**MINISTERUL EDUCAȚIEI ȘI CERCETĂRII**

**AL REPUBLICII MOLDOVA**

**Centrul de Excelenţă în Informatică şi Tehnologii Informaţionale**

**RAPORT**

**LA PRACTICA TEHNOLOGICĂ**

**SPECIALITATEA "Programare şi analiza produselor program"**

A elaborat elevul: Vatamaniuc Ion

Grupa: P-2032

Conducătorul practicii: Covali Eugenia

Chişinău 2023

**Cuprins:**

**[Sarcina:](#_Toc135997641)** [3](#_Toc135997641)

[**Procesul de proiectare a bazei de date** 4](#_Toc135997642)

[**Tabele** 4](#_Toc135997643)

[**Dbo.Agenti** 4](#_Toc135997644)

[**Dbo.Materiale.sql** 5](#_Toc135997645)

[**Dbo.Transporturi.sql** 6](#_Toc135997646)

[**Dbo.Agenti\_Exclusi.sql** 7](#_Toc135997647)

[**Dbo.Utilizatori.sql** 7](#_Toc135997648)

[**Descrierea interfeței** 8](#_Toc135997649)

[**Ghid de utilizare a aplicatiei** 13](#_Toc135997650)

[**Listingul produsului program** 14](#_Toc135997651)

[**Autentificare.cs** 14](#_Toc135997652)

[**FormAgent.cs** 15](#_Toc135997653)

[**FormMateriale.cs** 25](#_Toc135997654)

[**FormTransporturi.cs** 29](#_Toc135997655)

[**Demisionare.cs** 33](#_Toc135997656)

[**Agenti\_exclusi.cs** 35](#_Toc135997657)

[**AgentVanzari.cs** 38](#_Toc135997658)

[**reii.cs** 40](#_Toc135997659)

[**Baza de date SQL** 40](#_Toc135997660)

[**Concluzie** 43](#_Toc135997661)

[**Bibliografie** 44](#_Toc135997662)

# **Sarcina:**

**27. Agenți.** Informaţia despre materialele de construcţie dintr-un depozit şi despre activitatea agenţilor de vânzări este înregistrată într-o bază de date. Vânzarea materialelor şi transportarea lor este asigurată de către agenţii de vânzări (un agent nu poate transporta mai multe tipuri de materiale într-un singur transport).

Elaboraţi o bază de date, gestiunea căreia se va realiza dintr-o aplicaţie vizuală. În dependență de tematică se vor specifica criteriile de integritate pentru baza de date, administra obiecte ale bazei de date, stabili planul de întreținere a bazei de date și moduri de securizare a datelor din baza de date. Informațiile din baza de date se vor prelucra la solicitarea utilizatorului folosind meniuri, formulare și control-ere, ținând cont de următoarele funcţionalităţi:Înregistrează o nouă cursă;

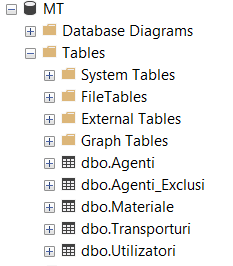
1. Înregistrează un nou agent de vânzări;
2. Înregistrează un nou produs la depozit;
3. Exclude informaţia, ce corespunde agentului, codul căruia este introdus de la tastatură şi o înscrie într-o altă tabelă;
4. Determină cel mai tânăr agent de vânzări;
5. Afişează la ecran lista agenţilor de vârstă pensionară (57 ani împliniţi la femei şi 62-la bărbaţi ), ordonată ascendent, conform vârstei acestora;
6. Determină salariul mediu lunar al agenţilor;
7. Exportă într-un fișier MS Word sau MS Excel lista agenţilor, care transportă produse;
8. Afişează la ecran datele despre materialele transportate de un anumit agent, numele căruia este introdus de la tastatură;
9. Determină preţul mediu al produsului ales.
10. Afișează la ecran lista celor mai vândute materiale de construcții.
11. Afișează la ecran vechimea în muncă a fiecărui angajat.

Afișarea la ecran a informațiilor acestora se va realiza prin intermediul rapoartelor.

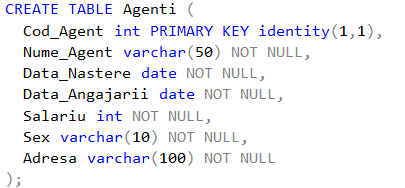
# **Procesul de proiectare a bazei de date**

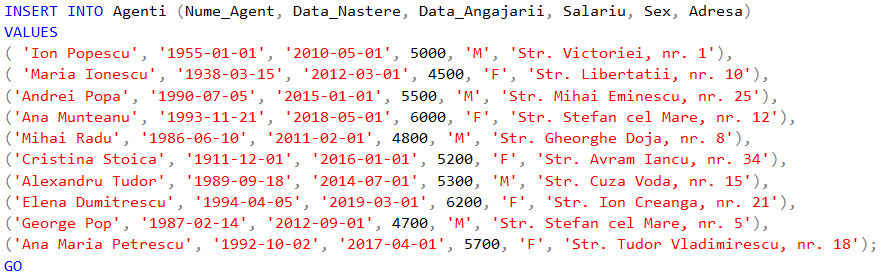
Pentru realizarea sarcina dată am avut nevoie de a proiecta baza de date „Depozit\_Materiale” pe care am creat-o prin intermediul aplicației Microsoft SQL Server 2018 Management Studio, cu 5 tabele. Dintre care 4 tabele sunt cu datele necesare pentru a îndeplini toate condițiile din sarcina noastră, iar una care se logează fiecare utilizator al aplicației.

# **Tabele**

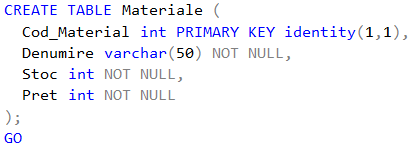


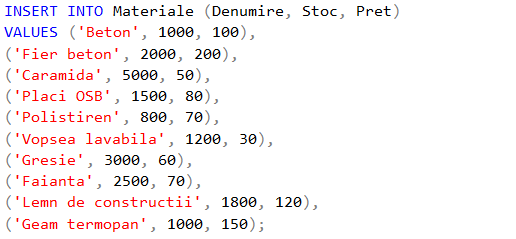
## **Dbo.Agenti**



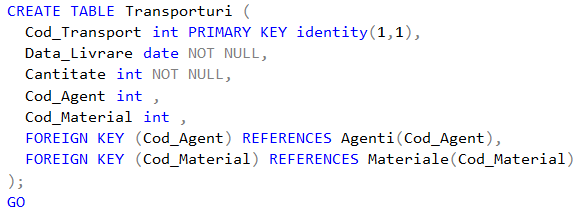


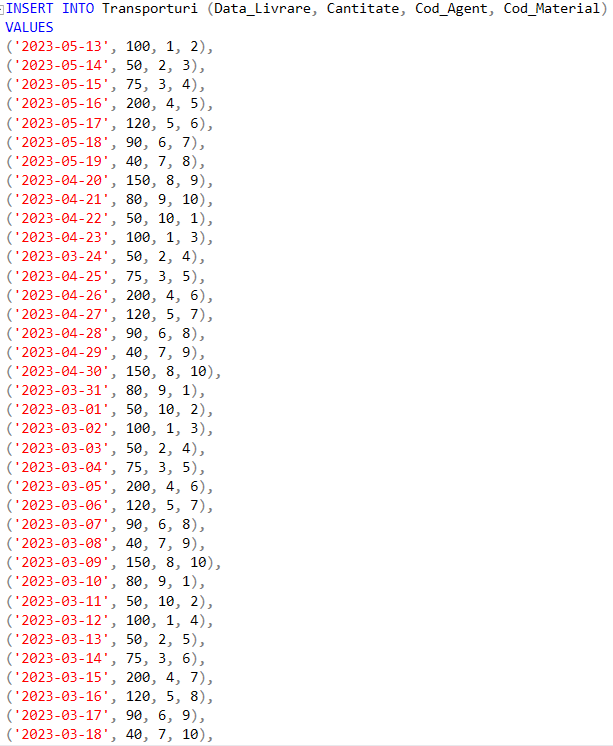
## **Dbo.Materiale.sql**



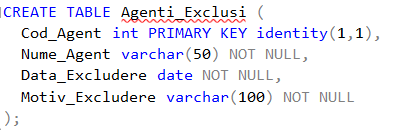


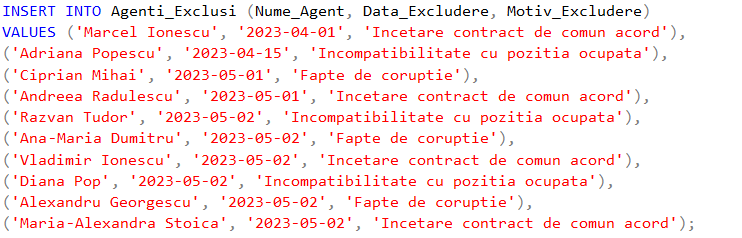
## **Dbo.Transporturi.sql**



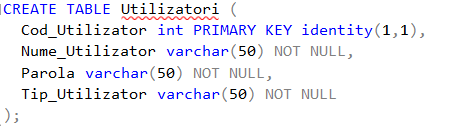


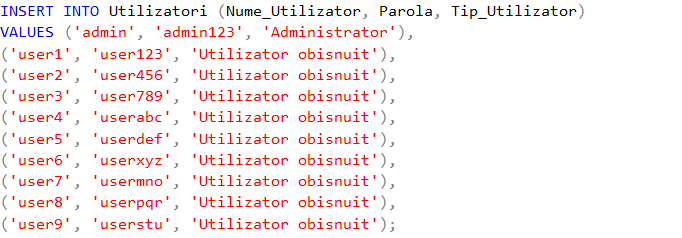
## **Dbo.Agenti\_Exclusi.sql**





## **Dbo.Utilizatori.sql**





# **Descrierea interfeței**

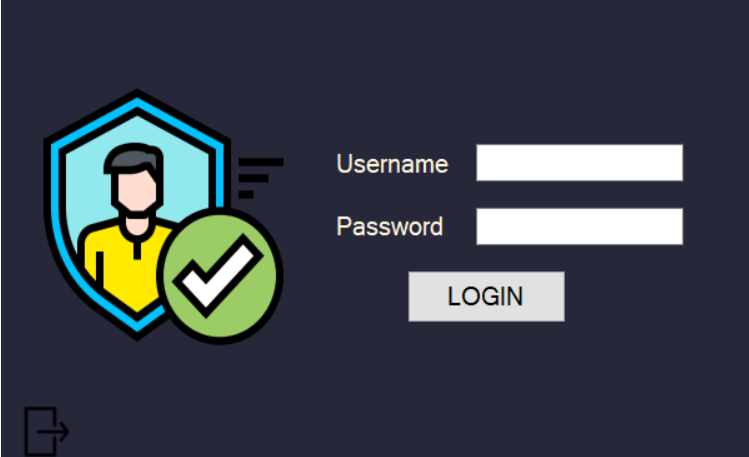
Pentru aplicația CAZAN.MD, am creat baza de date care vinde produse de construcții. Am creat o aplicație în C# și .NET cu o interfașă destul de plăcută pentru un utilizator obișnuit care lucrează în această companie.

După ce imaginea s-a încărat urmeaza un Forms care ne solicită logarea în sistemul de baze Aeroport.   
La lansarea aplicației ne solicită logarea în aplicația CAZAN.MD

Username: admin

Password: admin123

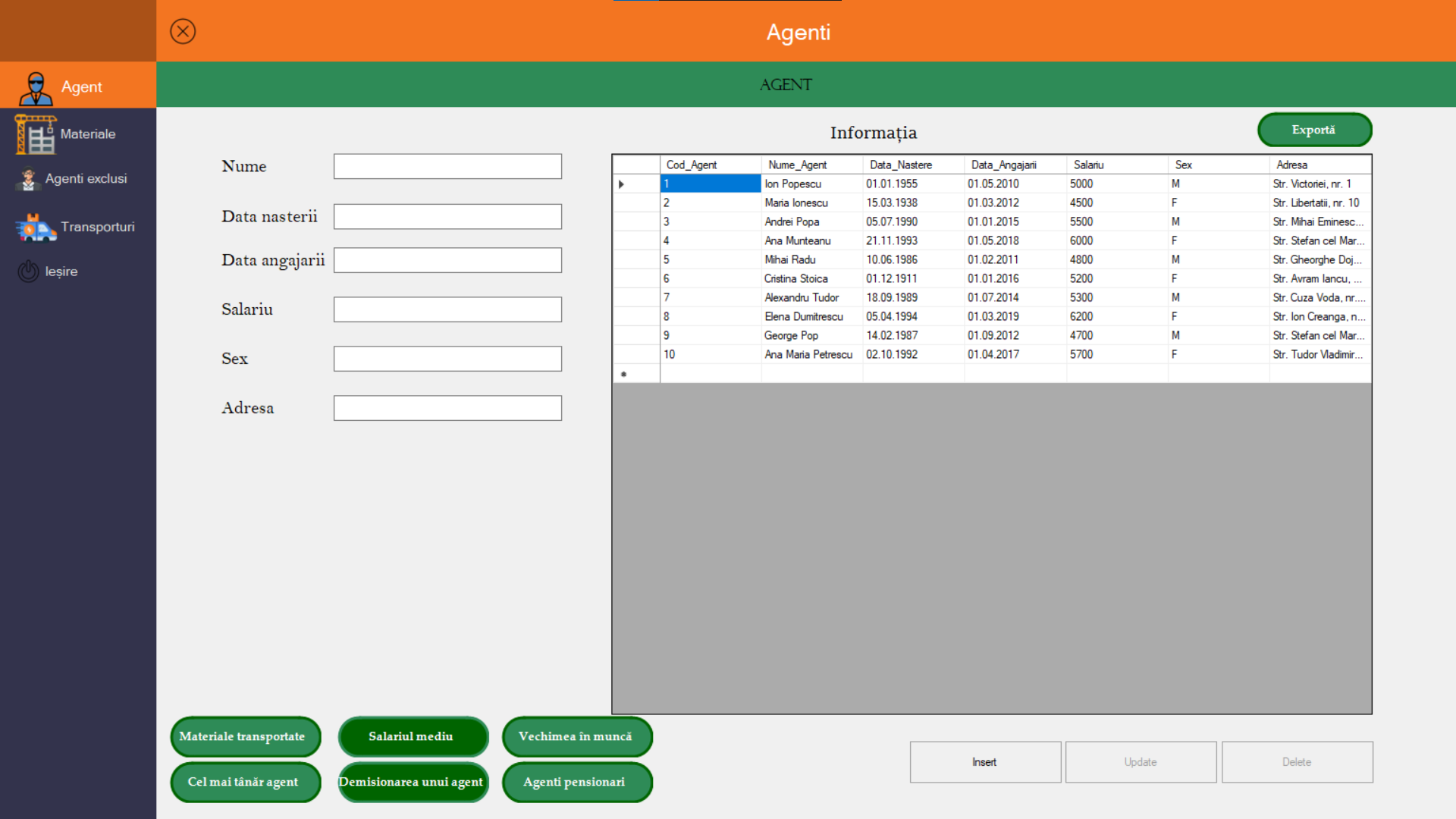
Accesăm buttonul Login dacă ne-am logat corect.



Aplicația CAZAN.MD după logarea prin fereastra login, apare interfața aplicației



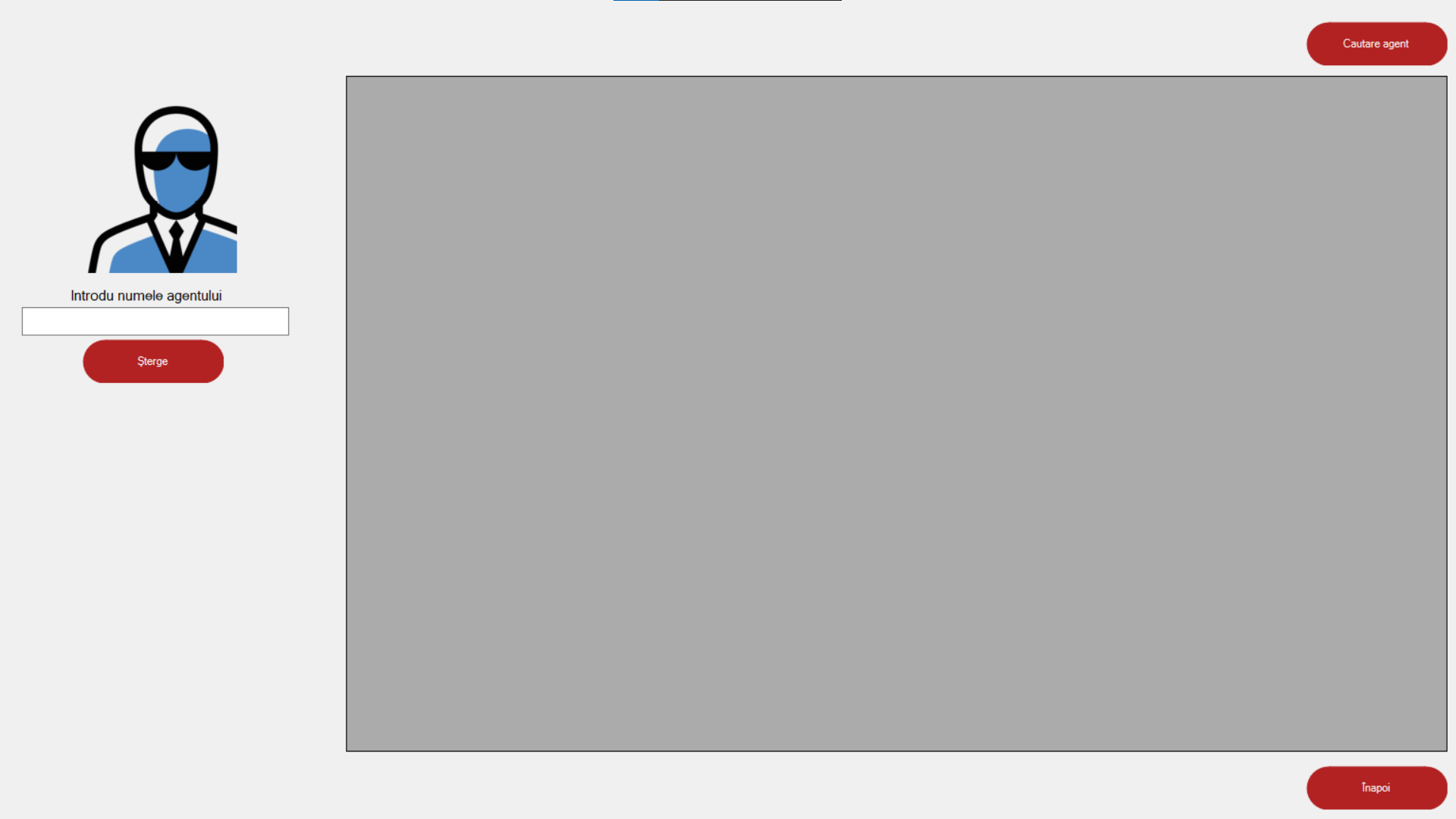
Interfața aplicației ne întâmpină cu un logo a aplicației, iar în partea stângă 5 butoane care își schimbă culoarea la accesarea lor. Fiecare buton ne duce la altă pagină care conține datele despre „Agent” „Materiale” „Agenti\_Exclusi” „Transporturi”, unde fiecare din ele au butoane și pagini adiționale. În partea dreaptă a fiecărui buton apare un tabel cu informația necesară, iar in colțul drept sus este butonul „Exportă”, care exportă datele din fiecare tabelă în Excel.



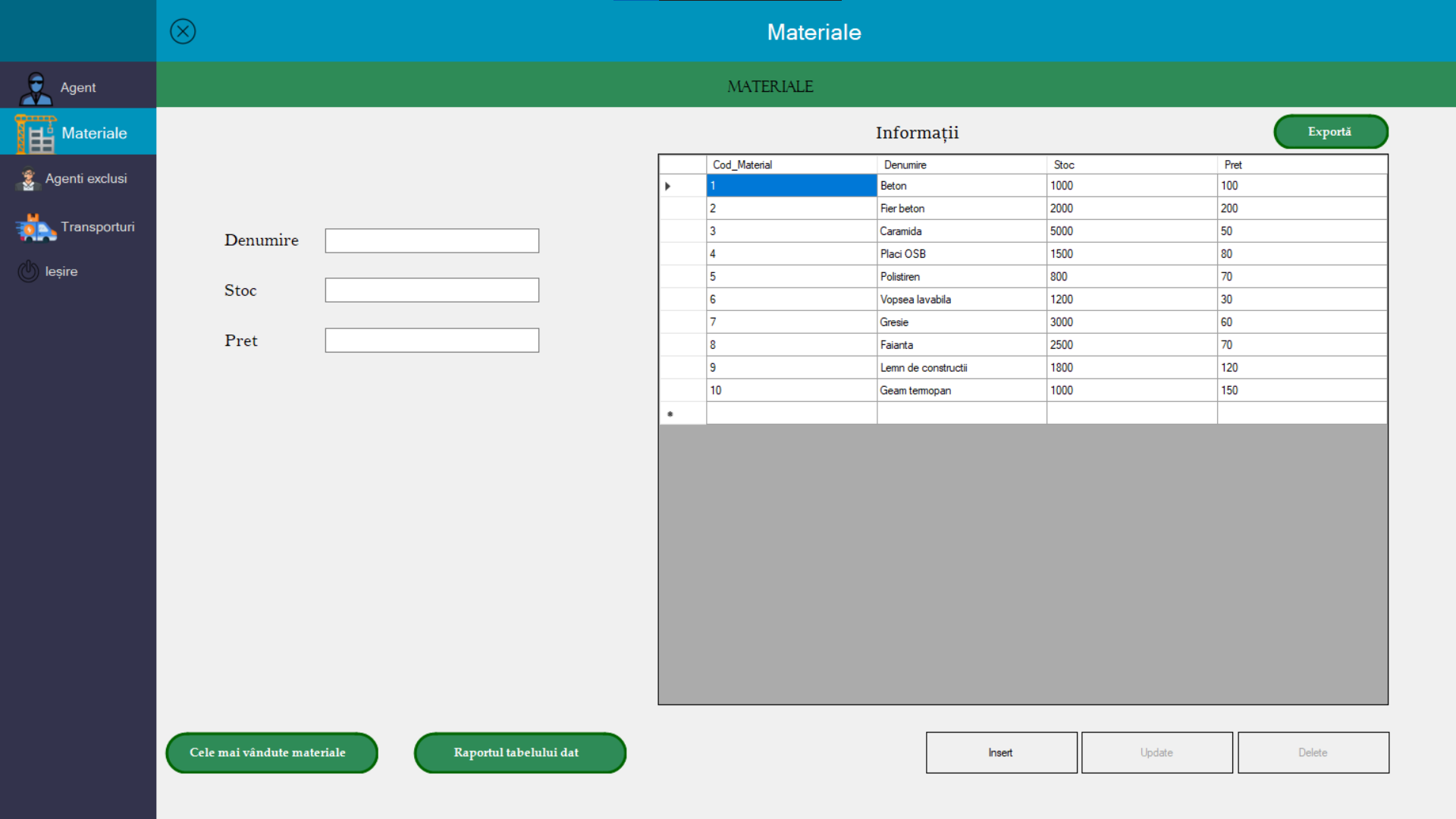
La accesarea Butonului Agent al interfeții, ne deschide o fereastră nouă cu datele despre angajați, care sunt afișate într-o tabelă în partea dreaptă. În partea jos în stânga, vedem 6 butoane care ne afișează informațiile despre angajați în mod diferit în tabel, iar butonul „Demisionarea unui agent”, care ne deschide o fereastră nouă.



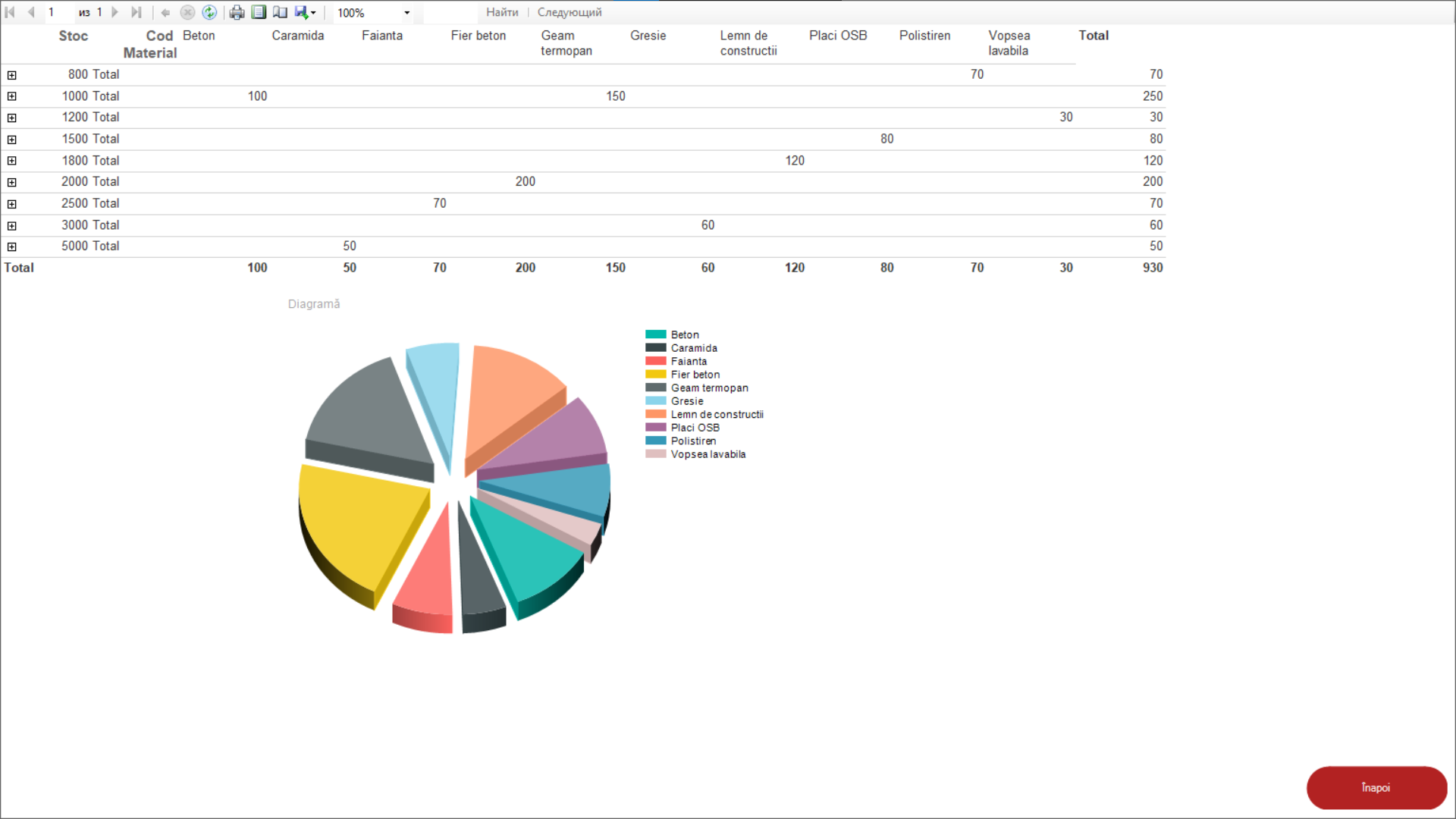
La accesarea butonului „Demisionarea unui agent” ne deschide o fereastră nouă, unde trebuie să introducem codul agentului (doar cifre acceptate), data și motivul demisionării din multi-select.



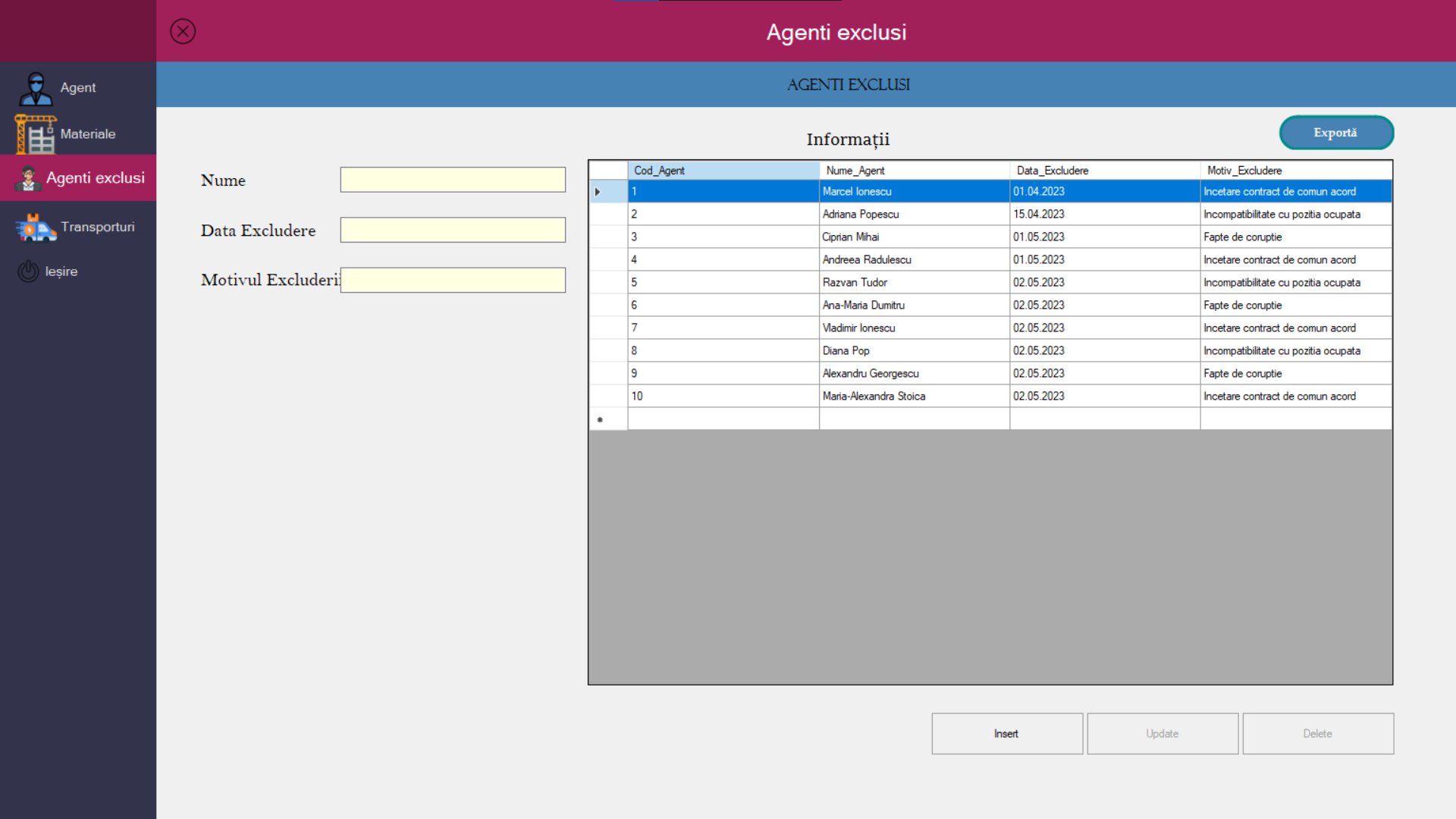
La accesarea buttonului „Materiale transportate” ne deschide altă fereastră nouă unde introducem numele agentului, iar în partea dreaptă a ecranului într-o tabelă ne afișează datele despre transportul agentului dorit.

****

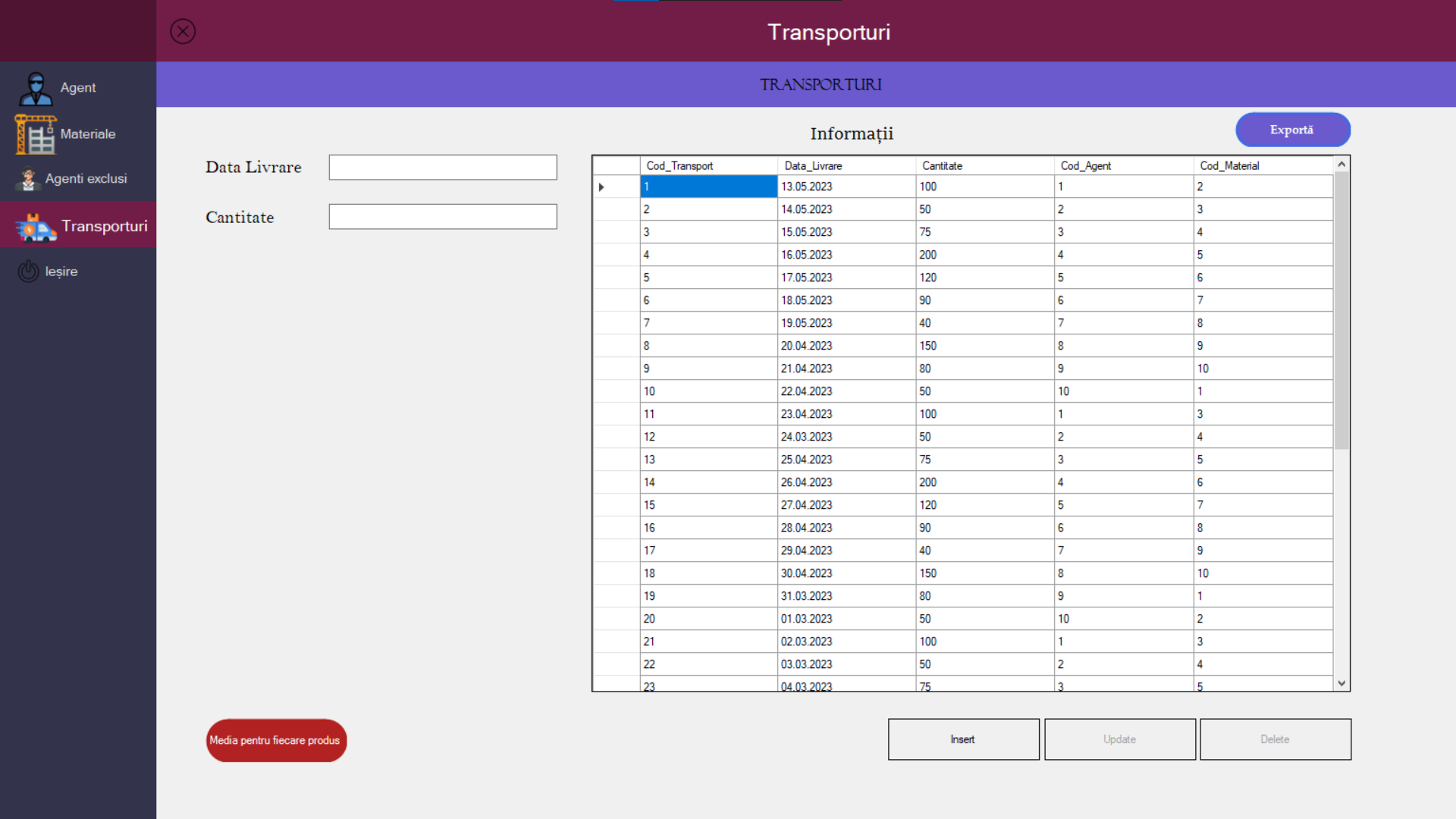
La accesarea butonului „Materiale” se deschide o fereastră nouă cu datele despre materialele de construcții. În partea stângă jos, avem 2 butoane care ne arată top 5 cele mai vândute materiale.

****

La accesarea butonului „Raportul tabelului dat”, ne deschide altă fereastră nouă cu raportul materialelor, și o diagramă cu suma materialelor de construcții.

****

La accesarea butonului „Agenti\_exclusi”, se dechide o fereastră nouă cu datele despre Agentii excluși sau demisionați.

****

La tastarea butonului „Transporturi” ne arată transportul de materiale pentru fiecare angajat după codul său și codul materialului transportat.

# **Ghid de utilizare a aplicatiei**

Această aplicație a fost creată pentru a monitoriza lucrul agenților de la compania CAZAN.MD și nivelul lor de lucru. La fel a fost 4 tabele cu date pentru fiecare material de construcții, transportul și o tabelă unde sunt stocați agenții demisionați.

Aplicația se instaleaza de catre furnizorul de software, care va prezenta aplicatia si va instrui personalul in folosirea ei.

Cu o conexiune active creată de către administratorul bazei de date și prin intermediul unor interogari realizate asupra baza de date putem extrage informatie din tabele.

Afișarea informației extrase din tabele BD se va face prin intermediul unor algoritmi de selectare și a dataGrid-urilor ce vor fi umplute cu date aduse de fiecare interogare, conform cerinței aplicate.

# **Listingul produsului program**

## **Autentificare.cs**

using studiul\_individual;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace studiul\_individual.Forms

{

public partial class Autentificare : Form

{

public Autentificare()

{

InitializeComponent();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-F8UOFGQ;Initial Catalog=MT;Integrated Security=True");

SqlCommand cmd = new SqlCommand(@"Select count (\*) From Utilizatori where Nume\_Utilizator = @numeUtilizator and Parola = @parola", con);

cmd.Parameters.AddWithValue("@numeUtilizator", textBox1.Text);

cmd.Parameters.AddWithValue("@parola", textBox2.Text);

con.Open();

int result = (int)cmd.ExecuteScalar();

con.Close();

// Verificarea credențialelor de autentificare

if (result > 0)

{

// Setarea rezultatului dialogului de autentificare ca OK si inchiderea formularului

DialogResult = DialogResult.OK;

Close();

}

else

{

MessageBox.Show("Nume de utilizator sau parolă incorecte!");

}

}

private void button2\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Sigur dorești să ieși?", "Question", MessageBoxButtons.YesNo) == DialogResult.Yes)

Application.Exit();

}

}

}

## **FormAgent.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Net.Mime.MediaTypeNames;

namespace studiul\_individual.Forms

{

public partial class FormAgent : Form

{

//Declararea variabilelor de conexiune și comandă pentru interogarea bazei de date

string path = @"Data Source=DESKTOP-F8UOFGQ;Initial Catalog=MT;Integrated Security=True";

SqlConnection con;

SqlCommand cmd;

SqlDataAdapter adpt;

DataTable dt;

int ID3;

//Constructorul clasei

public FormAgent()

{ //Inițializarea variabilelor de conexiune și comandă pentru interogarea bazei de date

InitializeComponent();

con = new SqlConnection(path);

display();

button2.Enabled = false;

button3.Enabled = false;

//Adăugarea unui event handler pentru evenimentul "Click" al butonului 2

button2.Click += button2\_Click;

//Adăugarea unui event handler pentru evenimentul "Click" al butonului 3

button3.Click += new EventHandler(button3\_Click);

}

private void button1\_Click(object sender, EventArgs e)

{

if (Numep.Text == "" || Prenumep.Text == "" || Adresap.Text == "" || Telefonp.Text == "" || Sexa.Text == "" || Adresaa.Text == "")

{

MessageBox.Show("Introduce-ti toate datele necesare!");

}

else

{

try

{

con.Open();

cmd = new SqlCommand("INSERT INTO Agenti( Nume\_Agent, Data\_Nastere, Data\_Angajarii, Salariu, Sex, Adresa) values ('" + Numep.Text + "','" + Prenumep.Text + "','" + Adresap.Text + "','" + Telefonp.Text + "','" + Sexa.Text + "','" + Adresaa.Text + "')", con);

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Inregistrarea unui Agent nou a decurs cu succes!");

clear();

display();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

public void clear()

{

Numep.Text = "";

Prenumep.Text = "";

Adresap.Text = "";

Telefonp.Text = "";

Sexa.Text = "";

Adresaa.Text = "";

}

//Metoda care ne afișează in dataGridView toate datele necesare.

public void display()

{

try

{

dt = new DataTable();

con.Open();

adpt = new SqlDataAdapter("SELECT \*FROM Agenti", con);

adpt.Fill(dt);

dataGridView1.DataSource = dt;

con.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void dataGridView1\_CellDoubleClick(object sender, DataGridViewCellEventArgs e)

{

ID3 = int.Parse(dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString());

Numep.Text = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();

Prenumep.Text = dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();

Adresap.Text = dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString();

Telefonp.Text = dataGridView1.Rows[e.RowIndex].Cells[4].Value.ToString();

Sexa.Text = dataGridView1.Rows[e.RowIndex].Cells[5].Value.ToString();

Adresaa.Text = dataGridView1.Rows[e.RowIndex].Cells[6].Value.ToString();

button2.Enabled = true;

button3.Enabled = true;

}

private void button3\_Click(object sender, EventArgs e)

{

con.Open();

cmd = new SqlCommand("update Agenti set Nume\_Agent = '" + Numep.Text + "', Data\_Nastere = '" + Prenumep.Text + "', Data\_Angajarii = '" + Adresap.Text + "', Salariu = '" + Telefonp.Text + "',Sex = '" + Sexa.Text + "',Adresa = '" + Adresaa.Text + "' where Cod\_Agent = " + ID3, con);

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Actualizarea unui client nou a decurs cu succes!");

display();

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

con.Open();

cmd = new SqlCommand("delete from Agenti where Cod\_Agent = '" + ID3 + "'", con);

cmd.ExecuteNonQuery();

MessageBox.Show("Stergerea unui Agent a decurs cu succes!");

display();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void btn\_tanar\_Click(object sender, EventArgs e)

{

try

{

con.Open();

cmd = new SqlCommand("SELECT TOP 1 Nume\_Agent, Data\_Nastere FROM Agenti WHERE Sex = 'M' ORDER BY Data\_Nastere ASC", con);

SqlDataReader reader = cmd.ExecuteReader();

if (reader.Read())

{

string numeAgent = reader.GetString(0);

DateTime dataNastere = reader.GetDateTime(1);

int varsta = DateTime.Today.Year - dataNastere.Year;

MessageBox.Show($"Cel mai tanar agent de vanzari este {numeAgent}, in varsta de {varsta} ani.");

}

else

{

MessageBox.Show("Nu exista agenti de vanzari inregistrati!");

}

reader.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void Excludere\_btn\_Click(object sender, EventArgs e)

{

Demisionare demisionare = new Demisionare();

demisionare.Show();

this.Hide();

}

private void button\_asc\_Click(object sender, EventArgs e)

{

try

{

con.Open();

// Definim interogarea SQL pentru a extrage agenții de vârstă pensionară

cmd = new SqlCommand("SELECT Nume\_Agent, Data\_Nastere, Sex FROM Agenti WHERE (DATEDIFF(YEAR, Data\_Nastere, GETDATE()) >= 57 AND Sex = 'F') OR (DATEDIFF(YEAR, Data\_Nastere, GETDATE()) >= 62 AND Sex = 'M') ORDER BY Data\_Nastere ASC", con);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void button\_mediu\_Click(object sender, EventArgs e)

{

try

{

con.Open();

// Definim interogarea SQL pentru a calcula media salariilor agenților

cmd = new SqlCommand("SELECT AVG(Salariu) FROM Agenti", con);

double mediaSalariilor = Convert.ToDouble(cmd.ExecuteScalar());

MessageBox.Show($"Salariul mediu lunar al agenților este {mediaSalariilor.ToString("C2")}.");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void button\_produse\_Click(object sender, EventArgs e)

{

Agent\_vanzari agent = new Agent\_vanzari();

agent.Show();

this.Hide();

}

private void button\_Vechime\_Click(object sender, EventArgs e)

{

con = new SqlConnection(path);

con.Open();

cmd = new SqlCommand("SELECT Nume\_Agent, DATEDIFF(YEAR, Data\_Angajarii, GETDATE()) AS Vechime FROM Agenti", con);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

con.Close();

}

private void exportToExcel(DataGridView dataGridView)

{

dataGridView1.SelectAll();

DataObject copydata = dataGridView1.GetClipboardContent();

if (copydata != null) Clipboard.SetDataObject(copydata);

Microsoft.Office.Interop.Excel.Application xlapp = new Microsoft.Office.Interop.Excel.Application();

xlapp.Visible = true;

Microsoft.Office.Interop.Excel.Workbook xlWbook;

Microsoft.Office.Interop.Excel.Worksheet xlsheet;

object miseddata = System.Reflection.Missing.Value;

xlWbook = xlapp.Workbooks.Add(miseddata);

xlsheet = (Microsoft.Office.Interop.Excel.Worksheet)xlWbook.Worksheets.get\_Item(1);

Microsoft.Office.Interop.Excel.Range xlr = (Microsoft.Office.Interop.Excel.Range)xlsheet.Cells[1, 1];

xlr.Select();

xlsheet.PasteSpecial(xlr, Type.Missing, Type.Missing, Type.Missing, Type.Missing, Type.Missing, true);

}

private void button\_excel\_Click(object sender, EventArgs e)

{

try

{

con.Open();

// Definim interogarea SQL pentru a extrage agenții care transportă produse

cmd = new SqlCommand("SELECT \* FROM Agenti ", con);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

// Exportăm datele într-un fișier Excel

exportToExcel(dataGridView1);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

}

## **FormMateriale.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Linq;

namespace studiul\_individual.Forms

{

public partial class FormMateriale : Form

{

string path = @"Data Source=DESKTOP-F8UOFGQ;Initial Catalog=MT;Integrated Security=True";

SqlConnection con;

SqlCommand cmd;

SqlDataAdapter adpt;

DataTable dt;

int ID;

public FormMateriale()

{

InitializeComponent();

con = new SqlConnection(path);

display();

button2.Enabled = false;

button3.Enabled = false;

button2.Click += button2\_Click;

button3.Click += new EventHandler(button3\_Click);

}

private void panel1\_Paint(object sender, PaintEventArgs e)

{

}

public void clear()

{

Nume.Text = "";

Adresa.Text = "";

Telefon.Text = "";

}

public void display()

{

try

{

dt = new DataTable();

con.Open();

adpt = new SqlDataAdapter("SELECT \*FROM Materiale", con);

adpt.Fill(dt);

dataGridView1.DataSource = dt;

con.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void dataGridView1\_CellDoubleClick(object sender, DataGridViewCellEventArgs e)

{

ID = int.Parse(dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString());

Nume.Text = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();

Adresa.Text = dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();

Telefon.Text = dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString();

button2.Enabled = true;

button3.Enabled = true;

}

private void button1\_Click(object sender, EventArgs e)

{

if (Nume.Text == "" || Adresa.Text == "" || Telefon.Text == "")

{

MessageBox.Show("Introduce-ti toate datele necesare!");

}

else

{

try

{

con.Open();

cmd = new SqlCommand("INSERT INTO Materiale (Denumire, Stoc, Pret) values ('" + Nume.Text + "','" + Adresa.Text + "','" + Telefon.Text + "')", con);

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Inregistrarea unui material de construcție nou a decurs cu succes!");

clear();

display();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

private void button3\_Click(object sender, EventArgs e)

{

con.Open();

cmd = new SqlCommand("update Materiale set Denumire = '" + Nume.Text + "', Stoc = '" + Adresa.Text + "', Pret = '" + Telefon.Text + "' where Cod\_Material = " + ID, con);

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Actualizarea unui material a decurs cu succes!");

display();

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

using (con = new SqlConnection(path))

{

con.Open();

cmd = new SqlCommand("delete from Materiale where Cod\_Material = '" + ID + "'", con);

cmd.ExecuteNonQuery();

MessageBox.Show("Stergerea unui material a decurs cu succes!");

display();

con.Close();

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void button\_produse\_Click(object sender, EventArgs e)

{

try

{

con = new SqlConnection(path);

con.Open();

string query = "SELECT TOP 5 M.Denumire, SUM(T.Cantitate) AS TotalCantitate " +

"FROM Materiale M " +

"JOIN Transporturi T ON M.Cod\_Material = T.Cod\_Material " +

"GROUP BY M.Cod\_Material, M.Denumire " +

"ORDER BY TotalCantitate DESC";

cmd = new SqlCommand(query, con);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

// afișează rezultatele într-un control DataGridView

dataGridView1.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void button\_WOC1\_Click(object sender, EventArgs e)

{

reii des = new reii();

des.Show();

}

private void exportToExcel(DataGridView dataGridView)

{

dataGridView1.SelectAll();

DataObject copydata = dataGridView1.GetClipboardContent();

if (copydata != null) Clipboard.SetDataObject(copydata);

Microsoft.Office.Interop.Excel.Application xlapp = new Microsoft.Office.Interop.Excel.Application();

xlapp.Visible = true;

Microsoft.Office.Interop.Excel.Workbook xlWbook;

Microsoft.Office.Interop.Excel.Worksheet xlsheet;

object miseddata = System.Reflection.Missing.Value;

xlWbook = xlapp.Workbooks.Add(miseddata);

xlsheet = (Microsoft.Office.Interop.Excel.Worksheet)xlWbook.Worksheets.get\_Item(1);

Microsoft.Office.Interop.Excel.Range xlr = (Microsoft.Office.Interop.Excel.Range)xlsheet.Cells[1, 1];

xlr.Select();

xlsheet.PasteSpecial(xlr, Type.Missing, Type.Missing, Type.Missing, Type.Missing, Type.Missing, true);

}

private void button\_excel\_Click(object sender, EventArgs e)

{

try

{

con.Open();

// Definim interogarea SQL pentru a extrage agenții care transportă produse

cmd = new SqlCommand("SELECT \* FROM Materiale ", con);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

// Exportăm datele într-un fișier Excel

exportToExcel(dataGridView1);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

}

## **FormTransporturi.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace studiul\_individual.Forms

{

public partial class FormTransporturi : Form

{

//Declararea variabilelor de conexiune și comandă pentru interogarea bazei de date

string path = @"Data Source=DESKTOP-F8UOFGQ;Initial Catalog=MT;Integrated Security=True";

SqlConnection con;

SqlCommand cmd;

SqlDataAdapter adpt;

DataTable dt;

int ID2;

//Constructorul clasei

public FormTransporturi()

{

//Inițializarea variabilelor de conexiune și comandă pentru interogarea bazei de date

InitializeComponent();

con = new SqlConnection(path);

display();

button2.Enabled = false;

button3.Enabled = false;

//Adăugarea unui event handler pentru evenimentul "Click" al butonului 2

button2.Click += button2\_Click;

//Adăugarea unui event handler pentru evenimentul "Click" al butonului 3

button3.Click += new EventHandler(button3\_Click);

}

private void button1\_Click(object sender, EventArgs e)

{

if (Data.Text == "" || Suma.Text == "")

{

MessageBox.Show("Introduce-ti toate datele necesare!");

}

else

{

try

{

con.Open();

//Crearea și executarea comenzii SQL pentru adăugarea unui nou contractîn baza de date

cmd = new SqlCommand("INSERT INTO Transporturi( Data\_Livrare, Cantitate) values ('" + Data.Text + "','" + Suma.Text + "')", con);

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Inregistrarea unui transport nou a decurs cu succes!");

clear();

display();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

public void clear()

{

Suma.Text = "";

Data.Text = "";

}

//Metoda care ne afișează in dataGridView toate datele necesare.

public void display()

{

try

{

dt = new DataTable();

con.Open();

adpt = new SqlDataAdapter("SELECT \* FROM Transporturi; ", con);

adpt.Fill(dt);

dataGridView1.DataSource = dt;

con.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void dataGridView1\_CellDoubleClick(object sender, DataGridViewCellEventArgs e)

{

ID2 = int.Parse(dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString());

// Se atribuie valorile din celulele selectate câmpurilor corespunzătoare din interfața grafică

Data.Text = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();

Suma.Text = dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();

// Se activează butoanele 2 și 3

button2.Enabled = true;

button3.Enabled = true;

}

private void button3\_Click(object sender, EventArgs e)

{

// Se deschide o conexiune la baza de date

con.Open();

// Se definește o comandă SQL de actualizare pentru contractul selectat

cmd = new SqlCommand("update Transporturi set Data\_Livrare = '" + Data.Text + "', Cantitate = '" + Suma.Text + "' where Cod\_Transport = " + ID2, con);

// Se execută comanda SQL

cmd.ExecuteNonQuery();

// Se închide conexiunea la baza de date

con.Close();

// Se afișează un mesaj de succes în interfața grafică

MessageBox.Show("Actualizarea unui transport a decurs cu succes!");

// Se afișează informațiile actualizate în interfața grafică

display();

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

// Se deschide o conexiune la baza de date

con.Open();

// Se definește o comandă SQL de ștergere pentru contractul selectat

cmd = new SqlCommand("delete from Transporturi where Cod\_Transport = " + ID2, con);

// Se execută comanda SQL

cmd.ExecuteNonQuery();

// Se afișează un mesaj de succes în interfața grafică

MessageBox.Show("Ștergerea unui transport a decurs cu succes!");

// Se afișează informațiile actualizate în interfața grafică

display();

}

catch (Exception ex)

{

// Se afișează un mesaj de eroare în interfața grafică în cazul în care apare o excepție

MessageBox.Show(ex.Message);

}

finally

{

// Se închide conexiunea la baza de date

con.Close();

}

}

private void button\_medie\_Click(object sender, EventArgs e)

{

try

{

// Se deschide conexiunea la baza de date

con.Open();

// Se definește o comandă SQL pentru a calcula venitul mediu al fiecărui agent

cmd = new SqlCommand("SELECT a.Cod\_Agent, a.Nume\_Agent, AVG(m.Pret \* t.Cantitate) AS Venit\_mediu " +

"FROM Agenti a " +

"JOIN Transporturi t ON a.Cod\_Agent = t.Cod\_Agent " +

"JOIN Materiale m ON t.Cod\_Material = m.Cod\_Material " +

"GROUP BY a.Cod\_Agent, a.Nume\_Agent " +

"ORDER BY a.Cod\_Agent;", con);

// Se creează un obiect DataTable și se completează cu datele obținute prin executarea comenzii SQL

dt = new DataTable();

adpt = new SqlDataAdapter(cmd);

adpt.Fill(dt);

// Se afișează rezultatele în interfața grafică

dataGridView1.DataSource = dt;

// Se închide conexiunea la baza de date

con.Close();

}

catch (Exception ex)

{

// Se afișează un mesaj de eroare în interfața grafică în cazul în care apare o excepție

MessageBox.Show(ex.Message);

}

finally

{

// Se închide conexiunea la baza de date (în cazul în care a fost deschisă)

if (con.State == ConnectionState.Open)

{

con.Close();

}

}

}

private void exportToExcel(DataGridView dataGridView)

{

dataGridView1.SelectAll();

DataObject copydata = dataGridView1.GetClipboardContent();

if (copydata != null) Clipboard.SetDataObject(copydata);

Microsoft.Office.Interop.Excel.Application xlapp = new Microsoft.Office.Interop.Excel.Application();

xlapp.Visible = true;

Microsoft.Office.Interop.Excel.Workbook xlWbook;

Microsoft.Office.Interop.Excel.Worksheet xlsheet;

object miseddata = System.Reflection.Missing.Value;

xlWbook = xlapp.Workbooks.Add(miseddata);

xlsheet = (Microsoft.Office.Interop.Excel.Worksheet)xlWbook.Worksheets.get\_Item(1);

Microsoft.Office.Interop.Excel.Range xlr = (Microsoft.Office.Interop.Excel.Range)xlsheet.Cells[1, 1];

xlr.Select();

xlsheet.PasteSpecial(xlr, Type.Missing, Type.Missing, Type.Missing, Type.Missing, Type.Missing, true);

}

private void button\_excel\_Click(object sender, EventArgs e)

{

try

{

con.Open();

// Definim interogarea SQL pentru a extrage agenții care transportă produse

cmd = new SqlCommand("SELECT \* FROM Transporturi ", con);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

// Exportăm datele într-un fișier Excel

exportToExcel(dataGridView1);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

}

## **Demisionare.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace studiul\_individual.Forms

{

public partial class Demisionare : Form

{

private SqlConnection con;

private SqlCommand cmd;

private SqlDataAdapter adpt;

private DataTable dt;

public Demisionare()

{

InitializeComponent();

con = new SqlConnection(@"Data Source=DESKTOP-F8UOFGQ;Initial Catalog=MT;Integrated Security=True");

// button\_Sterge.Click += button\_Sterge\_Click;

button\_WOC1.Click += button\_WOC1\_Click;

}

private void button\_Sterge\_Click(object sender, EventArgs e)

{

try

{

con.Open();

int codAgent = int.Parse(textBox1.Text);

DateTime dataExcludere = dateTimePicker1.Value;

string motivExcludere = comboBox1.Text;

// Verificăm dacă agentul există în tabelul Agenti

cmd = new SqlCommand("SELECT COUNT(\*) FROM Agenti WHERE Cod\_Agent = @codAgent", con);

cmd.Parameters.AddWithValue("@codAgent", codAgent);

int count = (int)cmd.ExecuteScalar();

if (count > 0)

{

// Introducem informațiile în tabelul Agenti\_Exclusi

cmd = new SqlCommand("INSERT INTO Agenti\_Exclusi (Nume\_Agent, Data\_Excludere, Motiv\_Excludere) SELECT Nume\_Agent, @dataExcludere, @motivExcludere FROM Agenti WHERE Cod\_Agent = @codAgent;", con);

cmd.Parameters.AddWithValue("@codAgent", codAgent);

cmd.Parameters.AddWithValue("@dataExcludere", dataExcludere);

cmd.Parameters.AddWithValue("@motivExcludere", motivExcludere);

cmd.ExecuteNonQuery();

MessageBox.Show("Agentul a fost exclus cu succes.");

}

else

{

MessageBox.Show("Agentul cu codul introdus nu există în baza de date.");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void button\_WOC1\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

## **Agenti\_exclusi.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace studiul\_individual.Forms

{

public partial class FormAgenti\_exclusi : Form

{

//Declararea variabilelor de conexiune și comandă pentru interogarea bazei de date

string path = @"Data Source=DESKTOP-F8UOFGQ;Initial Catalog=MT;Integrated Security=True";

SqlConnection con;

SqlCommand cmd;

SqlDataAdapter adpt;

DataTable dt;

int ID4;

//Constructorul clasei

public FormAgenti\_exclusi()

{ //Inițializarea variabilelor de conexiune și comandă pentru interogarea bazei de date

InitializeComponent();

con = new SqlConnection(path);

display();

button2.Enabled = false;

button3.Enabled = false;

//Adăugarea unui event handler pentru evenimentul "Click" al butonului 2

button2.Click += button2\_Click;

//Adăugarea unui event handler pentru evenimentul "Click" al butonului 3

button3.Click += new EventHandler(button3\_Click);

}

private void Agenți\_Load(object sender, EventArgs e)

{

}

//Metoda care este apelată atunci când utilizatorul apasă butonul "Adaugă Agent"

private void button1\_Click(object sender, EventArgs e)

{

if (Numea.Text == "" || Prenumea.Text == "" || IDNPa.Text == "")

{

MessageBox.Show("Introduce-ti toate datele necesare!");

}

else

{

try

{

con.Open();

//Crearea și executarea comenzii SQL pentru adăugarea unui nou agent în baza de date

cmd = new SqlCommand("INSERT INTO Agenti\_Exclusi( Nume\_Agent, Data\_Excludere, Motiv\_Excludere) values ('" + Numea.Text + "','" + Prenumea.Text + "','" + IDNPa.Text + "')", con);

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Inregistrarea unui agent nou a decurs cu succes!");

clear();

display();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

//Metoda care curăță textul din câmpurile de introducere a datelor pentru un nou agent

public void clear()

{

IDNPa.Text = "";

Prenumea.Text = "";

Numea.Text = "";

}

//Metoda care ne afișează in dataGridView toate datele necesare.

public void display()

{

try

{

dt = new DataTable();

con.Open();

adpt = new SqlDataAdapter("SELECT \*FROM Agenti\_Exclusi", con);

adpt.Fill(dt);

dataGridView1.DataSource = dt;

con.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

// Metoda apelată atunci când utilizatorul apasă butonul "Actualizează"

private void dataGridView1\_CellDoubleClick(object sender, DataGridViewCellEventArgs e)

{

ID4 = int.Parse(dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString());

Numea.Text = dataGridView1.Rows[e.RowIndex].Cells[1].Value.ToString();

Prenumea.Text = dataGridView1.Rows[e.RowIndex].Cells[2].Value.ToString();

IDNPa.Text = dataGridView1.Rows[e.RowIndex].Cells[3].Value.ToString();

button3.Enabled = true;

button2.Enabled = true;

}

private void button3\_Click(object sender, EventArgs e)

{

con.Open();

cmd = new SqlCommand("update Agenti\_Exclusi set Nume\_Agent = '" + Numea.Text + "', Data\_Excludere = '" + Prenumea.Text + "', Motiv\_Excludere = '" + IDNPa.Text + "' where Cod\_Agent = " + ID4, con);

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Actualizarea unui agent exclus a decurs cu succes!");

display();

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

con.Open();

cmd = new SqlCommand("delete from Agenti\_Exclusi where Cod\_Agent = '" + ID4 + "'", con);

cmd.ExecuteNonQuery();

MessageBox.Show("Ați șters un agent!");

display();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void exportToExcel(DataGridView dataGridView)

{

dataGridView1.SelectAll();

DataObject copydata = dataGridView1.GetClipboardContent();

if (copydata != null) Clipboard.SetDataObject(copydata);

Microsoft.Office.Interop.Excel.Application xlapp = new Microsoft.Office.Interop.Excel.Application();

xlapp.Visible = true;

Microsoft.Office.Interop.Excel.Workbook xlWbook;

Microsoft.Office.Interop.Excel.Worksheet xlsheet;

object miseddata = System.Reflection.Missing.Value;

xlWbook = xlapp.Workbooks.Add(miseddata);

xlsheet = (Microsoft.Office.Interop.Excel.Worksheet)xlWbook.Worksheets.get\_Item(1);

Microsoft.Office.Interop.Excel.Range xlr = (Microsoft.Office.Interop.Excel.Range)xlsheet.Cells[1, 1];

xlr.Select();

xlsheet.PasteSpecial(xlr, Type.Missing, Type.Missing, Type.Missing, Type.Missing, Type.Missing, true);

}

private void button\_excel\_Click(object sender, EventArgs e)

{

try

{

con.Open();

// Definim interogarea SQL pentru a extrage agenții care transportă produse

cmd = new SqlCommand("SELECT \* FROM Agenti\_exclusi ", con);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

// Exportăm datele într-un fișier Excel

exportToExcel(dataGridView1);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

}

}

## **AgentVanzari.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace studiul\_individual.Forms

{

public partial class Agent\_vanzari : Form

{

private SqlConnection con;

private SqlCommand cmd;

private SqlDataAdapter adpt;

private DataTable dt;

public Agent\_vanzari()

{

InitializeComponent();

con = new SqlConnection(@"Data Source=DESKTOP-F8UOFGQ;Initial Catalog=MT;Integrated Security=True");

button\_produse.Click += button\_produse\_Click;

}

private void button\_produse\_Click(object sender, EventArgs e)

{

string numeAgent = textBox\_nume.Text;

// verificăm dacă TextBox-ul a fost completat

if (string.IsNullOrWhiteSpace(numeAgent))

{

MessageBox.Show("Introduceți numele agentului.");

return;

}

try

{

con.Open();

// verificăm dacă agentul există în baza de date

cmd = new SqlCommand("SELECT COUNT(\*) FROM Agenti WHERE Nume\_Agent = @numeAgent", con);

cmd.Parameters.AddWithValue("@numeAgent", numeAgent);

int agentCount = (int)cmd.ExecuteScalar();

if (agentCount == 0)

{

MessageBox.Show("Agentul nu există în baza de date.");

return;

}

// continuăm cu interogarea originală

cmd = new SqlCommand("SELECT t.Data\_Livrare, m.Denumire, t.Cantitate, m.Pret \* t.Cantitate AS Valoare, a.Nume\_Agent FROM Transporturi t JOIN Materiale m ON t.Cod\_Material = m.Cod\_Material JOIN Agenti a ON t.Cod\_Agent = a.Cod\_Agent WHERE a.Nume\_Agent = @numeAgent", con);

cmd.Parameters.AddWithValue("@numeAgent", numeAgent);

adpt = new SqlDataAdapter(cmd);

dt = new DataTable();

adpt.Fill(dt);

dataGridView1.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

con.Close();

}

}

private void button\_inapoi\_Click(object sender, EventArgs e)

{

this.Close();

}

private void button\_Sterge\_Click(object sender, EventArgs e)

{

textBox\_nume.Text = "";

}

private void textBox\_nume\_KeyPress(object sender, KeyPressEventArgs e)

{

e.Handled = !char.IsLetter(e.KeyChar) && !char.IsWhiteSpace(e.KeyChar) && !char.IsControl(e.KeyChar);

}

}

}

## **reii.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 studiul\_individual.Forms

{

public partial class reii : Form

{

public reii()

{

InitializeComponent();

}

private void reii\_Load(object sender, EventArgs e)

{

// TODO: This line of code loads data into the 'mTDataSet1.Materiale' table. You can move, or remove it, as needed.

this.materialeTableAdapter.Fill(this.mTDataSet1.Materiale);

this.buttonCloseReport.RefreshReport();

}

private void button\_WOC1\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

# **Baza de date SQL**

set dateformat ymd

use master

go

if exists(select 'True' from sys.databases where name='MT')

begin

alter database MT set single\_user with rollback immediate

drop database MT

end

go

Create database MT

go

Alter authorization on database ::MT to sa

use MT

CREATE TABLE Agenti (

Cod\_Agent int PRIMARY KEY identity(1,1),

Nume\_Agent varchar(50) NOT NULL,

Data\_Nastere date NOT NULL,

Data\_Angajarii date NOT NULL,

Salariu int NOT NULL,

Sex varchar(10) NOT NULL,

Adresa varchar(100) NOT NULL

);

GO

-- Tabela "Materiale"

CREATE TABLE Materiale (

Cod\_Material int PRIMARY KEY identity(1,1),

Denumire varchar(50) NOT NULL,

Stoc int NOT NULL,

Pret int NOT NULL

);

GO

-- Tabela "Transporturi"

CREATE TABLE Transporturi (

Cod\_Transport int PRIMARY KEY identity(1,1),

Data\_Livrare date NOT NULL,

Cantitate int NOT NULL,

Cod\_Agent int ,

Cod\_Material int ,

FOREIGN KEY (Cod\_Agent) REFERENCES Agenti(Cod\_Agent),

FOREIGN KEY (Cod\_Material) REFERENCES Materiale(Cod\_Material)

);

GO

-- Tabela "Agenti\_Exclusi"

CREATE TABLE Agenti\_Exclusi (

Cod\_Agent int PRIMARY KEY identity(1,1),

Nume\_Agent varchar(50) NOT NULL,

Data\_Excludere date NOT NULL,

Motiv\_Excludere varchar(100) NOT NULL

);

GO

-- Tabela "Utilizatori"

CREATE TABLE Utilizatori (

Cod\_Utilizator int PRIMARY KEY identity(1,1),

Nume\_Utilizator varchar(50) NOT NULL,

Parola varchar(50) NOT NULL,

Tip\_Utilizator varchar(50) NOT NULL

);

GO

INSERT INTO Agenti (Nume\_Agent, Data\_Nastere, Data\_Angajarii, Salariu, Sex, Adresa)

VALUES

( 'Ion Popescu', '1955-01-01', '2010-05-01', 5000, 'M', 'Str. Victoriei, nr. 1'),

( 'Maria Ionescu', '1938-03-15', '2012-03-01', 4500, 'F', 'Str. Libertatii, nr. 10'),

('Andrei Popa', '1990-07-05', '2015-01-01', 5500, 'M', 'Str. Mihai Eminescu, nr. 25'),

('Ana Munteanu', '1993-11-21', '2018-05-01', 6000, 'F', 'Str. Stefan cel Mare, nr. 12'),

('Mihai Radu', '1986-06-10', '2011-02-01', 4800, 'M', 'Str. Gheorghe Doja, nr. 8'),

('Cristina Stoica', '1911-12-01', '2016-01-01', 5200, 'F', 'Str. Avram Iancu, nr. 34'),

('Alexandru Tudor', '1989-09-18', '2014-07-01', 5300, 'M', 'Str. Cuza Voda, nr. 15'),

('Elena Dumitrescu', '1994-04-05', '2019-03-01', 6200, 'F', 'Str. Ion Creanga, nr. 21'),

('George Pop', '1987-02-14', '2012-09-01', 4700, 'M', 'Str. Stefan cel Mare, nr. 5'),

('Ana Maria Petrescu', '1992-10-02', '2017-04-01', 5700, 'F', 'Str. Tudor Vladimirescu, nr. 18');

GO

INSERT INTO Materiale (Denumire, Stoc, Pret)

VALUES ('Beton', 1000, 100),

('Fier beton', 2000, 200),

('Caramida', 5000, 50),

('Placi OSB', 1500, 80),

('Polistiren', 800, 70),

('Vopsea lavabila', 1200, 30),

('Gresie', 3000, 60),

('Faianta', 2500, 70),

('Lemn de constructii', 1800, 120),

('Geam termopan', 1000, 150);

GO

INSERT INTO Transporturi (Data\_Livrare, Cantitate, Cod\_Agent, Cod\_Material)

VALUES

('2023-05-13', 100, 1, 2),

('2023-05-14', 50, 2, 3),

('2023-05-15', 75, 3, 4),

('2023-05-16', 200, 4, 5),

('2023-05-17', 120, 5, 6),

('2023-05-18', 90, 6, 7),

('2023-05-19', 40, 7, 8),

('2023-04-20', 150, 8, 9),

('2023-04-21', 80, 9, 10),

('2023-04-22', 50, 10, 1),

('2023-04-23', 100, 1, 3),

('2023-03-24', 50, 2, 4),

('2023-04-25', 75, 3, 5),

('2023-04-26', 200, 4, 6),

('2023-04-27', 120, 5, 7),

('2023-04-28', 90, 6, 8),

('2023-04-29', 40, 7, 9),

('2023-04-30', 150, 8, 10),

('2023-03-31', 80, 9, 1),

('2023-03-01', 50, 10, 2),

('2023-03-02', 100, 1, 3),

('2023-03-03', 50, 2, 4),

('2023-03-04', 75, 3, 5),

('2023-03-05', 200, 4, 6),

('2023-03-06', 120, 5, 7),

('2023-03-07', 90, 6, 8),

('2023-03-08', 40, 7, 9),

('2023-03-09', 150, 8, 10),

('2023-03-10', 80, 9, 1),

('2023-03-11', 50, 10, 2),

('2023-03-12', 100, 1, 4),

('2023-03-13', 50, 2, 5),

('2023-03-14', 75, 3, 6),

('2023-03-15', 200, 4, 7),

('2023-03-16', 120, 5, 8),

('2023-03-17', 90, 6, 9),

('2023-03-18', 40, 7, 10),

('2023-03-19', 150, 8, 1),

('2023-03-20', 80, 9, 2),

('2023-03-21', 50, 10, 3)

GO

INSERT INTO Agenti\_Exclusi (Nume\_Agent, Data\_Excludere, Motiv\_Excludere)

VALUES ('Marcel Ionescu', '2023-04-01', 'Incetare contract de comun acord'),

('Adriana Popescu', '2023-04-15', 'Incompatibilitate cu pozitia ocupata'),

('Ciprian Mihai', '2023-05-01', 'Fapte de coruptie'),

('Andreea Radulescu', '2023-05-01', 'Incetare contract de comun acord'),

('Razvan Tudor', '2023-05-02', 'Incompatibilitate cu pozitia ocupata'),

('Ana-Maria Dumitru', '2023-05-02', 'Fapte de coruptie'),

('Vladimir Ionescu', '2023-05-02', 'Incetare contract de comun acord'),

('Diana Pop', '2023-05-02', 'Incompatibilitate cu pozitia ocupata'),

('Alexandru Georgescu', '2023-05-02', 'Fapte de coruptie'),

('Maria-Alexandra Stoica', '2023-05-02', 'Incetare contract de comun acord');

GO

INSERT INTO Utilizatori (Nume\_Utilizator, Parola, Tip\_Utilizator)

VALUES ('admin', 'admin123', 'Administrator'),

('user1', 'user123', 'Utilizator obisnuit'),

('user2', 'user456', 'Utilizator obisnuit'),

('user3', 'user789', 'Utilizator obisnuit'),

('user4', 'userabc', 'Utilizator obisnuit'),

('user5', 'userdef', 'Utilizator obisnuit'),

('user6', 'userxyz', 'Utilizator obisnuit'),

('user7', 'usermno', 'Utilizator obisnuit'),

('user8', 'userpqr', 'Utilizator obisnuit'),

('user9', 'userstu', 'Utilizator obisnuit');

GO

SELECT a.Cod\_Agent, a.Nume\_Agent, AVG(m.Pret \* t.Cantitate) AS Venit\_mediu

FROM Agenti a

JOIN Transporturi t ON a.Cod\_Agent = t.Cod\_Agent

JOIN Materiale m ON t.Cod\_Material = m.Cod\_Material

GROUP BY a.Cod\_Agent, a.Nume\_Agent

ORDER BY a.Cod\_Agent;

# **Concluzie**

Sarcina dată implică crearea unei aplicații vizuale pentru gestionarea unei baze de date care conține informații despre materialele de construcție dintr-un depozit și activitatea agenților de vânzări. Aplicația trebuie să ofere funcționalități precum înregistrarea unui nou agent de vânzări, înregistrarea unui nou produs în depozit, excluderea informațiilor despre un agent specific și transferul acestora într-o altă tabelă, determinarea celui mai tânăr agent de vânzări, afișarea listei agenților care au atins vârsta de pensionare, determinarea salariului mediu lunar al agenților, exportul listei agenților care transportă produse într-un fișier MS Word sau MS Excel, afișarea informațiilor despre materialele transportate de un anumit agent, determinarea prețului mediu al unui produs ales, afișarea listei celor mai vândute materiale de construcții și afișarea vechimii în muncă a fiecărui angajat prin intermediul rapoartelor.

Pentru a implementa această aplicație, am folosit limbajului de programare C# și platformei .NET, precum și bazei de date SQL Management Studio pentru stocarea și gestionarea informațiilor. În cadrul aplicației vizuale, trebuie să fie create formulare, meniuri și controale pentru a permite utilizatorului să interacționeze cu baza de date și să acceseze informațiile dorite.

De asemenea, este important să se stabilească criterii de integritate pentru baza de date, să se administreze obiectele bazei de date, să se planifice întreținerea și să se asigure securitatea datelor din baza de date. Utilizarea rapoartelor va permite afișarea informațiilor într-un format structurat și ușor de înțeles pentru utilizator.

În concluzie, sarcina de a crea o aplicație vizuală pentru gestionarea unei baze de date privind materialele de construcție și activitatea agenților de vânzări implică implementarea unui set complex de funcționalități, cu accent pe interacțiunea cu utilizatorul și generarea de rapoarte pentru afișarea informațiilor. Acest proiect necesită cunoștințe solide de programare, utilizarea limbajului C# și .NET, precum și cunoștințe în gestionarea bazelor de date cu SQL Management Studio.

# **Bibliografie**

<https://learn.microsoft.com/en-us/sql/tools/command-prompt-utility-reference-database-engine?view=sql-server-ver16>

<http://www.c-sharpcorner.com/Internet.asp>

<https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/>

<https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/interfaces/explicit-interface-implementation>