

Olimpiada de Tehnologie a Informației (OTI) – județ 2022

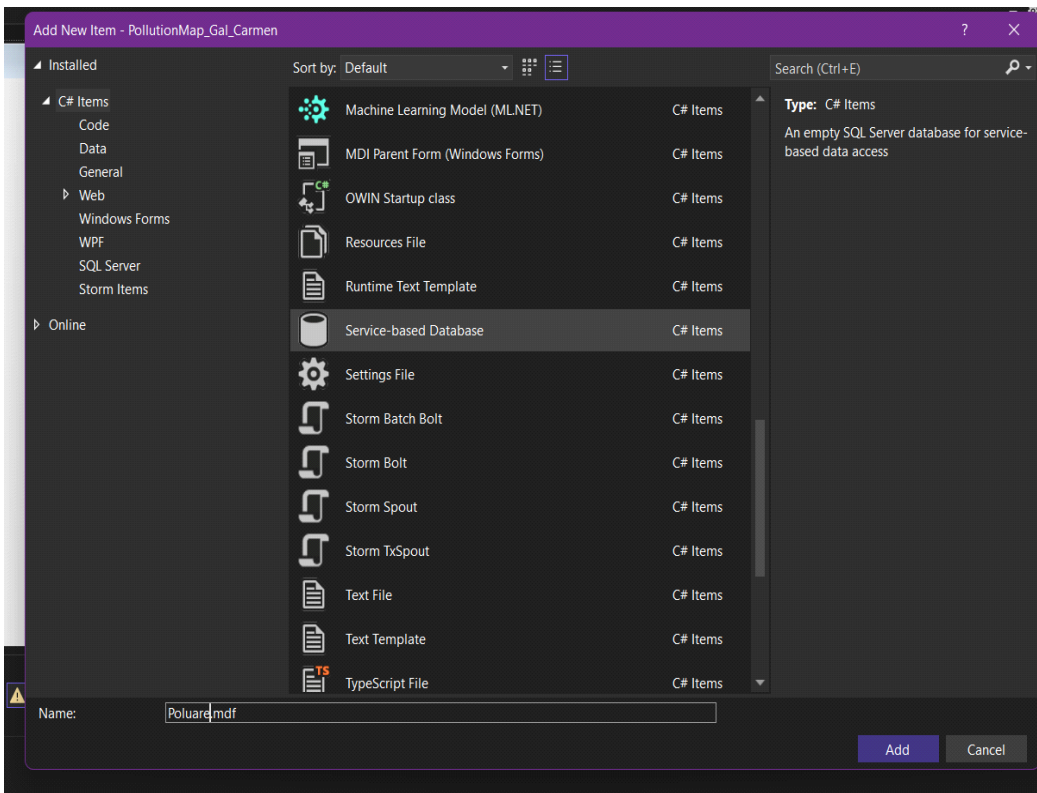
prof. Carmen Gal

O posibilă rezolvare (toate fișierele de tip imagine și text vor fi salvate în fișierul bin\debug)

Cuprins

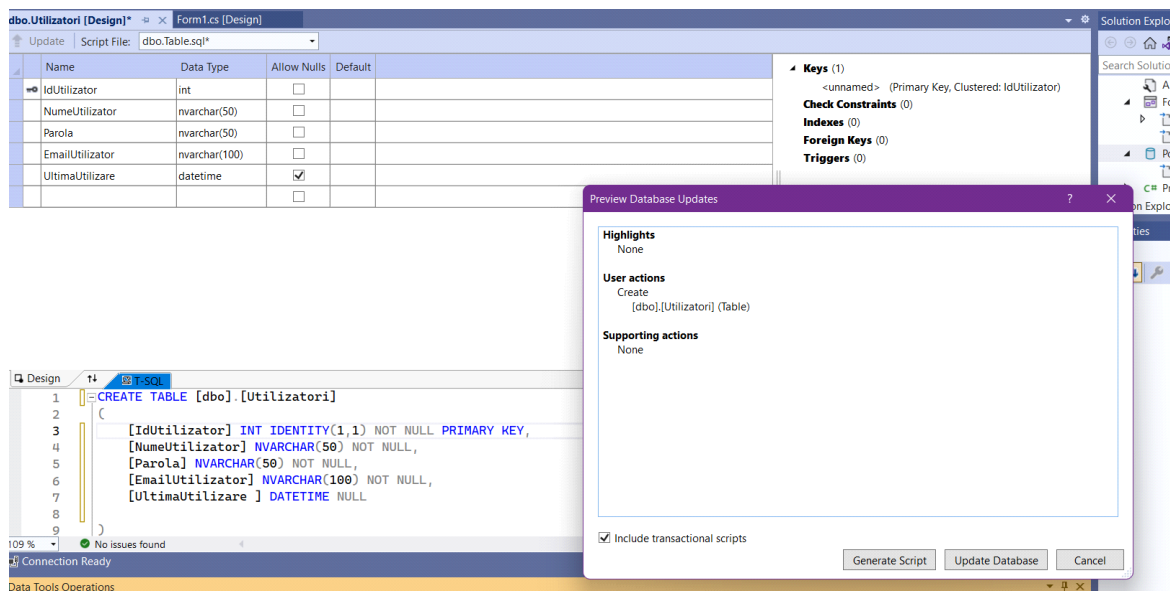
1. Crearea bazei de date Poluare	2
2. Adăugarea sursei de date	4
3. Crearea funcțiilor de Inserare în tabele	6
4. Adăugarea datelor în tabelul Utilizatori(manual)	8
5. Formularul Autentificare.....	8
5.1. Citire din fișierele de tip text și popularea tabelelor Harti și Masurare cu informațiile din fișiere	8
5.2 Verificare dacă citirea a fost făcută corect	9
5.2. Codul corespunzător din formularul Autentificare	10
6. Adăugarea formularului Înregistrare	12
7. Formularul Vizualizare	19
7.1. Adăugarea controalelor pe suprafața formularului.....	19
7.2. Introducere cod pentru afișare hartă/puncte în funcție de data și filtrul aplicat.....	23
7.3. Formularul AdăugaMasurare și stabilirea legăturii între cele două formulare	26
8. Fila Traseu și desenarea liniilor (distanța minimă) între punctul selectat și cele două puncte de valoare maximă	30

1. Crearea bazei de date Poluare



Crearea tabelor:

Tabelul Utilizatori:



Tabelul a fost adaugat la baza de date:

Table	Allow Nulls	Default
Utilizatori	<input checked="" type="checkbox"/>	
IdUtilizator	<input type="checkbox"/>	
NumeUtilizator	<input type="checkbox"/>	
Parola	<input type="checkbox"/>	
EmailUtilizator	<input type="checkbox"/>	
UltimaUtilizare	<input type="checkbox"/>	

[Utilizatori]

IDENTITY(1,1) NOT NULL PRIMARY KEY,
ARCHAR(50) NOT NULL,
0) NOT NULL,
VARCHAR(100) NOT NULL,
DATETIME NULL

Tabelul Harti:

Name	Data Type	Allow Nulls	Default
IdHarta	int	<input type="checkbox"/>	
NumeHarta	nvarchar(50)	<input type="checkbox"/>	
FisierHarta	nvarchar(256)	<input type="checkbox"/>	

Keys (1)
<unnamed> (Primary Key, Clustered: IdHarta)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

Preview Database Updates

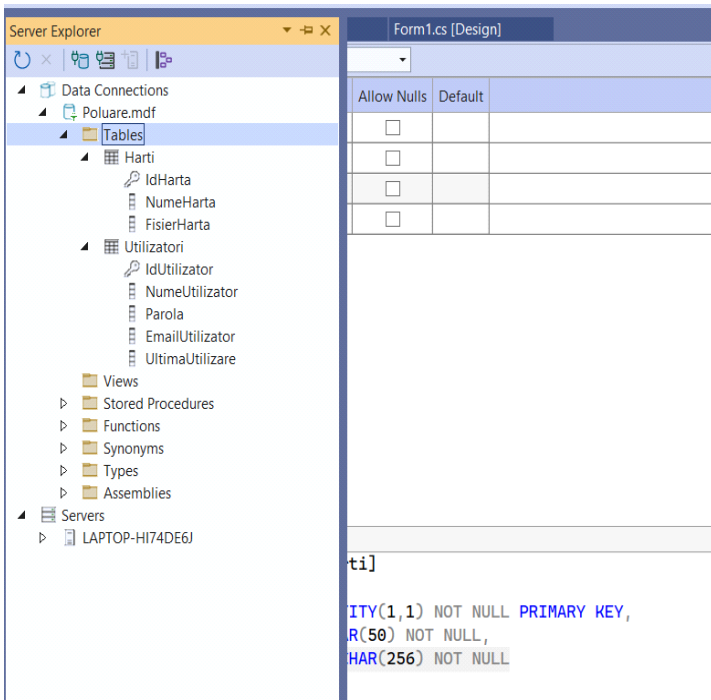
Highlights: None
User actions: None
Supporting actions: None

☒ Include transactional scripts

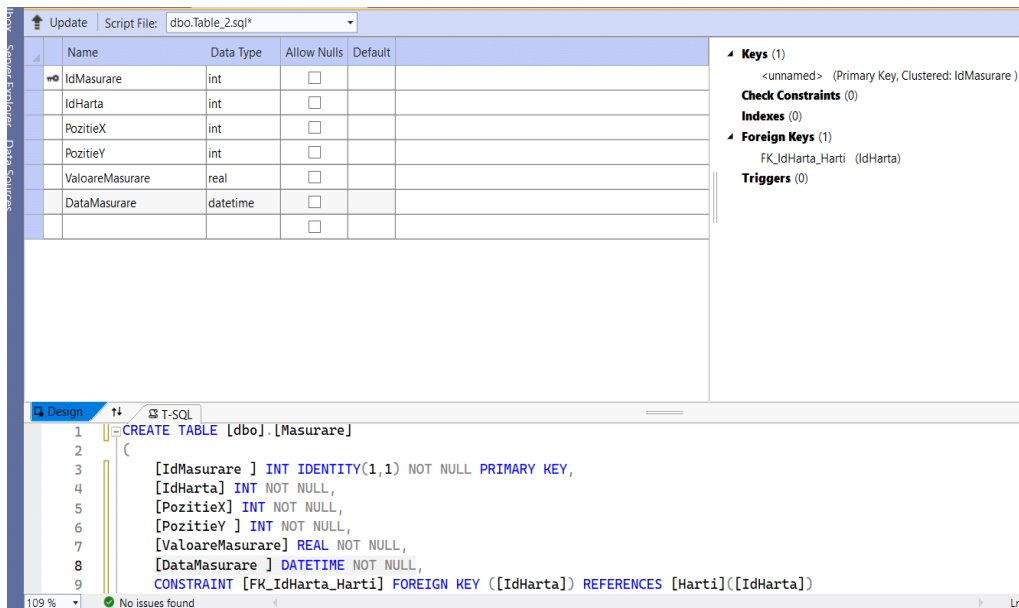
Generate Script Update Database Cancel

```
1 CREATE TABLE [dbo].[Harti]
2 (
3     [IdHarta] INT IDENTITY(1,1) NOT NULL PRIMARY KEY,
4     [NumeHarta] NVARCHAR(50) NOT NULL,
5     [FisierHarta] NVARCHAR(256) NOT NULL
6 )
7
```

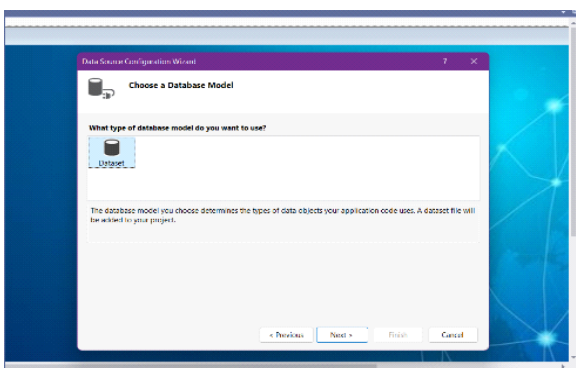
Daca mergem pe Update in ServerExplorer vom vedea noul tabel:



Crearea **tabelului Masurare** care va contine cheia straina IdHarta prin care tabelul va fi legat la tabelul Harti:

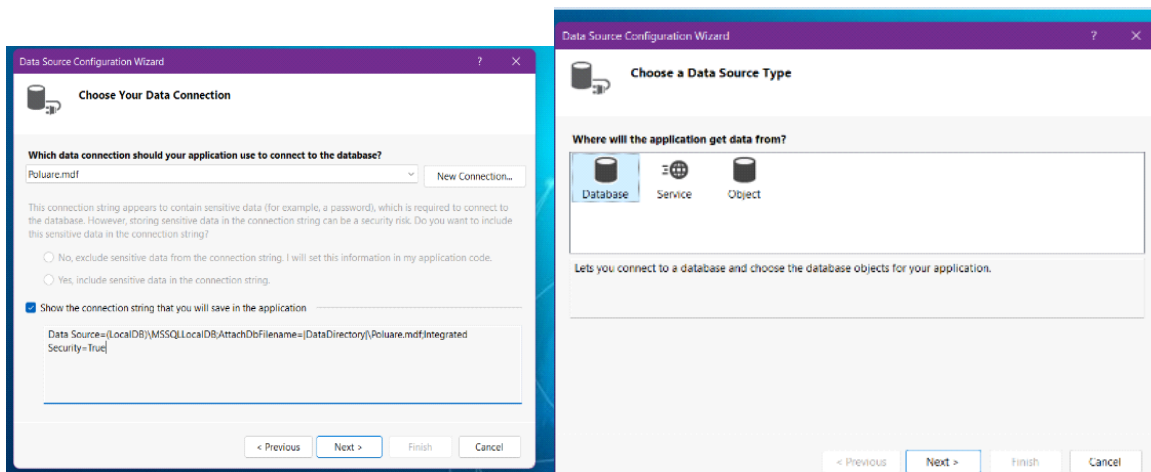


2. Adăugarea sursei de date



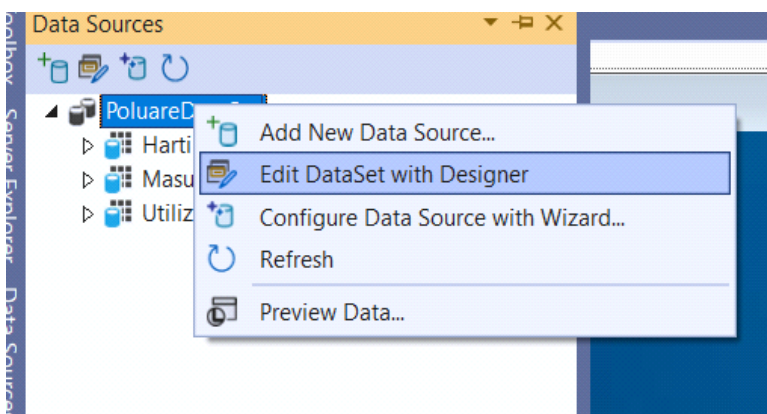
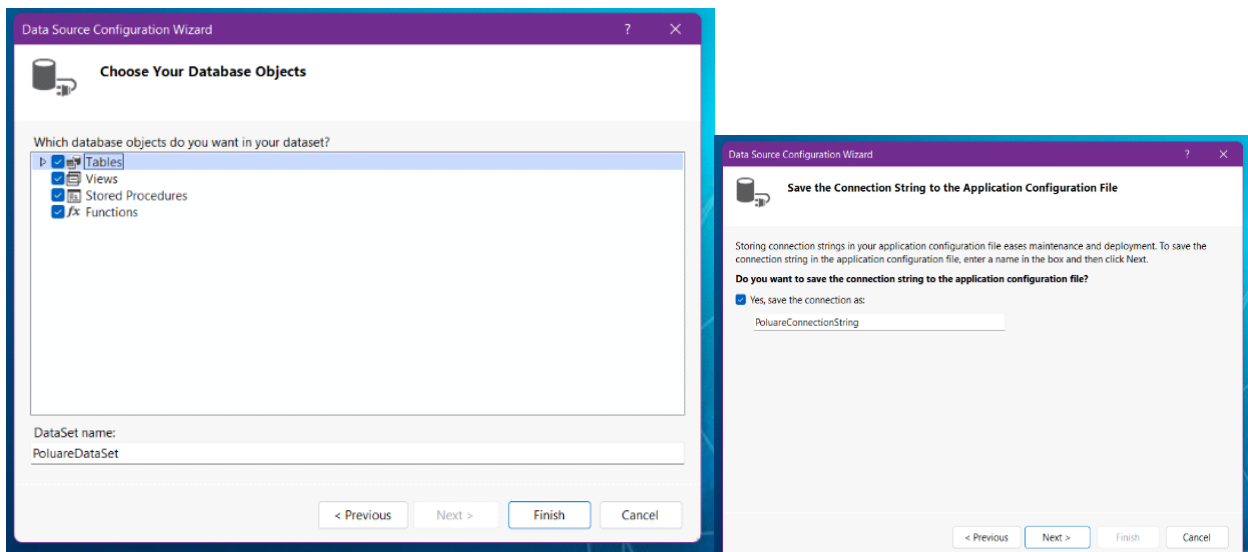
Your project currently has no data sources associated with it. Add a new data source, then data-bind items by dragging from this window onto forms or existing controls.

[Add New Data Source...](#)

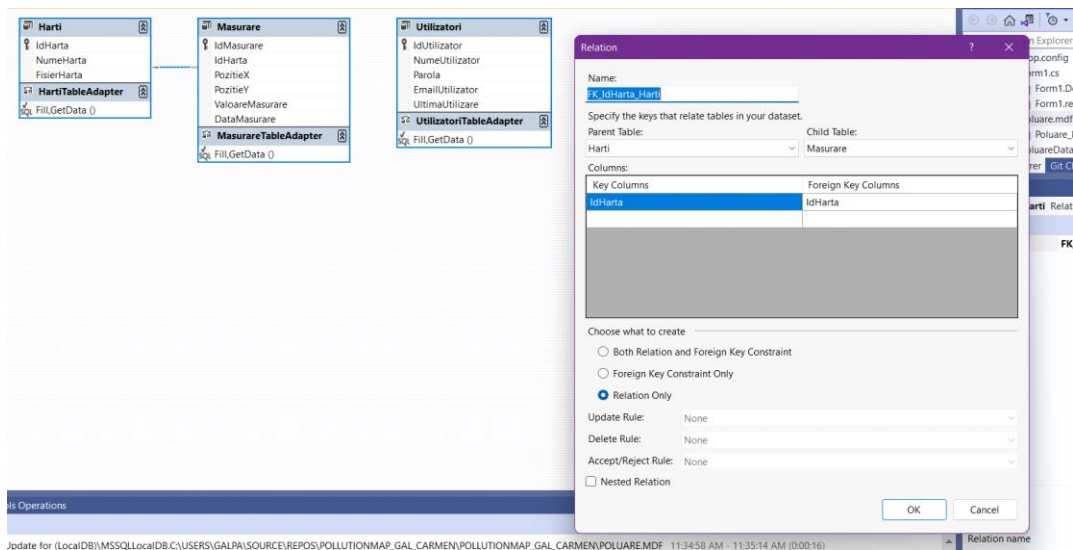


Connection string (este bine să fie copiat pentru a putea fi utilizat dacă e nevoie):

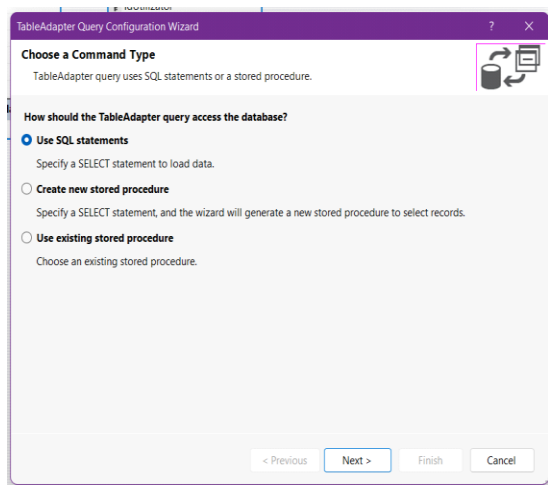
Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Poluare.mdf;Integrated Security=True



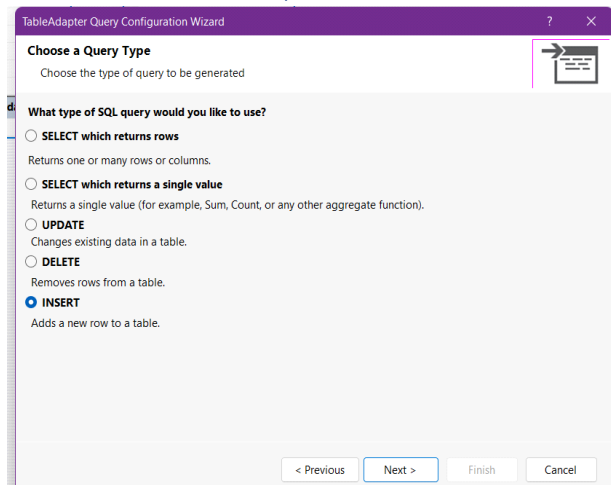
În versiunile mai noi relațiile sunt create automat (în cele mai vechi relația poate fi creată prin drag and drop)



Exista posibilitatea de a popula manual tabelele (caz in care se pierde din punctaj, dar nu foarte mult 4p din 100p) sau de a prelua prin cod din fisierele text existente in resurse – pentru punctaj maxim



3. Crearea funcțiilor de Inserare în tabele



TableAdapter Query Configuration Wizard

Specify a SQL INSERT statement

The INSERT statement will be used by the query.

Type your SQL statement or use the Query Builder to construct it. What data should be loaded into the table?

What data should the table load?

```
INSERT INTO [dbo].[Utilizatori] ([NumeUtilizator], [Parola], [EmailUtilizator], [UltimaUtilizare]) VALUES
(@NumeUtilizator, @Parola, @EmailUtilizator, @UltimaUtilizare);
SELECT IdUtilizator, NumeUtilizator, Parola, EmailUtilizator, [UltimaUtilizare] FROM Utilizatori WHERE (IdUtilizator =
SCOPE_IDENTITY())
```

Query Builder...

< Previous Next > Finish Cancel

TableAdapter Query Configuration Wizard

Specify a SQL INSERT statement

The INSERT statement will be used by the query.

Type your SQL statement or use the Query Builder to construct it. What data should be loaded into the table?

What data should the table load?

```
INSERT INTO [dbo].[Masurare] ([IdHarta], [PozitieX], [PozitieY], [ValoareMasurare], [DataMasurare]) VALUES
(@IdHarta, @PozitieX, @PozitieY, @ValoareMasurare, @DataMasurare);
SELECT [IdMasurare], IdHarta, PozitieX, [PozitieY], ValoareMasurare, [DataMasurare] FROM Masurare WHERE
(IdMasurare = SCOPE_IDENTITY())
```

Query Builder...

< Previous Next > Finish Cancel

TableAdapter Query Configuration Wizard

Choose Function Name

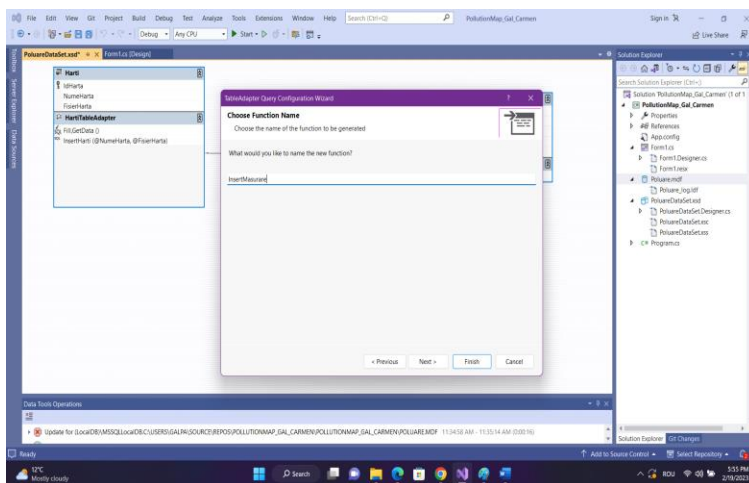
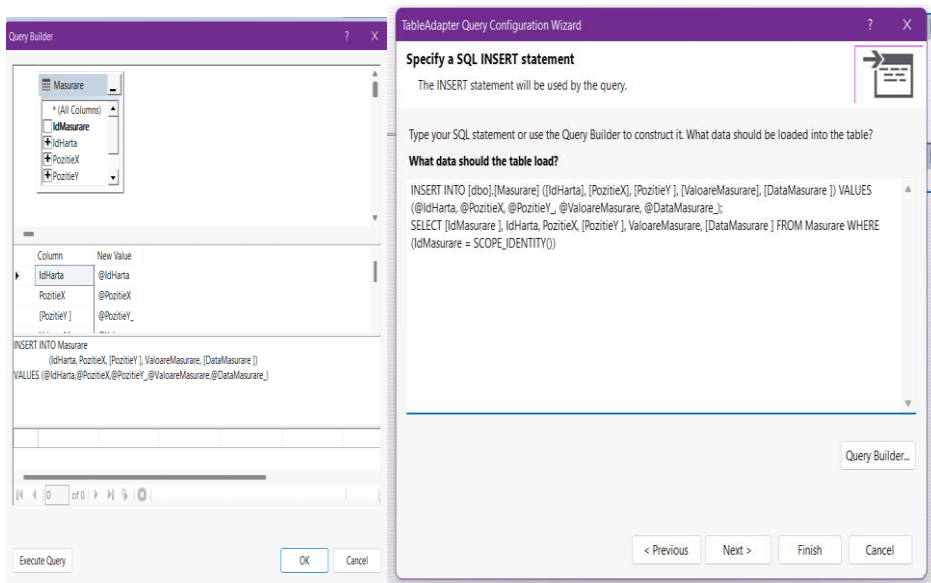
Choose the name of the function to be generated

What would you like to name the new function?

InsertHarta

< Previous Next > Finish Cancel

Inserare în tabelul Măsurare:

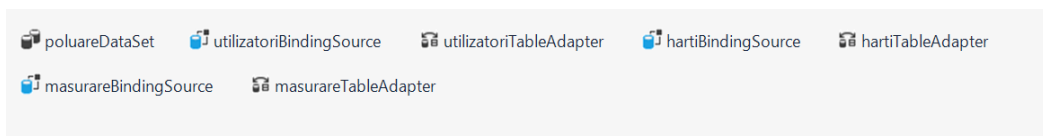


4. Adaugarea datelor in tabelul Utilizatori(manual)

IdUtilizator	NumeUtilizator	Parola	EmailUtilizator	UltimaUtilizare
1	oti2022	oti1234	oti2022@oti.com	NULL
2	carmen	123456	carmen@oti.com	NULL
NULL	NULL	NULL	NULL	NULL

5. Formularul Autentificare

in primul rand vom pune controalele pentru incarcarea bazei de date pe suprafata formularului Autentificare



5.1. Citire din fisierele de tip text si popularea tabelelor Harti si Masurare cu informatiile din fisiere

Pentru citirea din fisier vom folosi biblioteca:


```
using System.IO;
```

Fișierele din care se citesc datele vor fi salvate în bin\debug

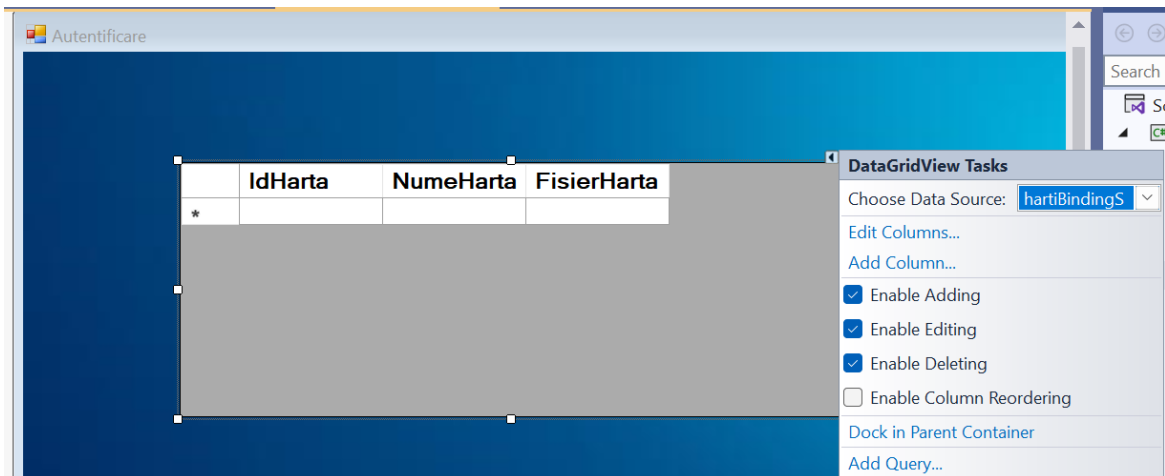
Pentru citirea din fișier:

```
private void IncarcareDate()
{
    string[] cuv = new string[10];
    string s;
    if(this.hartiTableAdapter.GetData().Rows.Count==0)//daca tabelul nu a fost populat
    {
        StreamReader sr = new StreamReader("harti.txt");//deschiderea fisierului pt citire
        while (!sr.EndOfStream)
        {
            s = sr.ReadLine();//citirea linie cu linie in variabila s
            cuv = s.Split('#');//impartirea si retinerea in vectorul de cuvinte cuv
            this.hartiTableAdapter.InsertHarti(cuv[0], cuv[1]);//inserare in tabelul harti
        }
        sr.Close();
    }
}

1 reference
private void Form1_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the 'poluareDataSet1.Utilizatori' table. You can move, or remove it, as
    this.utilizatoriTableAdapter.Fill(this.poluareDataSet1.Utilizatori);
    IncarcareDate();
    // TODO: This line of code loads data into the 'poluareDataSet1.Harti' table. You can move, or remove it, as needed
    this.hartiTableAdapter.Fill(this.poluareDataSet1.Harti);
    // TODO: This line of code loads data into the 'poluareDataSet1.Masurare' table. You can move, or remove it, as needed
    this.masurareTableAdapter.Fill(this.poluareDataSet1.Masurare);
}
```

5.2 Verificare dacă citirea a fost făcută corect

Pentru a verifica dacă datele au fost încărcate pe suprafața formularului vom pune un dataGridView pe care îl legăm la tabelul Harti. Acest control îl vom șterge după ce am verificat dacă operația de încărcare a datelor a fost realizată.



Autentificare

	IdHarta	NumeHarta	FisierHarta
▶	1	Harta Bucuresti	harta_bucurest...
	2	Harta Cluj-Nap...	harta_cluj.png
	3	Harta Constanta	harta_constant...
	4	Harta Iasi	harta_iasi.png
	5	Harta Sibiu	harta_sibiu.png
•			

La fel pentru verificare incarcare tabel Masurare:

	IdMasurare	IdHarta	PozitieX	PozitieY	ValoareMasurare
▶	1	1	10	30	15
	2	1	100	150	40
	3	1	50	340	41
	4	1	150	98	60
	5	1	10	300	35
	6	1	300	350	45
	7	1	450	340	48
	8	1	250	200	44
	9	1	263	112	33
	10	1	416	191	45
	11	1	145	232	47
	12	1	445	68	17
	13	1	55	153	15
	14	1	127	332	44

5.2. Codul corespunzător din formularul Autentificare

```
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;
using System.IO;

namespace PollutionMap_GalCarmen
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void IncarcareDate()
        {
            string[] cuv = new string[10];
            string s;
            if (this.hartiTableAdapter.GetData().Rows.Count == 0)//daca tabelul nu a fost populat
```

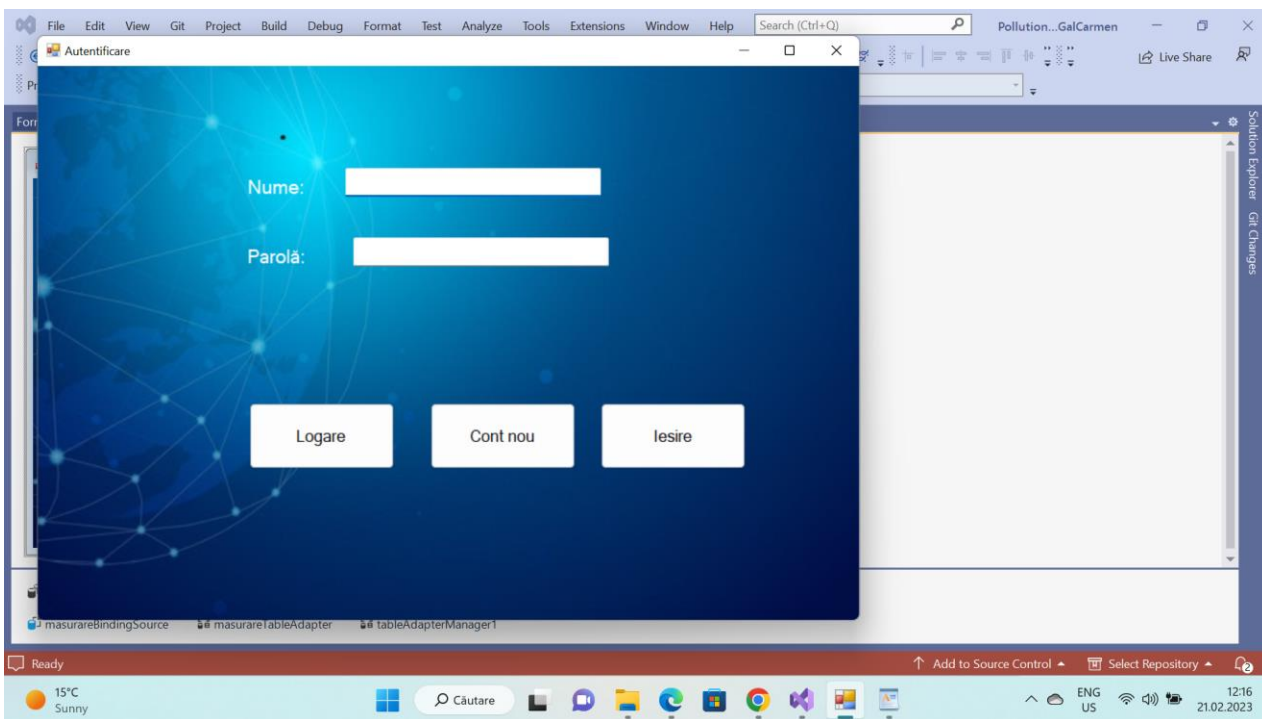
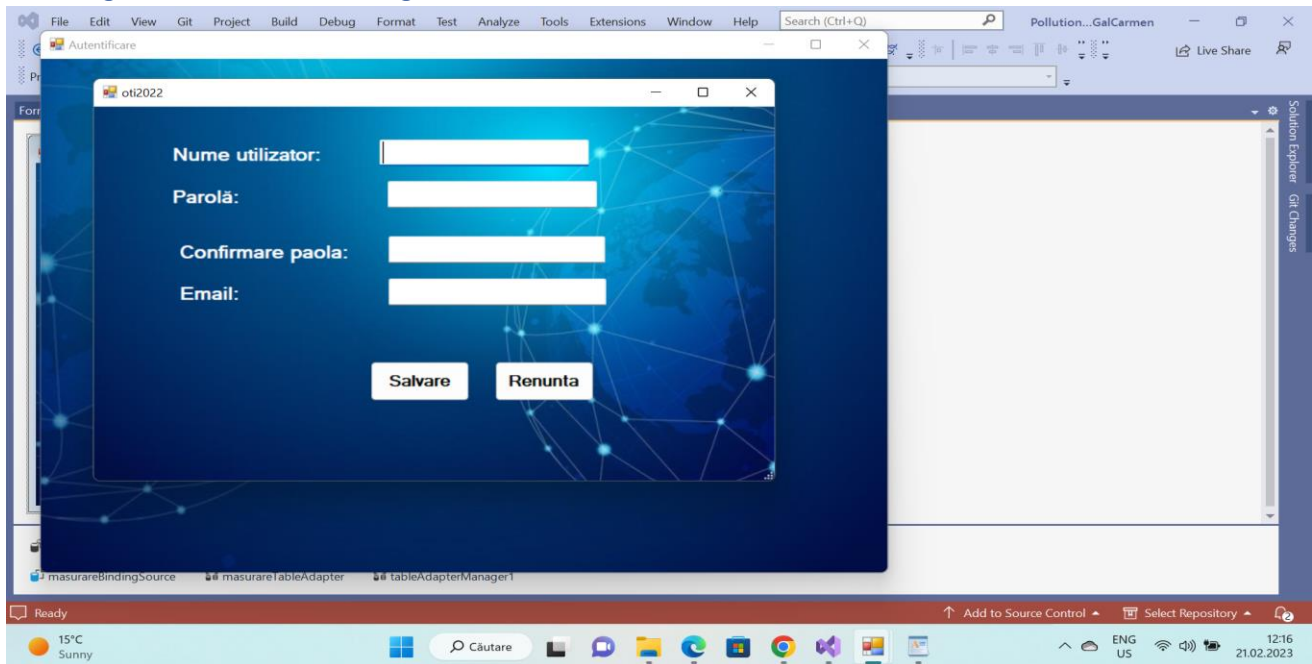
```

{
    StreamReader sr = new StreamReader("harti.txt");//deschiderea fisierului pt
citire
    while (!sr.EndOfStream)
    {
        s = sr.ReadLine();//citirea linie cu linie in variabila s
        cuv = s.Split('#');//impartirea si retinerea in vectorul de cuvinte cuv
        this.hartiTableAdapter.InsertHarti(cuv[0], cuv[1]);//inserare in tabelul
harti
    }
    sr.Close();
}
if (this.masurareTableAdapter.GetData().Rows.Count == 0)//daca tabelul nu a fost
populat
{
    StreamReader sr = new StreamReader("masurari.txt");//deschiderea fisierului pt
citire
    while (!sr.EndOfStream)
    {
        s = sr.ReadLine();//citirea linie cu linie in variabila s
        cuv = s.Split('#');//impartirea si retinerea in vectorul de cuvinte cuv
        int id=0;
        if (cuv[0] == "Harta Bucuresti") id = 1;
        else
            if (cuv[0] == "Harta Cluj-Napoca") id = 2;
        else
            if (cuv[0] == "Harta Constanta") id = 3;
        else
            if (cuv[0] == "Harta Iasi") id = 4;
        else
            if (cuv[0] == "Harta Sibiu") id = 5;
        this.masurareTableAdapter.InsertMasurare(id,
            Convert.ToInt32(cuv[1]), Convert.ToInt32(cuv[2]),
            float.Parse(cuv[3]), Convert.ToDateTime(cuv[4]));
        //inserare in tabelul masurare
    }
    sr.Close();
}
}
private void Form1_Load(object sender, EventArgs e)
{
    this.utilizatoriTableAdapter.Fill(this.poluareDataSet.Utilizatori);
    IncarcareDate();
    // TODO: This line of code loads data into the 'poluareDataSet.Harti' table. You can
move, or remove it, as needed.
    this.hartiTableAdapter.Fill(this.poluareDataSet.Harti);
    // TODO: This line of code loads data into the 'poluareDataSet.Masurare' table. You
can move, or remove it, as needed.
    this.masurareTableAdapter.Fill(this.poluareDataSet.Masurare);
}
}
}

```

Acum după ce am verificat dacă datele au fost introduse din cele două fișiere de tip text în tabele putem să ștergem Controlul de tipul DataGridView de pe suprafața formularului și să rezolvăm următoarele cerințe:

6. Adaugarea formularului Inregistrare



La apasarea butonului Cont nou din formularul principal se va deschide formularul inregistrare. Pentru aceasta la evenimentul Cont nou se introduce codul:

```
private void button2_Click(object sender, EventArgs e)
{
    Inregistrare f2= new Inregistrare();
    f2.ShowDialog();//se deschide formularul Inregistrare
    this.Hide();
}
```

Autentificare

oti2022

Nume utilizator: Patricia

Parolă: *****

Confirmare paola: *****

Email: patri@gmail.com

Salvare Renunta

oti2022

Nume utilizator: Patricia

Parolă: *****

Confirmare paola: *****

Email: patri@gmail.

Salvare Renunta

Utilizatorul a fost adaugat cu succes!

OK

Nume utilizator: Patricia

Parolă: *****

Confirmare paola: *****

Email: qa@oti.ro

Salvare Renunta

Utilizatorul exista! Alege alt nume

OK

Trebuie sa facem o serie de verificari:

- Lungimea numelui mai mare de 4 caractere

```
if (textBox1.Text.Length<=4)
{
```

```
    MessageBox.Show("Numele de utilizator trebuie sa aiba cel" +
```

```

        "putin 4 caractere");
        return;
    }
    • Parola să aibă minimum 6 caractere și să coincidă cu cea din câmpul de confirmare a parolei:
    if (textBox2.Text.Length < 6)//parola să aibă minimum 6 caractere
    {
        MessageBox.Show("Parola trebuie sa aiba cel putin 6 caractere");
        return;
    }
    if(textBox3.Text.Trim()!=textBox2.Text.Trim()) //și să coincidă
        //cu cea din câmpul de confirmare a parolei
    {
        MessageBox.Show("Cele doua parole trebuie sa coincida!");
        return;
    }
    • adresa de email să fie validă
    bibliotecile:
    using System.Net.Mail;
    using System.Text.RegularExpressions;//se putea folosi regex, dar am folosit varianta simpla
    Se apeleaza functia:

```

```

public bool IsValid(string emailaddress)
{
    try
    {
        MailAddress m = new MailAddress(emailaddress);

        return true;
    }
    catch (FormatException)
    {
        return false;
    }
}

if(!IsValid(textBox4.Text))//apelare functia de validare a parolei
{
    MessageBox.Show("Adresa de email incorecta!");
    return;
}

numele utilizatorului să fie unic la nivelul bazei de date:
foreach (DataRowView row in utilizatoriBindingSource.List)
{
    //numele utilizatorului să fie unic la nivelul bazei de date;
    // obtinerea valorii din campul 'NumeUtilizator'
    string nume = (string)row["NumeUtilizator"];
    if(nume.ToString().Trim()==textBox1.Text.Trim())
    {
        MessageBox.Show("Utilizatorul exista! Alege alt nume");
        return;
    }
}

```

```

daca a fost adaugat cu succes:
this.utilizatoriTableAdapter1.InsertUtilizatori(textBox1.Text,
        textBox2.Text, textBox4.Text, DateTime.Now);
this.Validate();
this.utilizatoriBindingSource.EndEdit();
this.tableAdapterManager1.UpdateAll(poluareDataSet);
MessageBox.Show("Utilizatorul a fost adaugat cu succes!");
Tot codul din formularul Inregistrare:
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;
using System.Net.Mail;
using System.Text.RegularExpressions;
using System.Data.SqlClient;
using System.Data.Sql;

namespace PollutionMap_GalCarmen
{
    public partial class Inregistrare : Form
    {
        public Inregistrare()
        {
            InitializeComponent();

            public bool IsValid(string emailaddress)
            {
                try
                {
                    MailAddress m = new MailAddress(emailaddress);

                    return true;
                }
                catch (FormatException)
                {
                    return false;
                }
            }

            public void adaugaUtilizator()
            {
                this.utilizatoriTableAdapter1.InsertUtilizatori(textBox1.Text,
                    textBox2.Text, textBox4.Text, DateTime.Now);
                this.Validate();
                this.utilizatoriBindingSource.EndEdit();
                this.tableAdapterManager1.UpdateAll(poluareDataSet);
                MessageBox.Show("Utilizatorul a fost adaugat cu succes!");
            }

            private void button1_Click(object sender, EventArgs e)
            {
                Form1 f1=new Form1();
                if (textBox1.Text.Length<=4)//lungimea parolei >4 caractere
                {
                    MessageBox.Show("Numele de utilizator trebuie sa aiba cel " +
                        "putin 4 caractere");
                    return;
                }
                if (textBox2.Text.Length < 6)//parola să aibă minimum 6 caractere
                {
                    MessageBox.Show("Parola trebuie sa aiba cel putin 6 caractere");
                    return;
                }
                if(textBox3.Text.Trim()!=textBox2.Text.Trim()) //și să coincidă
                    //cu cea din câmpul de confirmare a parolei
                {
                    MessageBox.Show("Cele doua parole trebuie sa coincidă!");
                    return;
                }
                if(!IsValid(textBox4.Text))//apelare functia de validare a parolei

```



```

    {
        MessageBox.Show("Adresa de email incorecta!");
        return;
    }
    foreach (DataRowView row in utilizatoriBindingSource.List)
    {
        //numele utilizatorului să fie unic la nivelul bazei de date;
        // obținerea valorii din campul 'NumeUtilizator'
        string nume = (string)row["NumeUtilizator"];
        if(nume.ToString().Trim()==textBox1.Text.Trim())
        {
            MessageBox.Show("Utilizatorul exista! Alege alt nume");
            return;
        }
    }
    adaugaUtilizator();
    f1.Show();
    this.Close();
}

private void Inregistrare_Load(object sender, EventArgs e)
{
    this.utilizatoriTableAdapter1.Fill(this.poluareDataSet.Utilizatori);
}


private void button2_Click(object sender, EventArgs e)
{
    Form1 f1=new Form1();
    f1.Show();
    this.Close();
}
}
}

```

Acum pentru logare va trebui sa verificam daca utilizatorul cu parola introduse exista in baza de date. Daca da, se va deschide formularul de vizualizare

Cream o interogare pt modificarea datei (de tip Update Query) UpdateUltimaUtilizare:

tableAdapter Query Configuration Wizard
?
X

Specify a SQL UPDATE statement
The UPDATE statement will be used by the query.


Type your SQL statement or use the Query Builder to construct it. What data should be loaded into the table?

What data should the table load?

```

UPDATE Utilizatori
SET     [UltimaUtilizare] = @UltimaUtilizare;
SELECT  [UltimaUtilizare] FROM Utilizatori WHERE (IdUtilizator = @IdUtilizator)

```

```

private bool existaUtilizator()//functia de verificare daca exista sau nu utilizatorul
{
    this.utilizatoriTableAdapter.Fill(this.poluareDataSet.Utilizatori);
    using (this.utilizatoriTableAdapter)
    {
        foreach (DataRowView row in utilizatoriBindingSource.List)
        {
            //numele utilizatorului să existe si parola sa fie corecte
            string nume = (string)row["NumeUtilizator"];
            string parola = (string)row["Parola"];
            if (nume.ToString().Trim() == textBox1.Text.Trim() &&
                parola.ToString().Trim() == textBox2.Text.Trim())
            {
                int idut = Convert.ToInt32(row["IdUtilizator"]);

```



```

        this.utilizatoriTableAdapter.UpdateUltimaUtilizare(DateTime.Now, idut);
        return true;
    }

    }
}
return false;
}
if (existaUtilizator() == true)
{
    Vizualizare f3= new Vizualizare();
    f3.ShowDialog();
    this.Hide();
}
else
    MessageBox.Show("Utilizatorul nu exista in baza de date");

```

Tot codul corespunzator (modificat) formularului Form1:

```

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;
using System.IO;
using System.Data.OleDb;
using System.Data.SqlClient;
using System.Data.Sql;

namespace PollutionMap_GalCarmen
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void IncarcareDate()
        {
            string[] cuv = new string[10];
            string s;
            if (this.hartiTableAdapter.GetData().Rows.Count == 0)//daca tabelul nu a fost populat
            {
                StreamReader sr = new StreamReader("harti.txt");//deschiderea fisierului pt
citire
                while (!sr.EndOfStream)
                {
                    s = sr.ReadLine();//citirea linie cu linie in variabila s
                    cuv = s.Split('#');//impartirea si retinerea in vectorul de cuvinte cuv
harti
                    this.hartiTableAdapter.InsertHarti(cuv[0], cuv[1]);//inserare in tabelul
                }
                sr.Close();
            }
            if (this.masurareTableAdapter.GetData().Rows.Count == 0)//daca tabelul nu a fost
populat
            {

```

```

StreamReader sr = new StreamReader("masurari.txt");//deschiderea fisierului pt
citire
while (!sr.EndOfStream)
{
    s = sr.ReadLine();//citirea linie cu linie in variabila s
    cuv = s.Split('#');//impartirea si retinerea in vectorul de cuvinte cuv
    int id=0;
    if (cuv[0] == "Harta Bucuresti") id = 1;
    else
        if (cuv[0] == "Harta Cluj-Napoca") id = 2;
    else
        if (cuv[0] == "Harta Constanta") id = 3;
    else
        if (cuv[0] == "Harta Iasi") id = 4;
    else
        if (cuv[0] == "Harta Sibiu") id = 5;
    this.masurareTableAdapter.InsertMasurare(id,
        Convert.ToInt32(cuv[1]), Convert.ToInt32(cuv[2]),
        float.Parse(cuv[3]), Convert.ToDateTime(cuv[4]));
    //inserare in tabelul masurare
}
sr.Close();
}
}
private void Form1_Load(object sender, EventArgs e)
{
    this.utilizatoriTableAdapter.Fill(this.poluareDataSet.Utilizatori);
    IncarcareDate();
    // TODO: This line of code loads data into the 'poluareDataSet.Harti' table. You can
move, or remove it, as needed.
    this.hartiTableAdapter.Fill(this.poluareDataSet.Harti);
    // TODO: This line of code loads data into the 'poluareDataSet.Masurare' table. You
can move, or remove it, as needed.
    this.masurareTableAdapter.Fill(this.poluareDataSet.Masurare);
}

private bool existaUtilizator()//functia de verificare daca exista sau nu utilizatorul
{
    this.utilizatoriTableAdapter.Fill(this.poluareDataSet.Utilizatori);
    using (this.utilizatoriTableAdapter)
    {
        foreach (DataRowView row in utilizatoriBindingSource.List)
        {
            //numele utilizatorului să existe si parola sa fie corecte
            string nume = (string)row["NumeUtilizator"];
            string parola = (string)row["Parola"];
            if (nume.ToString().Trim() == textBox1.Text.Trim() &&
                parola.ToString().Trim() == textBox2.Text.Trim())
            {
                int idut = Convert.ToInt32(row["IdUtilizator"]);
                this.utilizatoriTableAdapter.UpdateUltimaUtilizare(DateTime.Now, idut);
                return true;
            }
        }
    }
    return false;
}
private void button2_Click(object sender, EventArgs e)
{
    Inregistrare f2= new Inregistrare();
    f2.ShowDialog();
    this.Hide();
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    if (existaUtilizator() == true)
    {
        Vizualizare f3= new Vizualizare();
        f3.ShowDialog();
        this.Hide();
    }
    else
        MessageBox.Show("Utilizatorul nu exista in baza de date");
}

private void button3_Click(object sender, EventArgs e)
{
    Application.Exit();
}
}

```

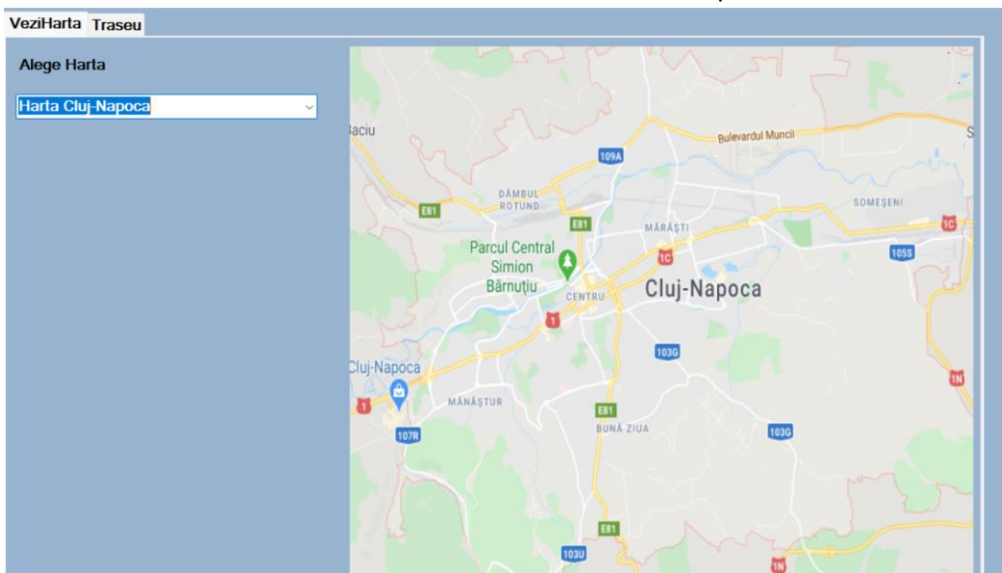
7. Formularul Vizualizare

7.1. Adăugarea controalelor pe suprafața formularului.

7.1.1. Legare ComboBox la câmpul Numeharta din tabelul Harti

The screenshot illustrates the process of binding a ComboBox to a data source in a Windows Forms application. The main window, titled "Vizualizare", contains a tab labeled "VeziHarta Traseu". A ComboBox control is selected, and the "ComboBox Tasks" dialog box is open. In this dialog, the "Use Data Bound Items" checkbox is checked. Under the "Data Binding Mode" section, the "Data Source" is set to "hartiBindingS", the "Display Member" is "NumeHarta", the "Value Member" is empty, and the "Selected Value" is "(none)". Below the dialog, a list of data sources is displayed: "poluareDataSet1", "hartitableAdapter1", "masurareTableAdapter1", "tableAdapterManager1", and "hartiBindingSource". At the bottom of the image, a preview of the ComboBox shows a list of cities: "Harta Bucuresti", "Harta Cluj-Napoca", "Harta Constanta", "Harta Iasi", and "Harta Sibiu".

La selectarea unei Harti se va afisa in PictureBox harta corespunzatoare:



Pentru realizarea acestui lucru am declarat o variabila globala s de tip String, o variabila IdHarta careia ii atribuim o valoare cuprinsa intre 1 si 5

```
string s="";  
int idHarta=0;
```

si am scris urmatorul cod:

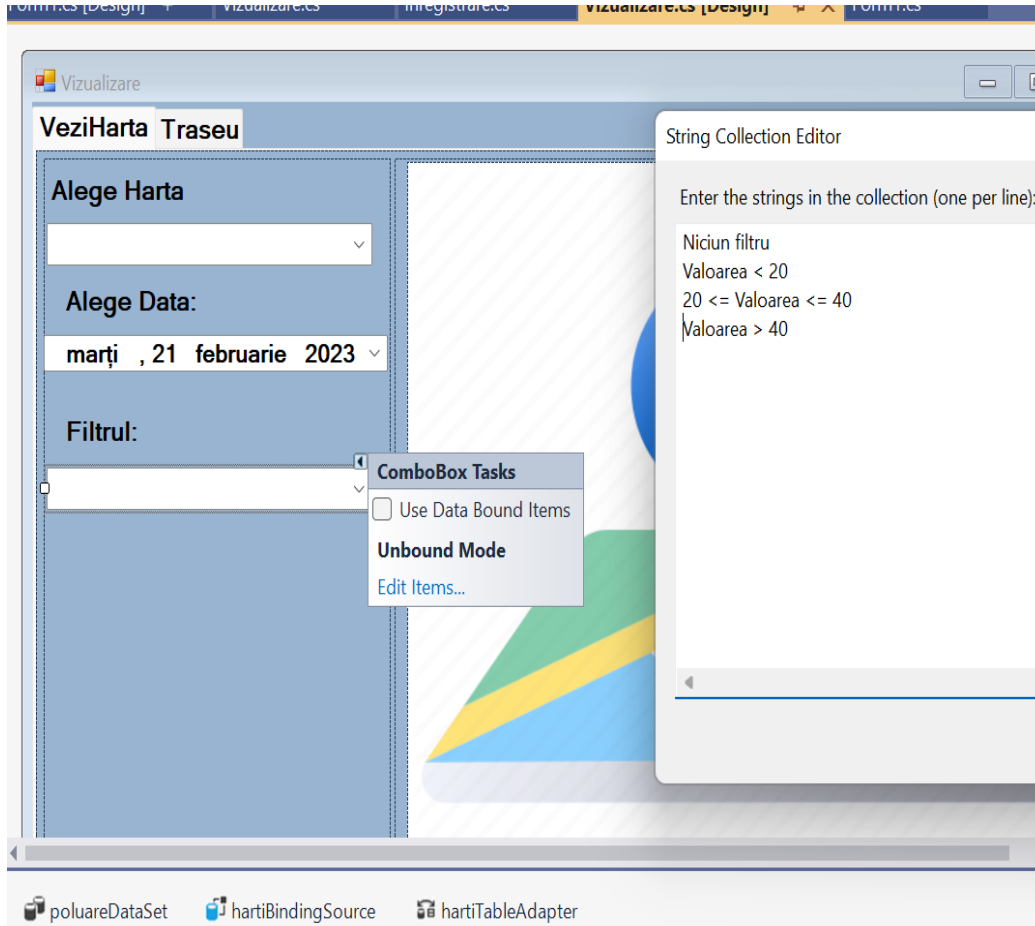
```
private void Vizualizare_Load(object sender, EventArgs e)  
{  
    // TODO: This line of code loads data into the 'poluareDataSet.Harti' table. You can  
    move, or remove it, as needed.  
    this.hartiTableAdapter.Fill(this.poluareDataSet.Harti);  
    pictureBox1.SizeMode=PictureBoxSizeMode.StretchImage;  
}  
private void AfisareHarta()  
{  
    if (s == "Harta Bucuresti")  
    {  
        pictureBox1.Image = Image.FromFile("harta_bucuresti.png");  
        idHarta = 1;  
    }  
    else  
        if (s == "Harta Cluj-Napoca")  
        {  
            pictureBox1.Image = Image.FromFile("harta_cluj.png");  
            idHarta = 2;  
        }  
    else  
        if (s == "Harta Constanta")  
        {  
            pictureBox1.Image = Image.FromFile("harta_constanta.png");  
            idHarta = 3;  
        }  
    else  
        if (s == "Harta Iasi")  
        {  
            pictureBox1.Image = Image.FromFile("harta_iasi.png");  
            idHarta = 4;  
        }  
    else  
        if (s == "Harta Sibiu")
```

```

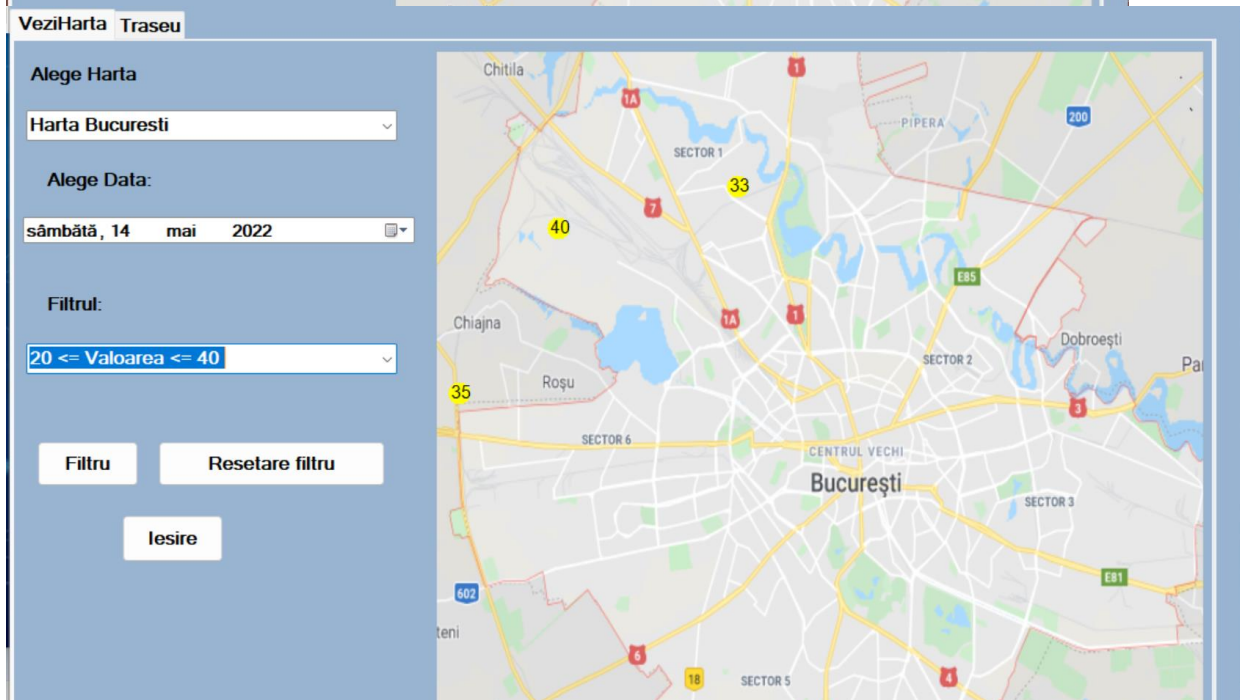
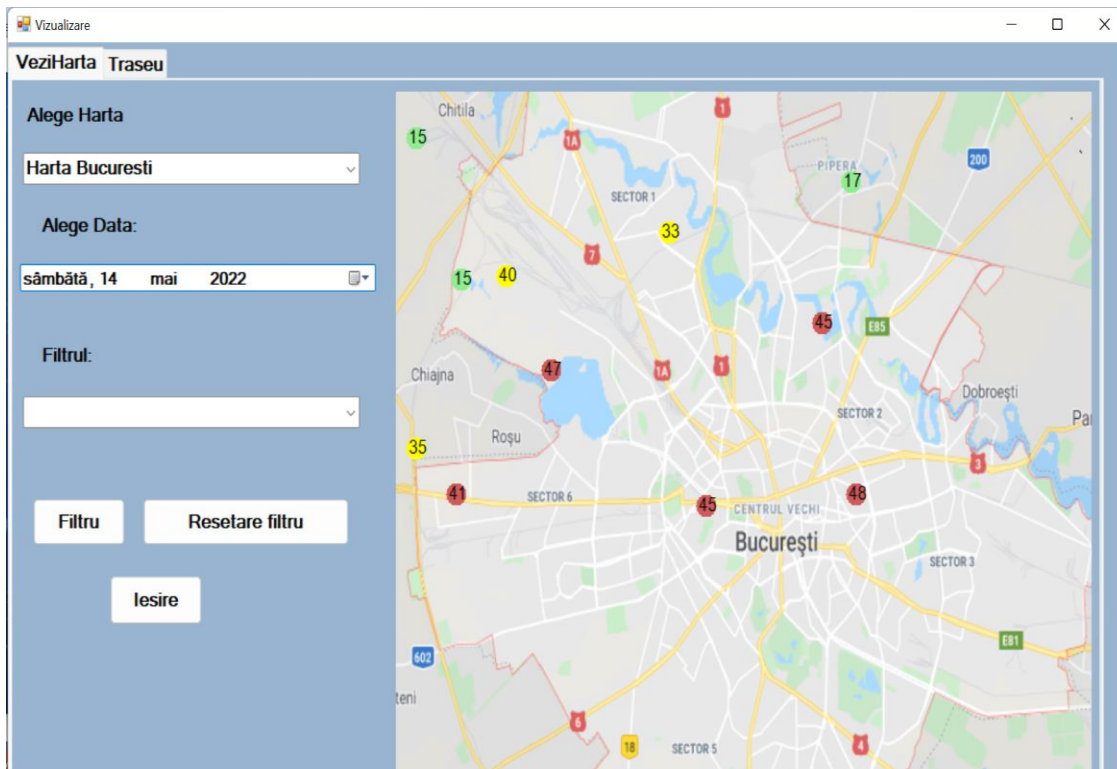
    {
        pictureBox1.Image = Image.FromFile("harta_sibiu.png");
        idHarta = 5;
    }
}
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    s= comboBox1.Text;
    AfisareHarta();
}
}
}

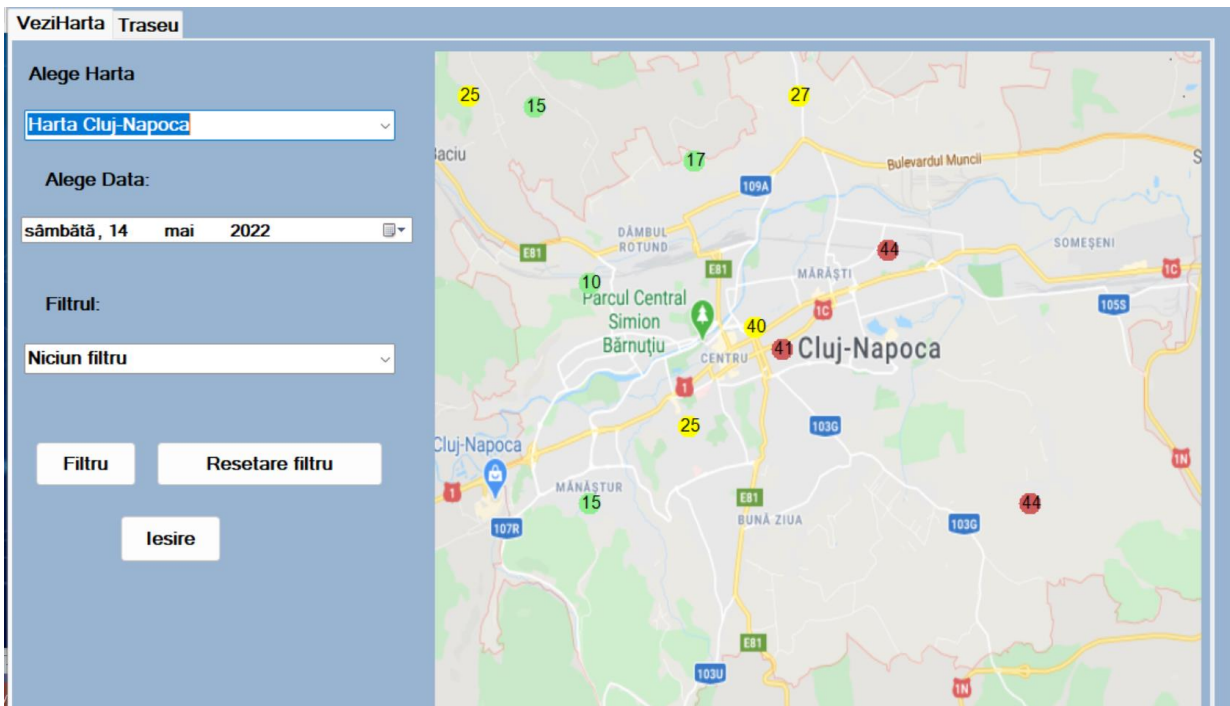
```

7.1.2. Adaugarea altor elemente (DateTimePicker si ComboBox):



7.1.3. Afișare puncte pe hartă și aplicare filtru





7.2. Introducere cod pentru afișare hartă/puncte în funcție de data și filtrul aplicat

```
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 PollutionMap_GalCarmen
{
    public partial class Vizualizare : Form
    {
        string s="",s1=""; //retin valorile din comboBox
        DateTime ds; //data din DateTimePicker
        int idHarta=0, filtru=0;
        Graphics g1=null, g2= null; //obiectele grafice pt deseneare
        int[] pozX = new int[100]; //vector pt retinerea coordonatelor pe Ox
        int[] pozY = new int[100]; //vector pt retinerea coordonatelor pe Oy
        int n = 0; //nr de puncte care vor fi desenate pe harta
        double[] vm=new double[100]; //valorile masurate coresp. punctelor

        public Vizualizare()
        {
            InitializeComponent();
        }

        private void Vizualizare_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the 'poluareDataSet.Harti' table. You can
            move, or remove it, as needed.
            this.hartiTableAdapter.Fill(this.poluareDataSet.Harti);
            this.masurareTableAdapter1.Fill(this.poluareDataSet.Masurare);
            pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage;
            ds = DateTime.Now;
        }
    }
}
```



```

        g1=this.pictureBox1.CreateGraphics();//desenare in PictureBox
        //g2 = this.pictureBox2.CreateGraphics();//desenare in PictureBox
    }
    private void AfisareHarta()
    {

```

```

        if (s == "Harta Bucuresti")//selectare imagine coresp in
        {//functie de optiunea aleasa in ComboBox1
            pictureBox1.Image = Image.FromFile("harta_bucuresti.png");
            idHarta = 1;//si incarcare in PictureBox1
        }
        else
            if (s == "Harta Cluj-Napoca")
            {
                pictureBox1.Image = Image.FromFile("harta_cluj.png");
                idHarta = 2;
            }
            else
                if (s == "Harta Constanta")
                {
                    pictureBox1.Image = Image.FromFile("harta_constanta.png");
                    idHarta = 3;
                }
                else
                    if (s == "Harta Iasi")
                    {
                        pictureBox1.Image = Image.FromFile("harta_iasi.png");
                        idHarta = 4;
                    }
                    else
                        if (s == "Harta Sibiu")
                        {
                            pictureBox1.Image = Image.FromFile("harta_sibiu.png");
                            idHarta = 5;
                        }

```

```

    }

    private void desenare(Graphics g, Brush b,int px,int py, double valmas)
    {//desenarea punctelor pe suprafata hartii
        if ((idHarta >= 1 && idHarta <= 5) &&
            (filtru >= 1 && filtru <= 4))
        {
            Rectangle re;
            re = new Rectangle(px-10, py-10, 20, 20);
            g.FillEllipse(b,re);//desenare cercuri
            g.DrawString(Convert.ToString(valmas),
                new Font(new FontFamily("Arial"), 12),Brushes.Black, px-10, py-10);
        }
    }
    //scrierea valorilor

```

```

    private void AfisareValori(Graphics g)
    {
        DateTime dm;
        int id=0;
        if ((idHarta >= 1 && idHarta <= 5)&&
            (filtru>=1&&filtru<=4))
        {//parcurgem tabelul harti rand cu rand a tabelului masurare
            n = 0;
            foreach (DataRow r in poluareDataSet.Masurare)//parcurerea tabelului Masurare
            {//linie cu linie
                dm = Convert.ToDateTime(Convert.ToDateTime(r["DataMasurare"]));//se preia
                data
                id = Convert.ToInt32(r["IdHarta"]);//id-ul hartii
                din tabel
                if (dm.Date==ds.Date && id==idHarta)//daca data selectata corespunde cu cea
                {
                    //si sa corespunda hartii selectate in ComboBox
                }
            }
        }
    }
}

```



```

n++; //creste nr de puncte
pozX[n] = Convert.ToInt32(r["PozitieX"]); //se retin coord punctelor
pozY[n] = Convert.ToInt32(r["PozitieY"]);
vm[n] = Convert.ToDouble(r["ValoareMasurare"]); //se retine valoarea
if (filtru <= 2 && vm[n] < 20) //in functie de filtru si valoare se apeleaza
functia de desenare
    desenare(g, Brushes.LightGreen, pozX[n], pozY[n], vm[n]);
else
    if ((filtru == 3 || filtru == 1) && (vm[n] >= 20 && vm[n] <= 40))
        desenare(g, Brushes.Yellow, pozX[n], pozY[n], vm[n]);
    else
        if ((filtru == 4 || filtru == 1) && vm[n] > 40)
            desenare(g, Brushes.IndianRed, pozX[n], pozY[n], vm[n]);
        }
    }
}
}
private void button3_Click(object sender, EventArgs e)
{
    Application.Exit(); //iesirea din aplicatie
}

private void OperatiiAfisare()
{
    AfisareHarta();
    if (idHarta >= 1 && idHarta <= 5)
    {
        this.Refresh();
        AfisareValori(g1); //daca a fost selectata harta se afiseaza valorile
    }
}

private void button1_Click(object sender, EventArgs e)
{
    OperatiiAfisare();
}

private void button2_Click(object sender, EventArgs e)
{
    filtru = 1;
    s1 = comboBox2.Text = "Niciun filtru";
    OperatiiAfisare();
}

private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
    ds = dateTimePicker1.Value; //preluam data si ora
    ds = Convert.ToDateTime(ds);
    OperatiiAfisare();
}

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    //selectare harta
    s = comboBox1.Text.Trim();
    filtru = 1;
    s1 = comboBox2.Text = "Niciun filtru";
    OperatiiAfisare();
}

private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
{
    //selectare filtru- variabila filtru retine optiunea
    s1 = comboBox2.Text.Trim();
    if (s1 == "Niciun filtru")
        filtru = 1;
}

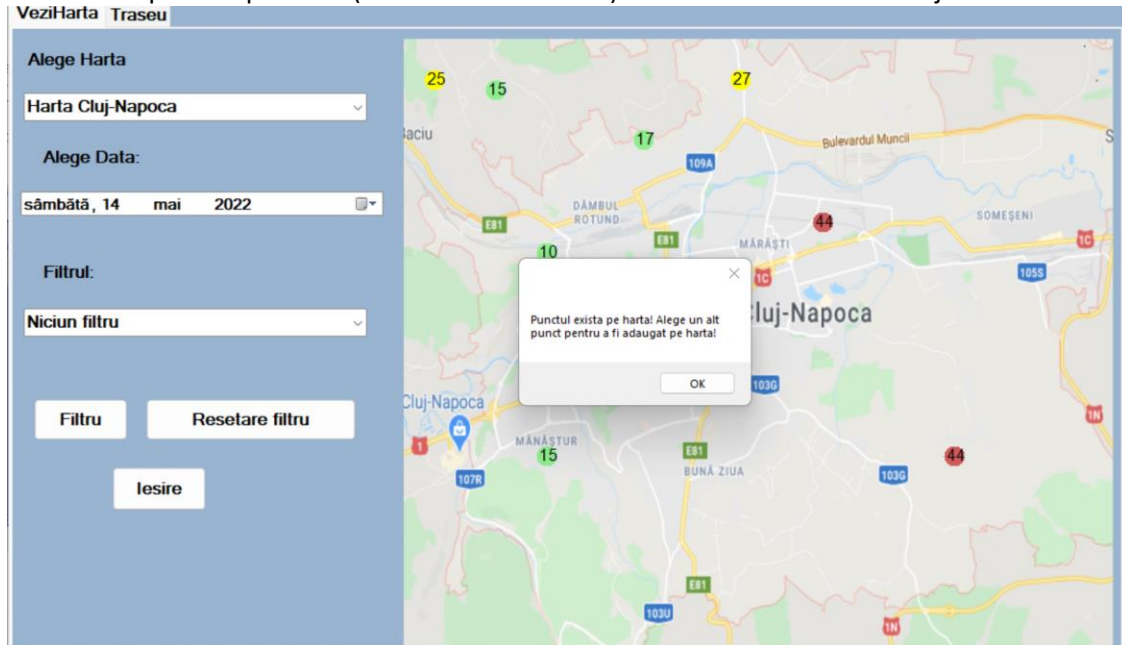
```

```

else
    if(s1== @"Valoarea < 20")
        filtru = 2;
else
    if(s1== @"20 <= Valoarea <= 40")
        filtru = 3;
else
    if (s1 == @"Valoarea > 40")
        filtru = 4;
    OperatiiAfisare();
}
}
}

```

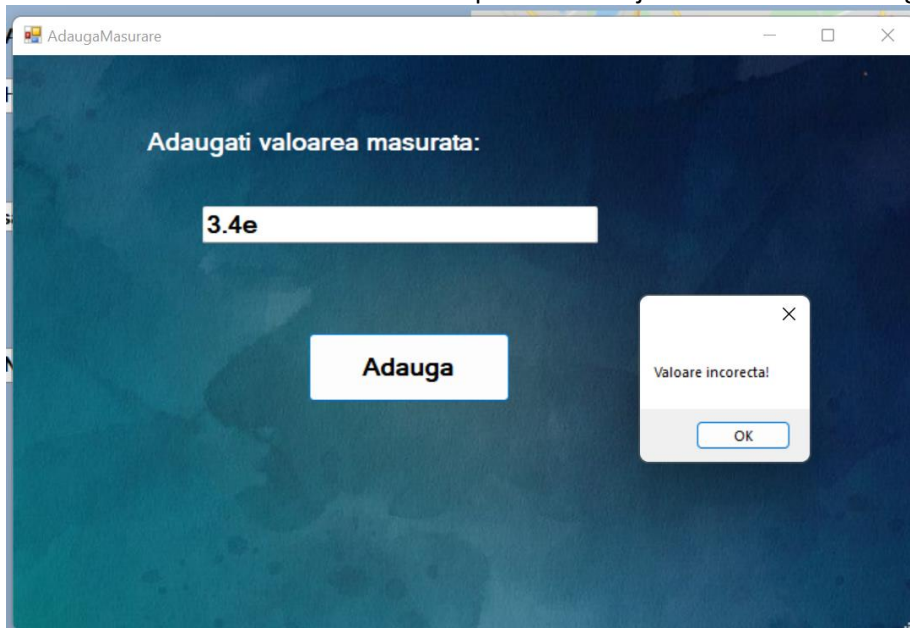
Daca exista punctul pe harta (cand se executa click) atunci se va afisa un mesaj:



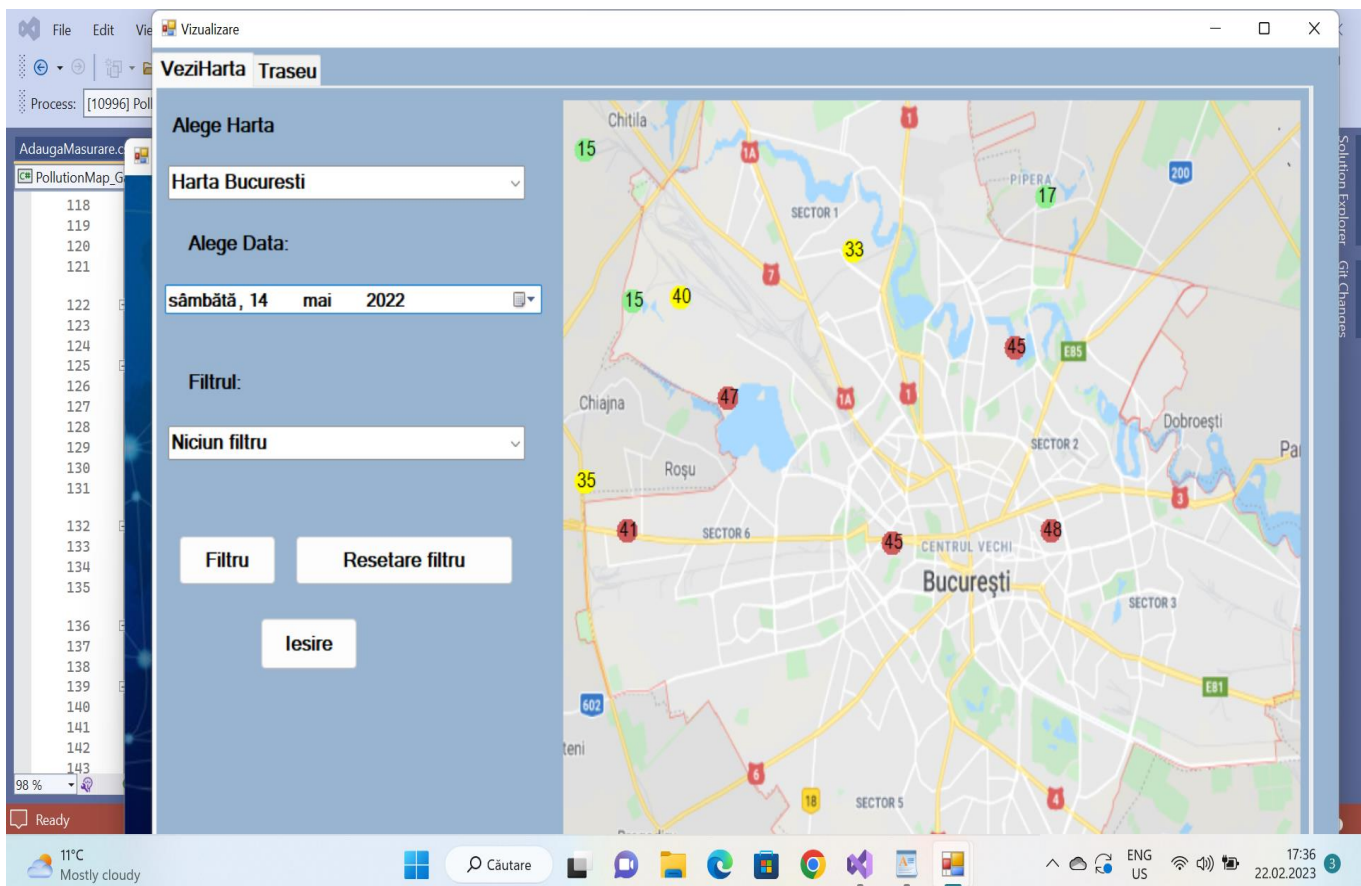
7.3. Formularul AdaugaMasurare si stabilirea legăturii între cele două formulare

Daca se executa click unde nu exista un punct se deschide formularul AdaugaMasurare:

Daca se introduce o valoare incorecta apare un mesaj si caseta de text va fi golita de continut:



Executam click pe harta:



Valoarea adaugata pe harta va fi:

Alege Harta

Harta Bucuresti

Alege Data:

sâmbătă, 14 mai 2022

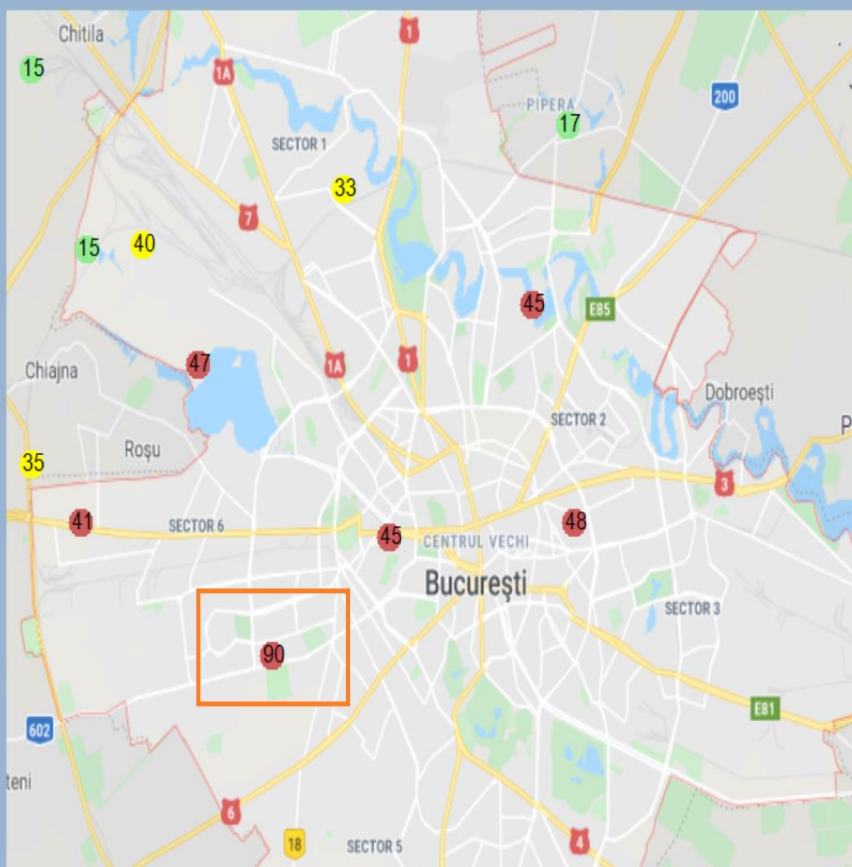
Filtrul:

Niciun filtru

Filtru

Resetare filtru

Iesire



Alege Harta

Harta Bucuresti

Alege Data:

sâmbătă, 14 mai 2022

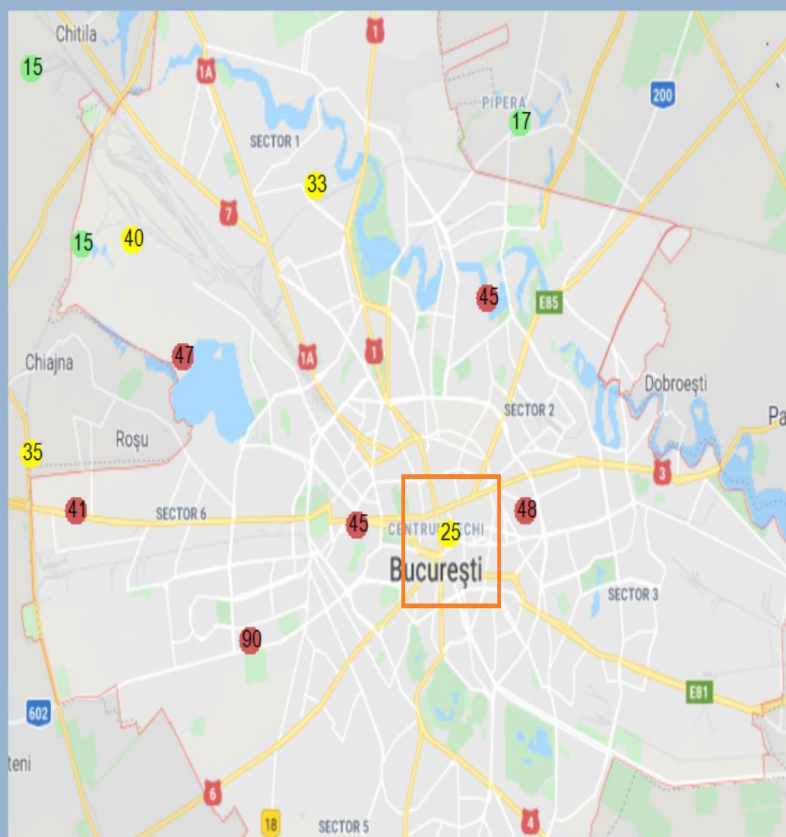
Filtrul:

Niciun filtru

Filtru

Resetare filtru

Iesire



Am adaugat doua valori:

In formularul Vizualizare se adauga variabilele:


```

int xc, yc, xmouse, ymouse; //coordonatele punctului la care se face click pe PictureBox1
//si coordonatele mouse-ului
double valm; //valoarea care va fi inserata
in form load scriem:
g2 = this.pictureBox2.CreateGraphics(); //desenare in PictureBox
Si functiile:

```

```

private bool ExistaPunctPeHarta(int x, int y)
{
    for (int i = 1; i <= n; i++)
        if (Math.Abs(x - pozX[i]) <= 20 && Math.Abs(y - pozY[i]) <= 20)
        { //se verifica daca punctul exista sau nu pe harta
            xc = pozX[i]; yc = pozY[i]; //se retin coordonatele exacte ale punctului
            valpc = vm[i];
            return true; //punctul exista pe harta
        }
    return false; //punctul nu exista pe harta
}

private void pictureBox1_MouseClick(object sender, MouseEventArgs e)
{
    xmouse = e.X; ymouse = e.Y; //coordonatele la care se face click
    if (ExistaPunctPeHarta(xmouse, ymouse) == true)
        MessageBox.Show("Punctul exista pe harta! Alege un alt\n" +
            "punct pentru a fi adaugat pe harta!");
    else
    {
        AdaugaMasurare f4 = new AdaugaMasurare();
        f4.ShowDialog();
        valm = f4.ValoareMasurare; //se retine valoarea introdusa in formularul de
adaugare

        MessageBox.Show(Convert.ToString("Se pune punctul avand valoarea:" + valm));
        Brush b;
        if (valm < 20) b = Brushes.LightGreen; //determinam culoarea cercului
        else if (valm <= 40) b = Brushes.Yellow; //care va fi desenat
        else b = Brushes.IndianRed;
        desenare(g1, b, xmouse, ymouse, valm); //se apeleaza functia de desenare
        this.masurareTableAdapter1.InsertMasurare(idHarta, xmouse, ymouse
            , (float)valm, Convert.ToDateTime(dateTimePicker1.Text));
        //se completeaza tabelul cu noua valoare
        this.masurareTableAdapter1.Fill(this.poluareDataSet.Masurare);
        this.masurareTableAdapter1.Update(this.poluareDataSet.Masurare);
    }
}

```

In formularul AdaugaMasurare vom introduce variabila publica ValoareMasurare si codul corespunzator:

```

namespace PollutionMap_GalCarmen
{
    public partial class AdaugaMasurare : Form
    {
        public double ValoareMasurare;
        public AdaugaMasurare()
        {
            InitializeComponent();
        }
        private void button1_Click(object sender, EventArgs e)
        {
            if (textBox1.Text != "" && double.TryParse(textBox1.Text, out double result))
            {
                ValoareMasurare = Convert.ToDouble(textBox1.Text);
                if (ValoareMasurare <= 0)

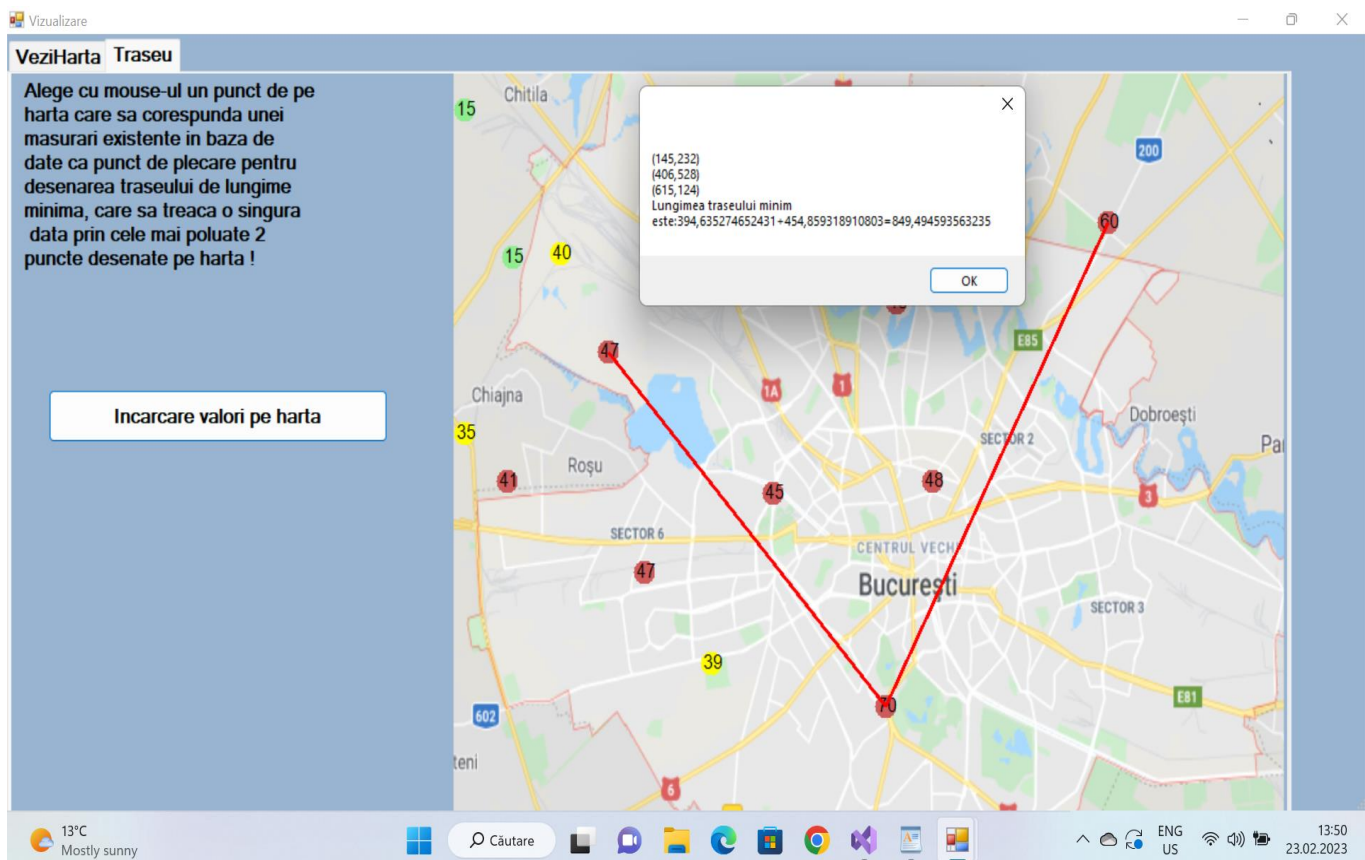
```

```

        return;
        this.Close();
    }
    else
    {
        MessageBox.Show("Valoare incorecta!");
        textBox1.Clear();
    }
}
}
}

```

8. Fila Traseu și desenarea liniilor (distanța minimă) între punctul selectat și cele două puncte de valoare maximă



```

private void DesenareLinie(int x1,int y1, int x2, int y2)
{
    g2.DrawLine(new Pen(Color.Red,3),x1, y1, x2, y2);
}
private void determinare_distante()
{
    d1 = Math.Sqrt(Math.Pow(xc - xmax1, 2) + Math.Pow(yc - ymax1, 2));
    d2 = Math.Sqrt(Math.Pow(xc - xmax2, 2) + Math.Pow(yc - ymax2, 2));
    d3 = Math.Sqrt(Math.Pow(xmax1 - xmax2, 2) + Math.Pow(ymax1 - ymax2, 2));
    if (d1<=d3 && d2<=d3)
    {

        lungimetraseu = d1 + d2;
        DesenareLinie(xmax1, ymax1, xc, yc);
        DesenareLinie(xc, yc, xmax2, ymax2);
        MessageBox.Show("(" + Convert.ToString(xmax1) + "," + Convert.ToString(ymax1) +
")\n" +
        "(" + Convert.ToString(xc) + "," + Convert.ToString(yc) + ")\n" +
        "(" + Convert.ToString(xmax2) + "," + Convert.ToString(ymax2) + ")\n" +

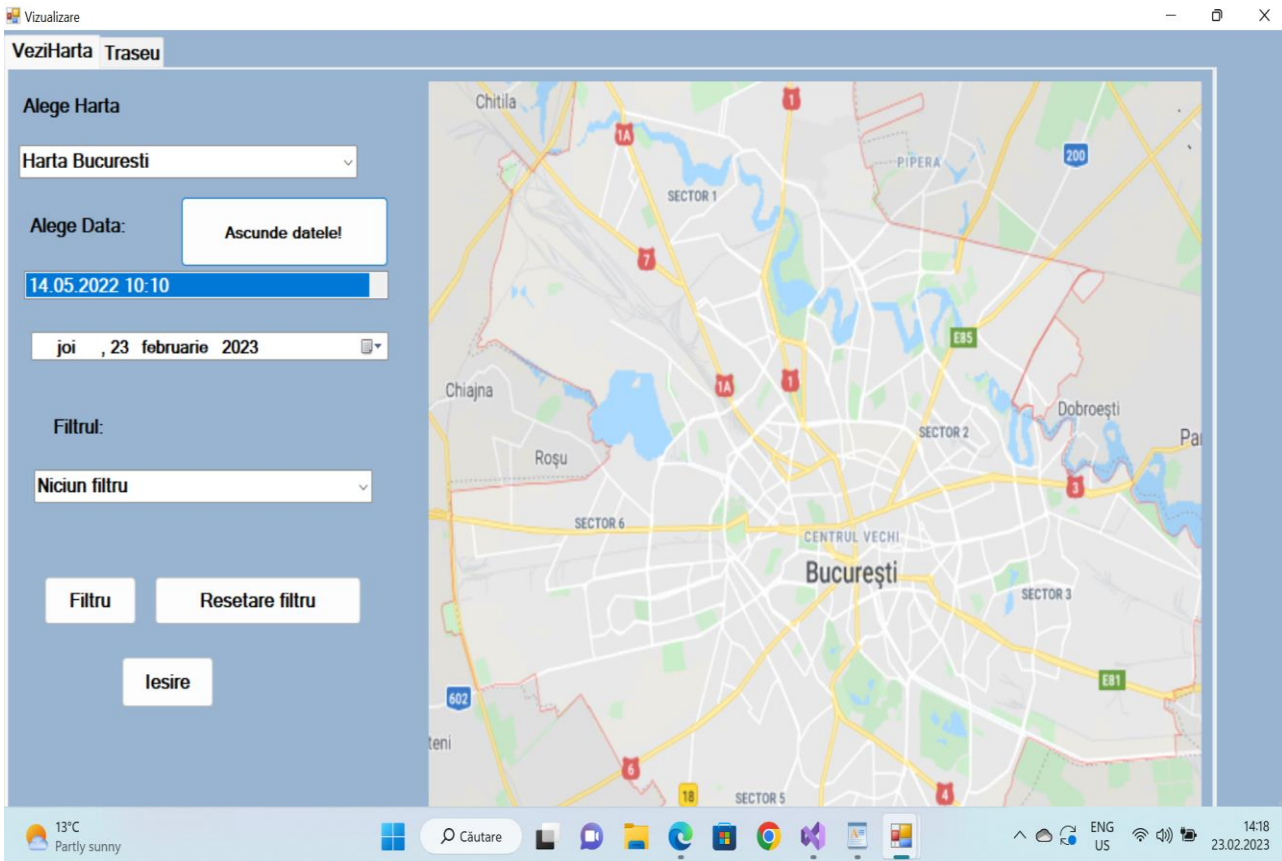
```

```

        "Lungimea traseului minim este:" + Convert.ToString(d1)+"+" +
        Convert.ToString(d2) + "=" + Convert.ToString(lungimetraseu));
    }
    else if(d1<=d2 && d3<=d2)
    {
        lungimetraseu = d1 + d3;
        DesenareLinie(xc, yc, xmax1, ymax1);
        DesenareLinie(xmax1, ymax1, xmax2, ymax2);
        MessageBox.Show("(" + Convert.ToString(xc) + "," + Convert.ToString(yc) + ")\n" +
            "(" + Convert.ToString(xmax1) + "," + Convert.ToString(ymax1) + ")\n" +
            "(" + Convert.ToString(xmax2) + "," + Convert.ToString(ymax2) + ")\n" +
            "Lungimea traseului minim este:" + Convert.ToString(d1) + "+" +
            Convert.ToString(d3) + "=" + Convert.ToString(lungimetraseu));
    }
    else
        if (d2<=d1 && d3<=d1)
        {
            lungimetraseu = d2 + d3;
            DesenareLinie(xc, yc, xmax2, ymax2);
            DesenareLinie(xmax2, ymax2, xmax1, ymax1);
            MessageBox.Show("(" + Convert.ToString(xc) + "," + Convert.ToString(yc) + ")\n" +
                "(" + Convert.ToString(xmax1) + "," + Convert.ToString(ymax1) + ")\n" +
                "(" + Convert.ToString(xmax2) + "," + Convert.ToString(ymax2) + ")\n" +
                "Lungimea traseului minim este:" + Convert.ToString(d2) + "+" +
                Convert.ToString(d3) + "=" + Convert.ToString(lungimetraseu));
        }
    }
}
private void pictureBox2_MouseClick(object sender, MouseEventArgs e)
{
    xmouse=e.X; ymouse=e.Y;
    if (ExistaPunctPeHarta(xmouse, ymouse) == true)//punctul exista pe harta
    {
        if (valpc <=40)//valoarea punctului la care se face click >40
        {
            MessageBox.Show("Selectați un punct de pe hartă corespunzător \n" +
                "unei măsurări existente în baza de date!(valoare > 40)");
        }
        else
        {
            determinare_distante();
        }
    }
    else MessageBox.Show("Selectati un punct de pe harta!");
}

```

In plus, am adaugat un buton si un listBox populat cu datele din tabelul Masurare care permite vizualizarea datelor disponibile din baza de date. Selectarea se va face tot de la dateTimePicker.



```

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 PollutionMap_GalCarmen
{
    public partial class Vizualizare : Form
    {
        string s="",s1="";//retin valorile din comboBox
        DateTime ds;//data din DateTimePicker
        int idHarta=0,filtru=0;
        Graphics g1=null,g2= null;//obiectele grafice pt desenare
        int[] pozX = new int[100];//vector pt retinerea coordonatelor pe Ox
        int[] pozY = new int[100];//vector pt retinerea coordonatelor pe Oy
        int n = 0;//nr de puncte care vor fi desenate pe harta
        double[] vm=new double[100];//valorile masurate coresp. punctelor
        int xc, yc,xmouse,ymouse;//coordonatele punctului la care se face click pe PictureBox1
        //si coordonatele mouse-ului
        double valm;//valoarea care va fi inserata
        double valpc=0,valmax1=0,valmax2=0;//valoarea punctului selectat, primele 2 valori maxime
        int xmax1,ymax1,xmax2,ymax2;//coordonatele punctelor avand valoare maxima
        double d1, d2, d3, lungimetraseu=0;//distanțele între punctul curent si punctele de
        valori maxime
        //si distanta între cele doua maxime si la final in d1 si d2 vom retine distanțele maxime
        public Vizualizare()
        {
            InitializeComponent();
        }
    }
}

```



```

private void Vizualizare_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the 'poluareDataSet.Harti' table. You can
move, or remove it, as needed.
    this.hartiTableAdapter.Fill(this.poluareDataSet.Harti);
    this.masurareTableAdapter1.Fill(this.poluareDataSet.Masurare);
    pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage;
    pictureBox2.Image = pictureBox1.Image;
    ds = DateTime.Now;
    g1=this.pictureBox1.CreateGraphics();//desenare in PictureBox
    g2 = this.pictureBox2.CreateGraphics();//desenare in PictureBox
}
private void AfisareHarta()
{
    if (s == "Harta Bucuresti")//selectare imagine coresp in
    {
        //functie de optiunea aleasa in ComboBox1
        pictureBox1.Image = Image.FromFile("harta_bucuresti.png");
        idHarta = 1;//si incarcare in PictureBox1
    }
    else
    {
        if (s == "Harta Cluj-Napoca")
        {
            pictureBox1.Image = Image.FromFile("harta_cluj.png");
            idHarta = 2;
        }
        else
        {
            if (s == "Harta Constanta")
            {
                pictureBox1.Image = Image.FromFile("harta_constanta.png");
                idHarta = 3;
            }
            else
            {
                if (s == "Harta Iasi")
                {
                    pictureBox1.Image = Image.FromFile("harta_iasi.png");
                    idHarta = 4;
                }
            }
            else
            {
                if (s == "Harta Sibiu")
                {
                    pictureBox1.Image = Image.FromFile("harta_sibiu.png");
                    idHarta = 5;
                }
            }
        }
    }

private void desenare(Graphics g, Brush b,int px,int py, double valmas)
{
    //desenarea punctelor pe suprafata hartii
    if ((idHarta >= 1 && idHarta <= 5) &&
        (filtru >= 1 && filtru <= 4))
    {
        Rectangle re;
        re = new Rectangle(px-10, py-10, 20, 20);
        g.FillEllipse(b,re);//desenare cercuri
        g.DrawString(Convert.ToString(valmas),
            new Font(new FontFamily("Arial"), 12),Brushes.Black, px-10, py-10);
    }
    //scrierea valorilor
}

private void button5_Click(object sender, EventArgs e)
{
    if (listBox1.Visible == false)
    {
        listBox1.Visible = true;
        button5.Text = "Ascunde datele!";
    }
}

```

```

    }
    else
    {
        listBox1.Visible = false;
        button5.Text = "Vizualizare date disponibile";
    }
}

private void AfisareValori(Graphics g)
{
    DateTime dm;
    int id=0;
    if ((idHarta >= 1 && idHarta <= 5)&&
        (filtru>=1&&filtru<=4))
    {
        //parcurem tabelul harti rand cu rand a tabelului masurare
        n = 0;valmax1= 0;valmax2 = 0;
        foreach (DataRow r in poluareDataSet.Masurare)//parcurerea tabelului Masurare
        {
            //linie cu linie
            dm = Convert.ToDateTime(Convert.ToDateTime(r["DataMasurare"])); //se preia
data
            id = Convert.ToInt32(r["IdHarta"]); //id-ul hartii
din tabel
            if (dm.Date==ds.Date && id==idHarta) //daca data selectata corespunde cu cea

                {
                    //si sa corespunda hartii selectate in ComboBox
                    n++; //creste nr de puncte
                    pozX[n] = Convert.ToInt32(r["PozitieX"]); //se retin coord punctelor
                    pozY[n] = Convert.ToInt32(r["PozitieY"]);
                    vm[n] = Convert.ToDouble(r["ValoareMasurare"]); //se retine valoarea
                    if (filtru <= 2 && vm[n]<20) //in functie de filtru si valoare se apeleaza
functia de desenare
                        desenare(g,Brushes.LightGreen, pozX[n], pozY[n], vm[n]);
                    else
                        if((filtru==3 || filtru == 1) && (vm[n] >= 20 && vm[n]<=40))
                            desenare(g, Brushes.Yellow, pozX[n], pozY[n], vm[n]);
                    else
                        if((filtru==4 || filtru == 1) && vm[n]>40)
                            desenare(g, Brushes.IndianRed, pozX[n], pozY[n], vm[n]);
                    //determinarea punctelor de pe harta avand valori maxime
                    if (vm[n] >= valmax1)
                    {
                        valmax2 = valmax1;xmax2= xmax1;ymax2= ymax1;
                        valmax1 = vm[n]; xmax1 = pozX[n];ymax1 = pozY[n];
                    }else
                        if (vm[n]>valmax2)
                        {
                            valmax2 = vm[n]; xmax2 = pozX[n]; ymax2 = pozY[n];
                        }
                }
            }
        }
    }
}

private void button3_Click(object sender, EventArgs e)
{
    Application.Exit();//iesirea din aplicatie
}

private void OperatiiAfisare()
{
    AfisareHarta();
    if (idHarta >= 1 && idHarta <= 5)
    {
        this.Refresh();
        AfisareValori(g1); //daca a fost selectata harta se afiseaza valorile
    }
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    OperatiiAfisare();
}

private void button4_Click(object sender, EventArgs e)
{
    pictureBox2.Image = pictureBox1.Image;
    if (idHarta >= 1 && idHarta <= 5)
    {
        MessageBox.Show("Se incarca valorile din pagina VeziHarta");
        AfisareValori(g2);
    }
    else
        MessageBox.Show("Harta din pagina VeziHarta nu a fost incarcata");
}

private bool ExistaPunctPeHarta(int x, int y)
{
    for (int i = 1; i <= n; i++)
        if (Math.Abs(x - pozX[i]) <= 20 && Math.Abs(y - pozY[i]) <= 20)
        {
            //se verifica daca punctul exista sau nu pe harta
            xc = pozX[i]; yc = pozY[i]; //se retin coordonatele exacte ale punctului
            valpc = vm[i];
            return true; //punctul exista pe harta
        }
    return false; //punctul nu exista pe harta
}

private void pictureBox1_MouseClick(object sender, MouseEventArgs e)
{
    xmouse = e.X; ymouse=e.Y; //coordonatele la care se face click
    if (ExistaPunctPeHarta(xmouse, ymouse) == true)
        MessageBox.Show("Punctul exista pe harta! Alege un alt\n" +
            "punct pentru a fi adaugat pe harta!");
    else
    {
        AdaugaMasurare f4=new AdaugaMasurare();
        f4.ShowDialog();
        valm = f4.ValoareMasurare; //se retine valoarea introdusa in formularul de
        //adaugare
        MessageBox.Show(Convert.ToString("Se pune punctul avand valoarea:"+valm));
        Brush b;
        if (valm < 20) b = Brushes.LightGreen; //determinam culoarea cercului
        else if (valm <= 40) b = Brushes.Yellow; //care va fi desenat
        else b = Brushes.IndianRed;
        desena(g1, b, xmouse, ymouse, valm); //se apeleaza functia de desena
        this.masurareTableAdapter1.InsertMasurare(idHarta, xmouse, ymouse
            , (float)valm, Convert.ToDateTime(dateTimePicker1.Text));
        //se completeazatabelul cu noua valoare
        this.masurareTableAdapter1.Fill(this.poluareDataSet.Masurare);
        this.masurareTableAdapter1.Update(this.poluareDataSet.Masurare);
    }
}

private void DesenareLinie(int x1,int y1, int x2, int y2)
{
    g2.DrawLine(new Pen(Color.Red,3),x1, y1, x2, y2);
}

private void determinare_distante()
{
    d1 = Math.Sqrt(Math.Pow(xc - xmax1, 2) + Math.Pow(yc - ymax1, 2));
    d2 = Math.Sqrt(Math.Pow(xc - xmax2, 2) + Math.Pow(yc - ymax2, 2));
    d3 = Math.Sqrt(Math.Pow(xmax1 - xmax2, 2) + Math.Pow(ymax1 - ymax2, 2));
}

```

```

if (d1<=d3 && d2<=d3)
{
    lungimetraseu = d1 + d2;
    DesenareLinie(xmax1, ymax1, xc, yc);
    DesenareLinie(xc, yc, xmax2, ymax2);
    MessageBox.Show("(" + Convert.ToString(xmax1) + "," + Convert.ToString(ymax1) +
")\n" +
    "(" + Convert.ToString(xc) + "," + Convert.ToString(yc) + ")\n" +
    "(" + Convert.ToString(xmax2) + "," + Convert.ToString(ymax2) + ")\n" +
    "Lungimea traseului minim este:" + Convert.ToString(d1)+"+" +
    Convert.ToString(d2) + "=" + Convert.ToString(lungimetraseu));
}
else if(d1<=d2 && d3<=d2)
{
    lungimetraseu = d1 + d3;
    DesenareLinie(xc, yc, xmax1, ymax1);
    DesenareLinie(xmax1, ymax1, xmax2, ymax2);
    MessageBox.Show("(" + Convert.ToString(xc) + "," + Convert.ToString(yc) + ")\n" +
    "(" + Convert.ToString(xmax1) + "," + Convert.ToString(ymax1) + ")\n" +
    "(" + Convert.ToString(xmax2) + "," + Convert.ToString(ymax2) + ")\n" +
    "Lungimea traseului minim este:" + Convert.ToString(d1) + "+" +
    Convert.ToString(d3) + "=" + Convert.ToString(lungimetraseu));
}
else
    if (d2<=d1 && d3<=d1)
    {
        lungimetraseu = d2 + d3;
        DesenareLinie(xc, yc, xmax2, ymax2);
        DesenareLinie(xmax2, ymax2, xmax1, ymax1);
        MessageBox.Show("(" + Convert.ToString(xc) + "," + Convert.ToString(yc) + ")\n" +
        "(" + Convert.ToString(xmax1) + "," + Convert.ToString(ymax1) + ")\n" +
        "(" + Convert.ToString(xmax2) + "," + Convert.ToString(ymax2) + ")\n" +
        "Lungimea traseului minim este:" + Convert.ToString(d2) + "+" +
        Convert.ToString(d3) + "=" + Convert.ToString(lungimetraseu));
    }
}
private void pictureBox2_MouseClick(object sender, MouseEventArgs e)
{
    xmouse=e.X; ymouse=e.Y;
    if (ExistaPunctPeHarta(xmouse, ymouse) == true)//punctul exista pe harta
    {
        if (valpc <= 40)//valoarea punctului la care se face click >40
        {
            MessageBox.Show("Selectați un punct de pe hartă corespunzător \n" +
            "unei măsurări existente în baza de date!(valoare > 40)");
        }
        else
        {
            determinare_distante();
        }
    }
    else MessageBox.Show("Selectati un punct de pe harta!");
}
private void button2_Click(object sender, EventArgs e)
{
    filtru = 1;
    s1=comboBox2.Text = "Niciun filtru";
    OperatiiAfisare();
}

private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
    ds = dateTimePicker1.Value;//preluam data si ora

```

```

        ds = Convert.ToDateTime(ds);
        OperatiiAfisare();
    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
        //selectare harta
        s = comboBox1.Text.Trim();
        filtru = 1;
        s1 = comboBox2.Text = "Niciun filtru";
        OperatiiAfisare();
    }

    private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
    {
        //selectare filtru- variabila filtru retine optiunea
        s1=comboBox2.Text.Trim();
        if (s1 == "Niciun filtru")
            filtru = 1;
        else
            if(s1== @"Valoarea < 20")
                filtru = 2;
            else
                if(s1== @"20 <= Valoarea <= 40")
                    filtru = 3;
                else
                    if (s1 == @"Valoarea > 40")
                        filtru = 4;
                    OperatiiAfisare();
    }
}
}

```