***Object Oriented Program***

***Final Project***



**Submitted to: Madam Erum Ashraf**

**Submitted by**: **Saad Atif Enrolment #** **01-134212-156**

**Submitted by**: **Qazi Muhammad Usman Enrolment #** **01-134212-149**

**Submitted by**: **Rohan Ahmed Enrolment #** **01-134212-153**

**Department of Computer Science**

**Bahria University Islamabad**

**Project Report:**

Designed and developed an application to automate a **Patient Management System**. The detail of the application is written below

* Created a class patient which has data members patient name, age, address, and patient status. A patient status could be regular or corona virus.
* Created a child class ‘Corona Patient’ which has data members immunity level, symptoms, and city name.
* Created another child class ‘Regular Patient’ which has data members symptoms, disease name and city name.
* A display function which can show corona or regular patient record depending on the user choice.
* A file handler class which will contain all the modules of add, search, update and delete record of the patient in the text file. Copy and paste the code that is available in the sample project, now your task is to alter and connect the code to the respective classes.

**Source Code:**

#include<iostream>

#include<fstream>

#include<string>

#include<cstring>

#include<iomanip>

using namespace std;

#define MIN\_IMMUNITY\_LEVEL(10); //minimum immunity level, below this value, patient would be

// considered as corona patient

class Error

{

int b;

string msg;

public:

Error(string a, int b)

{

msg = a;

this->b = b;

}

void display()

{

cout << b<<" " << msg;

}

};

class Patient // ptient class with some basic attributes

{

public:

char PatientName[30];

char Age[10];

char Address[30];

char Status[10];

Patient() //default constructor

{

strcpy\_s(PatientName, "");

strcpy\_s(Age, "");

strcpy\_s(Address, "");

strcpy\_s(Status, "");

}

void addRegularPatient(char statusPatient[]) // Function for the regular patient

// Regular patient would have some known diseases

{

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out); //text file is made

//would add the record of

//patient in the file

if (!patient) //to check if file exists or not

{

cout << "File Opening Error:" << endl;

exit(1);

}

cin.ignore();

cout << "\t\t\t\t\tEnter Patient Name: ";

cin.getline(PatientName, 30);

cin.getline(PatientName, 30);

cout << "\t\t\t\t\tEnter Patient Age: ";

cin.getline(Age, 10);

cout << "\t\t\t\t\tEnter Address: ";

cin.getline(Address, 30);

strcpy\_s(Status, statusPatient);

patient << PatientName << " " << Age << " " << Address << " " << Status;

patient.close(); //file is being closed here and data would get saved in the file

}

};

class CoronaPatient : public Patient // inherited class from patient

// corona patient class with some extra attributes

// and having just corona patients

{

public:

char ImmunityLevel[30];

char Symptoms[30];

char CityName[30];

CoronaPatient()

{

strcpy\_s(ImmunityLevel, "");

strcpy\_s(Symptoms, "");

strcpy\_s(CityName, "");

}

};

class RegularPatient : public Patient //this is inherited class from patient

//this class would have just regular patients

{

public:

char Symptoms[30];

char DiseaseName[30];

char CityName[30];

RegularPatient()

{

strcpy\_s(DiseaseName, "");

strcpy\_s(Symptoms, "");

strcpy\_s(CityName, "");

}

};

// FileHandle class inherited from all three previous classes

//All the data of three classes would get saved in ths file

// Regular patients would have separate data saved

//Corona patients would have separate data saved

class fileHandle :public CoronaPatient, public RegularPatient, public Patient

{

public:

void AddPatient(int i)

{

cout << "\n\n\n";

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out);

if (!patient)

{

cout << "File opening error:" << endl;

exit(1);

}

Patient pat;

char patientstatus[10];

if (i == 1)

{

RegularPatient RP; //regular patients all data would get saved in the file

strcpy\_s(patientstatus, "Regular");

pat.addRegularPatient(patientstatus);

cin.ignore();

cout << "\t\t\t\t\tEnter Symptom (Tiredness,Cough,Fever): ";

cin.getline(RP.Symptoms, 30);

cout << "\t\t\t\t\tEnter Diseases Name: ";

cin.getline(RP.DiseaseName, 30);

cout << "\t\t\t\t\tEnter City Name: ";

cin.getline(RP.CityName, 30);

patient << " " << RP.Symptoms << " " << RP.DiseaseName << " " << RP.CityName << endl;

}

else if (i == 2)

{

CoronaPatient CP; // Corona patients all data would get saved in the file

strcpy\_s(patientstatus, "Corona");

pat.addRegularPatient(patientstatus);

cin.ignore();

cout << "\t\t\t\t\tEnter Symptom(Tiredness,Cough,Fever): ";

cin.getline(CP.Symptoms, 30);

h:

cout << "\t\t\t\t\tEnter Immunity Level (BELOW 10): ";

cin.getline(CP.ImmunityLevel, 30);

int immunity = atoi(CP.ImmunityLevel);

if (immunity >= MIN\_IMMUNITY\_LEVEL)

{

cout << "\n\t\t\t\t\tEnter Invalid Immunity\n";

goto h;

}

cout << "\t\t\t\t\tEnter City Name: ";

cin.getline(CP.CityName, 30);

patient << " " << CP.Symptoms << " " << CP.ImmunityLevel << " " << CP.CityName << endl;

}

patient.close();

cout << "\n\n\t\t\t\t\tRecord Added Successfully\n";

//system("pause");

cout << "\n\n";

}

void DeletePatientRecord() //deleting the record of the patient from the file

//Record of the Regular patients would be deleted separately

//Record of the Corona patients would be deleted separately

{

char NameP[30];

cin.ignore();

cout << "\t\t\t\t\tEnter Patient Name: ";

cin.getline(NameP, 30);

cout << "\t\t\t\t\t1 - CoronaPatient\t2 - RegularPatient: ";

int t;

cout << "\n\t\t\t\t\tSelect Option: ";

cin >> t;

Patient p;

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out);

if (!patient)

{

cout << "File Opening Error:" << endl;

exit(1);

}

fstream temp("Temp.txt", ios::app); //temporary file is created to put data temporary

//while deleting

if (!temp)

{

cout << "File opening error:" << endl;

exit(1);

}

int a = 0;

if (t == 1)

{

CoronaPatient CP; //Corona patient data would be deleted

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> CP.Symptoms >>CP. ImmunityLevel >> CP.CityName)

{

if ((strcmp(p.PatientName, NameP) == 0) && (strcmp(p.Status, "Corona") == 0))

{

a = 1;

cout << "\n\t\t\t\t\tRecord Successfully Deleted\n";

}

else

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << CP.Symptoms << " " << CP.ImmunityLevel << " " << CP.CityName << endl;

}

}

else if (t == 2)

{

RegularPatient RP; //Regular patient data would be deleted

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> RP.Symptoms >> RP.DiseaseName >> RP.CityName)

{

if (strcmp(p.PatientName, NameP) == 0 && strcmp(p.Status, "Regular") == 0)

{

a = 1;

cout << "\n\t\t\t\t\tRecord Successfully Deleted\n";

}

else

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << RP.Symptoms << " " << RP.DiseaseName << " " << RP.CityName << endl;

}

}

temp.close();

patient.close();

remove("PATIENT.txt");

rename("Temp.txt", "PATIENT.txt");

if (a == 0)

{

cout << "\n\t\t\t\t\tRecord Not Found\n";

}

system("pause");

}

int Menus()

{

system("cls");

cout << "\n\n\t\t\t>>>>>Press -1 for Display All Corona Patient<<<<<\n";

cout << "\t\t\t>>>>>Press -2 for Display All Regular Patient<<<<<\n";

cout << "\t\t\t>>>>>Press -3 for Search Regular Patient By Name<<<<<\n";

cout << "\t\t\t>>>>>Press -4 for Search Corona Patient By Name<<<<<<\n";

cout << "\t\t\t>>>>>Press -5 Quit application<<<<<<<<<<<<<<<<<<<<<<<<\n";

int option;

cout << "\n\n\t\t\tEnter Your Choice: ";

cin >> option;

return option;

}

void displayAllCoronaPatients() //Displaying all Corona patient data which would be in the file

{

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out);

if (!patient)

{

cout << "File opening error:" << endl;

exit(1);

}

cin.ignore();

cout << "\n\n\n";

Patient p;

CoronaPatient CP;

int a = 0;

cout << setiosflags(ios::left) << setw(30) << "Name" << setw(10) << "Age" << setw(30) << "Address" << setw(15) << "Status" << setw(30) << "Symptoms" << setw(15) << "Imm. Level" << setw(15) << "City\n\n" << endl;

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> CP.Symptoms >> CP.ImmunityLevel >> CP.CityName)

{

if (strcmp(p.Status, "Corona") == 0)

{

a = 1;

cout << setiosflags(ios::left) << setw(30) << p.PatientName << setw(10) << p.Age << setw(30) << p.Address << setw(15) << p.Status << setw(30) << CP.Symptoms << setw(15) << CP.ImmunityLevel << setw(15) << CP.CityName << endl << endl;

}

}

patient.close();

if (a == 0)

{

cout << "\n\t\t\tRecord Not Found\n";

}

system("pause");

}

void displayAllRegularPatients() //All regular patient data would be displayed from file

{

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out);

if (!patient)

{

cout << "File opening error:" << endl;

exit(1);

}

cin.ignore();

int a = 0;

Patient p;

RegularPatient RP;

cout << "\n\n\n";

cout << setiosflags(ios::left) << setw(30) << "Name" << setw(10) << "Age" << setw(30) << "Address" << setw(15) << "Status" << setw(30) << "Symptoms" << setw(30) << "Disease" << setw(15) << "City\n\n" << endl;

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> RP.Symptoms >> RP.DiseaseName >> RP.CityName)

{

if (strcmp(p.Status, "Regular") == 0)

{

a = 1;

cout << setiosflags(ios::left) << setw(30) << p.PatientName << setw(10) << p.Age << setw(30) << p.Address << setw(15) << p.Status << setw(30) << RP.Symptoms << setw(30) << RP.DiseaseName << setw(15) << RP.CityName << endl << endl;

}

}

patient.close();

if (a == 0)

{

cout << "\n\t\t\tRecord Not Found\n";

}

system("pause");

}

void RegulardisplayByName() // Displaying the record of the Regular patients by their name

{

cin.ignore();

char RName[30];

cout << "Enter Patient Name: ";

cin.getline(RName, 30);

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out);

if (!patient)

{

cout << "File opening error:" << endl;

exit(1);

}

int a = 0;

Patient p;

RegularPatient RP;

cout << "\n\n\n";

cout << setiosflags(ios::left) << setw(30) << "Name" << setw(10) << "Age" << setw(30) << "Address" << setw(15) << "Status" << setw(15) << "Symptoms" << setw(30) << "Disease" << setw(15) << "City\n\n" << endl;

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> RP.Symptoms >> RP.DiseaseName >> RP.CityName)

{

if (strcmp(p.Status, "Regular") == 0 && strcmp(p.PatientName, RName) == 0)

{

cout << setiosflags(ios::left) << setw(30) << p.PatientName << setw(10) << p.Age << setw(30) << p.Address << setw(15) << p.Status << setw(15) << RP.Symptoms << setw(30) << RP.DiseaseName << setw(15) << RP.CityName << endl << endl;

a = 1;

}

}

patient.close();

if (a == 0)

{

cout << "\n\n\t\t\tRecord Not Found\n";

}

system("pause");

}

void CoronaDisplayByName() //Displaying the record of the Corona patients by their name

{

cin.ignore();

char CName[30];

cout << "Enter the patient name: ";

cin.getline(CName, 30);

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out);

if (!patient)

{

cout << "File opening error:" << endl;

exit(1);

}

int a = 0;

cout << "\n\n\n";

Patient p;

CoronaPatient CP;

cout << setiosflags(ios::left) << setw(30) << "Name" << setw(10) << "Age" << setw(30) << "Address" << setw(15) << "Status" << setw(15) << "Symptoms" << setw(10) << "Imm. Level" << setw(15) << "City\n\n" << endl;

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> CP.Symptoms >> CP.ImmunityLevel >> CP.CityName)

{

if (strcmp(p.Status, "Corona") == 0 && strcmp(p.PatientName, CName) == 0)

{

a = 1;

cout << setiosflags(ios::left) << setw(30) << p.PatientName << setw(10) << p.Age << setw(30) << p.Address << setw(15) << p.Status << setw(15) << CP.Symptoms << setw(10) << CP.ImmunityLevel << setw(15) << CP.CityName << endl << endl;

}

}

if (a == 0)

{

cout << "\n\t\t\tRecord Not Found\n";

}

patient.close();

system("pause");

}

void search() //To search the data of both the categories patients from the record

{

int Option;

while ((Option = Menus()) != 5)

{

if (Option == 1)

{

displayAllCoronaPatients();

}

if (Option == 2)

{

displayAllRegularPatients();

}

if (Option == 3)

{

RegulardisplayByName();

}

if (Option == 4)

{

CoronaDisplayByName();

}

if (Option < 1 || Option > 5)

{

cout << "\n\nIncorrect Choice\n";

}

system("pause");

}

}

void updatePatientRecord() //To updatae the record of any patient

//if any corona patient gets nagative his record would be modified

//if any of regular patient would get well his record would get modified

{

cin.ignore();

char Name[30];

cout << "Enter Patient Name: ";

cin.getline(Name, 30);

int option;

top:

cout << "Select Option: (1 - Regular \t 2 - Corona Patient): ";

cin >> option;

if (option > 2 || option < 1)

{

cout << "\nInvalid Number\n";

goto top;

}

fstream patient("PATIENT.txt", ios::app | ios::in | ios::out);

if (!patient)

{

cout << "File opening error:" << endl;

exit(1);

}

fstream temp("Temp.txt", ios::app);

if (!temp)

{

cout << "File opening error:" << endl;

exit(1);

}

int a = 0;

Patient p;

CoronaPatient CP;

RegularPatient RP;

cout << "\n\n";

if (option == 1)

{

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> RP.Symptoms >> RP.DiseaseName >> RP.CityName)

{

if (strcmp(p.Status, "Regular") == 0 && strcmp(p.PatientName, Name) == 0)

{

a = 1;

cout << setiosflags(ios::left) << setw(30) << "Name" << setw(10) << "Age" << setw(30) << "Address" << setw(15) << "Status" << setw(15) << "Symptoms" << setw(15) << "Disease" << setw(15) << "City\n\n" << endl;

cout << setiosflags(ios::left) << setw(30) << p.PatientName << setw(10) << p.Age << setw(30) << p.Address << setw(15) << p.Status << setw(15) << RP.Symptoms << setw(15) << RP.DiseaseName << setw(15) << RP.CityName << endl << endl;

above:

cout << "Enter Patient Status(1 - Regular\t 2 - Corona Patient): ";

int op;

cin >> op;

if (op > 2 || op < 1)

{

cout << "\nInvalid Number\n";

goto above;

}

if (op == 1)

{

cin.ignore();

strcpy\_s(p.Status, "Regular");

cout << "Enter Disease Name: ";

cin.getline(RP.DiseaseName, 30);

cout << "Enter Symptoms(Fever,Cough, Tiredness): ";

cin.getline(RP.Symptoms, 30);

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << RP.Symptoms << " " << RP.DiseaseName << " " << RP.CityName << endl;

}

else if (op == 2)

{

cin.ignore();

strcpy\_s(p.Status, "Corona");

cout << "Enter Immunity Level(BELOW 10): ";

cin.getline(CP.ImmunityLevel, 30);

cout << "Enter Symptoms(Fever,Cough, Tiredness): ";

cin.getline(CP.Symptoms, 30);

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << CP.Symptoms << " " << CP.ImmunityLevel << " " << RP.CityName << endl;

}

}

else

{

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << RP.Symptoms << " " << RP.DiseaseName << " " << RP.CityName << endl;

}

}

}

else if (option == 2)

{

while (patient >> p.PatientName >> p.Age >> p.Address >> p.Status >> CP.Symptoms >> CP.ImmunityLevel >> CP.CityName)

{

if (strcmp(p.Status, "Corona") == 0 && strcmp(p.PatientName, Name) == 0)

{

a = 1;

cout << setiosflags(ios::left) << setw(30) << "Name" << setw(10) << "Age" << setw(30) << "Address" << setw(15) << "Status" << setw(15) << "Symptoms" << setw(10) << "Imm. Level" << setw(15) << "City\n\n" << endl;

cout << setiosflags(ios::left) << setw(30) << p.PatientName << setw(10) << p.Age << setw(30) << p.Address << setw(15) << p.Status << setw(15) << CP.Symptoms << setw(10) << CP.ImmunityLevel << setw(15) << CP.CityName << endl << endl;

abov:

cout << "Enter Patient Status(1- Regular \t 2 - Corona Patient): ";

int op;

cin >> op;

if (op > 2 || op < 1)

{

cout << "\nInvalid Number\n";

goto abov;

}

if (op == 1)

{

cin.ignore();

strcpy\_s(p.Status, "Regular");

cout << "Enter Disease Name: ";

cin.getline(RP.DiseaseName, 30);

cout << "Enter Symptoms(Faver,Cough, Tiredness): ";

cin.getline(RP.Symptoms, 30);

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << RP.Symptoms << " " << RP.DiseaseName << " " << CP.CityName << endl;

}

else if (op == 2)

{

cin.ignore();

strcpy\_s(p.Status, "Corona");

cout << "Enter Immunity Level(BELOW 10): ";

cin.getline(CP.ImmunityLevel, 30);

cout << "Enter Symptoms(Faver,Cough, Tiredness): ";

cin.getline(CP.Symptoms, 30);

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << CP.Symptoms << " " << CP.ImmunityLevel << " " << CP.CityName << endl;

}

}

else

{

temp << p.PatientName << " " << p.Age << " " << p.Address << " " << p.Status << " " << CP.Symptoms << " " << CP.ImmunityLevel << " " << CP.CityName << endl;

}

}

}

patient.close();

temp.close();

remove("PATIENT.txt");

rename("Temp.txt", "PATIENT.txt");

if (a == 0)

{

cout << "\n\t\t\tRecord Not Found\n";

}

else

cout << "\n\t\t\tRecord Updated Successfully\n\n\n";

system("pause");

}

};

int Menu() //Whole menu driven function of the application made

{

system("cls");

cout << "\n\n\t\t\t>>>>>Press - 1 For Enter New Regular Patient<<<<<\n";

cout << "\t\t\t>>>>>Press - 2 For Enter New Corona Patient<<<<<<\n";

cout << "\t\t\t>>>>>Press - 3 For Update Patient Record<<<<<<<<<\n";

cout << "\t\t\t>>>>>Press - 4 For Delete Patient Record<<<<<<<<<\n";

cout << "\t\t\t>>>>>Press - 5 For Searching/ Display Patient Data<<<<<\n";

cout << "\t\t\t>>>>>Press - 6 For Quit Application<<<<<<<<<<<<<<\n";

int Option;

cout << "\n\n\t\t\tEnter Your Choice: ";

cin >> Option;

if (Option<1 || Option > 6)

{

throw Error("Is invalid Option ", Option);

}

return Option;

}

bool Login()

{

cout << "\n\n\n\n\n\t\t\t\tWelcome to Patient Management System \n\n\n\n";

cout << "\n\t\t\tProject Members:\n\n";

cout << "\t\t\t\t\Saad Atif ---> 01-134212-156 \n\n";

cout << "\t\t\t\tQazi Muhammad Usman---> 01-134212-156 \n\n";

cout << "\t\t\t\tRohan Ahmed---> 01-134212-153 \n\n\n\n";

system("pause");

system("cls");

string str1, str2;

cout << "\n\n\n\n\t\t\tLogin\n\n";

cout << "\n\nTo Access the system please Enter Your Username and Password. \n\n";

cout << "\n\n\t\tUsername: ";

cin >> str1;

cout << "\n\n\t\tPassword: ";

cin >> str2;

cout << endl << endl << endl;

if (str1 == "User" && str2 == "1234")

{

cout << "Access Granted.\n\n";

system("pause");

system("cls");

return 1;

}

else

{

cout << "Access Denied......Good Bye\n\n";

system("pause");

return 0;

}

}

int main()

{

if (Login())

{

fileHandle FH; //object of the class made

int Option;

//Calling of all the functions starts from here

try {

while ((Option = Menu()) != 6)

{

if (Option == 1)

{

FH.AddPatient(1);

}

if (Option == 2)

{

FH.AddPatient(2);

}

if (Option == 3)

{

FH.updatePatientRecord();

}

if (Option == 4)

{

FH.DeletePatientRecord();

}

if (Option == 5)

{

FH.search();

}

system("pause");

}

}

catch (Error e)

{

e.display();

}

}

}

**Outputs:**

**Text

Description automatically generated** **Text

Description automatically generated** Text

Description automatically generated A screenshot of a computer

Description automatically generated with medium confidence Text

Description automatically generated