



**UNIVERSITATEA TEHNICĂ**  
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

# ADT CLINIC

Proiect Informatică Industrială

Studentii din grupa 30133 :

- + Team leader : Cighi Andreea - Maria
- + Developer : Bolba Denisa - Georgiana
- + Tester : Hinsă Tudor

**2022**



## Sinteza

proiectului cu titlul:

**«ADT Clinic»**

### 1. **Cerințele temei:**

Dezvoltati o aplicatie complexa in C# care sa contina o baza de date, un server si un client. Server-ul ofera acces la baza de date care stocheaza informatii, iar clientul utilizeaza metodele descrise in server.

### 2. **Soluții alese:**

Serverul este de tip Web Service, iar clientul este de tip Windows Form Application. Exista doua tipuri de utilizator, medic si pacient, fiecare avand o serie de optiuni.

### 3. **Rezultate obținute:**

Aplicatia “ADT Clinic” are ca scop administrarea activitatii unei clinici. Din punct de vedere structural, aplicatia are trei componente: server, client si baza de date.

Server-ul este de tip Web Service si contine metodele: loginMedic, registerMedic, getMedicData, changeMedicData, deleteMedicAccount, loginPacient, registerPacient, getPacientDetails, updatePacientData, deletePacient, programari, fisaMedicala, DoctorsfromSpecialization, listSpecialization, dataProgramarii, listPacienti, dateMedic, datePacient, problemaCurenta, listaPacientiProgramati, datePacient, antecedenteMedicale si problemaCurenta.

Baza de date contine 6 tabele: datamedic, FisaMedicala, MedicDetail, PacientDetail, Pacienti si Programare. Tabelele dataMedic si MedicDetail stocheaza informatii despre medicii din sistem, iar Pacienti si PacientDetail stocheaza informatii despre pacientii care apeleaza la serviciile clinicii. In tabelul Fisa medicala se regasesc diagnosticul si recomandarile medicului pentru pacienti, iar tabelul Programare contine datele personale ale pacientilor care isi fac programari, data programarii si simptomele acestora.

Clientul este de tip Windows Form Application si poate fi accesat de doua tipuri de utilizatori: pacienti si medici. Ambii utilizatori au posibilitatea de a se inregistra in sistem, de a se loga si de a accesa mai multe informatii precum datele personale, lista de



pacienti si lista de medici. De asemenea, isi pot sterge contul de utilizator. In plus, medicii isi pot vedea programul zilnic, iar pacientii pot efectua programari.

#### **4. Testări și verificări:**

Problema identificata in timpul testarii consta in posibilitatea crearii unui nou cont de utlizator fara a fi completate informatii necesare pentru logare.

Aceasta problema s-a rezolvat prin adaugarea unei conditii care sa verifice prezenta acestor informatii in Text Box-uri.



# Cuprins

<b>CAPITOLUL 1 .....</b>	<b>2</b>
<b>INTRODUCERE .....</b>	<b>2</b>
1.1    ENUNTUL PROBLEMEI .....	2
1.2    OBIECTIVE .....	2
1.3    SPECIFICATII .....	2
<b>CAPITOLUL 2 .....</b>	<b>3</b>
<b>ARHITECTURA APLICATIEI .....</b>	<b>3</b>
2.1. DIAGRAMA CLASELOR.....	3
2.2. DIAGRAMA USE-CASE .....	4
2.3. DIAGRAMA BAZEI DE DATE .....	4
2.4. DIAGRAMA DE ACTIVITATE.....	5
<b>CAPITOLUL 3 .....</b>	<b>5</b>
<b>IMPLEMENTAREA APLICATIEI .....</b>	<b>5</b>
<b>CAPITOLUL 4.....</b>	<b>26</b>
<b>TESTAREA APLICATIEI .....</b>	<b>26</b>
<b>ANEXA 1 (COD SURSA).....</b>	<b>27</b>

---

# Capitolul 1

## INTRODUCERE

### 1.1 Enuntul problemei

Dezvoltati o aplicatie complexa in C# care sa realizeze interogarea unei baze de date, permitand popularea si actualizarea acesteia. Nivelul logic va fi implementat in Web Service, iar nivelul de prezentare in format Windows Form Application.

### 1.2 Obiective

Obiectivul proiectului este de a crea o aplicatie care sa administreze activitatea unei clinici.

### 1.3 Specificatii

Prin intermediul aplicatiei, atat medicii cat si pacientii au posibilitatea de a se inregistra in sistem, dupa care se pot loga. In functie de tipul de utilizator, pacient sau medic, exista mai multe optiuni.

Funcțiile pe care medicul le poate realiza:

1. Vizualizarea datelor personale → numele de utilizator, parola, CNP, adresa de e-mail, numele si prenumele, adresa, data nasterii si specializarea;
2. Actualizarea datelor personale → parola, adresa de e-mail, numele si prenumele, adresa si specializarea;
3. Vizualizarea listei de pacienti → aceasta va contine toti pacientii inregistrati in sistem;
4. Vizualizarea listei de medici → aceasta va contine toti medicii inregistrati in sistem;
5. Stergerea contului de utilizator → dupa confirmarea stergerii contului, acesta va disparea din sistem;
6. Vizualizarea programului zilnic → la accesarea acestei optiuni, vor aparea toate programarile din ziua respectiva ale medicului;
7. Punerea diagnosticului pacientului si posibilitatea de a-i face recomandari in functie de acesta → pe baza problemei curente, dar si a celorlalte probleme ale pacientului, medicul va pune un diagnostic si va face recomandari;

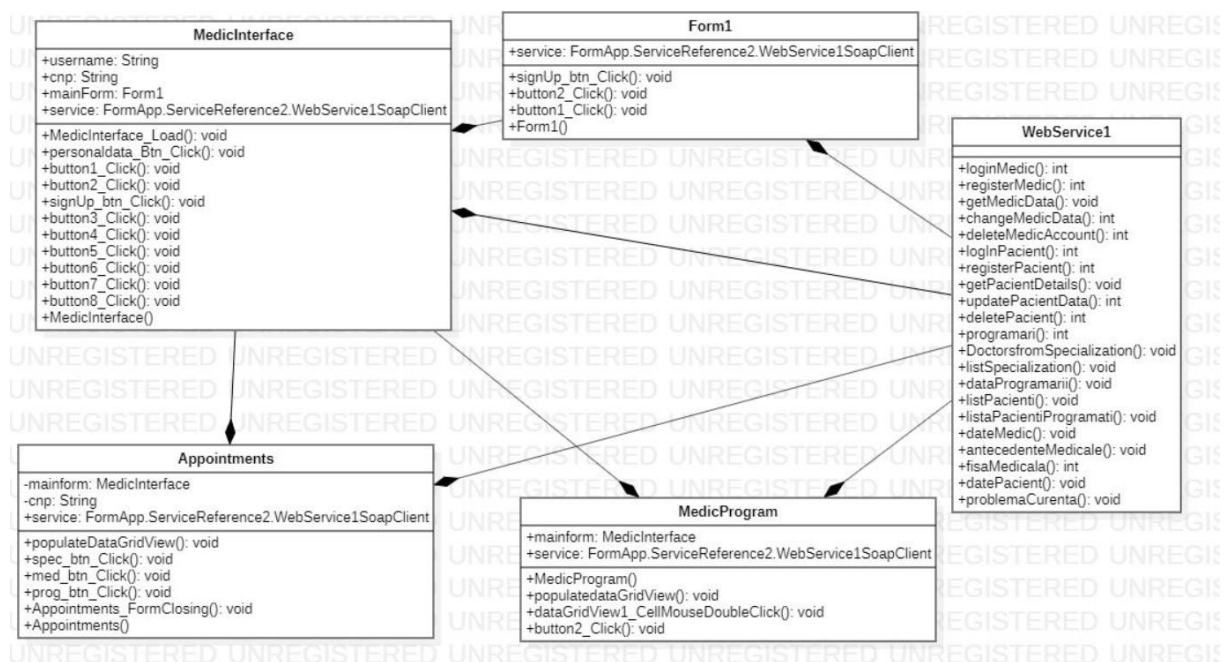
Funcțiile pe care pacientul le poate realiza:

1. Vizualizarea datelor personale → numele de utilizator, parola, CNP, adresa de e-mail, numele si prenumele, adresa si data nasterii;
2. Actualizarea datelor personale → parola, adresa de e-mail, numele si prenumele, adresa;
3. Vizualizarea listei de pacienti → aceasta va contine toti pacientii inregistrati in sistem;
4. Vizualizarea listei de medici → aceasta va contine tot medicii inregsitrati in sistem;
5. Stergerea contului de utilizator → dupa confirmarea stingerii contului, acesta va dispaea din sistem;
6. Efectuarea unor programari la anumiti medici, intr-un anumit interval orar → pacientul are posibilitatea de a vedea in DataGridView programarile din saptamana respectiva, ulterior alege specializarea medicului la care doreste sa faca programarea, alege medicul, descrie simptomele pe care le are si alege ziua si ora pe care le prefera pentru programare;

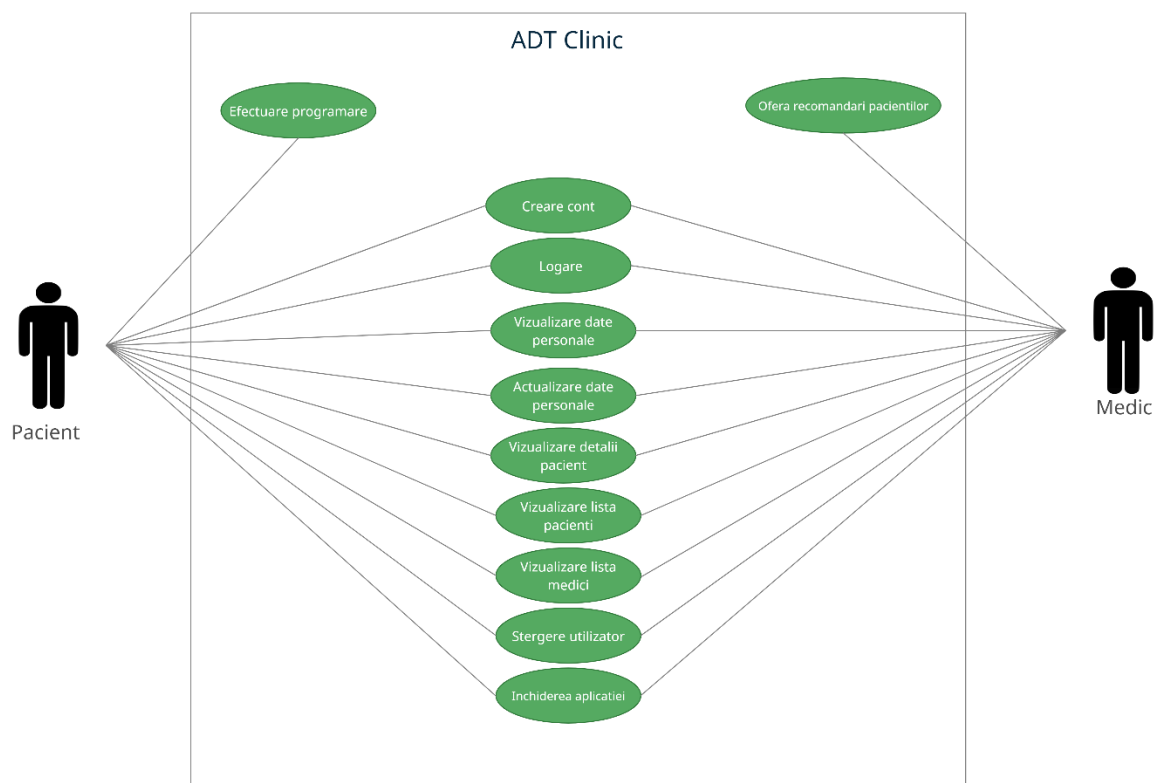
## Capitolul 2

# ARHITECTURA APLICATIEI

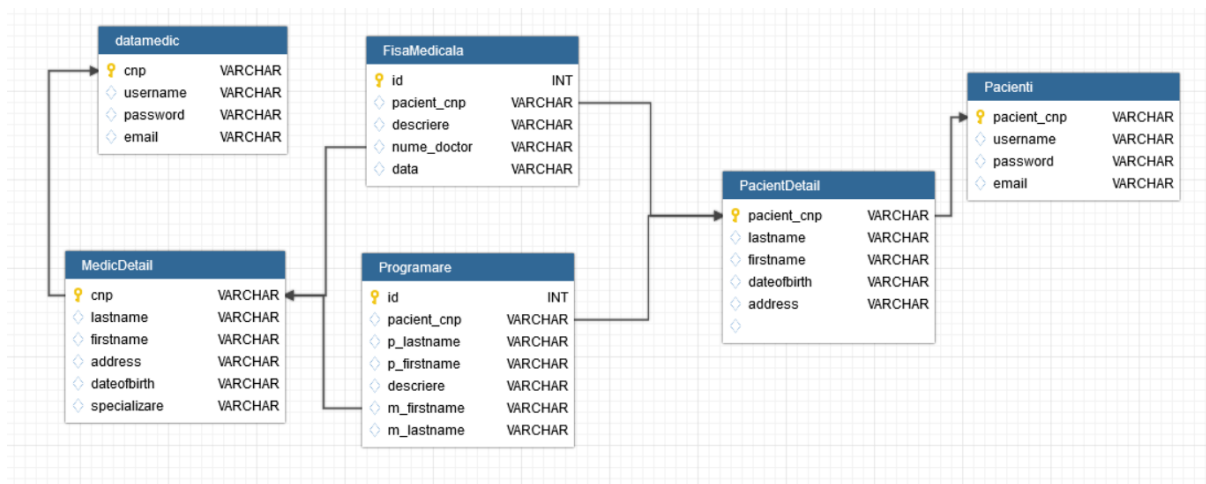
### 2.1. Diagrama claselor



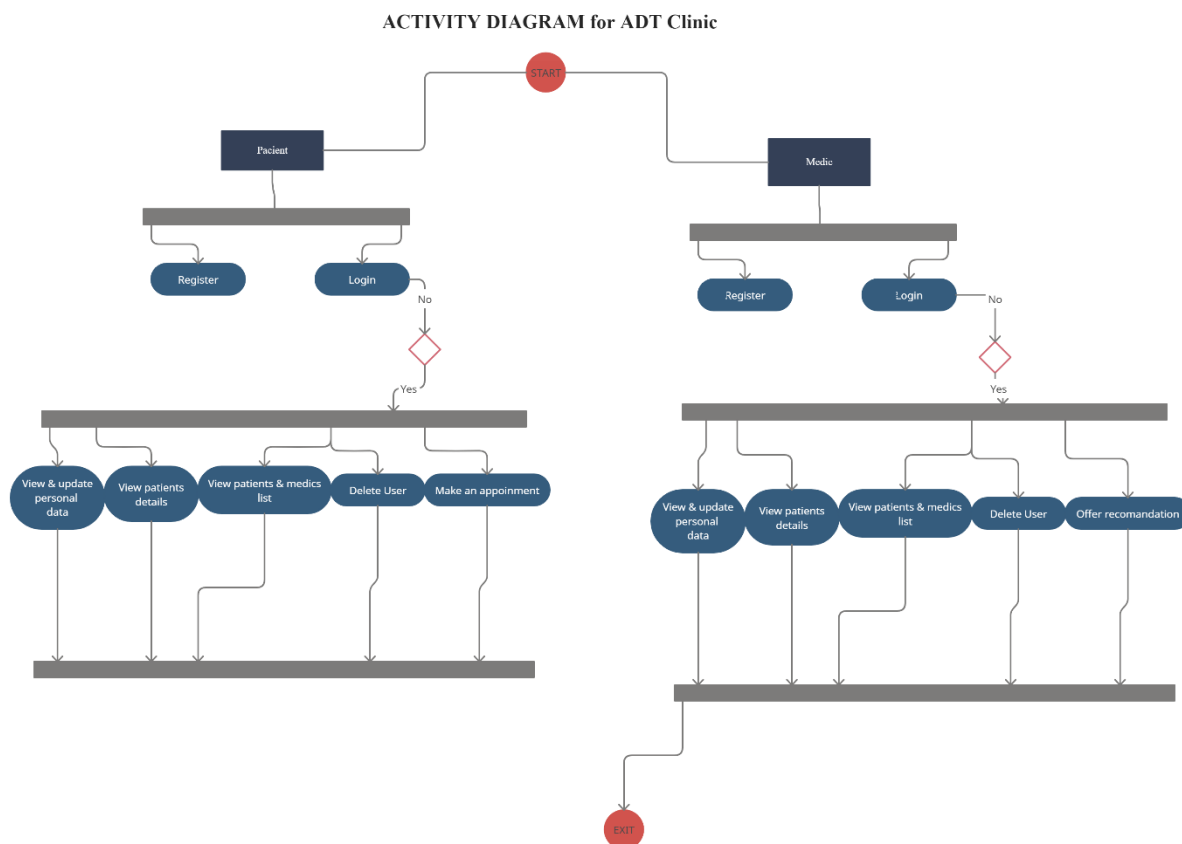
## 2.2. Diagrama use-case



## 2.3. Diagrama bazei de date



## 2.4. Diagrama de Activitate



## Capitolul 3

### IMPLEMENTAREA APLICATIEI

Aplicatia a fost dezvoltata in Visual Studio , folosind limbajul C#, iar baza de date a fost creata prin intermediul Sql Server.

In continuare, vor fi prezentate functionalitatile aplicatiei.

Pagina principala a aplicatiei este urmatoarea:



WELCOME TO ADT Clinic !



Login

Nume de utilizator:

Parolă:

CNP:

☐ Medic ☐ Pacient



ADT app by Bolba, Cighi, Hinsa

În calitate de pacient, se pot efectua următoarele operațiuni:

- Înregistrarea în sistem:
  - Primul pas este de a selecta Radio Button-ul “Pacient”, apoi se apasă butonul Înregistrare

WELCOME TO ADT Clinic !



Login

Nume de utilizator:

Parolă:

CNP:

☐ Medic ☒ Pacient



ADT app by Bolba, Cighi, Hinsa

- Apoi se introduc datele in Text Box-uri, iar la final se apasa butonul de Inregistrare

Form1

WELCOME TO AD7 Clinic !



Inregistrare

Nume de utilizator: florea\_adriana

Parolă: \*\*\*\*\*

CNP: 2890918193995

Adresă de e-mail: florea.adriana@gmail.com

Nume: Florea

Prenume: Adriana

Adresă: Strada Fagetului Nr.6

Data nașterii: 18.09.1989

Inregistrare



ADT app by Bolba, Cighi, Hinsu

- Logarea in sistem:
- Se selecteaza Radio Buttonu-ul “Pacient” si se introduc datele in Text Box-uri

Form1

WELCOME TO AD7 Clinic !



Login

Nume de utilizator: florea\_adriana

Parolă: \*\*\*\*\*

CNP: 2890918193995

☐ Medic ☒ Pacient

Login Inregistrare

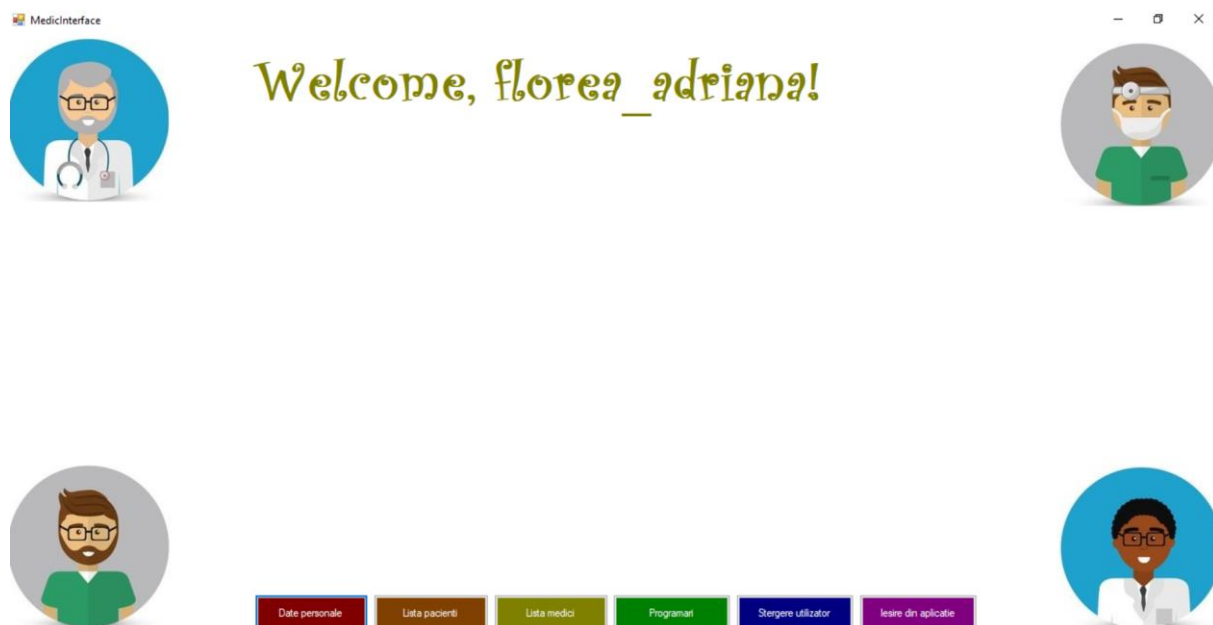
Bine ati venit!

OK



ADT app by Bolba, Cighi, Hinsu

- Se apasa butonul Login si se deschide un alt form care ii va oferi pacientului mai multe optiuni

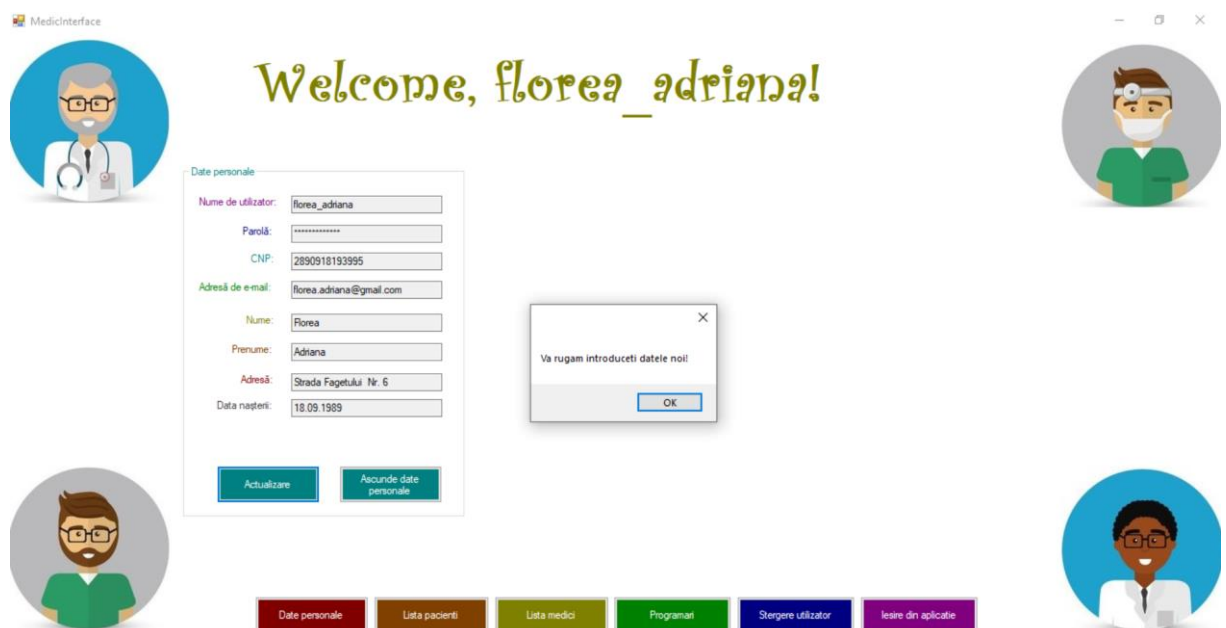


➤ Vizualizarea datelor persoanele:

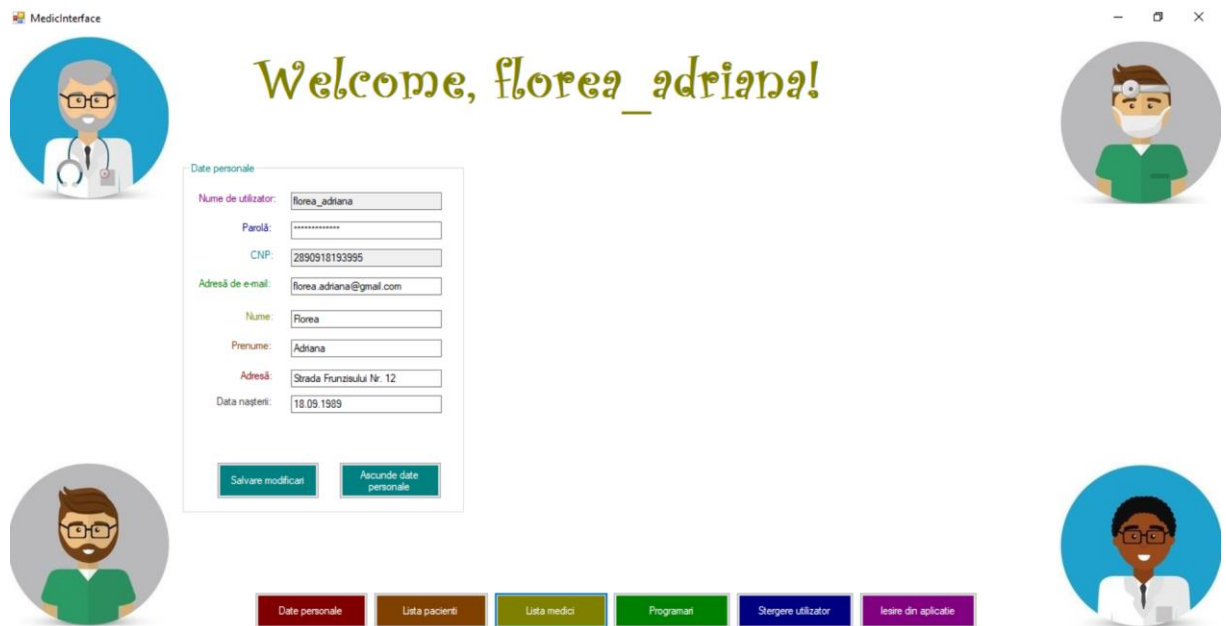
- Se apasa butonul Date personale



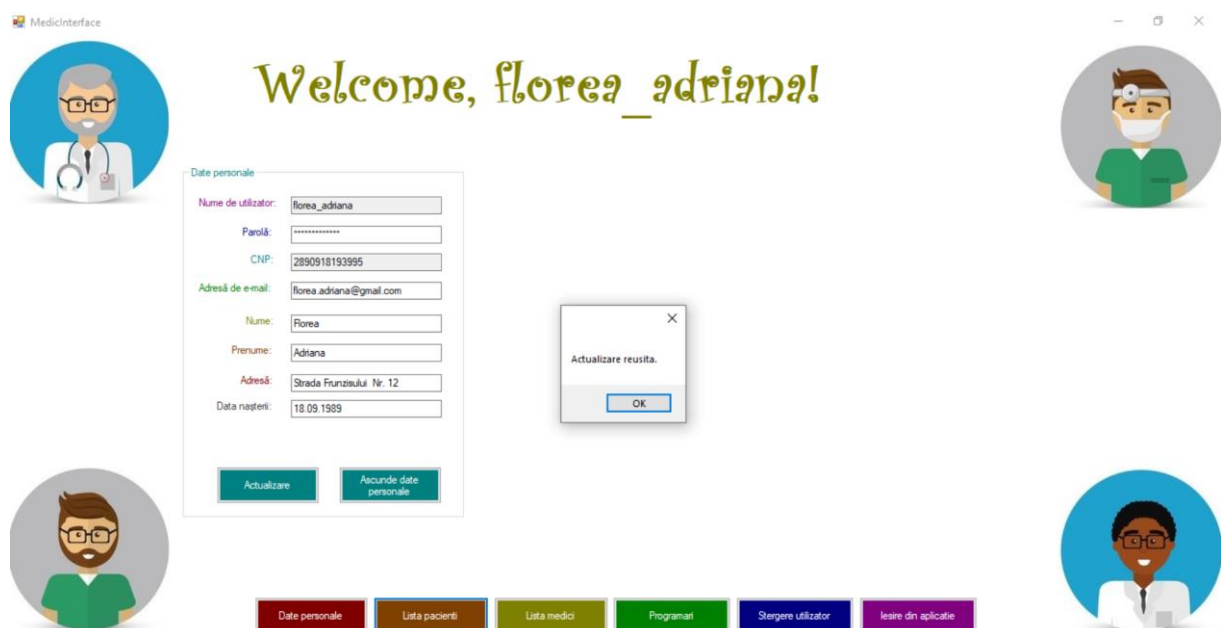
- Actualizarea datelor personale:
  - Se apasa butonul Actualizare din Group Box-ul Date Personale



- Se modifica datele dorite(in acest caz, Adresa)



- Se apasa butonul Salvare modificari



- Vizualizarea listei de pacienti:
  - Se apasa butonul Lista Pacienti



- Vizualizarea listei de medici:
  - Se apasa butonul Lista Medici:



- Efectuarea programarilor:
  - Se apasa butonul Programari si se deschide un nou form



Appointments

ora	Luni	Marti	Miercuri	Joi	Vineri
8:9		dr. Nistor StefanaPaci...			
9:10					
10:11			dr. Pavel GabrielaPaci...		
11:12					dr. Constantin MihaP...
12:13	dr. Iorga DoruPacient...				
13:14					
14:15				dr. Ciorteanu Roxana...	
15:16					

May 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Today: 5/24/2020

Simptome:

Data(Zina hh1:hh2):

Programare

Specializari

- Se apasa butonul Specializari si se selecteaza din List Box specializarea medicului la care se va efectua programarea

Appointments

ora	Luni	Marti	Miercuri	Joi	Vineri
8:9		dr. Nistor StefanaPaci...			
9:10					
10:11			dr. Pavel GabrielaPaci...		
11:12					dr. Constantin MihaP...
12:13	dr. Iorga DoruPacient...				
13:14					
14:15				dr. Ciorteanu Roxana...	
15:16					

Specializari

Oftamologie

Pneumologie

Endocrinologie

Gastroenterologie

Cardiologie

Ortopedie

Psihiatrie

Diabet zaharat, nutritie si boli m...

Neurologie

Specializari

Medici

May 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Today: 5/24/2020

Simptome:

Data(Zina hh1:hh2):

Programare

- Se apasa butonul Medici si se selecteaza din List Box medicul dorit

Appointments

ora	Luni	Marti	Miercuri	Joi	Vineri
8:9		dr. Nistor StefanaPaci...			
9:10			dr. Pavel GabrielaPaci...		
10:11					dr. Constantin MihaP...
11:12					
12:13	dr. Iorga DoruPacient...				
13:14					
14:15				dr. Corteanu Roxana...	
15:16					

HOSPITAL

Lista Medici

Pavel Bianca

Specializari

Medici

May 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Today: 5/24/2020

Simptome:

Data(Ziua hh1:hh2):

Programare

- Se introduc datele in Text Box-urile Simptome si Data(Ziua hh1:hh2)

Appointments

ora	Luni	Marti	Miercuri	Joi	Vineri
8:9		dr. Nistor StefanaPaci...			
9:10			dr. Pavel GabrielaPaci...		
10:11					dr. Constantin MihaP...
11:12					
12:13	dr. Iorga DoruPacient...				
13:14					
14:15				dr. Corteanu Roxana...	
15:16					

HOSPITAL

Lista Medici

Pavel Gabriela

Specializari

Medici

May 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Today: 5/24/2020

Simptome:

Data(Ziua hh1:hh2):

Programare

- Se apasa butonul Programare



Appointments

ora	Luni	Marti	Miercuri	Joi	Vineri
8:9		dr. Nistor StefanaPaci...			
9:10			dr. Pavel GabrielaPaci...		
10:11					dr. Constantin MihaP...
11:12					
12:13	dr. Iorga DoruPacient...				
13:14					
14:15				dr. Corteanu Roxana...	
15:16					

HOSPITAL

Lista Medici

Pavel Gabriela

Programarea a fost realizata cu succes!

OK

May 2020

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Today: 5/24/2020

Symptoms: a mobilitati piciorului umflatura

Data(Ziua hh1:hh2): Joi 9:10

Programare

Specializari

Medici

- o DataGridView-ul se va actualiza cu programarea efectuata

Appointments

ora	Luni	Marti	Miercuri	Joi	Vineri
8:9		dr. Nistor StefanaPaci...		dr. Pavel GabrielaPaci...	
9:10			dr. Pavel GabrielaPaci...		
10:11					dr. Constantin MihaP...
11:12					
12:13	dr. Iorga DoruPacient...				
13:14					
14:15				dr. Corteanu Roxana...	
15:16					

HOSPITAL

Lista Medici

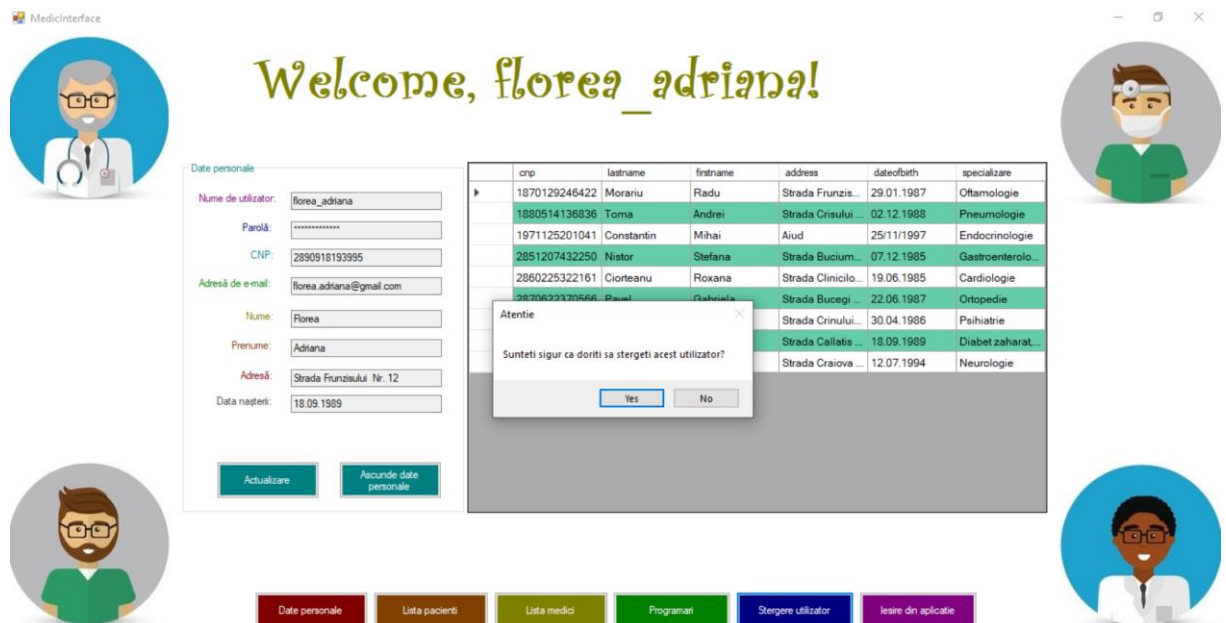
Pavel Gabriela

Programare

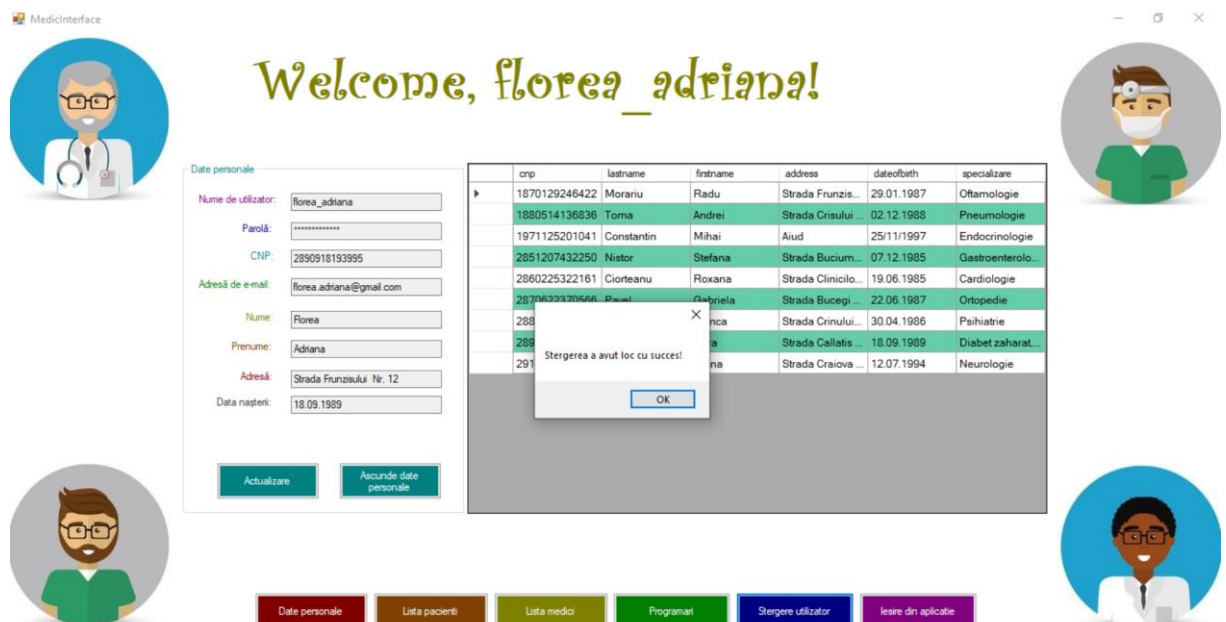
Specializari

Medici

- Stergerea contului de utilizator:
  - o Se apasa butonul Stergere utilizator



- Confirmarea stingerii contului



- Se revine la pagina principala

WELCOME TO AD7 Clinic !



Login

Nume de utilizator:

Parolă:

CNP:

☐ Medic ☐ Pacient



ADT app by Bolba, Cighi, Hinsu

- Iesirea din aplicatie:
  - Se apasa butonul Iesire din aplicatie



Welcome, florea \_adriana!



Atentie

Sunteti sigur ca doriti sa iesiti din program?



- Confirmarea iesirii din aplicatie

In calitate de medic, se pot efectua urmatoarele operatiuni:

- Inregistrarea in sistem:
  - Primul pas este de a selecta Radio Button-ul “Medic”, apoi se apasa butonul Inregistrare

Form1

WELCOME TO ADT Clinic !

Login

Nume de utilizator:

Parolă:

CNP:

☒ Medic ☐ Pacient

Login Înregistrare

ADT app by Bolba, Cighi, Hinsa

- Se introduc datele in Text Box-uri, iar pe urma se apasa pe butonul de Inregistrare

Form1

WELCOME TO AD7 Clinic !



**Înregistrare**

Nume de utilizator:

Parolă:

CNP:

Adresă de e-mail:

Nume:

Prenume:

Adresă:

Data nașterii:

Specializare:



ADT app by Bolba, Cighi, Hinsu

- Logarea in sistem:
- Se selecteaza Radio Button-ul “Medic” si se introduc datele in Text Box-uri

Form1

WELCOME TO AD7 Clinic !



**Login**

Nume de utilizator:

Parolă:

CNP:

☒ Medic
 ☐ Pacient



ADT app by Bolba, Cighi, Hinsu

- Se apasa butonul Login si se deschide un alt form care ii va oferi medicului mai multe optiuni

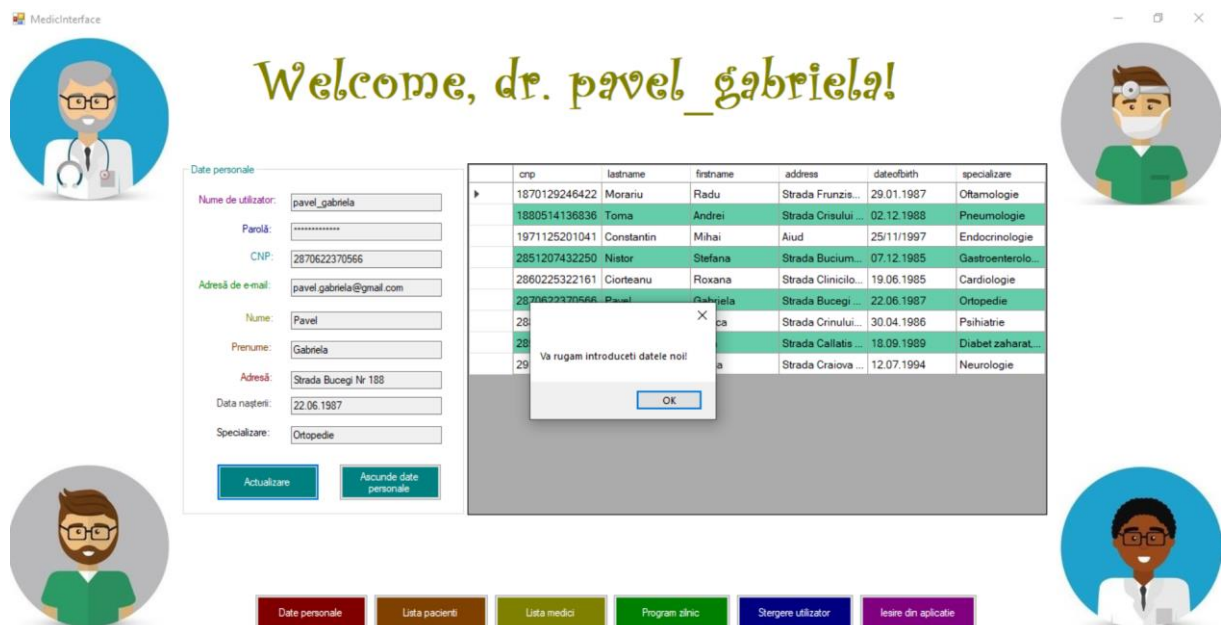


- Vizualizarea datelor persoanelor:
  - Se apasa butonul Date personale





- Actualizarea datelor personale:
  - Se apasa butonul Actualizare din Group Box-ul Date Personale



- Se modifica datele dorite(in acest caz, se mai adauga un prenume)



- Se apasa butonul Salvare modificari



- Vizualizarea listei de pacienti:
  - Se apasa butonul Lista pacienti





- Vizualizarea listei de medici:
  - Se apasa butonul Lista medici

MedicInterface

Welcome, dr. pavel\_gabriela!

Date personale

Nume de utilizator: pavel\_gabriela

Parolă: \*\*\*\*\*

CNP: 2870622370566

Adresă de e-mail: pavel.gabriela@gmail.com

Nume: Pavel

Prenume: Gabriela Maria

Adresă: Strada Bucegi Nr 188

Data nașterii: 22.06.1987

Specializare: Ortopedie

Actualizare Ascunde date personale

cnp	lastname	firstname	address	dateofbirth	specializare
1870129246422	Morariu	Radu	Strada Frunzis...	29.01.1987	Oftamologie
1880514136836	Toma	Andrei	Strada Crisului...	02.12.1988	Pneumologie
1971125201041	Constantin	Mihai	Aiud	25/11/1997	Endocrinologie
2651207432250	Nistor	Stefana	Strada Bucium...	07.12.1985	Gastroenterolo...
2860225322161	Ciorțeanu	Roxana	Strada Clinicio...	19.06.1985	Cardiologie
2870622370566	Pavel	Gabriela Maria	Strada Bucegi...	22.06.1987	Ortopedie
2881129281446	Pavel	Bianca	Strada Crinului...	30.04.1986	Psihiatrie
2890918415319	Iorga	Dora	Strada Callatis...	18.09.1989	Diabet zaharat...
2910614065713	Cristea	Diana	Strada Craiova...	12.07.1994	Neurologie

Date personale Lista pacienti Lista medici Program zilnic Stergere utilizator Inire din aplicatie

- Vizualizarea programului zilnic:
  - Se apasa butonul Program zilnic

MedicProgram

Nume_Pacient	Prenume_Pacient	Data_Ora_Programari
Olean	Anca	Miercuri 10:11
Florea	Adriana	Joi 9:10

Detali Pacient

Probleme medicale:

Problema curenta:

Recomandari

Nume Pacient:

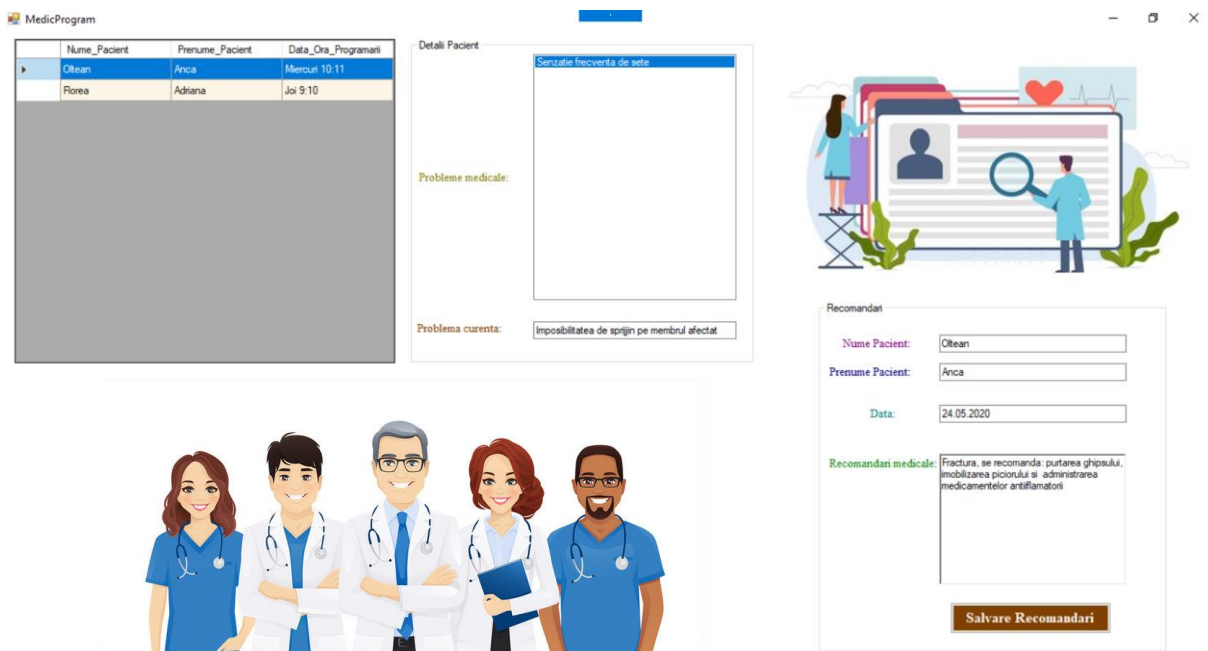
Prenume Pacient:

Data:

Recomandari medicale:

Salvare Recomandari

- Se selectează din DataGridView pacientul despre care se doresc mai multe informații. Apoi, în Group Box-ul Detalii Pacient, la secțiunea Probleme Medicale vor apărea toate problemele pacientului, iar la Problema curentă va apărea problema actuală pentru care a fost făcută programarea. Ulterior, în Group Box-ul Recomandări, medicul va introduce mai multe date în Text Box-uri, numele și prenumele pacientului, data la care a fost făcută programarea, iar la secțiunea Recomandări medicale, va introduce diagnosticul și recomandările în conformitate cu acesta



The screenshot displays the 'MedicProgram' application window. It features a DataGridView on the left with patient data, a 'Detalii Pacient' section in the center, and a 'Recomandari' section on the right. Below the application window is an illustration of five medical professionals.

	Nume_Pacient	Prenume_Pacient	Data_Ora_Programari
	Oltean	Anca	Miercuri 10:11
	Florea	Adriana	Joi 9:10

**Detalii Pacient**

Senzatie frecventa de sete

**Probleme medicale:**

**Problema curenta:** Imposibilitatea de sprijin pe membrul afectat

**Recomandari**

**Nume Pacient:** Oltean

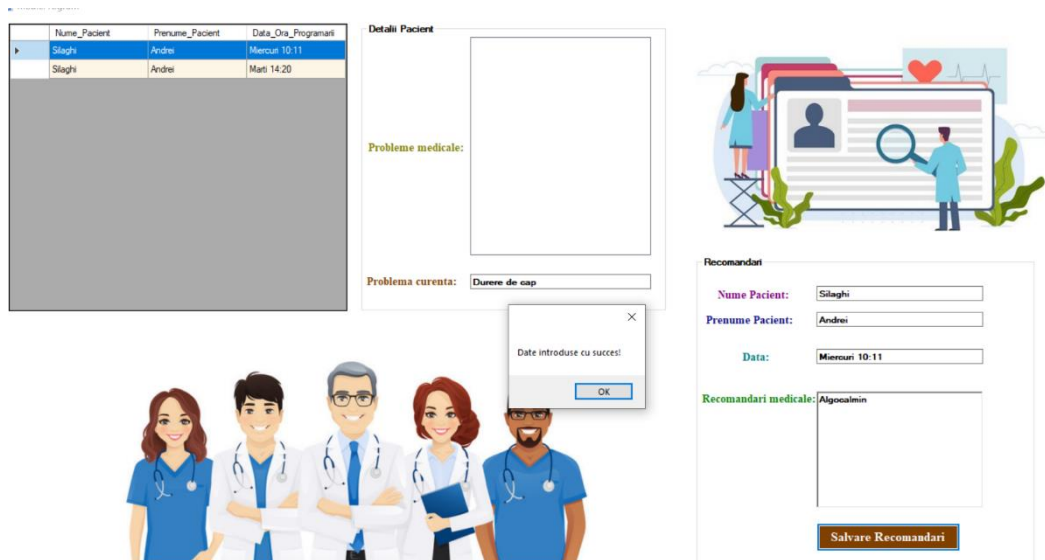
**Prenume Pacient:** Anca

**Data:** 24.05.2020

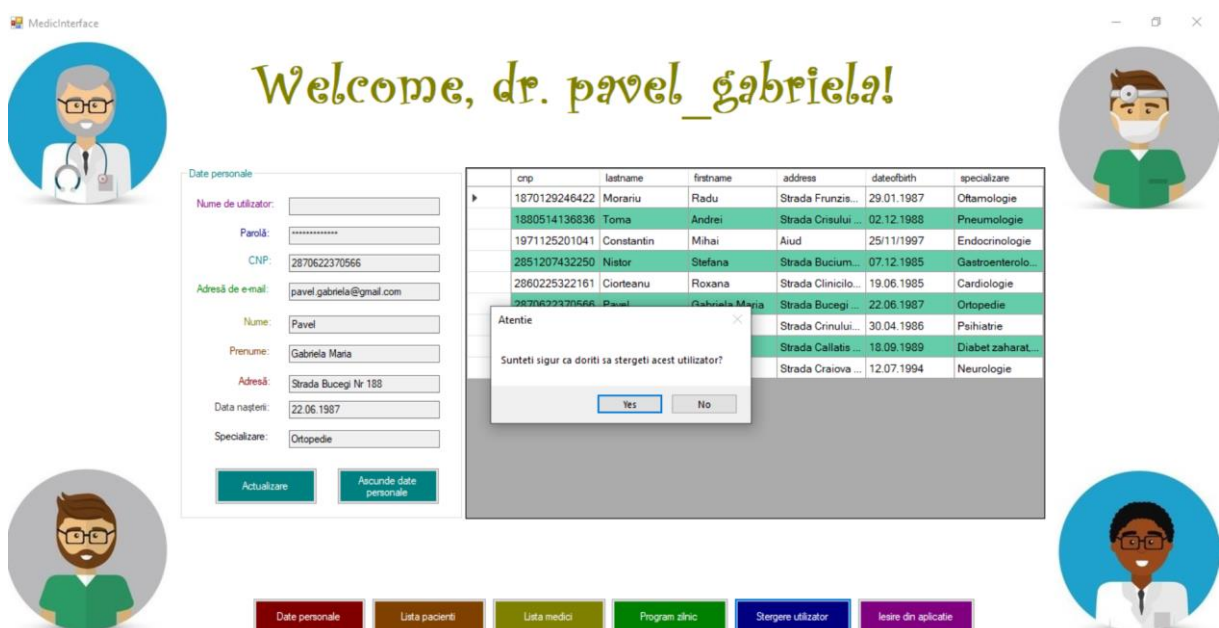
**Recomandari medicale:** Fractura, se recomanda: punerea gipsului, imobilizarea piciorului si administrarea medicamentelor antiinflamatori

**Salvare Recomandari**

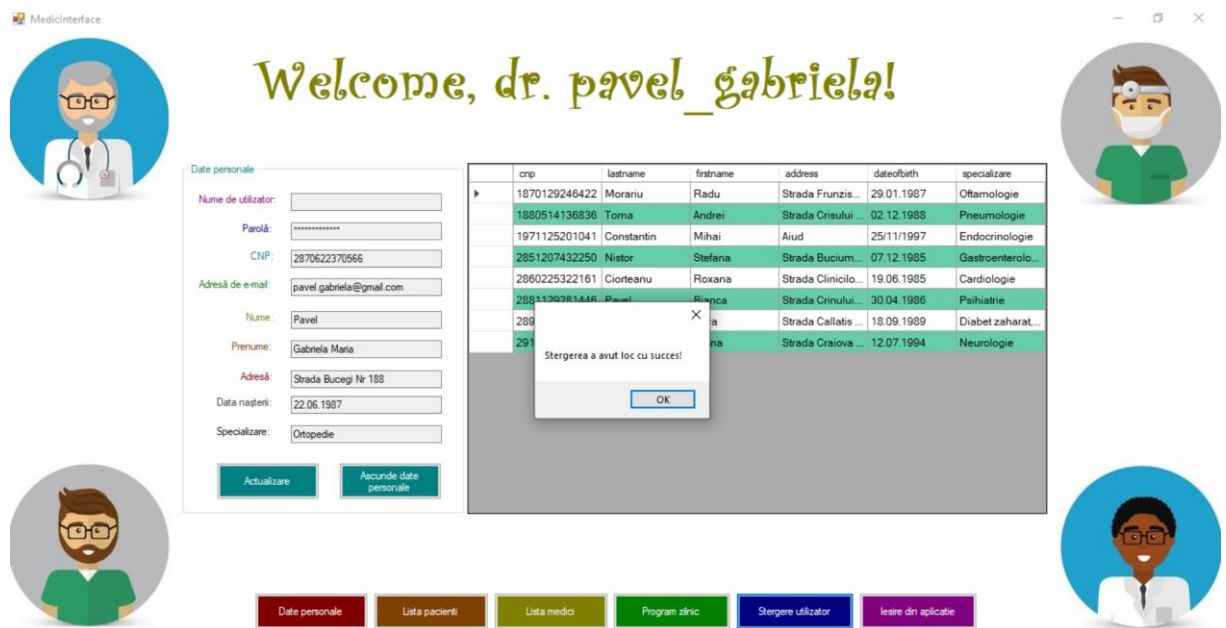
- Se apasă butonul Salvare Recomandari din Group Box-ul Recomandari



- Stergerea contului de utilizator:
  - Se apasa butonul Stergere utilizator



- Confirmarea stergerii contului

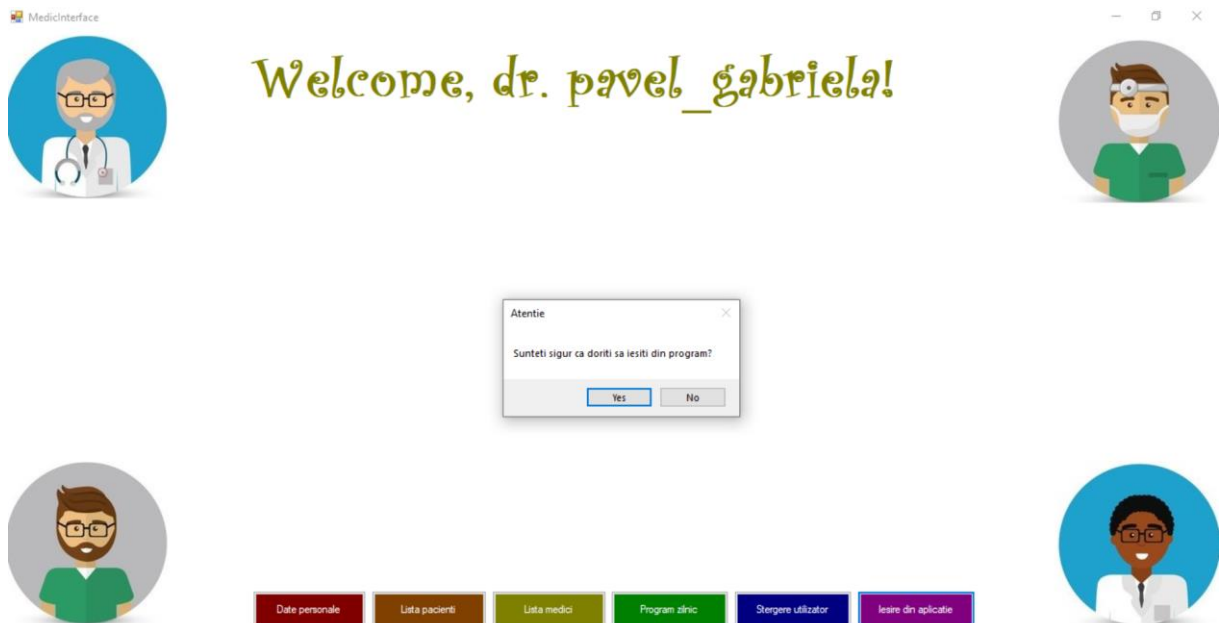


- Se revine la pagina principala



- Iesirea din aplicatie:

- Se apasa butonul Iesire din aplicatie



- Se inchide aplicatia

## Capitolul 4

### TESTAREA APLICATIEI

In urma testarii aplicatiei, am identificat urmatoarea problema: chiar daca Text Box-urile responsabile pentru username, password si cnp sunt necomplete, apasarea butonului Inregistrare determina crearea unui cont pentru un utilizator nou. Aceste informatii vor fi insa folosite si pentru logarea ulterioara si sunt, deci absolut necesare. Restul detaliilor ar putea fi adaugate sau modificate si dupa ce inregistrarea s-a efectuat.

Pentru a rezolva aceasta problema, am adaugat o conditie care verifica daca Text Box-urile responsabile pentru username, password si cnp sunt completate si doar daca aceasta conditie este indeplinita, contul noului utilizator va fi creat.

*WELCOME TO ADT Clinic !*



**Înregistrare**

Nume de utilizator:

Parolă:

Adresă de domiciliu:

Adresă:

Data nașterii:

Specializare:



ADT app by Bolba, Cighi, Hinsa

## Anexa 1 (cod sursa)

### Cod pentru Webservice :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace Webservice
{
    /// <summary>
    /// Summary description for Webservice1
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET AJAX,
    // uncomment the following line.
    // [System.Web.Script.Services.ScriptService]
    public class Webservice1 : System.Web.Services.WebService
    {
```

---

```

[WebMethod]
public int loginMedic(string iusername, string ipassword, string icnp)
{
    datamedic medic = new datamedic();

    using (MedicDBEntities db = new MedicDBEntities())
    {
        medic = db.datamedics.Where(x => x.cnp == icnp).FirstOrDefault();
        if (medic == null) return -1; //medic==null => utilizatorul nu
        exista in baza de date;
        else
        {
            if (medic.username == iusername)
            {
                if (medic.password == ipassword)
                {
                    return 1; //medicul exista si parola e buna =>
                    access granted;
                }
                else return -2; //parola gresita
            }
        }
        return 0; //medicul exista, dar parola si username sunt gresite
    }
} //loginMedic

```

```

[WebMethod]
public int registerMedic(string username, string password, string cnp,
string email, string lastname, string firstname, string address, string
dateofbirth, string specializare)
{
    datamedic medic1 = new datamedic();
    datamedic m1 = new datamedic(); //m1 se foloseste pentru cautare in
    baza de date
    //daca medicul nu exista in baza de
    date, atunci am fi avut un null pointer in medic
    //si intr-un null pointer nu se pot
    insera date (throws NullPointerException)

    MedicDetail detail = new MedicDetail();
    detail.lastname = lastname;
    detail.firstname = firstname;
    detail.address = address;
    detail.dateofbirth = dateofbirth;
    detail.cnp = cnp;
    detail.specializare = specializare;

    using (MedicDBEntities db = new MedicDBEntities())
    {
        m1 = db.datamedics.Where(x => x.cnp == cnp).FirstOrDefault();
        if (m1 != null) return 0; //it means that the given CNP already
        exists, registration is not possible
        //otherwise we create the medic object and insert it into the
        datamedic tabel
        medic1.username = username;
        medic1.password = password;
        medic1.cnp = cnp;
        medic1.email = email;
        try
        {
            db.datamedics.Add(medic1);
            db.SaveChanges();
        }
    }
}

```

---

```

        catch
        {
            return -1;
        }

        try
        {
            db.MedicDetails.Add(detail);
            db.SaveChanges();
        }
        catch
        {
            return -2;
        }

        return 1;
    }

    } //using
} //registerMedic

[WebMethod]
public void getMedicData(ref string username, ref string password, string
cnp, ref string email, ref string lastname, ref string firstname, ref string
address, ref string dateofbirth, ref string specializare)
{
    datamedic medic = new datamedic();
    MedicDetail detail = new MedicDetail();
    using (MedicDBEntities db = new MedicDBEntities())
    {
        medic = db.datamedics.Where(x => x.cnp == cnp).FirstOrDefault();
        password = medic.password;
        cnp = medic.cnp;
        email = medic.email;
        //search in datamedics by cnp
        detail = db.MedicDetails.Where(x => x.cnp ==
medic.cnp).FirstOrDefault();
        lastname = detail.lastname;
        firstname = detail.firstname;
        address = detail.address;
        dateofbirth = detail.dateofbirth;
        specializare = detail.specializare;
        //search in medicdetails based on cnp
    }
}

[WebMethod]
public int changeMedicData(string username, string password, string cnp,
string email, string lastname, string firstname, string address, string
dateofbirth, string specializare)
{
    datamedic medic = new datamedic();
    MedicDetail detail = new MedicDetail();

    using (MedicDBEntities db = new MedicDBEntities())
    {
        try
        {
            medic = db.datamedics.Where(x => x.cnp ==
cnp).FirstOrDefault();

            medic.password = password;

```



---

```

        //changing cnp is prohibited, because this is key information
(for identifying)
        medic.email = email;
        medic.username = username;

        db.Entry(medic).State =
System.Data.Entity.EntityState.Modified;
        db.SaveChanges();

        detail = db.MedicDetails.Where(x => x.cnp ==
medic.cnp).FirstOrDefault();
        detail.lastname = lastname;
        detail.firstname = firstname;
        detail.address = address;
        detail.dateofbirth = dateofbirth;
        detail.specializare = specializare;
        db.Entry(detail).State =
System.Data.Entity.EntityState.Modified;
        db.SaveChanges();
    }
    catch
    {
        return -1;
    }
    return 1;

}
}

} //using
} //changeMedicData()

[WebMethod]
public int deleteMedicAccount(string username, string cnp)
{
    datamedic medic = new datamedic();
    MedicDetail detail = new MedicDetail();

    try
    {
        using (MedicDBEntities db = new MedicDBEntities())
        {
            medic = db.datamedics.Where(x => x.username ==
username).FirstOrDefault();
            if (medic.cnp != cnp) return -1;

            detail = db.MedicDetails.Where(x => x.cnp ==
medic.cnp).FirstOrDefault();
            var entry = db.Entry(detail);
            if (entry.State == System.Data.Entity.EntityState.Detached)
                db.MedicDetails.Attach(detail);
            db.MedicDetails.Remove(detail);
            db.SaveChanges();

            var entry1 = db.Entry(medic);
            if (entry1.State == System.Data.Entity.EntityState.Detached)
                db.datamedics.Attach(medic);
            db.datamedics.Remove(medic);
            db.SaveChanges();
        }
    }
    catch
    {
        return -2;
    }
}

```

---

---

```

        } //catch
        return 1;

    } //function

    [WebMethod]

    public int logInPacient(string pUser, string pPass, string pCNP)
    {
        Pacienti p = new Pacienti();

        using (MedicDBEntities md = new MedicDBEntities())
        {
            p = md.Pacientis.Where(x => x.pacient_cnp ==
pCNP).FirstOrDefault();

            if (p == null) return -1; //pacientul nu e in baza de date
            else
            {
                if (p.username == pUser)
                {
                    if (p.password == pPass)
                        return 1; //se logheaza pacientul
                    else
                        return -2; //parola gresita
                }
            }
            return 0; //username si parola gresite
        }
    } //logInPacient

    [WebMethod]

    public int registerPacient(string pUser, string pPass, string pCNP,
string pEmail, string pLName, string pFName, string pBirth, string pAddress)
    {
        Pacienti p = new Pacienti();
        Pacienti p1 = new Pacienti(); //variabila temporara de cautare

        PacientDetail detail = new PacientDetail();

        detail.address = pAddress;
        detail.dateofbirth = pBirth;
        detail.firstname = pFName;
        detail.latname = pLName;
        detail.pacient_cnp = pCNP;

        using (MedicDBEntities md = new MedicDBEntities())
        {
            p1 = md.Pacientis.Where(x => x.pacient_cnp ==
pCNP).FirstOrDefault();

            if (p1 != null) //cnp exista deja
            {
                return 0;
            }
            else
            {
                p.username = pUser;
                p.password = pPass;
                p.email = pEmail;
            }
        }
    }

```

---

```

        p.pacient_cnp = pCNP;
        try
        {
            md.Pacientis.Add(p);
            md.SaveChanges();
        }
        catch
        {
            return -1;
        }
        try
        {
            md.PacientDetails.Add(detail);
            md.SaveChanges();
        }
        catch
        {
            return -2;
        }
    }
    return 1;
}

[WebMethod]
public void getPacientDetails(ref string pUser, ref string pPass, string
pCNP, ref string pEmail, ref string pLName, ref string pFName, ref string pBirth,
ref string pAddress)
{
    PatientDetail detail = new PatientDetail();
    Pacienti patient = new Pacienti();

    using (MedicDBEntities md = new MedicDBEntities())
    {
        //cautare date de login in tabelul pacienti
        patient = md.Pacientis.Where(x => x.pacient_cnp ==
pCNP).FirstOrDefault();

        pUser = patient.username;
        pPass = patient.password;
        pCNP = patient.pacient_cnp;
        pEmail = patient.email;

        //cautare date personale in tabelul patientDetail

        detail = md.PacientDetails.Where(x => x.pacient_cnp ==
patient.pacient_cnp).FirstOrDefault();
        pAddress = detail.address;
        pLName = detail.latname;
        pFName = detail.firstname;
        pBirth = detail.dateofbirth;
    }
}

[WebMethod]
public int updatePacientData(string pUser, string pPass, string pCNP,
string pEmail, string pLName, string pFName, string pBirth, string pAddress)
{
    Pacienti patient = new Pacienti();

```

---

```

        PacientDetail detail = new PacientDetail();

        using (MedicDBEntities md = new MedicDBEntities())
        {
            try
            {
                pacient = md.Pacientis.Where(x => x.pacient_cnp ==
pCNP).FirstOrDefault();

                pacient.username = pUser;
                pacient.password = pPass;
                pacient.email = pEmail;

                //nu vom schimba cnp-ul pentru ca este o informatie esentiala
pt cautare

                md.Entry(pacient).State =
System.Data.Entity.EntityState.Modified;
                md.SaveChanges();

                detail = md.PacientDetails.Where(x => x.pacient_cnp ==
pacient.pacient_cnp).FirstOrDefault();
                detail.latname = pLName;
                detail.firstname = pFName;
                detail.address = pAddress;
                detail.dateofbirth = pBirth;

                md.Entry(detail).State =
System.Data.Entity.EntityState.Modified;
                md.SaveChanges();

            }
            catch
            {
                return -1;
            }
            return 1;
        }
    }

    [WebMethod]
    public int deletePacient(string pUser, string pCNP)
    {
        Pacienti pacient = new Pacienti();
        PacientDetail detail = new PacientDetail();

        using (MedicDBEntities md = new MedicDBEntities())
        {
            try
            {
                pacient = md.Pacientis.Where(x => x.username ==
pUser).FirstOrDefault();

                if (pacient.pacient_cnp != pCNP) return -1;
                detail = md.PacientDetails.Where(x => x.pacient_cnp ==
pacient.pacient_cnp).FirstOrDefault();
                var entry = md.Entry(detail);
                if (entry.State == System.Data.Entity.EntityState.Detached)
                    md.PacientDetails.Attach(detail);
                md.PacientDetails.Remove(detail);
                md.SaveChanges();
            }
            catch
            {
                return -1;
            }
            return 1;
        }
    }
}

```

---

```

        var entry1 = md.Entry(pacient);
        if (entry1.State == System.Data.Entity.EntityState.Detached)
            md.Pacientis.Attach(pacient);
        md.Pacientis.Remove(pacient);
        md.SaveChanges();
    }
    catch
    {
        return -2;
    }
    return 1;
}
}
[WebMethod]
public int programari(string pCNP, string pLName, string pFName, string
descriere, string date, string mFName, string mLName)
{
    Programare programare = new Programare();
    PacientDetail p_detail = new PacientDetail();
    MedicDetail m_detail = new MedicDetail();
    Programare p1 = new Programare();

    using (MedicDBEntities md = new MedicDBEntities())
    {
        try
        {
            p_detail = md.PacientDetails.Where(x => x.pacient_cnp ==
pCNP).FirstOrDefault();
            m_detail = md.MedicDetails.Where(x => x.lastname == mLName &&
x.firstname == mFName).FirstOrDefault();

            programare = md.Programares.Where(x => x.data == date &&
x.m_lastname == m_detail.lastname && x.m_firstname ==
m_detail.firstname).FirstOrDefault();

            if (programare != null)
                return 0; //exista o programare
            else
            {
                p1.pacient_cnp = pCNP;
                p1.p_lastname = pLName;
                p1.p_firstname = pFName;
                p1.descriere = descriere;
                p1.data = date;
                p1.m_firstname = mFName;
                p1.m_lastname = mLName;

                md.Programares.Add(p1);
                md.SaveChanges();
            }
        }
        catch
        {
            return -1;
        }
        return 1;
    }
}
}

```

---

```

[WebMethod]

    public void DoctorsfromSpecialization(string specializare, ref
List<string> list)
    {
        MedicDetail detail = new MedicDetail();

        using (MedicDBEntities md = new MedicDBEntities())
        {
            List<MedicDetail> mList = new List<MedicDetail>();

            mList = md.MedicDetails.Where(x => x.specializare ==
specializare).ToList();

            foreach (var det in mList)
            {
                list.Add(det.lastname + " " + det.firstname);
            }
        }
    }

[WebMethod]

    public void listSpecialization(ref List<string> list)
    {

        using (MedicDBEntities md = new MedicDBEntities())
        {
            List<MedicDetail> mList = new List<MedicDetail>();

            mList = md.MedicDetails.ToList();

            foreach (var det in mList)
            {
                list.Add(det.specializare);
            }
        }
    }

[WebMethod]

    public void dataProgramarii(ref List<string> list)
    {
        Programare prog = new Programare();
        List<Programare> plist = new List<Programare>();
        using (MedicDBEntities md = new MedicDBEntities())
        {
            plist = md.Programares.ToList();

            foreach (var det in plist)
            {
                list.Add(det.data + " " + det.m_lastname + " " +
det.m_firstname + " " + det.p_lastname + " " + det.p_firstname);
            }
        }
    }
}

```

---

```

        [WebMethod]
        public void listPacienti(string pCnp, ref string pFName, ref string
pLName)
        {
            PacientDetail pacient = new PacientDetail();

            using (MedicDBEntities md = new MedicDBEntities())
            {
                pacient = md.PacientDetails.Where(x => x.pacient_cnp ==
pCnp).FirstOrDefault();

                pFName = pacient.firstname;
                pLName = pacient.latname;
            }
        }

        [WebMethod]
        public void listaPacientiProgramati(string mLNum, string mFNum, ref
List<string> list)
        {
            List<Programare> pList = new List<Programare>();
            using (MedicDBEntities md = new MedicDBEntities())
            {
                pList = md.Programares.Where(x => x.m_firstname == mFNum &&
x.m_lastname == mLNum).ToList();

                foreach (var el in pList)
                {
                    list.Add(el.p_lastname + " " + el.p_firstname + " " +
el.data);
                }
            }
        }

        [WebMethod]
        public void dateMedic(string mCNP, ref string mLName, ref string mFName)
        {
            MedicDetail medic = new MedicDetail();

            using (MedicDBEntities md = new MedicDBEntities())
            {
                medic = md.MedicDetails.Where(x => x.cnp ==
mCNP).FirstOrDefault();
                mLName = medic.lastname;
                mFName = medic.firstname;
            }
        }

        [WebMethod]
        public void antecedenteMedicale(string pLname, string pFname, ref
List<string> list, string mLname, string mFname)
        {
            List<Programare> pList = new List<Programare>();
            using (MedicDBEntities md = new MedicDBEntities())
            {
                pList = md.Programares.Where(x => x.p_firstname == pFname &&
x.p_lastname == pLname && x.m_firstname != mFname && x.m_lastname !=
mLname).ToList();
            }
        }

```

---

```

        foreach (var el in pList)
        {
            list.Add(el.descriere);
        }
    }
}

[WebMethod]

public int fisaMedicala(string pCNP, string descriere, string data,
string numeD)
{
    FisaMedicala fs = new FisaMedicala();
    FisaMedicala fs1 = new FisaMedicala();

    using (MedicDBEntities md = new MedicDBEntities())
    {
        try
        {
            fs1 = md.FisaMedicalas.Where(x => x.data == data &&
x.nume_doctor == numeD).FirstOrDefault();
            if (fs1 != null)
                return 0;
            fs.pacient_cnp = pCNP;
            fs.descriere = descriere;
            fs.nume_doctor = numeD;
            fs.data = data;

            md.FisaMedicalas.Add(fs);
            md.SaveChanges();
        }
        catch
        {
            return -1;
        }
        return 1;
    }
}

[WebMethod]

public void datePacient(string lastname, string firstname, ref string
cnp)
{
    PacientDetail pacient = new PacientDetail();

    using (MedicDBEntities md = new MedicDBEntities())
    {
        pacient = md.PacientDetails.Where(x => x.firstname == firstname
&& x.lastname == lastname).FirstOrDefault();
        cnp = pacient.pacient_cnp;
    }
}

[WebMethod]

public void problemaCurenta(string pLname, string pFname, string mLname,
string mFname, ref string descriere)
{
    Programare prog = new Programare();
    using (MedicDBEntities md = new MedicDBEntities())

```



---

```

        {
            prog = md.Programares.Where(x => x.p_firstname == pFname &&
x.p_lastname == pLname && x.m_firstname == mFname && x.m_lastname ==
mLname).FirstOrDefault();
            descriere = prog.descriere;
        }
    }
}

```

## Cod pentru FormApplication :

### 1. Form1.cs:

```

using FormApp.ServiceReference2;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace FormApp
{
    public partial class Form1 : Form
    {
        FormApp.ServiceReference2.WebService1SoapClient service =
            new FormApp.ServiceReference2.WebService1SoapClient();
        public Form1()
        {
            InitializeComponent();
        }
        public String GetUsername()
        {
            return txtUsername.Text;
        }
        private void signUp_btn_Click(object sender, EventArgs e)
        {
            //datamedic:
            string username = txt_username.Text.ToString();
            string password = txt_password.Text.ToString();
            string cnp = txt_CNP.Text.ToString();
            string email = txt_email.Text.ToString();

            //MedicDetails:
            //cnp as foreign key
            string lastname = txt_nume.Text.ToString();
            string firstname = txt_prenume.Text.ToString();
            string address = txt_address.Text.ToString();
            string dateofbirth = txt_data.Text.ToString();
            string specializare = sp_txt.Text.ToString();
            if (m_radioBtn.Checked == false && p_radioBtn.Checked == false)
            {
                MessageBox.Show("Va rugam alegeti o optiune: PACIENT/MEDIC!");
                groupbox_register.Visible = false;
                groupbox_login.Visible = true;
            }
        }
    }
}

```

---

```

    }
    else if (m_radioBtn.Checked == true)
    {
        int opt = service.registerMedic(username, password, cnp, email,
lastname, firstname, address, dateofbirth, specializare);
        if (opt == 1)
        {
            MessageBox.Show("Contul a fost creat. Bine ati venit!");
            MedicInterface medicInf = new MedicInterface(username, this,
cnp);

            medicInf.Show();
            this.Visible = false;

        }
        else if (opt == -1) MessageBox.Show("Error code: -1. Could not
insert to DataMedics");
        else if (opt == -2) MessageBox.Show("Error code: -2. Could not
insert to MedicDetail");
        else
        {
            MessageBox.Show("Acest CNP exista deja in baza de date. Va
rugam sa va autentificati");
            groupbox_register.Visible = false;
            groupbox_login.Visible = true;
        }
    }
    else if (p_radioBtn.Checked == true)
    {
        int opt_p = service.registerPacient(username, password, cnp,
email, lastname, firstname, dateofbirth, address);
        if (opt_p == 1)
        {
            MessageBox.Show("Contul a fost creat. Bine ati venit!");
            MedicInterface medicInf = new MedicInterface(username, this,
cnp);

            medicInf.Show();
            this.Visible = false;

        }
        else if (opt_p == -1) MessageBox.Show("Error code: -1. Could not
insert to Pacienti.");
        else if (opt_p == -2) MessageBox.Show("Error code: -2. Could not
insert to PacientDetail.");
        else
        {
            MessageBox.Show("Acest CNP exista deja in baza de date. Va
rugam sa va autentificati!");
            groupbox_register.Visible = false;
            groupbox_login.Visible = true;
        }
    }
}
private void button2_Click(object sender, EventArgs e)
{
    //register button from login group
    groupbox_login.Visible = false; //hide the login window => flawless
swap between the interfaces
    groupbox_register.Visible = true; //make sure to show the
registration window
    if (p_radioBtn.Checked == true)
    {
        sp_txt.Visible = false;
    }
}

```

---

```

        sp_label.Visible = false;
    }
}
private void button1_Click(object sender, EventArgs e)
{
    //login button from login group
    string iusername, ipassword, icnp;
    iusername = txtUsername.Text.ToString();
    ipassword = txtPassword.Text.ToString(); //preluam datele de la
fereastră Login
    icnp = cnp_log.Text.ToString();

    if (m_radioBtn.Checked == false && p_radioBtn.Checked == false)
    {
        MessageBox.Show("Va rugam sa alegeti o optiune: PACIENT/MEDIC!");
    }
    else if (m_radioBtn.Checked == true)
    {
        int opt = service.loginMedic(iusername, ipassword, icnp);
        if (opt == -1)
        {
            MessageBox.Show("Acest CNP nu exista in baza de date.");
        }
        else
        {
            if (opt == -2) MessageBox.Show("Nume de utilizator si CNP
existente, dar parola e gresita.");
            else if (opt == 1)
            {
                MessageBox.Show("Bine ati venit!");
                MedicInterface medicInf = new MedicInterface(iusername,
this, icnp);

                medicInf.Show();
                this.Visible = false;
            }
            else
            {
                MessageBox.Show("Nume de utilizator gresit!");
            }
        }
    }
}
else if (p_radioBtn.Checked == true)
{
    int opt_p = service.logInPacient(iusername, ipassword, icnp);
    if (opt_p == -1)
    {
        MessageBox.Show("Acest CNP nu exista in baza de date.");
    }
    else
    {
        if (opt_p == -2) MessageBox.Show("Nume de utilizator si CNP
existente, dar parola e gresita.");
        else if (opt_p == 1)
        {
            MessageBox.Show("Bine ati venit!");
            MedicInterface medicInf = new MedicInterface(iusername,
this, icnp);

            medicInf.Show();

```

---

```

        this.Visible = false;
    }
    else
    {
        MessageBox.Show("Nume de utilizator gresit!");
    }
}

txtUsername.Text = ""; txtPassword.Text = ""; cnp_log.Text = "";
}

private void pictureBox1_Click(object sender, EventArgs e)
{
}
}
}

```

## 2. MedicProgram.cs:

```

using FormApp.ServiceReference2;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace FormApp
{
    public partial class MedicProgram : Form
    {
        FormApp.ServiceReference2.WebService1SoapClient service =
            new FormApp.ServiceReference2.WebService1SoapClient();
        MedicInterface mainform = null;
        public string cnp;
        public MedicProgram(string cnp, Form callingForm)
        {
            InitializeComponent();
            this.cnp = cnp;
            mainform = callingForm as MedicInterface;
            populatedataGridView();
        }
        public void populatedataGridView()
        {
            string mLName = " ", mFName = " ", mcnp = this.cnp;
            service.dateMedic(mcnp, ref mLName, ref mFName);

            List<string> list = new List<string>();
            ArrayOfString array = new ArrayOfString();
            array.AddRange(list);
            service.listaPacientiProgramati(mLName, mFName, ref array);
            string[] data;

```

---

```

        string date;

        foreach (var el in array)
        {
            data = el.Split(' ');
            date = data[2] + " " + data[3];
            dataGridView1.Rows.Add(data[0], data[1], date);
        }
    }

    private void button2_Click(object sender, EventArgs e)
    {
        string numep = textBox2.Text;
        string prenumep = textBox3.Text;

        string descriere = richTextBox1.Text;
        string data = textBox4.Text;
        string pcnp = " ";
        string numem = " ", prenumem = " ";
        service.datePacient(numep, prenumep, ref pcnp);
        service.dateMedic(cnp, ref numem, ref prenumem);

        int opt;
        opt = service.fisaMedicala(pcnp, descriere, data, numem);
        if (opt == 0)
        {
            MessageBox.Show("S-au mai introdus date la aceasta programare!");
        }
        else if (opt == 1)
        {
            MessageBox.Show("Date introduse cu succes!");
        }
        else
        {
            MessageBox.Show("ERROR -1: Eroare de sistem!");
        }
    }

    private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
    {
        string pLname, pFname, mLname = " ", mFname = " ", descriere = " ";
        service.dateMedic(this.cnp, ref mLname, ref mFname);
        pLname = dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString();
        pFname = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();
        List<string> list = new List<string>();
        ArrayOfString array = new ArrayOfString();
        array.AddRange(list);
        service.antecedenteMedicale(pLname, pFname, ref array, mLname,
mFname);
        // var l = array.Take(array.Count - 1).ToList();
        service.problemaCurenta(pLname, pFname, mLname, mFname, ref
descriere);
        listBox1.DataSource = array;
        textBox1.Text = descriere;
    }
}
}

```

MedicInterface.cs:

---

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace FormApp
{
    public partial class MedicInterface : Form
    {
        FormApp.ServiceReference2.WebService1SoapClient service =
            new FormApp.ServiceReference2.WebService1SoapClient();
        public string username;
        public string cnp;

        private Form1 mainForm = null;
        public MedicInterface(string username, Form callingForm, string cnp)
        {
            InitializeComponent();
            this.username = username;
            this.cnp = cnp;

            mainForm = callingForm as Form1;          //accesam elemente din Form1,
                                                    //dupa ce le-am modificat
        }

        vizibilitatea in public
        if (mainForm.m_radioBtn.Checked == true)
        {
            label1.Text = "Welcome, dr. " + username.ToString() + "!";
            button7.Visible = false;
        }
        else if (mainForm.p_radioBtn.Checked == true)
        {
            label1.Text = "Welcome, " + username.ToString() + "!";
            spec_lab.Visible = false;
            spec_txtBox.Visible = false;
            button8.Visible = false;
        }

        groupbox_personal_medic.Visible = false;
    }
    private void MedicInterface_Load(object sender, EventArgs e)
    {
        // TODO: This line of code loads data into the
        'medicDBDataSet1.MedicDetail' table. You can move, or remove it, as needed.
        this.medicDetailTableAdapter.Fill(this.medicDBDataSet1.MedicDetail);
        // TODO: This line of code loads data into the
        'medicDBDataSet.PacientDetail' table. You can move, or remove it, as needed.
        this.pacientDetailTableAdapter.Fill(this.medicDBDataSet.PacientDetail);
    }

    private void personaldata_Btn_Click(object sender, EventArgs e)
    {
        if (groupbox_personal_medic.Visible == false)
            groupbox_personal_medic.Visible = true;
        else if (groupbox_personal_medic.Visible == true)
            groupbox_personal_medic.Visible = false;
        string username = this.username, password = "", cnp = this.cnp, email
= "";

```

---

```

        string lastname = "", firstname = "", address = "", dateofbirth = "",
specializare = "";

        if (mainForm.m_radioBtn.Checked == true)
        {
            service.getMedicData(ref username, ref password, this.cnp, ref
email, ref lastname, ref firstname, ref address, ref dateofbirth, ref
specializare);
            txt_username.Text = username;
            txt_password.Text = password;
            txt_email.Text = email;
            txt_CNP.Text = cnp;

            txt_nume.Text = lastname;
            txt_prenume.Text = firstname;
            txt_address.Text = address;
            txt_data.Text = dateofbirth;
            spec_txtBox.Text = specializare;
        }
        else
        {
            service.getPacientDetails(ref username, ref password, this.cnp,
ref email, ref lastname, ref firstname, ref dateofbirth, ref address);
            txt_username.Text = username;
            txt_password.Text = password;
            txt_email.Text = email;
            txt_CNP.Text = cnp;

            txt_nume.Text = lastname;
            txt_prenume.Text = firstname;
            txt_address.Text = address;
            txt_data.Text = dateofbirth;
        }
    }
    private void button1_Click(object sender, EventArgs e)
    {
        groupbox_personal_medic.Visible = false;//date personale
    }
    private void button2_Click(object sender, EventArgs e)
    {
        if (pacient_dataGridView.Visible == false)
        {
            pacient_dataGridView.Visible = true;
            medic_dataGridView.Visible = false;
        }
        else if (pacient_dataGridView.Visible == true)
            pacient_dataGridView.Visible = false;
    }
    private void button6_Click(object sender, EventArgs e)
    {
        if (medic_dataGridView.Visible == false)
        {
            medic_dataGridView.Visible = true;
            pacient_dataGridView.Visible = false;
            medic_dataGridView.DataSource = this.medicDBDataSet1.Tables[0];
        }
        else if (medic_dataGridView.Visible == true)
        {
            medic_dataGridView.Visible = false;
        }
    }

```

---

```

    }
}

private void button8_Click(object sender, EventArgs e)
{
    if (mainForm.m_radioBtn.Checked == true)
    {
        MedicProgram appl = new MedicProgram(cnp, this);
        appl.Show();
    }
}

private void button5_Click(object sender, EventArgs e)
{
    int opt = 0;
    string username = "", password = "", cnp = this.cnp, email = "";
    string lastname = "", firstname = "", address = "", dateofbirth = "",
specializare = "";

    if (mainForm.m_radioBtn.Checked == true)
        service.getMedicData(ref username, ref password, this.cnp, ref
email, ref lastname, ref firstname, ref address, ref dateofbirth, ref
specializare);
    else if (mainForm.p_radioBtn.Checked == true)
        service.getPacientDetails(ref username, ref password, this.cnp,
ref email, ref lastname, ref firstname, ref dateofbirth, ref address);

    txt_username.Text = username;
    txt_password.Text = password;
    txt_email.Text = email;
    txt_CNP.Text = cnp;

    txt_nume.Text = lastname;
    txt_prenume.Text = firstname;
    txt_address.Text = address;
    txt_data.Text = dateofbirth;
    DialogResult dialogResult = MessageBox.Show("Sunteti sigur ca doriti
sa stergeti acest utilizator?", "Atentie", MessageBoxButtons.YesNo);
    if (dialogResult == DialogResult.Yes)
    {
        if (mainForm.m_radioBtn.Checked == true)
        {
            opt = service.deleteMedicAccount(this.username, this.cnp);

this.medicDetailTableAdapter.Fill(this.medicDBDataSet1.MedicDetail);
        }
        else if (mainForm.p_radioBtn.Checked == true)
        {
            opt = service.deletePacient(this.username, this.cnp);

this.pacientDetailTableAdapter.Fill(this.medicDBDataSet.PacientDetail);
        }

        if (opt == -1 || opt == -2)
            MessageBox.Show("Problema la stergerea utilizatorului");
        else
        {
            MessageBox.Show("Stergerea a avut loc cu succes!");

            Form1 form = new Form1();
            form.Show();
        }
    }
}

```

---



---

```

        this.Close();
    }
}
else if (dialogResult == DialogResult.No)
{
    //do something else
}
}

private void button4_Click(object sender, EventArgs e)
{
    DialogResult dialogResult = MessageBox.Show("Sunteti sigur ca doriti
sa iesiti din program?", "Atentie", MessageBoxButtons.YesNo);
    if (dialogResult == DialogResult.Yes)
    {
        Form1 loginWindow = new Form1();
        this.Close();
    }
    else if (dialogResult == DialogResult.No)
    {
        //do something else
    }
}

private void button3_Click(object sender, EventArgs e)
{
    button3.Visible = false;
    update_Btn.Visible = true;

    string username = txt_username.Text.ToString();
    string password = txt_password.Text.ToString();
    string cnp = txt_CNP.Text.ToString();
    string email = txt_email.Text.ToString();

    string lastname = txt_nume.Text.ToString();
    string firstname = txt_prenume.Text.ToString();
    string address = txt_address.Text.ToString();
    string dateofbirth = txt_data.Text.ToString();
    string specializare = spec_txtBox.Text.ToString();

    if (mainForm.m_radioBtn.Checked == true)
    {
        int opt = service.changeMedicData(this.username, password, cnp,
email, lastname, firstname, address, dateofbirth, specializare);
        if (opt == -1)
        {
            MessageBox.Show("A aparut o problema la actualizare.");
        }
        else
        {
            MessageBox.Show("Actualizare reusita.");
        }
    }

    this.medicDetailTableAdapter.Fill(this.medicDBDataSet1.MedicDetail);

    }//else
}
else if (mainForm.p_radioBtn.Checked == true)
{
    int opt_p = service.updatePacientData(username, password, cnp,
email, lastname, firstname, dateofbirth, address);

```

---

```

        if (opt_p == -1)
        {
            MessageBox.Show("A aparut o problema la actualizare.");
        }
        else
        {
            MessageBox.Show("Actualizare reusita.");
        }
    }
    this.pacientDetailTableAdapter.Fill(this.medicDBDataSet.PacientDetail);
    }
}

txt_password.ReadOnly = true;
txt_email.ReadOnly = true;
txt_nume.ReadOnly = true;
txt_prenume.ReadOnly = true;
txt_address.ReadOnly = true;
txt_data.ReadOnly = true;
}
private void signUp_btn_Click(object sender, EventArgs e)
{
    MessageBox.Show("Va rugam introduceti datele noi!");

    button3.Visible = true;
    update_Btn.Visible = false;
    txt_password.ReadOnly = false;
    txt_email.ReadOnly = false;
    txt_nume.ReadOnly = false;
    txt_prenume.ReadOnly = false;
    txt_address.ReadOnly = false;
    txt_data.ReadOnly = false;

}

private void button7_Click(object sender, EventArgs e)
{
    if (mainForm.p_radioBtn.Checked == true)
    {
        Appointments app = new Appointments(this, cnp);
        app.Show();
    }
}
}
}

```

### 3. Appointments.cs :

```

using FormApp.ServiceReference2;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace FormApp
{
    public partial class Appointments : Form
    {

```

---

```

FormApp.ServiceReference2.WebService1SoapClient service =
    new FormApp.ServiceReference2.WebService1SoapClient();
private MedicInterface mainform = null;
private string cnp;
public Appointments(Form callingform, string cnp)
{
    InitializeComponent();
    this.cnp = cnp;
    mainform = callingform as MedicInterface;
    this.cnp = mainform.cnp;
    med_btn.Visible = false;
    int ora1, ora2;

    for (int i = 0; i < 8; i++)
    {
        ora1 = 8 + i;
        ora2 = 9 + i;
        string ore = ora1.ToString() + ":" + ora2.ToString();
        dataGridView1.Rows.Add(ore, " ", " ", " ", " ", " ", " ");
    }

    populateDataGridView();
}
public void populateDataGridView()
{
    List<string> list = new List<string>();
    ArrayOfString array = new ArrayOfString();
    array.AddRange(list);
    service.dataProgramarii(ref array);
    string day, hour, data1;
    int positionCol = 0, positionRow = 0;
    foreach (var el in array)
    {
        string[] data = el.Split(' ');
        day = data[0];
        hour = data[1];
        for (int i = 1; i < 6; i++)
        {
            if (dataGridView1.Columns[i].Name == day)
            {
                positionCol = i;
            }
        }
        for (int i = 0; i < 8; i++)
        {
            if (dataGridView1.Rows[i].Cells[0].Value.ToString() == hour)
            {
                positionRow = i;
            }
        }
        string recuperata =
dataGridView1.Rows[positionRow].Cells[positionCol].Value.ToString();
        data1 = recuperata + "\n" + "dr. " + data[2] + " " + data[3] +
"\nPacient: " + " " + data[4] + " " + data[5];
        dataGridView1.Rows[positionRow].Cells[positionCol].Value = data1;

dataGridView1.Rows[positionRow].Cells[positionCol].Style.BackColor =
changeColor();
    }
}
}

```

---

```

private void spec_btn_Click(object sender, EventArgs e)
{
    if (spec_groupBox.Visible == false)
    {
        List<string> list = new List<string>();
        ArrayOfString arrString = new ArrayOfString();
        arrString.AddRange(list);
        service.listSpecialization(ref arrString);

        listBox1.DataSource = arrString;
        med_btn.Visible = true;
        m_groupBox.Visible = false;
        spec_groupBox.Visible = true;
    }
    else if (spec_groupBox.Visible == true)
        spec_groupBox.Visible = false;
}

private void med_btn_Click(object sender, EventArgs e)
{
    if (m_groupBox.Visible == false)
    {
        List<string> list = new List<string>();
        ArrayOfString arrString = new ArrayOfString();
        arrString.AddRange(list);
        string spec = listBox1.SelectedItem.ToString();
        service.DoctorsfromSpecialization(spec, ref arrString);

        m_list.DataSource = arrString;

        spec_groupBox.Visible = false;
        m_groupBox.Visible = true;
    }
    else if (m_groupBox.Visible == true)
        m_groupBox.Visible = false;
}

private void prog_btn_Click(object sender, EventArgs e)
{
    string descriere, data, m_lName, m_fName, p_Lname = " ", p_fName = "
", pCnp = this.cnp;
    descriere = desc_txtBox.Text;
    data = date_txtBox.Text;
    service.listPacienti(pCnp, ref p_fName, ref p_Lname);
    string[] numeMedic = m_list.SelectedItem.ToString().Split(' ');
    m_lName = numeMedic[0];
    m_fName = numeMedic[1];
    int opt;
    opt = service.programari(pCnp, p_Lname, p_fName, descriere, data,
m_fName, m_lName);
    if (opt == 1)
    {
        MessageBox.Show("Programarea a fost realizata cu succes!");
        populateDataGridView();
    }
    else if (opt == 0)
    {
        MessageBox.Show("Doctorul " + m_lName + " " + m_fName + " este
ocupat la data aleasa!");
    }
}

```

---

```

        else
        {
            MessageBox.Show("ERROR -1: Nu se poate realiza programarea! Au
aparut probleme de sistem.");
        }
    }

    private void Appointments_Load(object sender, EventArgs e)
    {

    }

    private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
    {

    }

    private void Appointments_FormClosing(object sender, FormClosingEventArgs
e)
    {
        Properties.Settings.Default.Save();
    }
    private Color changeColor()
    {
        Random r = new Random();
        int R, G, B;
        R = r.Next(0, 255);
        G = r.Next(0, 255);
        B = r.Next(0, 255);
        Color myRGB = new Color();
        myRGB = Color.FromArgb(R, G, B);

        return myRGB;
    }

    private void label1_Click(object sender, EventArgs e)
    {

    }
}

```