Project 1

Object Oriented Programming (OOP) Using (C++)

CONTACT MANAGER APPLICATION

Programmed by:

- ***Sherif Muhammad Muhammad .***
- ***Group : A2***
- ***Section: 13***

Under Supervisor:

- ***Dr. Mahmoud A. Mahdi***

Faculty Of Computer Science And Artificial Intelligence ZagaZig University (FCIZU).

Application Description:

This is a brief description of the application. A contact management application tracks user contact information. It is a database for the communications you make daily, including names, addresses, emails, and phone numbers. You can use a contact manager to easily organize and find all the information linked to your users.

To construct application using (C++ OOP), it was divided to subproblems (Classes) to create it alone such as Phone, Email And address for each user then connecting them to User Other information like name, id …etc. And the Contact Problem which has the ability to deal with Person Who want to use the application and save his data. The contact class contain the main feathers of system like adding, editing, search, deleting or show all users saved data.

Data Structures Used:

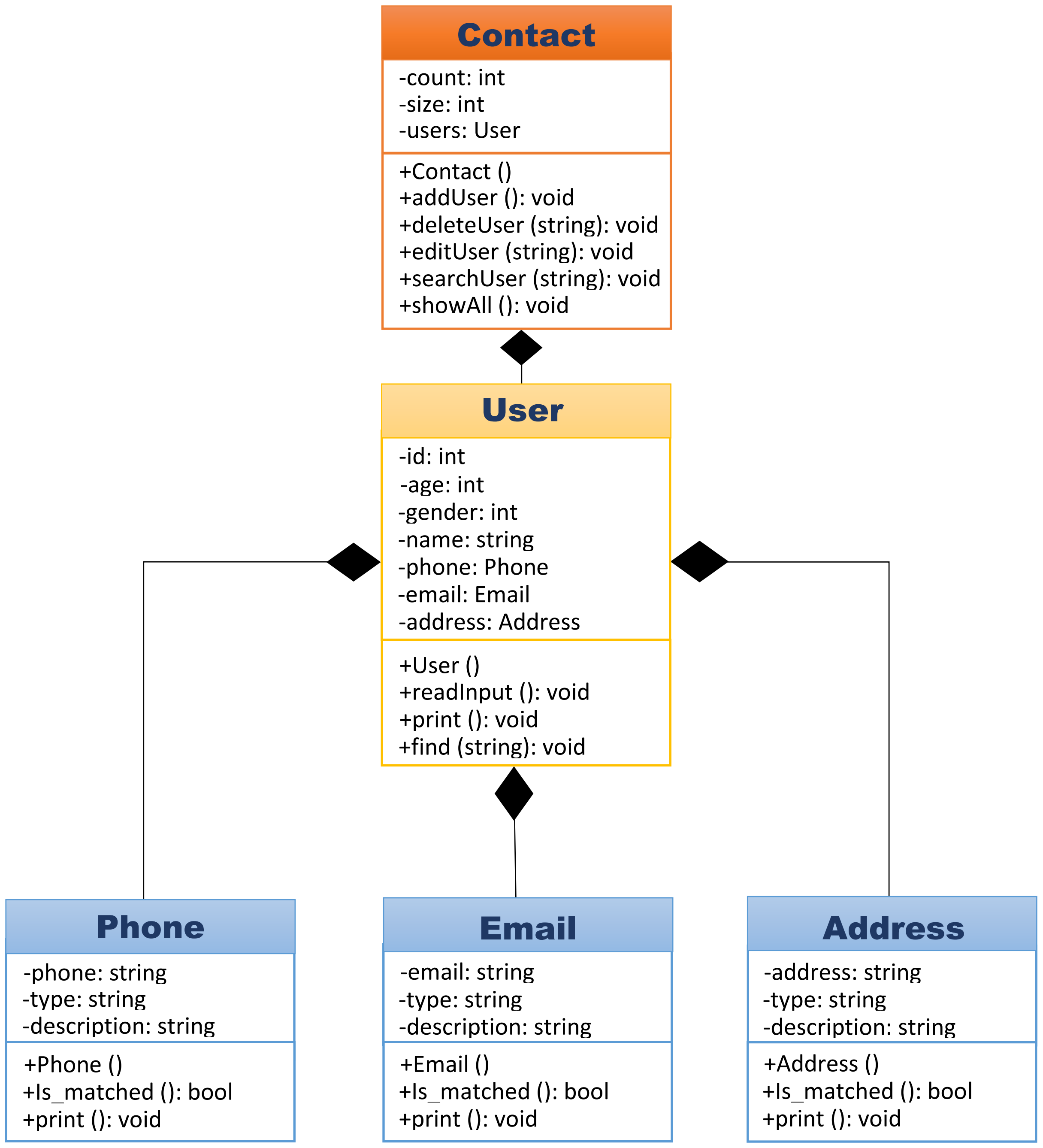
Arrays:

Arrays has a variety of datatypes like built in datatypes (char, int, string…. etc.) and other are user defined data types (phone, email, address and user classes).

These data were collected to provide the best solution for collecting data together to be easily reached and used.

To give the best performance and save the memory we choose dynamic array to be used due to its performance and because it can fit automatically after insert or deleting objects or data from the array.

UML Class Diagram



Description for Classes and Functions Used:

Phone, Email and Address classes:

These classes are created to store the user information from phone, email and address. Its Contain functions to set them with the type and description for each. And Functions to get these sored data in case you want to check on it.

User class:

This class has some function which can be accessed by user. The (read Input) function is used to take the data from user (name, id, age, gender) and stored in built in data types like (int, string). Furthermore (some of phones, some of emails and some of addresses) stored in user defined data types (Phone, Email, Address). Another function called (print) used to show stored data of the user. And function call (find) which take a string keyword which user want to search and show print its data.

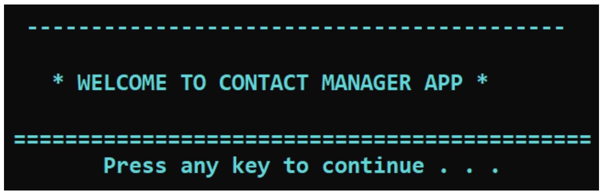
Contact class:

The contact class holds the main system. This due to decomposing it from functions called (add User, edit User, delete User, search User, show All). This class used to store many of users from different places and nationalities by using methods mentioned above.

So, this is the controller class of the program.

Samples Output from Console Screen:

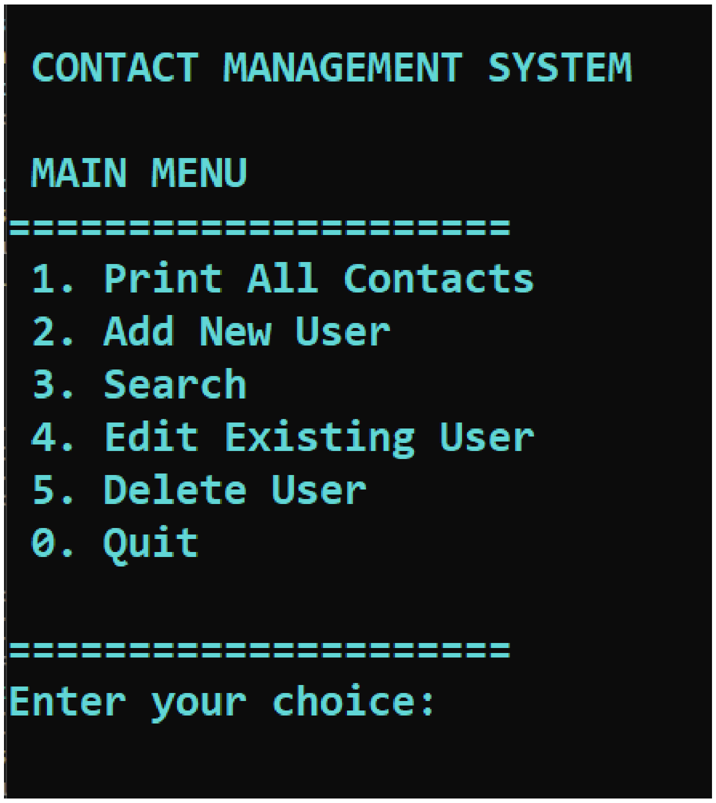
The Welcome Screen



```
 ---------------------------------------

   * WELCOME TO CONTACT MANAGER APP *

=============================================
        Press any key to continue . . .
```

Figure no: 1

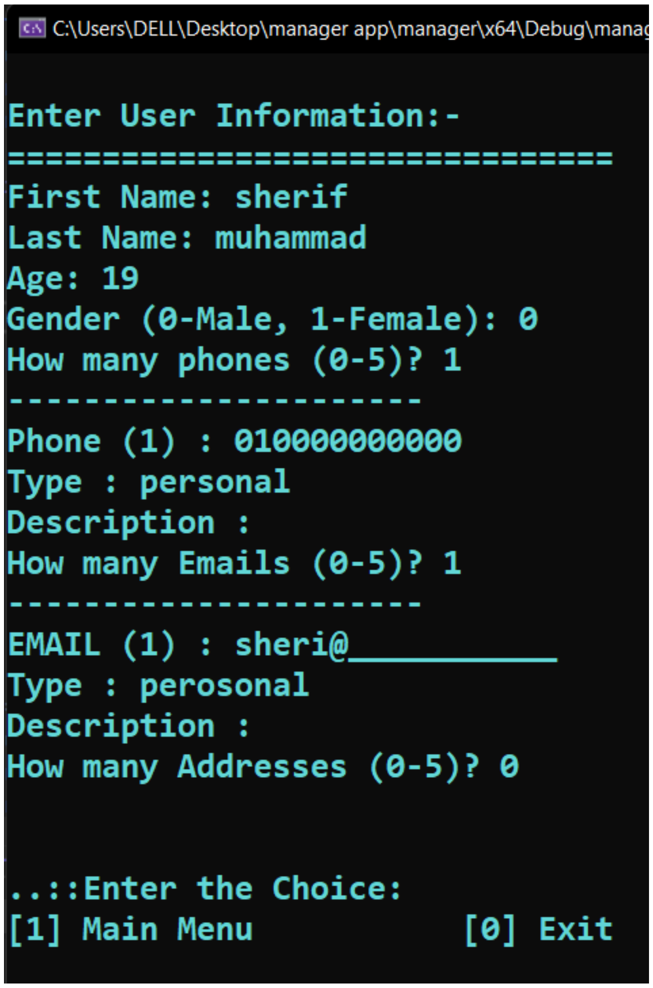
User Interface



```
 CONTACT MANAGEMENT SYSTEM

 MAIN MENU
=====================
 1. Print All Contacts
 2. Add New User
 3. Search
 4. Edit Existing User
 5. Delete User
 0. Quit

=====================
Enter your choice:
```

Figure no: 2

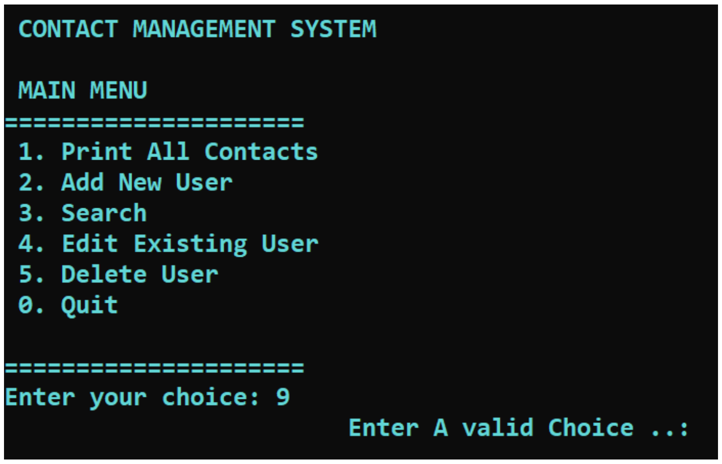
Add New User



```
C:\Users\DELL\Desktop\manager app\manager\x64\Debug\manag

Enter User Information:-
================================
First Name: sherif
Last Name: muhammad
Age: 19
Gender (0-Male, 1-Female): 0
How many phones (0-5)? 1
----------------------
Phone (1) : 010000000000
Type : personal
Description :
How many Emails (0-5)? 1
----------------------
EMAIL (1) : sheri@___________
Type : perosonal
Description :
How many Addresses (0-5)? 0


..::Enter the Choice:
[1] Main Menu                [0] Exit
```

Figure no: 4

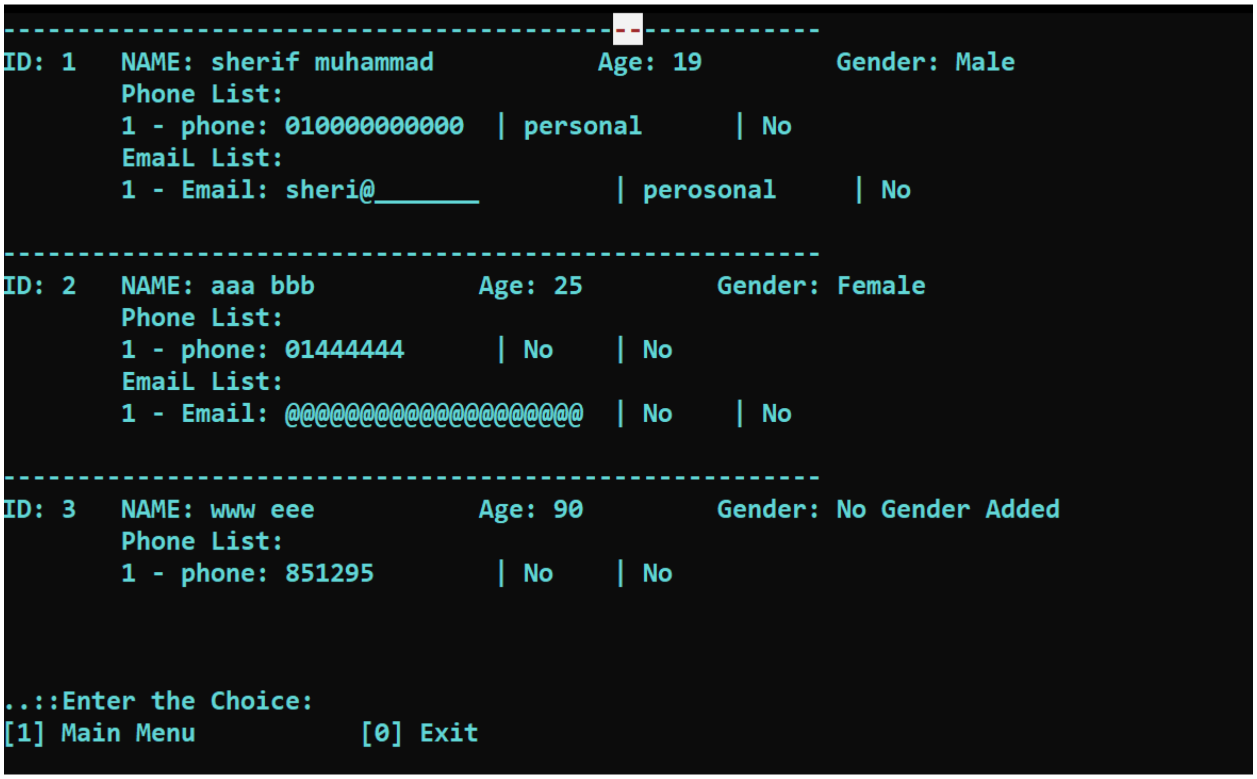
If the user entered an invalid choice, the program sleeps form seconds, then work again.



```
 CONTACT MANAGEMENT SYSTEM

 MAIN MENU
====================
 1. Print All Contacts
 2. Add New User
 3. Search
 4. Edit Existing User
 5. Delete User
 0. Quit

====================
Enter your choice: 9
                        Enter A valid Choice ..:
```

Figure no: 3

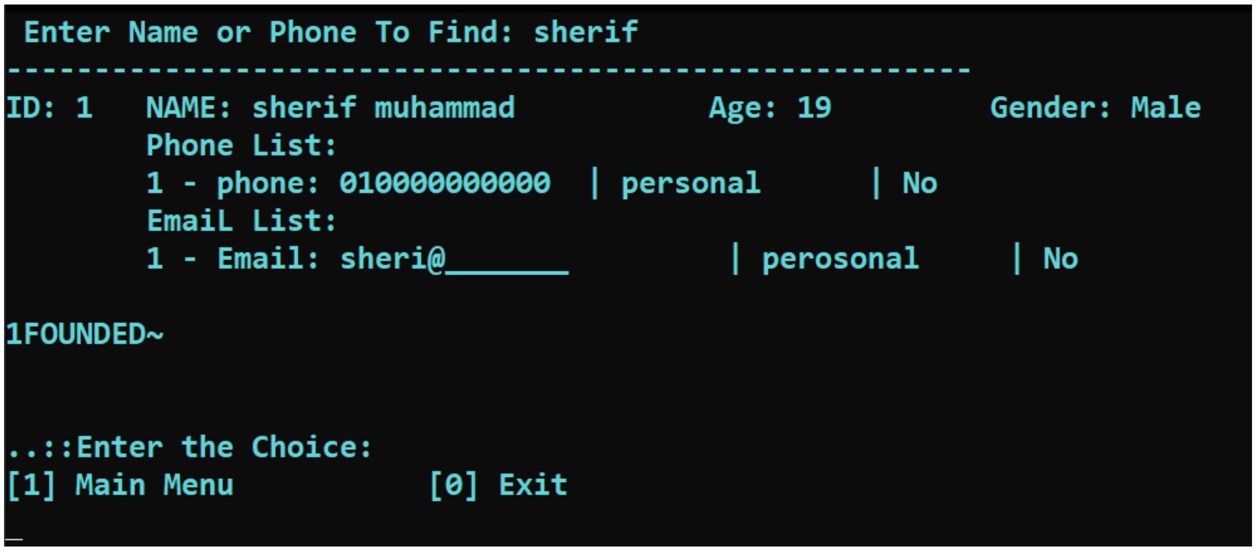
Show All Users



```
-------------------------------------------------------
ID: 1   NAME: sherif muhammad          Age: 19        Gender: Male
        Phone List:
        1 - phone: 010000000000  | personal      | No
        EmaiL List:
        1 - Email: sheri@_______         | perosonal     | No

-------------------------------------------------------
ID: 2   NAME: aaa bbb          Age: 25        Gender: Female
        Phone List:
        1 - phone: 01444444      | No     | No
        EmaiL List:
        1 - Email: @@@@@@@@@@@@@@@@@@@@  | No     | No

-------------------------------------------------------
ID: 3   NAME: www eee          Age: 90        Gender: No Gender Added
        Phone List:
        1 - phone: 851295        | No     | No



..::Enter the Choice:
[1] Main Menu           [0] Exit
```

Figure no: 5

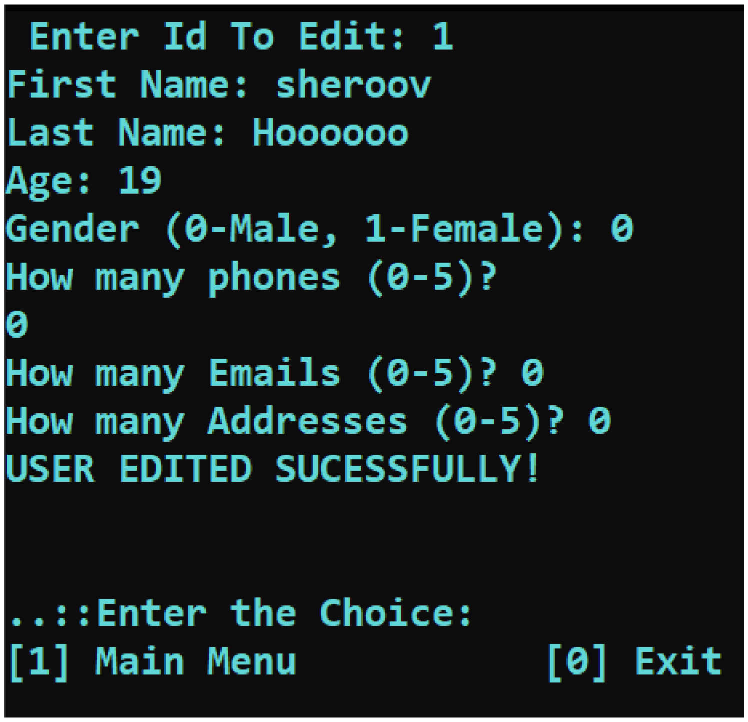
Search For Users



```
 Enter Name or Phone To Find: sherif
-------------------------------------------------------
ID: 1   NAME: sherif muhammad          Age: 19        Gender: Male
        Phone List:
        1 - phone: 010000000000  | personal      | No
        EmaiL List:
        1 - Email: sheri@_______         | perosonal     | No

1FOUNDED~


..::Enter the Choice:
[1] Main Menu           [0] Exit
_
```

Figure no: 6

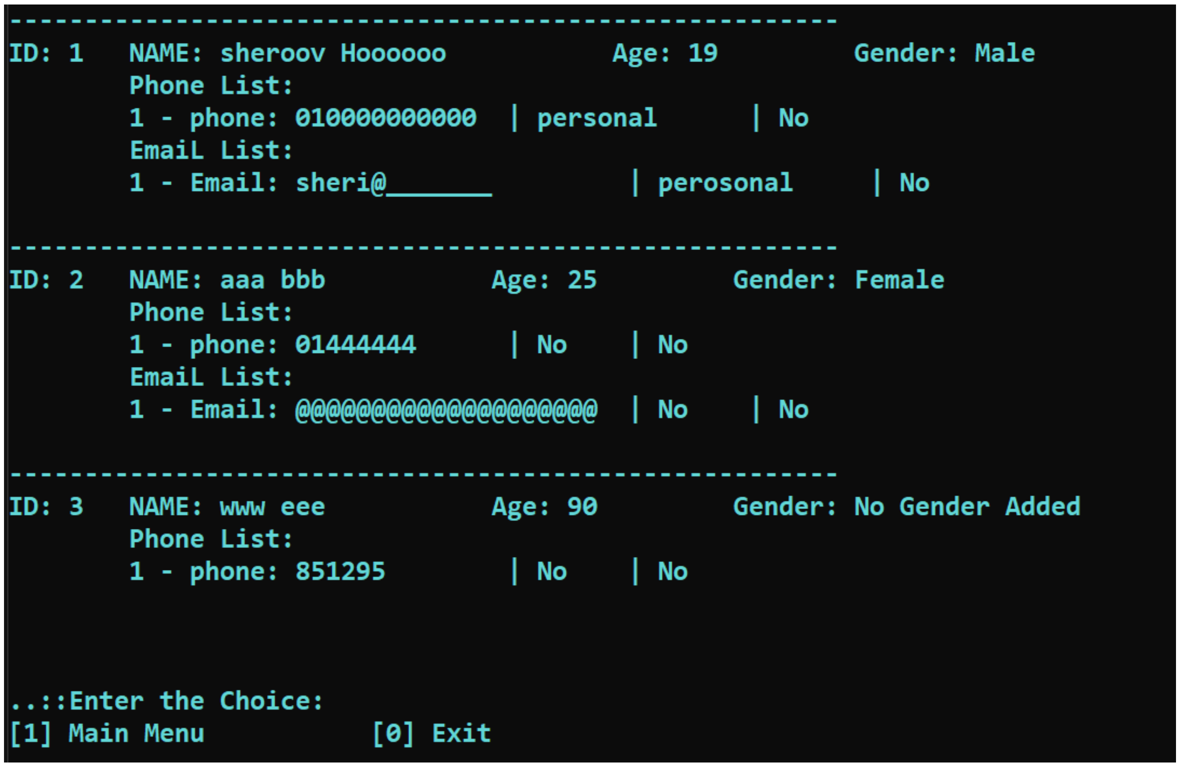
Edit an existing User



```
 Enter Id To Edit: 1
First Name: sheroov
Last Name: Hoooooo
Age: 19
Gender (0-Male, 1-Female): 0
How many phones (0-5)?
0
How many Emails (0-5)? 0
How many Addresses (0-5)? 0
USER EDITED SUCESSFULLY!


..::Enter the Choice:
[1] Main Menu           [0] Exit
```

Figure no: 7

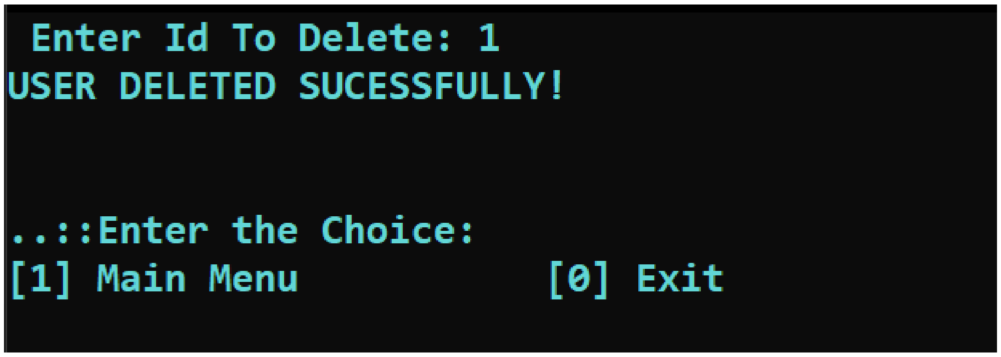
Show All Users After Editiing



```
-------------------------------------------------------
ID: 1   NAME: sheroov Hoooooo          Age: 19         Gender: Male
        Phone List:
        1 - phone: 010000000000  | personal      | No
        EmaiL List:
        1 - Email: sheri@_______         | perosonal     | No

-------------------------------------------------------
ID: 2   NAME: aaa bbb          Age: 25         Gender: Female
        Phone List:
        1 - phone: 01444444      | No     | No
        EmaiL List:
        1 - Email: @@@@@@@@@@@@@@@@@@@@  | No     | No

-------------------------------------------------------
ID: 3   NAME: www eee          Age: 90         Gender: No Gender Added
        Phone List:
        1 - phone: 851295        | No     | No



..::Enter the Choice:
[1] Main Menu           [0] Exit
```

Figure no: 8

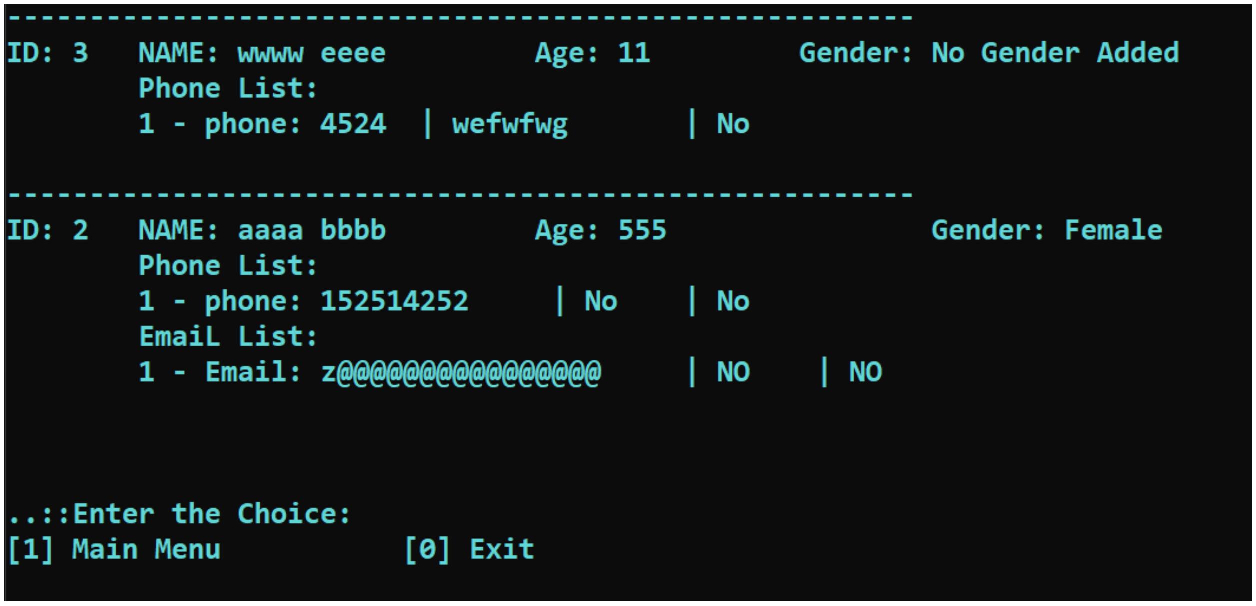
Delete User



```
 Enter Id To Delete: 1
USER DELETED SUCESSFULLY!


..::Enter the Choice:
[1] Main Menu           [0] Exit
```

Figure no: 9

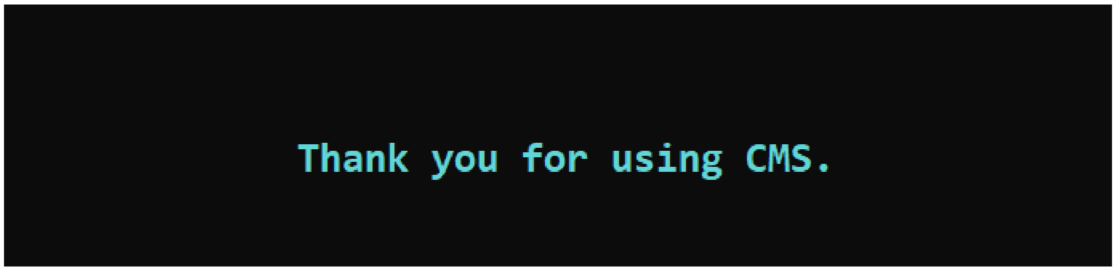
Show All Users After Deleting



```
-------------------------------------------------------
ID: 3   NAME: wwww eeee        Age: 11        Gender: No Gender Added
        Phone List:
        1 - phone: 4524  | wefwfwg      | No

-------------------------------------------------------
ID: 2   NAME: aaaa bbbb        Age: 555               Gender: Female
        Phone List:
        1 - phone: 152514252     | No    | No
        EmaiL List:
        1 - Email: z@@@@@@@@@@@@@@@@     | NO    | NO



..::Enter the Choice:
[1] Main Menu           [0] Exit
```

Figure no: 10

Exit Screen



```
Thank you for using CMS.
```

Figure no: 11

Copy Of C++ Code:

Main source code

```
#include <iostream>
#include "contact.h"
using namespace std;
void remain() {
    int opt;
    cout << "\n\n.::Enter the Choice:\n[1] Main Menu\t\t[0] Exit\n";
    cin >> opt;
    if (opt == 0) {
        system("cls");
        cout << "\n\n\n\t\t\tThank you for using CMS." << endl << endl;
        exit(0);
    }
}
int main()
{
    system("cls");
    system("Color 0B");
    cout << "\n\n\n\n\n\n\n\n\n\n\n\t\t\t\t   ------------------------------------------"
        << "\n\n\t\t\t\t     * WELCOME TO CONTACT MANAGER APP *"
        << "\n\n\t\t\t\t  =============================================";
    cout << "\n\t\t\t\t\t ";
    system("pause");
    int c = -1;
    contact contact(100);
    while (c != 0) {
        system("cls"); system("Color 0B");
        cout << "\n CONTACT MANAGEMENT SYSTEM";
        cout << "\n\n MAIN MENU";
        cout << "\n======================\n";
        cout << " 1. Print All Contacts\n"<< " 2. Add New User\n" << " 3. Search \n"
            << " 4. Edit Existing User\n"<< " 5. Delete User\n"<< " 0. Quit\n";
        cout << "\n======================\n";
        cout << "Enter your choice: ";
        cin >> c;
        if (c == 1) {
            system("cls");contact.print();remain();
        }
        else if (c == 2) {
            system("cls");contact.addNewUser();remain();
        }
        else if (c == 3) {
            system("cls");
            string key;cout << " Enter Name or Phone To Find: ";
             cin >> key;contact.findAll(key);remain();
        }
      else if (c == 4) {
            system("cls");int editid;cout << " Enter Id To Edit: ";cin >> editid;
            contact.editeUser(editid); remain();
        }
        else if (c == 5) {
            system("cls");
           int delid; cout << " Enter Id To Delete: ";
            cin >> delid;contact.editeUser(delid);remain();
        }
      else {
            cout << "\t\t\tEnter A valid Choice ..:";
            Sleep(2000);
        }
    }
    system("cls");
    cout << "\n\n\n\t\t\tThank you for using CMS." << endl << endl;
    return 0;
}
```

Contact Class Code: __contact.h__

```
#pragma once
#include"user.h"
#include<iostream>
using namespace std;
class contact
{
private:
      int count,size;
      user* users;
public:
      contact(int);
      ~contact();
      void addNewUser();
      void deleteUser(int);
      void editeUser(int);
      void print() const;
      void findAll(string) const;
};
```

Contact Class Code: __contact.cpp__

```
#include "contact.h"
contact::contact(int contactSize)
{
      count = 0;size = contactSize;users = new user[size];
}
contact::~contact(){delete[]users;}
void contact::addNewUser()
{
      user* newUser = new user(); newUser->setUserId(count + 1);
      newUser->readInput();users[count++] = *newUser;
      delete newUser;
}
```

```
else if (c == 4) {
            system("cls");int editid;cout << " Enter Id To Edit: ";cin >> editid;
contact.editeUser(editid); remain();
        }
        else if (c == 5) {
            system("cls");
            int delid;cout << " Enter Id To Delete: ";cin >>
delid;contact.editeUser(delid);remain();
        }
    }
    system("cls");
    cout << "\n\n\n\t\t\tThank you..:)" << endl << endl;
    return 0;
}

void contact::deleteUser(int userid)
{
      if (count == 0) { cout << " NO USER EXIXT!!"<< endl; return; }
      bool del = false;
      for (size_t i = 0; i < count; i++)
      {
            if (users[i].getUserId() == userid) {
                  if (i == count - 1) count--;
                  else { users[i] = users[count - 1];count--;}
                  del = true;break;
            }
      }
      if (del) cout << " USER DELETED SUCCESFULLY" << endl;
      else cout << " NO USER EXIST" << endl;
}

void contact::editeUser(int userid)
{

      if (count == 0) {cout << " NO USER EXIXT!!" << endl;return;}
      bool edit = false;
      for (size_t i = 0; i < count; i++)
      {
         if (users[i].getUserId() == userid) {users[i].readInput();
         edit = true;break;}}
      if (edit) cout << " USER EDITED SUCCESFULLY" << endl;
      else cout << " NO USER EXIST" << endl;
}

void contact::print() const
{if (count == 0) { cout << " No User Found!!!" << endl; return;}
      for (size_t i = 0; i < count; i++) {users[i].print();}}

void contact::findAll(string key) const
{
      bool found=false;
      for (size_t i = 0; i < count; i++)
      { if (users[i].findAny(key)) { users[i].print(); found = true;}}
      if (!found) cout << " NO USER FOUNDED!!" << endl;
}
```

User Class Code: __user.h__

```
#include <iostream>
#include <string>
#include "phone.h"
#include "email.h"
#include "address.h"
using namespace std;
class user{
      string fname, lname; int gender, age, id;
      phone* v_phone; email* v_email; address* v_address;
      int phone_size; int address_size; int email_size;
      int phone_counter; int address_counter;int email_counter;
public:
      user(); virtual ~user();
      void setUserId(int);void setFname(string); void setlname(string);
      void setAge(int); void setGender(int); int  getUserId() const;
      void print() const;void readInput();bool findAny(string);
};
```

User Class Code: __user.cpp__

```
#include<iostream>
#include "user.h"
user::user(){
setAge(age); setFname(fname); setlname(lname); setGender(gender); setUserId(id);
phone_size = 5; address_size = 5; email_size = 5;
phone_counter = 0; address_counter = 0; email_counter = 0;
v_phone = new phone[phone_size]; v_email = new email[email_size];
v_address = new address[address_size];
}

user::~user(){delete[]v_address;delete[]v_email;delete[]v_phone;}
void user::setUserId(int id) {this->id = id;}
void user::setFname(string fname) {this->fname = fname;}
void user::setlname(string lname) {this->lname = lname;}
void user::setAge(int age) {this->age = age;}
void user::setGender(int gender) {this->gender = gender;}
int  user::getUserId() const { return id; }

void user::print() const
{
    cout << "\n================================\n";
    cout << " Id: " << id << "\n Name: " << fname << " " << lname << endl;
    if (age > 0)  cout << " Age:  " << age << endl;
    if (gender == 0) cout << " Gender: Male";
    else cout << " Gender: Female";
    if (phone_counter > 0) {
        cout << " List Phones: " << endl;
        for (size_t i = 0; i < phone_counter; i++)
        {cout << " " << i + 1 << ") "; v_phone[i].print();}
    }
    if (email_counter > 0) {
        cout << " List Emails: " << endl;
        for (size_t i = 0; i < email_counter; i++)
        {cout << " " << i + 1 << ") "; v_email[i].print();}
    }
```

```
if (address_counter > 0) {
        cout << " List Addresses: " << endl;
        for (size_t i = 0; i < address_counter; i++)
        {
            cout << " " << i + 1 << ") "; v_address[i].print();}
    }
    cout << "\n=====================================================" << endl;
}

void user::readInput() {
    cout << " Enter User Informatiion: " << endl;
    cout << "===========================" << endl;
    cout << " First Name: ";
    cin.ignore(); getline(cin, fname);
    cout << " Last Name: ";
    cin.ignore(); getline(cin, lname);
    cout << " Age: ";cin >> age;
    cout << " Enter '0' For Male And '1' For Female: "; cin >> gender;
    int n = 0; cout << " How Many Phones (0-5): "; cin >> n;
    for (size_t i = 0; i < n; i++){
        string data, type, desc;
        cout << " Phone " << i + 1 << " : ";cin >> data;
        cout << " Type: ";cin >> type;
        cout << " Describtion: ";cin >> desc;
        phone* new_phone = new phone(data, type, desc);
        v_phone[phone_counter++] = *new_phone;delete new_phone;
    }
    int e = 0; cout << " How Many Emails (0-5): ";cin >> e;
    for (size_t i = 1; i <= e; i++){
        string data, type, desc;
        cout << " Email " << i << " : ";cin >> data;
        cout << " Type: ";cin >> type;
        cout << " Describtion: ";cin >> desc;
        email* new_email = new email(data, type, desc);
        v_email[email_counter++] = *new_email;delete new_email;
    }
    int ad = 0;cout << " How Many Places (0-5): ";cin >> ad;
    for (size_t i = 1; i <= ad; i++){
        string data, type, desc;
        cout << " Address " << i << " : ";cin >> data;
        cout << " Type: ";cin >> type;
        cout << " Describtion: ";cin >> desc;
        address* new_address = new address(data, type, desc);
        v_address[address_counter++] = *new_address;delete new_address;
    }
}

bool user::findAny(string key) {
    if (fname.compare(key) == 0 || lname.compare(key) == 0)
        return true;
    if (phone_counter > 0) {
        for (size_t i = 0; i < phone_counter; i++)
        {
            if (v_phone[i].is_Matched(key))
                return true;
        }
    }
    return false;
}
```

Phone Class Code: __phone.h__

```
#include <string>
using namespace std;
#ifndef PHONE_H
#define PHONE_H
class phone{
    string m_phone;string m_type;string m_description;
public:
    phone();phone(string phone, string type, string desc);virtual ~phone();
    void setPhone(string);void setType(string);void setDesc(string);
    bool is_Matched(string) const;void print() const;
};
#endif // PHONE_H
```

Phone Class Code: __phone.cpp__

```
#include "phone.h"
#include<iostream>
using namespace std;
phone::phone() {}
phone::phone(string phone, string type, string desc)
{ setPhone(phone); setType(type); setDesc(desc);}
phone::~phone() {};
void phone::setPhone(string phone) { m_phone = phone;}
void phone::setType(string type) { m_type = type; }
void phone::setDesc(string desc) { m_description = desc; }

bool phone::is_Matched(string key) const
{ return (m_phone.compare(key) == 0); }
void phone::print() const{
    cout << " Phone:  " << m_phone << " \t " << m_type << " \t " <<m_description << endl;
}
```

Email Class Code: __email.h__

```
#include <string>
using namespace std;
#ifndef EMAIL_H
#define EMAIL_H
class email{
    string m_email;
    string m_type;
    string m_description;
public:
    email();
    email(string, string, string);
    virtual ~email();
    void setEmail(string);
    void setType(string);
    void setDesc(string);
    bool is_Matched(string);
    void print() const;
};
#endif // EMAIL_H
```

Email Class Code: __email.cpp__

```
#include "email.h"
#include<iostream>
email::email() {}
email::email(string email, string type, string desc){
       setEmail(email);setType(type);setDesc(desc);
}
email::~email(){}
void email::setEmail(string email) {m_email = email;}
void email::setType(string type) { m_type = type;}
void email::setDesc(string desc) { m_description = desc;}
bool email::is_Matched(string key){return (m_email.compare(key) == 0);}
void email::print() const
{
std::cout << " Email:   " << m_email << " \t " << m_type << " \t " <<m_description <<endl;
}
```

Address Class Code: __address.h__

```
#include <string>
using namespace std;
#ifndef ADDRESS_H
#define ADDRESS_H
class address{
    string m_address, m_type,m_description;
public:
    address();
    address(string, string, string);
   ~address();
    void setAddress(string);
    void setType(string);
    void setDesc(string);
    void print() const;
};
#endif // ADDRESS_H
```

Address Class Code: __address.cpp__

```
#include "address.h"
#include<iostream>
address::address() {}
address::address(string add, string type, string desc) {
       setAddress(add);
       setType(type); setDesc(desc);
};
address::~address() {}
void address::setAddress(string address) { m_address = address; }
void address::setType(string type) { m_type = type; }
void address::setDesc(string desc) { m_description = desc; }
void address::print() const {
       std::cout << " Address: " << m_address <<
             " \t " << m_type << " \t " << m_description << endl;
}
```