

            int pos = 0;

            fp.seekp(pos, ios::cur);

            fp.write((char \*)&m1, sizeof(m1));

            cout << endl

                 << endl

                 << "\t Record Successfully Updated...";

            found = true;

        }

    }

    fp.close();

    if (found == false)

        cout << endl

             << endl

             << "Record Not Found...";

    getchar();

}

**Delete member:**

void delete\_member()

{

    int num;

    system("cls");

    cout << endl

         << endl

         << "Please Enter The member number: ";

    cin >> num;

    fp.open("list\_of\_members.txt", ios::in | ios::out);

    fstream fp2;

    fp2.open("Temp.txt", ios::out);

    fp.seekg(0, ios::beg);

    while (fp.read((char \*)&m1, sizeof(m1)))

    {

        if (m1.getmem() != num)

        {

            fp2.write((char \*)&m1, sizeof(m1));

        }

    }

    fp2.close();

    fp.close();

    remove("list\_of\_members.txt");

    rename("Temp.txt", "list\_of\_members.txt");

    cout << endl

         << endl

         << "\tRecord Deleted...";

    getchar();

}

**Trainer:**

void fntrainer()

{

    ifstream t1;

    ifstream t2;

    for (;;)

    {

        int tom;

        cout << endl<<"1.trainer1\n2.trainer2\n";

        cout << endl

             << "enter your trainer number" << endl;

        cin >> tom;

        switch(tom)

        {

        case 1: cout<<"You are alloted to the ones who choosen gold membership"<<endl;

                   t1.open("trainer1.txt",ios::in);

                   if(t1.is\_open()){

                string tr;

                while(getline(t1,tr)){

                    cout<<tr<<"\n";

                }

            }

                  // cout<<mem\_name;

                   break;

        case 2: cout<<"You are alloted to the ones who has chosen silver membership"<<endl;

            t2.open("trainer2.txt",ios::in);

            if(t2.is\_open()){

                string tr2;

                while(getline(t2,tr2)){

                    cout<<tr2<<"\n";

                }

            }

            //cout<<mem\_name;

            break;

            t1.close();

            t2.close();

        }

    }

}

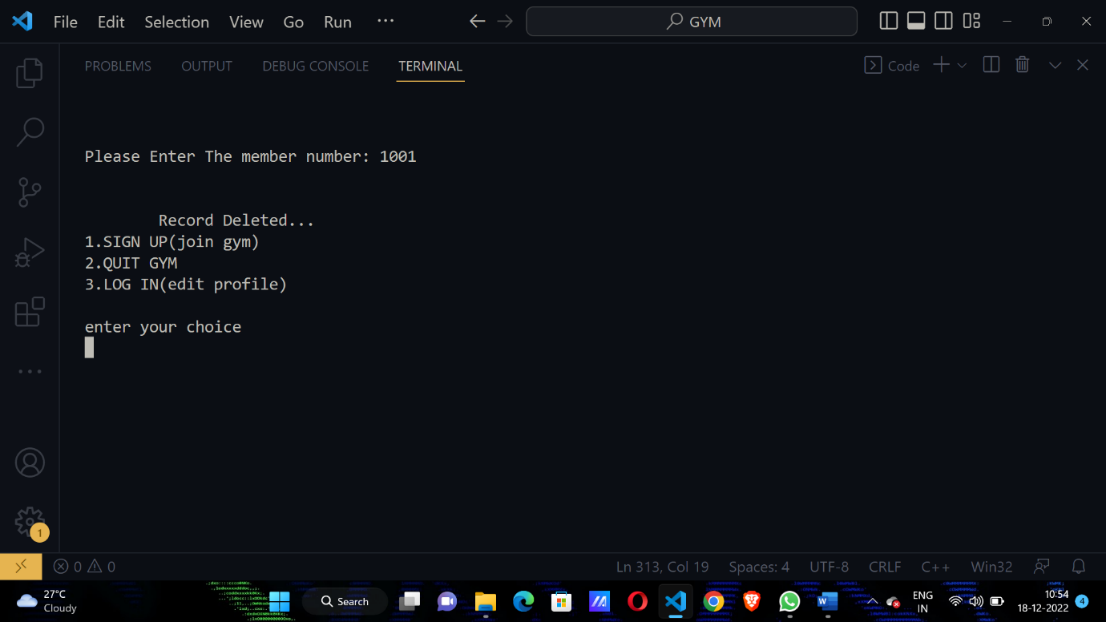
### OUTPUT

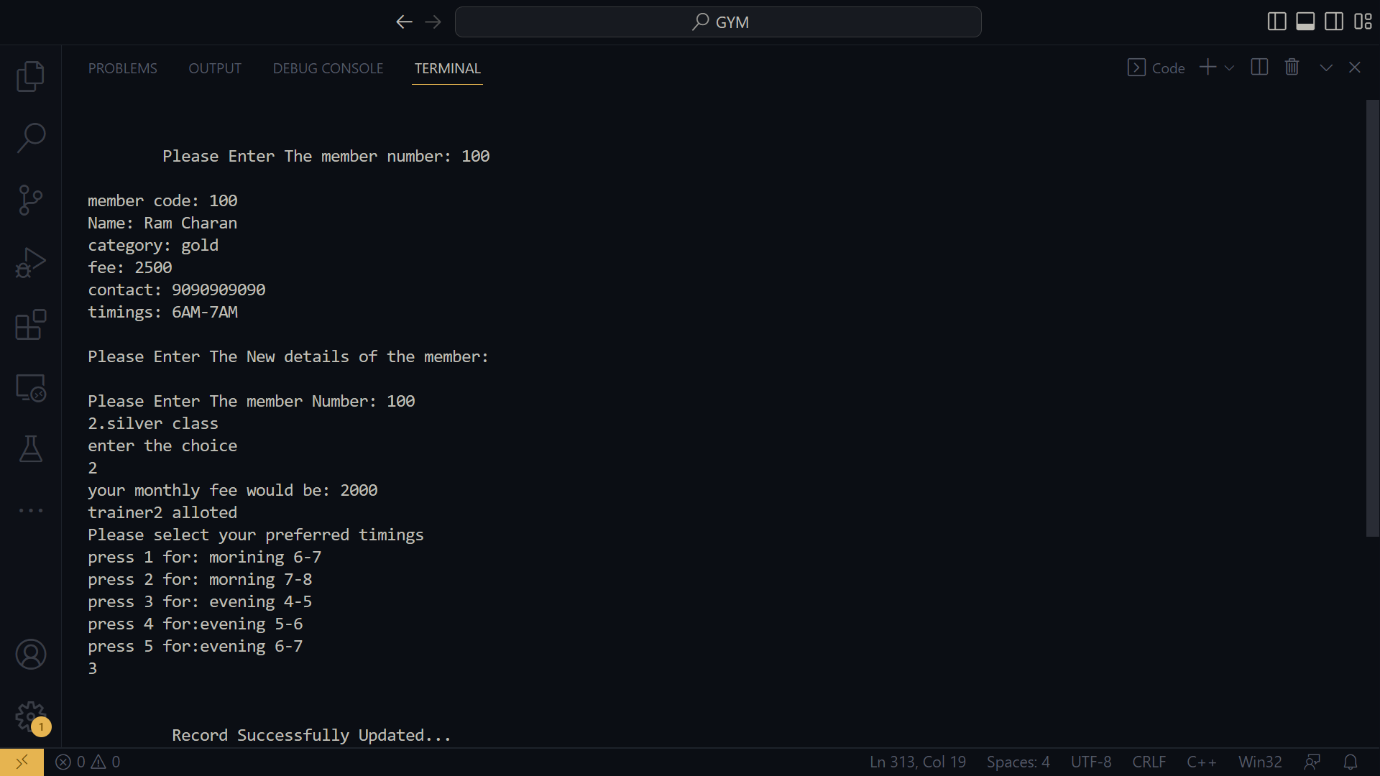


USER Mode:

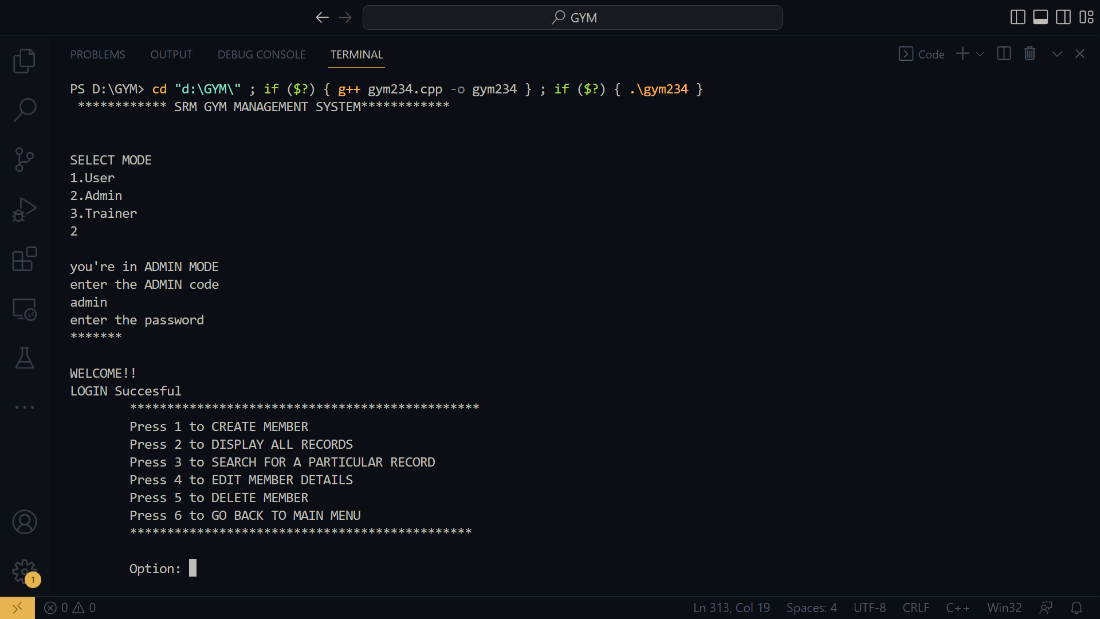


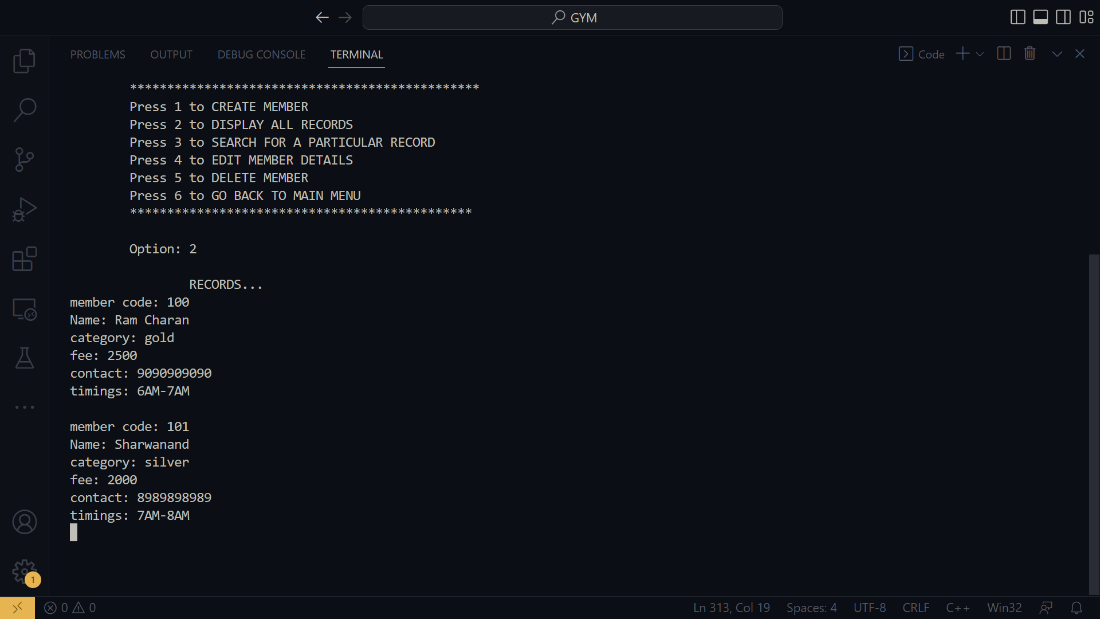


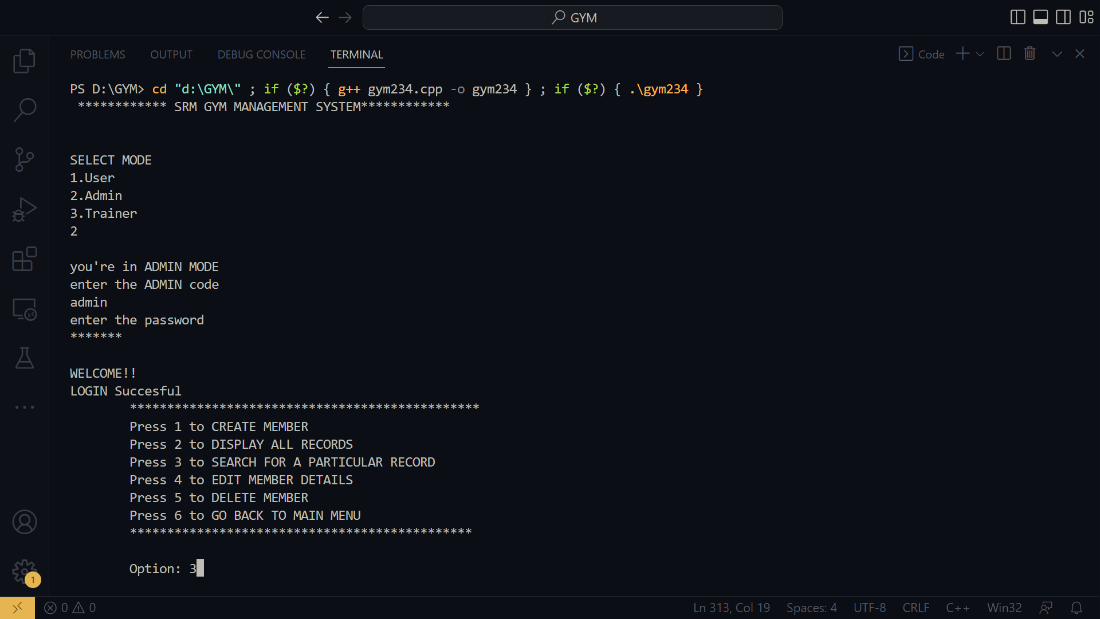


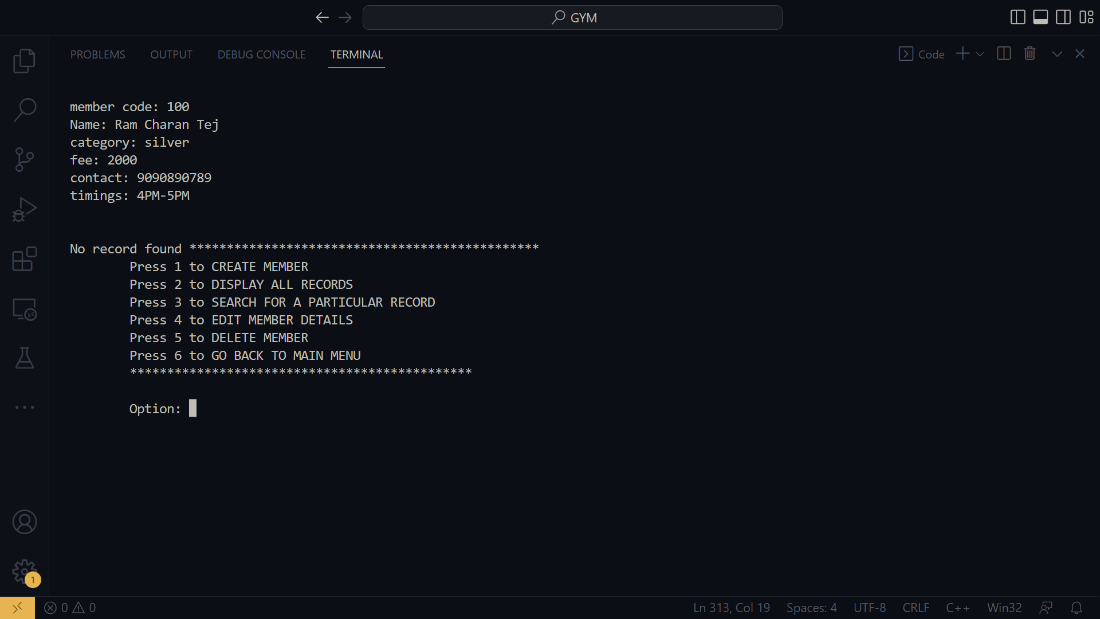


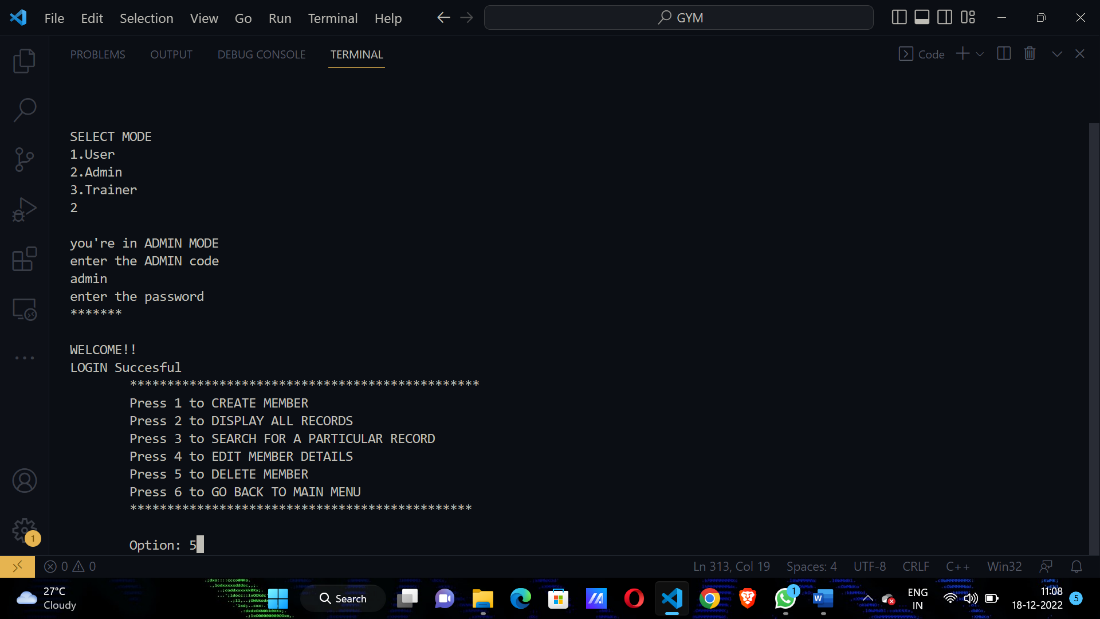
ADMIN Mode:

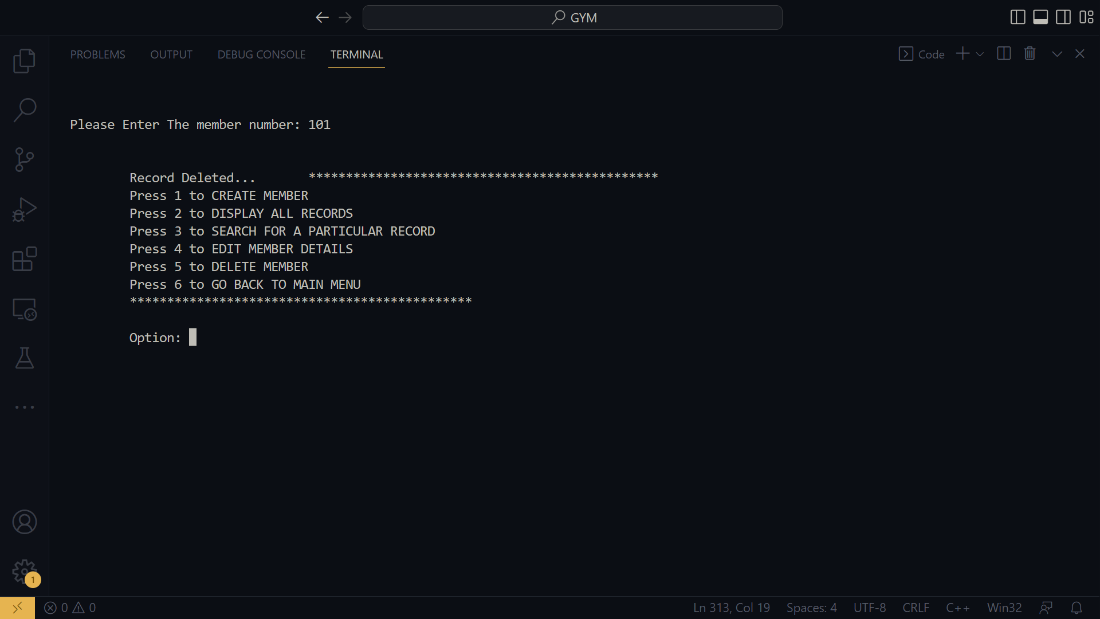




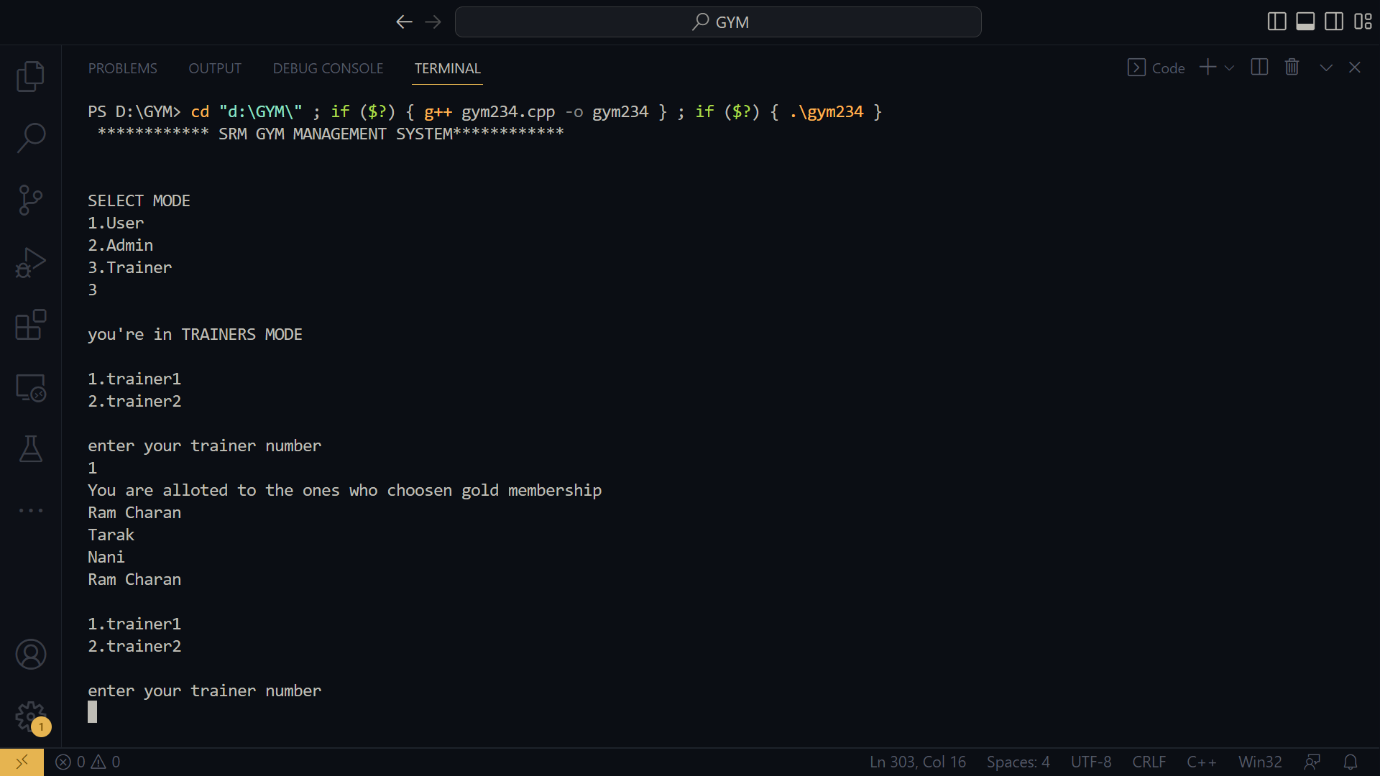








TRAINER Mode:



# Conclusion

The main moto of this project is to digitalize the data logging process, initialization of this project is done by c++ due to versatility and liability, The chance of integrating it with many classes.

By this project we understood the usage of file handling and classes usage in c++. We have created the file and given access in different modes like read only file and write only file to the main file.

Through this we understood the OOPS concept and how to use it in appropriate way.

# References

**Books referred:**

**1.** The C++ programming Language(4th Edition) By Bjarne Stroustrup.

**Websites referred:**

**1**.<https://www.geeksforgeeks.org/c-plus-plus/>

**2.**[W3Schools Online Web Tutorials](https://www.w3schools.com/)

**3.**[C++ Tutorial (tutorialspoint.com)](https://www.tutorialspoint.com/cplusplus/index.htm)