**Abstract:**

* **The Problem** –

Coronavirus disease 2019 (COVID-19) has had a catastrophic impact in this world.Self-Isolation is the only judicious solution to counter its devastating impact.

* **The Solution-**

Self – Isolation Monitoring System is used to monitor the health parameters of self-quarantined patients remotely. A wrist band measures the parameters like blood pressure, body Temperature, pulse rate, location, etc. along with allowing the symptoms like cough, running nose and short breath to be entered-which are stored in files, manipulated and displayed.

* **Technical Concepts –**

Here, we’ve used the concept of Object Oriented Programming along with C++ as the base language. The concepts are listed and implemented as follows:

1. **Classes and objects:** We have created 4 classes and an object for every record.
2. **Functions:** Functions are used along with member functions to calculate the received readings and store them in files.
3. **Inheritance**: We have a base class Person. Regulate and Symptoms inherit the base class Person (Multi-level inheritance). Then ’Calculate’ class inherits them both using virtual classes (Hybrid Inheritance).
4. **Constructors and Destructors**: They automatically measure the readings and decide which file to add to. Destructors delete the object when control goes out of the main function.
5. **Abstraction:** We have used private, protected and public class modes.
6. **Polymorphism:** We have eliminated redundancy in the code.
7. **Pointers:** We dynamically allocate a new object every time a new record of a person is created.
8. **Exception handling:** We have implemented exceptions in the main function in case the user enters the wrong option accidently.
9. **File handling:** Every person’s record is stored using the concept of File Handling i.e., libraries like ‘ifstream’ and ’ofstream’.

* **Use –**

This System will be used by Doctors and para-medical staff to monitor the health conditions of quarantined patients in their homes/rooms. This System will detect record a patient’s health readings, store, manipulate and allow them to be viewed later.

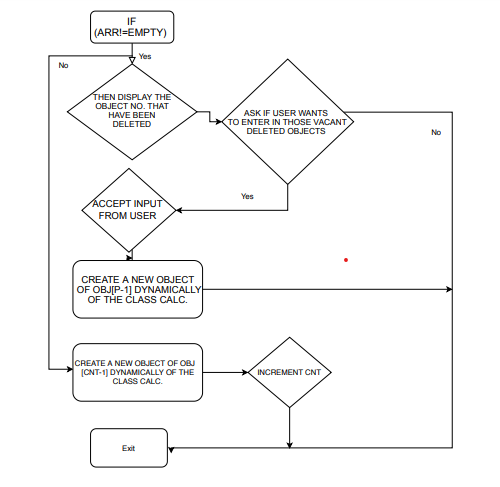
**Table of contents:**

1. Introduction
2. Flow Chart
3. Algorithm
4. Coding
5. Output
6. Conclusion
7. References
8. **Introduction:-**

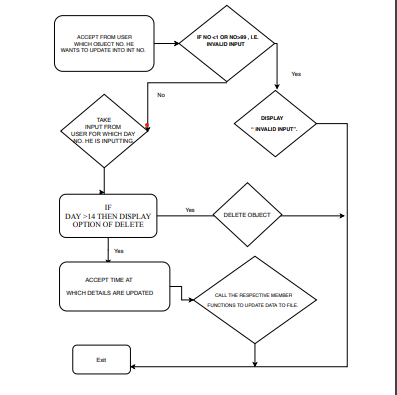
* The outbreak of coronavirus disease 2019 (COVID-19) has created a global health crisis that has had a deep impact on the way we perceive our world and our everyday lives. The world is grappling with an invisible, deadly enemy, trying to understand how to live with the threat posed by a virus. Moreover, there still isn’t a sustainable cure to this disease.
* The only way to defeat it is to prevent it. Across the globe people are being tested and quarantined. At the same time, it isn’t practical to quarantine everyone at hospitals as they may get over-populated and thus may contribute to the problem. Also, there lies a huge risk for the people in the vicinity of the person kept under observation (such as his relatives or para-medical staff). But, keeping them under observation is obligatory and at the same time inevitable. Thus, we have tried to come up with a solution of a “Self-Isolation Monitoring System” which may be further developed.
* When a person gets self-isolated at his home/in a room, we assume that he wears a wrist band which measures his health parameters like temperature, blood- pressure, location, etc. The band would periodically update his readings to the system. The system collects this data and stores it in a file. Every time the band updates the readings, it gets appended to the individual’s record. Then it is automatically calculated if the readings show abnormal range. If it is so, then the readings are added to a file called as Critical.txt. Also if he is out of his location area, we the details of the person appended to a file called location.txt.
* An option is provided to the medical staff to view either the individual’s data, the group lying in Critical.txt (which need special attention), or the people who were recorded to be out of their location at a particular instance of time. After the readings exceed Day-14, we also provide an option to delete the record to the medical staff.

1. **FLOWCHARTS:**

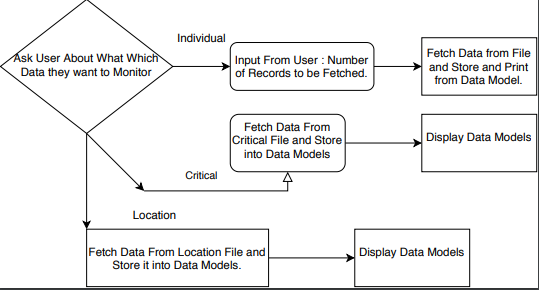
**CREATE**

****

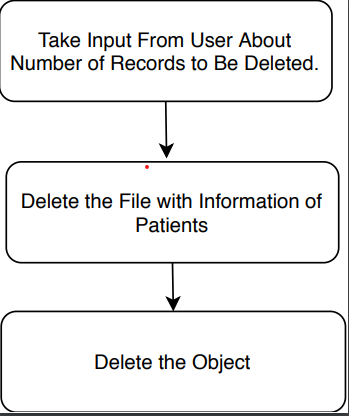
**UPDATE**

****

**VIEW**



**DELETE**



1. **ALGORITHMS: -**
2. **Create Record:**
3. IF (ARR=NOT EMPTY) I.E., THERE ARE EXISTING OBJECTS IN THE SYSTEM.GOTO 4
4. THEN DISPLAY THE OBJECT NO. THAT HAVE BEEN DELETED AND CAN BE REUSED.
5. ASK IF USER WANTS TO ENTER IN THOSE VACANT DELETED OBJECTS (RECORDS).
6. ACCEPT INPUT FROM USER INTO CHARACTER A.
7. IF A=’Y’ THEN GOTO 6 ELSE GOTO 8
8. ACCEPT INTO CHAR ‘P’ WHICH NO. VACANT OBJECT HE WANTS TO CREATE RECORD INTO.
9. CREATE A NEW OBJECT OF OBJ[P-1] DYNAMICALLY OF THE CLASS CALC. GOTO 10
10. CREATE A NEW OBJECT OF OBJ [CNT-1] DYNAMICALLY OF THE CLASS CALC.
11. INCREMENT CNT.
12. EXIT.
13. **Update Record:**
14. ACCEPT FROM USER WHICH OBJECT NO. HE WANTS TO UPDATE INTO INT NO.
15. IF NO <1 OR NO>99 , I.E. INVALID INPUT THEN GOTO 3
16. DISPLAY “INVALID INPUT”.
17. ELSE TAKE INPUT FROM USER FOR WHICH DAY NO. HE IS INPUTTING.
18. IF DAY >14 THEN DISPLAY OPTION OF DELETE. TAKE INPUT INTO ‘N’.
19. IF USER WANTS TO CONTINUE ENTERING THEN GOTO 7 ELSE GOTO 10.
20. END IF.
21. ACCEPT TIME AT WHICH DETAILS ARE UPDATED.
22. CALL THE RESPECTIVE MEMBER FUNCTIONS TO UPDATE DATA TO FILE.
23. EXIT.
24. **View Record:**
25. Take Input from User about Which Record They Want to Monitor
26. If User Want to Observe Individual Record: Input Number of Persons:
27. Fetch Data from File and Store it in Data Models.
28. Show Details of Individual Person.
29. If User Wants to Observe Critical Records:
30. Fetch data From Critical File and Store it in Data Models
31. Display Content of Critical Record.
32. If User Wants to Observe People Outside of their Location:
33. Fetch data From Location File and Store it in Data Models
34. Display Content of Patients which are Outside of their Location**.**
35. **Delete Record:**
36. Take Input from User about Number of Records to be deleted.
37. Delete the File which contents of Patient.
38. Delete the Object.
39. **INPUT CODE: -**

#include<iostream>

#include <stdlib.h>

#include <string.h>

#include<conio.h>

#include <fstream>

#include <cstdlib>

#include <stack>

using namespace std;

class Person;

class Regulate;

class Symptoms;

class Calc;

stack<int> arr;

// Class declarations and initializations

class Person{

protected:

char firstname[30],lastname[30],gender[10];

char id[20];

int age;

public:

char time[20],filename[20];int Count\_Day=1;

Person(){

cout << "Enter your id :(\'Note: It can't be changed later.\') :";

cin >> filename;

strcpy(id,filename);

strcat(filename,".txt");

ofstream File(filename);

get\_details();

set\_details();

}

void get\_details();

void set\_details();

};

void Person :: get\_details(){

cout<<"Enter day number: "; cin>>Count\_Day;

cout<<"Enter time : "; cin>>time;

cout<<"Enter firstname : "; cin>>firstname; firstname[0]=toupper(firstname[0]);

cout<<"Enter lastname : "; cin>>lastname; lastname[0]= toupper(lastname[0]);

cout<<"Enter gender : "; cin>>gender; gender[0]= toupper(gender[0]);

}

void Person :: set\_details(){

ofstream File;

File.open(filename, ios::out | ios::app);

if(File.is\_open()){

File<<Count\_Day<<endl;

File<<time<<endl;

File<<id<<endl;

File<<firstname<<endl;

File<<lastname<<endl;

File<<gender<<endl;

}

}

class Regulate : virtual public Person{

protected:

float bp\_h,bp\_l,temperature,pulse\_rate;

public:

char location[5];

Regulate(){

get\_health();

write\_health\_to\_file();

}

void get\_health();

void write\_health\_to\_file();

};

void Regulate :: get\_health(){

cout<<"Within location boundary? (y/n) : "; cin>>location;

cout<<"Enter high blood pressure Normal(120-80): "; cin>>bp\_h;

cout<<"Enter low Blood Pressure Normal(90-60) :";cin>> bp\_l;

cout<<"Enter temperature.Normal(98.6-99.9) : "; cin>>temperature;

cout<<"Enter pulse\_rate.Normal(60-100) : "; cin>>pulse\_rate;

}

void Regulate :: write\_health\_to\_file(){

ofstream File(filename, ios::out | ios::app);

if( File.is\_open() ) {

File<<location<<endl;

File<<bp\_h<<endl;

File<<bp\_l<<endl;

File<<temperature<<endl;

File<<pulse\_rate<<endl;

}

};

class Symptoms : virtual public Person{

protected:

char dry\_cough[5],running\_nose[5],short\_breath[5];

public:

Symptoms(){

get\_symp();

write\_symp\_to\_file();

}

void get\_symp();

void write\_symp\_to\_file();

};

void Symptoms :: get\_symp(){

cout<<"Enter dry cough (y/n): "; cin>>dry\_cough;

cout<<"Enter running nose (y/n): "; cin>>running\_nose;

cout<<"Enter short breath (y/n): "; cin>>short\_breath;

}

void Symptoms :: write\_symp\_to\_file(){

ofstream File;

File.open(filename, ios::out | ios::app);

if(File.is\_open()){

File<<dry\_cough<<endl;

File<<running\_nose<<endl;

File<<short\_breath<<endl;

}

}

class Calc : public Regulate , public Symptoms{

public:

int turn=0;int n=0;

Calc(){

cout<<"\nRecord created."<<endl;

}

~Calc(){

cout<<"Destructor called ~ Object destroyed."<<endl;

}

void addto\_file();

void show\_details();

void update\_critical();

void display\_critical();

void check\_location();

void show\_location();

};

void Calc :: addto\_file(){

set\_details();

write\_health\_to\_file();

write\_symp\_to\_file();

update\_critical();

check\_location();

cout<<"\n\'Details updated.\'"<<endl;

}

void Calc :: show\_details(){

ifstream File(filename);

if(File.is\_open()){

while(File>>Count\_Day>>time>>id>>firstname>>lastname>>gender>>location>>bp\_h>>bp\_l>>temperature>>pulse\_rate>>dry\_cough>>running\_nose>>short\_breath){

cout<<"The details are as follows : "<<endl;

cout<<"\n--------\t Day Number : " <<Count\_Day<<"\t---------"<<endl;

cout<<"\n--------\tTime when recorded : " <<time<<"\t---------"<<endl;

cout<<"ID : "<<id<<endl;

cout<<"Firstname : "<<firstname<<endl;

cout<<"Lastname : "<<lastname<<endl;

cout<<"Gender : "<<gender<<endl;

cout<<"Within Location ? : "<<location<<endl;

cout<<"High blood pressure : "<<bp\_h<<endl;

cout<<"Lower blood pressure : "<<bp\_l<<endl;

cout<<"Temperature : "<<temperature<<endl;

cout<<"Pulse rate : "<<pulse\_rate<<endl;

cout<<"Dry cough : "<<dry\_cough<<endl;

cout<<"Running nose : "<<running\_nose<<endl;

cout<<"Short breath : "<<short\_breath<<endl;

}

}

else {cout<<"File not open."<<endl;}

}

void Calc :: update\_critical(){

ifstream File(filename);

if ( File.is\_open()) {

while(File>>Count\_Day>>time>>id>>firstname>>lastname>>gender>>location>>bp\_h>>bp\_l>>temperature>>pulse\_rate>>dry\_cough>>running\_nose>>short\_breath)

{

if(n>=turn){

if(temperature>99.9 || bp\_l <60 || bp\_l>90 || bp\_h <80 || bp\_h>120 || pulse\_rate<60 || pulse\_rate>100

|| dry\_cough=="y"|| dry\_cough=="Y" || running\_nose=="y" ||running\_nose=="Y" || short\_breath=="y" || short\_breath=="Y")

{

ofstream FILE("Critical.txt", ios::out | ios::app);

FILE<<Count\_Day<<endl;

FILE<<time<<endl;

FILE<<id<<endl;

FILE<<firstname<<endl;

FILE<<lastname<<endl;

FILE<<gender<<endl;

FILE<<location<<endl;

FILE<<bp\_h<<endl;

FILE<<bp\_l<<endl;

FILE<<temperature<<endl;

FILE<<pulse\_rate<<endl;

FILE<<dry\_cough<<endl;

FILE<<running\_nose<<endl;

FILE<<short\_breath<<endl;

}

}

n++;

}

n=0;

}

}

void Calc :: display\_critical(){

ifstream File("Critical.txt");

if(File.is\_open()){

cout<<"\n--------\tThe following people are in critical file : "<<endl;

while(File>>Count\_Day>>time>>id>>firstname>>lastname>>gender>>location>>bp\_h>>bp\_l>>temperature>>pulse\_rate>>dry\_cough>>running\_nose>>short\_breath){

cout<<"The details are as follows : "<<endl;

cout<<"\n--------\t Day Number : " <<Count\_Day<<"\t---------"<<endl;

cout<<"\n--------\tTime when recorded : " <<time<<"\t---------"<<endl;

cout<<"ID : "<<id<<endl;

cout<<"Firstname : "<<firstname<<endl;

cout<<"Lastname : "<<lastname<<endl;

cout<<"Gender : "<<gender<<endl;

cout<<"Within Location ? : "<<location<<endl;

cout<<"High blood pressure : "<<bp\_h<<endl;

cout<<"Lower blood pressure : "<<bp\_l<<endl;

cout<<"Temperature : "<<temperature<<endl;

cout<<"Pulse rate : "<<pulse\_rate<<endl;

cout<<"Dry cough : "<<dry\_cough<<endl;

cout<<"Running nose : "<<running\_nose<<endl;

cout<<"Short breath : "<<short\_breath<<endl;

}

}

else {cout<<"Critical file not open."<<endl;}

}

void Calc :: check\_location(){

ofstream file;file.open("location.txt",ios::app);

if(strcmp(location,"n")==0){

file<<Count\_Day<<endl;

file<<time<<endl;

file<<id<<endl;

file<<firstname<<endl;

file<<lastname<<endl;

}

}

void Calc :: show\_location(){

ifstream file("location.txt");

while(file>>Count\_Day>>time>>id>>firstname>>lastname){

cout<<"---------------------- The following people were found out of their location. Details are:- ----------------------------"<<endl;

cout<<"ID : "<<id<<endl;

cout<<"Day no. : "<<Count\_Day<<endl;

cout<<"Time : "<<time<<endl;

cout<<"Firstname : "<<firstname<<endl;

cout<<"Lastname : "<<lastname<<endl;

}

}

int main() {

ofstream FILE("Critical.txt", ios::ate); // If critical exists then truncate, if not exist

ofstream file("location.txt",ios::ate); // then create an empty critical file.

int cnt=0; int temp=0;

Calc \*obj[100];

while(1){

system("cls");

cout<<"Choose an option :"<<endl;

cout<<"1.Create a new record."<<endl;

cout<<"2.Update a record ( for the system )."<<endl;

cout<<"3.View a record. ( for the doctor )"<<endl;

cout<<"4.Delete record."<<endl;

cout<<"5.Exit"<<endl;

int n=0;int no;

cin>>n;

switch(n){

case 1: {

if(arr.empty()== false){

cout<<"Note: Record number/s : ";

for(int i=0;i<arr.size();i++){

cout<<arr.top()<<" ";

}

cout<<" is/are empty."<<endl;

cout<<"Do you want to insert in them ? (y/n)"<<endl;

char a;

cin>>a;

if (a='y'){

int p;

cout<<"Enter the record number : "<<endl;

cin >> p;

obj[p-1]= new Calc();

obj[p-1]->update\_critical();

break;

}

}

obj[cnt]= new Calc();

obj[cnt]->check\_location();

obj[cnt]->update\_critical();

cnt++; temp++;

break;

}

case 2:

try{

cout<<"Enter number of Person's record you want to update (eg. 1,2,3..): ";

cin>>no;

if(no<1 || no>cnt)

{

throw 99;

}

cout<<"Enter Day number : "; cin>>obj[no-1]->Count\_Day;

if(obj[no-1]->Count\_Day >= 14){

cout << "The 14 day Isolation Period of this Patient is Over. \nIf You feel the Patient's Health condition is OK then You can Delete his/her \nRecord.Choose Delete Record Option in Main Menu." <<endl;

cout<<"Do you want to continue entering data (y/n)?"; char u;cin>>u;

if(u=='n'){break;}

}

cout<<"Enter time : "; cin>>obj[no-1]->time;

obj[no-1]->get\_health();

obj[no-1]->get\_symp();

obj[no-1]->turn++;

obj[no-1]->addto\_file();

}

catch(int y){

cout<<"Enter valid record number !. Error :"<<y<<endl;

cout <<"Total registered records are "<<temp<<endl;

}

break;

case 3: cout << "Press\n1.See individual record.\n2.See critical record.\n3.People out of location"<<endl;

int k;

cin>>k;

switch(k){

case 1:

{

try{

cout<<"Enter number of Person's record you want to see (eg. 1,2,3..): ";

cin>>no;

if(no<1 || no>cnt) {

throw 99;

}

obj[no-1]->show\_details();

}

catch(int y){

cout<<"Enter valid record number !. Error :"<<y<<endl;

cout <<"Total registered records are "<<temp<<endl;

}

break;

}

case 2:

{ if(cnt<1)

{

cout<<"There are no records in the system !"<<endl;

break;

}

else {

obj[cnt-1]->display\_critical();

}

break;

}

case 3:{

obj[cnt-1]->show\_location();

break;

}

}

break;

case 4: {

try{

cout<<"Enter the number of the record you want to delete : ";

cin >>no;

if(no<1 || no>cnt)

{

throw 99;

}

ofstream File(obj[no-1]->filename);

char c[20];

strcpy(c,obj[no-1]->filename);

File.close();

delete obj[no-1];

if( remove(c) != 0 )

perror( "Error deleting file" );

else{

if (no!=cnt)

{arr.push(no);}

cout<<"Record number "<<no<<" and file "<<c<<" deleted"<<endl;

temp--;

}

if(no==cnt)

cnt--;

}

catch(int y){

cout<<"Enter valid record number !. Error :"<<y<<endl;

cout <<"Total registered records are "<<temp<<endl;

}

break;

}

case 5: return 0;

}

getch();

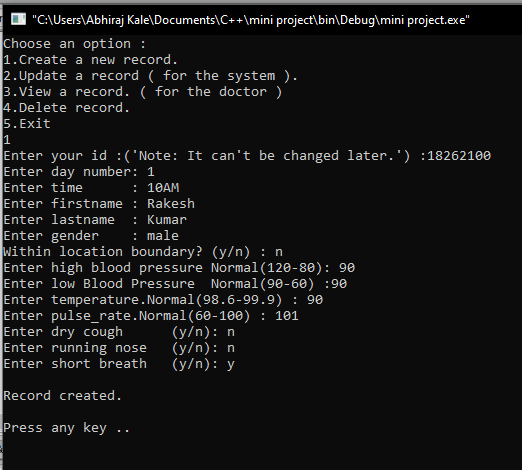
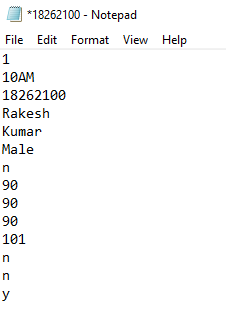
cout<<"\nPress any key .."<<endl;

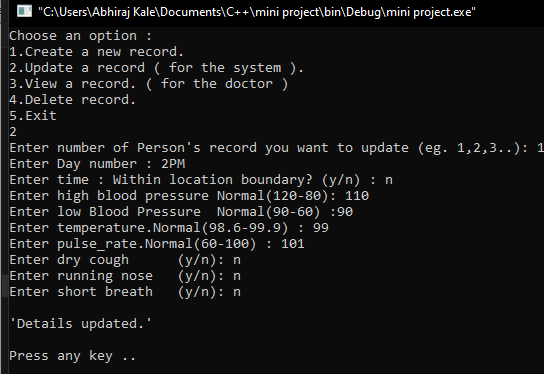
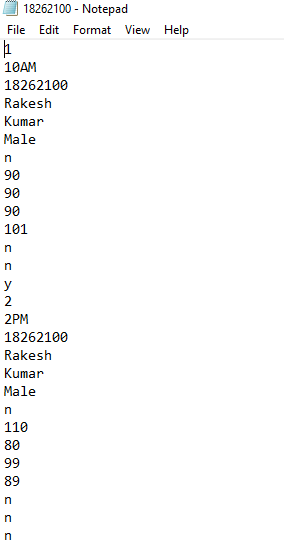
}

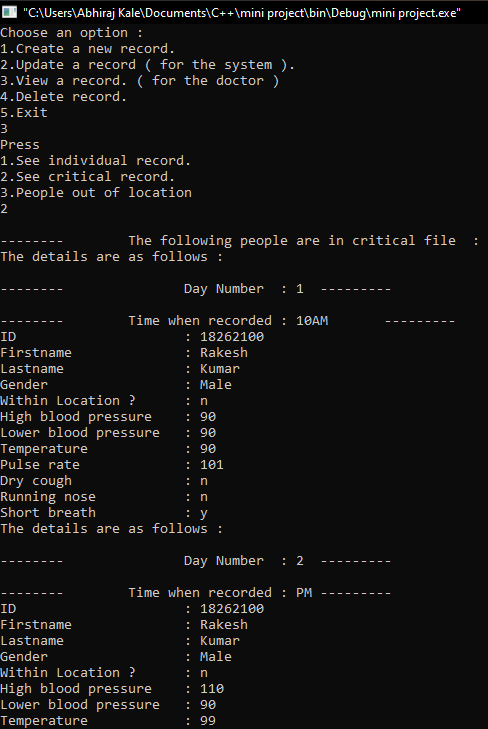
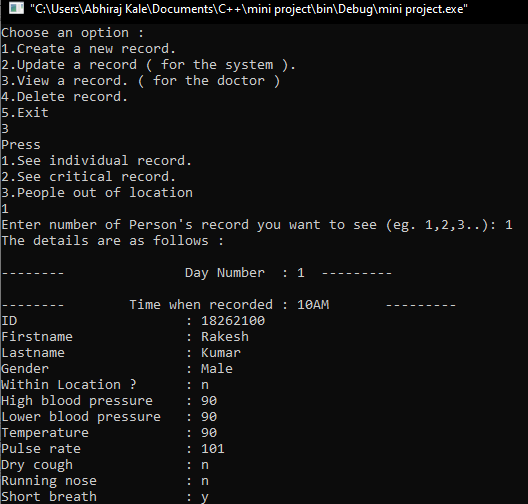
return 0;

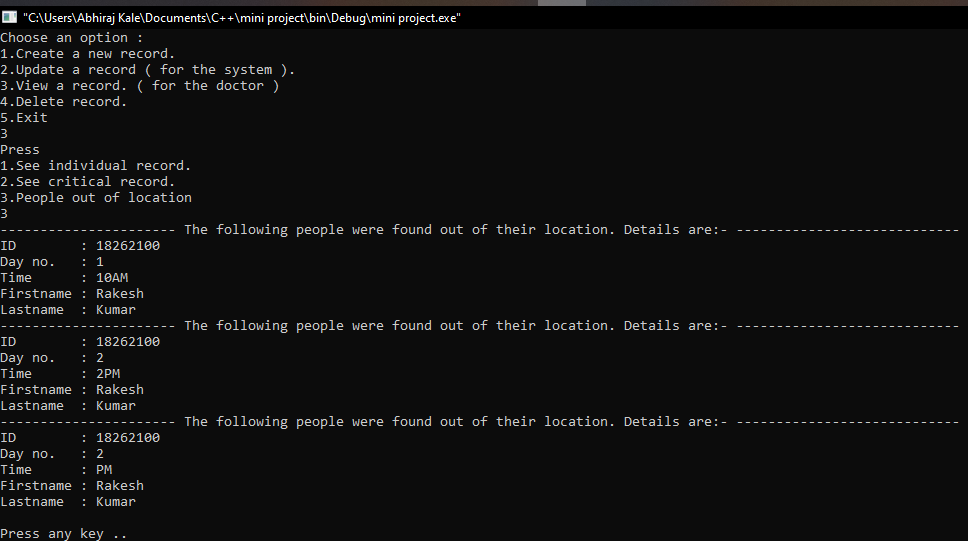
}

1. **OUTPUT:**
2. **Creating a record**

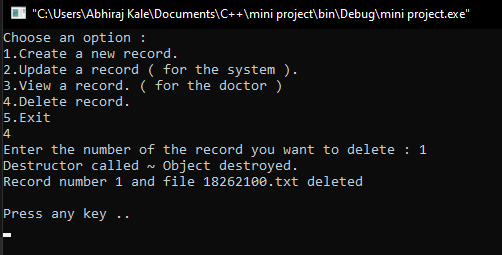
**** ****

1. **Updating a record** **** 
2. **Viewing a record**



****

1. **Deleting a record**



1. **REFERENCES :**

* Ms. Khushboo Sathawane (Mini - Project Guide).
* Object Oriented Programming in C++ - McGraw Hill
* Colleagues and Elders (IT Proffesionals).
* Articles on Internet and Videos on YouTube.

1. **Conclusions:**

Thus, in this way we have successfully developed **“Covid-19 Self Isolation Monitoring System”**, which is a constructive solution to confront the Covid-19 Pandemic. We applied almost all the concepts in Object Oriented Programming in C++ [Subject CM-388)].