

Liceul Tehnologic „Școala Națională de Gaz”-Mediaș

Proiect pentru susținerea atestatului profesional

-Steam: platformă de distribuție digitală a jocurilor video-

2018

Profesor coordonator: Sîrbu Margareta

Elev: Folea Roxana-Andreea

Cuprins

1. Motivul și scopul proiectului.....	3
2. Resurse hardware și software necesare	4
3. Prezentarea generală a temei	5
4. Teoria programării orientate pe obiect și a bazelor de date	6
4.1. Principiile programării orientate pe obiect	6
4.2. Concepte de bază ale programării vizuale	6
4.3. Accesarea și prelucrearea datelor prin intermediul SQL Server	7
5. Descrierea aplicației.....	9
5.1. Baza de date	9
5.2. Primul formular	9
5.3. Al doilea formular	11
5.4. Al treilea formular	12
5.4.1. Formularul „Profil”	13
5.4.1.1. Formularul de schimbare a parolei.....	13
5.4.1.2. Formularul de schimbare a adresei de email	14
5.4.2. Formularul „Listă_Jocuri”	15
5.4.2.1. Formularul „Detalii comandă”	16
5.4.3. Formularul „Info”	18
5.4.4. Formularul „News”	18
6. Bibliografie și webografie.....	19

1. Motivul și scopul proiectului

În ultimii ani, dezvoltarea sistemelor de baze de date reprezintă unul dintre cele mai importante aspecte în domeniul tehnologiei informației, având un impact decisiv asupra modului de organizare și funcționare a numeroaselor instituții și servicii. Acestea sunt dependente de funcționarea corectă și neîntreruptă a sistemelor de baze de date. Pentru a ilustra modul de funcționare a unei baze de date, am creat aplicația *Steam: platformă de distribuție digitală a jocurilor video* în programul Visual Studio Professional 2017, limbajul C#.

Scopul proiectului este de a servi ca atestat profesional la informatică.

2. Resurse hardware și software necesare

Hardware

- 1.8 GHz or faster processor. Dual-core or better recommended
- 2 GB of RAM; 4 GB of RAM recommended (2.5 GB minimum if running on a virtual machine)
- Hard disk space: up to 130 GB of available space, depending on features installed; typical installations require 20-50 GB of free space.
- Hard disk speed: to improve performance, install Windows and Visual Studio on a solid state drive (SSD).
- Video card that supports a minimum display resolution of 720p (1280 by 720); Visual Studio will work best at a resolution of WXGA (1366 by 768) or higher.

Supported Operating Systems

Visual Studio 2017 will install and run on the following operating systems:

- Windows 10 version 1507 or higher: Home, Professional, Education, and Enterprise (LTSC and S are not supported)
- Windows Server 2016: Standard and Datacenter
- Windows 8.1 (with [Update 2919355](#)): Core, Professional, and Enterprise
- Windows Server 2012 R2 (with [Update 2919355](#)): Essentials, Standard, Datacenter
- Windows 7 SP1 (with latest Windows Updates): Home Premium, Professional, Enterprise, Ultimate

3. Prezentarea generală a temei

Aplicația reprezintă o versiune mult simplificată a platformei globale de distribuție digitală a jocurilor video pentru *Windows*, dezvoltată de *Valve Corporation: Steam*. Aceasta efectuează operațiuni de bază specifice unei astfel de platforme cum ar fi: logarea, crearea unui nou cont, afișarea detaliilor privitoare la jocul selectat, schimbarea parolei sau/și a adresei de email și cumpărarea jocurilor din lista de jocuri. Am adăugat de asemenea o filă cu noutăți și una care prezintă o descriere a platformei reale.

4. Teoria programării orientate pe obiect și a bazelor de date

4.1. Principiile programării orientate pe obiect

Ideea POO este de a crea programele ca o colecție de obiecte, unități individuale de cod care interacționează unele cu altele, în loc de simple liste de instrucțiuni sau de apeluri de proceduri.

Obiectele POO sunt, de obicei, reprezentări ale obiectelor din viața reală (domeniul problemei), astfel încât programele realizate prin tehnica POO sunt mai ușor de înțeles, de depanat și de extins decât programele procedurale. Aceasta este adevărată mai ales în cazul proiectelor software complexe și de dimensiuni mari.

Principiile POO sunt:

- Abstractizarea - principiu care permite identificarea caracteristicilor și comportamentului obiectelor ce țin nemijlocit de domeniul problemei. Rezultatul este un model. În urma abstractizării, entităților din domeniul problemei se definesc prin clase.
- Încapsularea – numită și ascunderea de informații, este caracterizată prin 2 aspecte:
a. Gruparea comportamentelor și caracteristicilor într-un tip abstract de date b. Definirea nivelului de acces la datele unui obiect
- Moștenirea – organizează și facilitează polimorfismul și încapsularea permițând definirea și crearea unor clase specializate plecând de la clase (generale) care sunt deja definite - acestea pot împărtăși (și extinde) comportamentul lor fără a fi nevoie de redefinirea aceluiași comportament.
- Polimorfismul - posibilitatea mai multor obiecte dintr-o ierarhie de clase de a utiliza denumiri de metode cu același nume dar, cu un comportament diferit.

4.2. Concepte de bază ale programării vizuale

Programarea vizuală trebuie privită ca un mod de proiectare a unui program prin operare directă asupra unui set de elemente grafice (de aici vine denumirea de programare vizuală). Această operare are ca efect scrierea automată a unor secvențe de program, secvențe care, împreună cu secvențele scrise textual vor forma programul.

Spunem că o aplicație este vizuală dacă dispune de o interfață grafică sugestivă și pune la dispoziția utilizatorului instrumente specifice de utilizare (drag, clic, hint etc.)

Realizarea unei aplicații vizuale nu constă doar în desenare și aranjare de controale, ci presupune în principal stabilirea unor decizii arhitecturale, decizii ce au la bază unul dintre modelele arhitecturale de bază.

În realizarea aplicației mai trebuie respectate și principiile proiectării interfețelor:

- **Simplitatea:** Interfața trebuie să fie cât mai ușor de înțeles și de învățat de către utilizator și să permită acestuia să efectueze operațiile dorite în timp cât mai scurt. În acest sens, este vitală culegerea de informații despre utilizatorii finali ai aplicației și a modului în care aceștia sunt obișnuiți să lucreze.
- **Poziția controalelor:** Locația controalelor dintr-o fereastră trebuie să reflecte importanța relativă și frecvența de utilizare. Astfel, când un utilizator trebuie să introducă niște informații – unele obligatorii și altele opționale – este indicat să organizăm controalele astfel încât primele să fie cele care preiau informații obligatorii.
- **Consistența:** Ferestrele și controalele trebuie să fie afișate după un design asemănător („template”) pe parcursul utilizării aplicației. Înainte de a implementa interfața, trebuie decidem cum va arăta aceasta, să definim „template”-ul.
- **Estetica:** Interfața trebuie să fie pe cât posibil plăcută și atrăgătoare.

4.3. Accesarea și prelucrearea datelor prin intermediul SQL Server

Accesarea și prelucrarea datelor cu ajutorul mediului vizual Mediul de dezvoltare Visual Studio dispune de instrumente puternice și sugestive pentru utilizarea bazelor de date în aplicații. Conceptual, în spatele unei ferestre în care lucrăm cu date preluate dintr-una sau mai multe tabele ale unei baze de date se află obiectele din categoriile Connection, Command, DataAdapter și DataSet prezentate.

După crearea unei baze de date în SQL informațiile înregistrate în tabela sau tabellele bazei de date pot fi utilizate într-o aplicație din Visual C# într-un formular sau într-o aplicație consolă. Vom prezenta acum modul în care se poate utiliza o bază de date într-un formular creat în Windows Forms.

Atunci când într-un formular utilizăm un tabel trebuie să avem posibilitatea de a utiliza funcțiile ce operează asupra datelor incluse în el. Toate instrucțiunile prezentate în capitolul Introducere în limbajul SQL pot fi accesate și pe un formular. Prin "tragerea" unor obiecte din fereastra Data Sources în fereastra noastră nouă, se creează automat obiecte specifice. În partea de jos a figurii se pot observa obiectele de tip Dataset, TableAdapter, BindingSource, BindingNavigator și, în fereastră, TableGridView. BindingNavigator este un tip ce permite, prin instanțiere, construirea barei de navigare care facilitează operații de deplasare, editare, ștergere și adăugare în table etc.

De obicei, sursa de date este o bază de date, dar ar putea de asemenea să fie un fișier text, o foaie Excel, un fișier Access sau un fișier XML. În aplicațiile tradiționale cu baze de date, clienții stabilesc o conexiune cu baza de date și mențin această conexiune deschisă până la încheierea executării aplicației. Conexiunile deschise necesită alocarea de resurse sistem. Atunci când menținem mai multe conexiuni deschise server-ul de baze de date va răspunde mai lent la comenzile clienților întrucât cele mai multe baze de date permit un număr foarte mic de conexiuni concurente. ADO.NET permite și lucrul în stil conectat dar și lucrul în stil deconectat,

aplicațiile conectându-se la server-ul de baze de date numai pentru extragerea și actualizarea datelor. Acest lucru permite reducerea numărului de conexiuni deschise simultan la sursele de date. ADO.NET oferă instrumentele de utilizare și reprezentare XML pentru transferul datelor între aplicații și surse de date, furnizând o reprezentare comună a datelor, ceea ce permite accesarea datelor din diferite surse de diferite tipuri și prelucrarea lor ca entități, fără să fie necesar să convertim explicit datele în format XML sau invers.

Aceste caracteristici sunt determinate în stabilirea beneficiilor furnizate de ADO.NET: Interoperabilitate. ADO.NET poate interacționa ușor cu orice componentă care suportă XML.

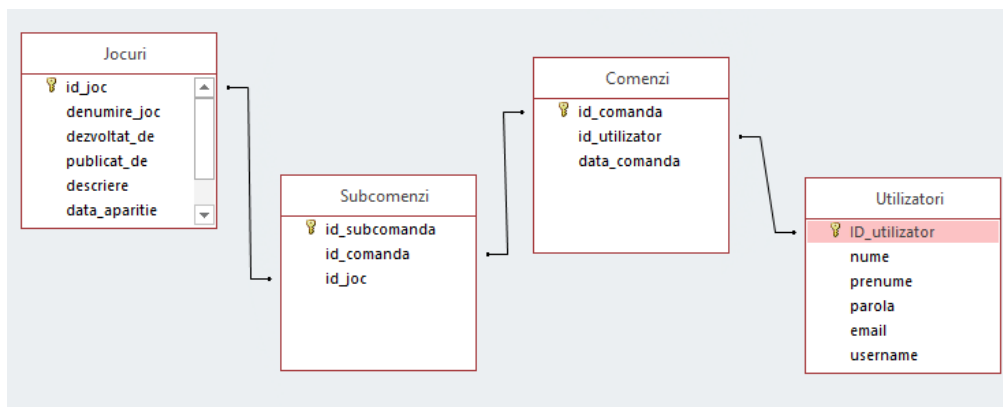
- Durabilitate. ADO.NET permite dezvoltarea arhitecturii unei aplicații datorită modului de transfer a datelor între nivelele arhitecturale.
- Programabilitate. ADO.NET simplifică programarea pentru diferite task-uri cum ar fi comenzile SQL, ceea ce duce la o creștere a productivității și la o scădere a numărului de erori.
- Performanță. Nu mai este necesară conversia explicită a datelor la transferul între aplicații, fapt care duce la crește performanțelor acestora.
- Accesibilitate. Utilizarea arhitecturii deconectate permite accesul simultan la același set de date. Reducerea numărului de conexiuni deschise simultan determină utilizarea optimă a resurselor.

5. Descrierea aplicației

Această aplicație este alcătuită din 11 Formulare Windows și este conectată la o bază de date Access.

5.1. Baza de date

Baza de date la care este conectat proiectul este creată în Access și conține următoarele tabele:



5.2. Primul formular

Primul formular permite logarea utilizatorului utilizând un cont existent, sau crearea unui cont nou.

The screenshot shows a Windows application window with a dark background and a Steam logo in the top right corner. The form contains the following elements:

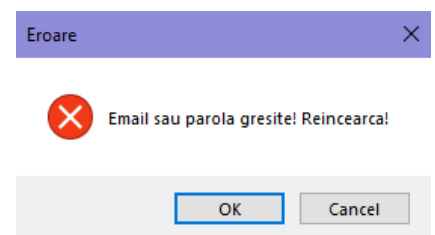
- Username**: A text input field.
- Parola**: A password input field.
- Login**: A button.
- Cancel**: A button.
- Nu ai încă un cont?**: A text label.
- Creează un cont nou**: A button.

Logarea:

```
private void loginbutton_Click(object sender, EventArgs e)
{
    int ct = 0;
    con.Open();
    string sql = "select ID_utilizator from Utilizatori " +
        "where username='" + usertextBox.Text + "'" +
        " and parola='" + passtextBox.Text + "'";
    OleDbCommand cmd = new OleDbCommand(sql, con);
    OleDbDataReader rdr = cmd.ExecuteReader();

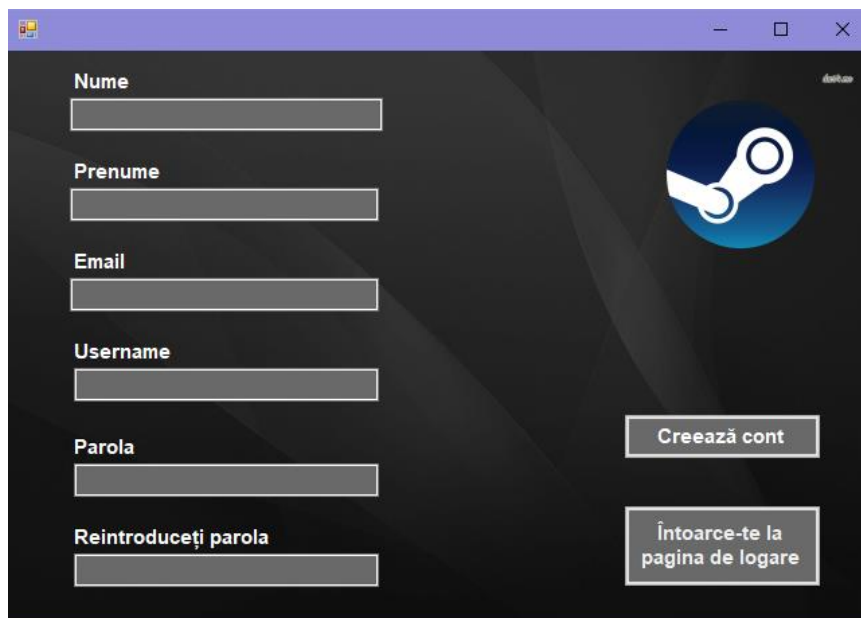
    while (rdr.Read())
    {
        ct++;
        id = int.Parse(rdr[0].ToString());
    }
    if (ct > 0)
    {
        con.Close();
        Meniu f = new Meniu(id);
        f.Show();
        Hide();
    }
}
```

În cazul în care numele sau parola nu sunt corecte
se va afișa un MessageBox care ne va avertiza:



5.3. Al doilea formular

Cel de-al doilea formular este cel de înregistrare. Aici utilizatorul poate crea un cont care îi va fi folositor ulterior pentru logare.



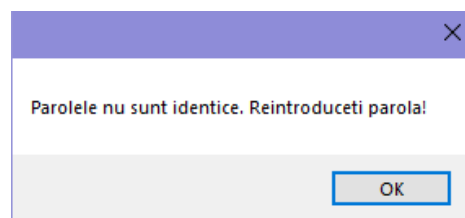
The screenshot shows a registration form window with a dark background and a blue circular logo on the right. The form contains the following fields and buttons:

- Nume:
- Prenume:
- Email:
- Username:
- Parola:
- Reintroduceți parola:
- Creează cont:
- Întoarce-te la pagina de logare:

Inserarea datelor utilizatorului în baza de date se face astfel:

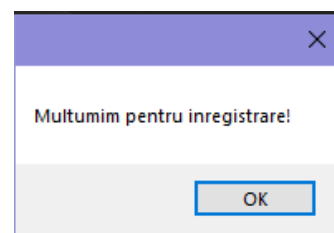
```
OleDbCommand cmd=new OleDbCommand("insert into  
Utilizatori(num,prenume,email,username,parola)" +  
" values('" + numetextBox.Text + "','" + prenumetextBox.Text + "','" +  
+ emailtextBox.Text + "','" + usernametextBox.Text + "','" +  
parola1textBox.Text + "')" ,con);  
cmd.ExecuteNonQuery();
```

În cazul în care parolele nu coincid se va afișa mesajul următor:



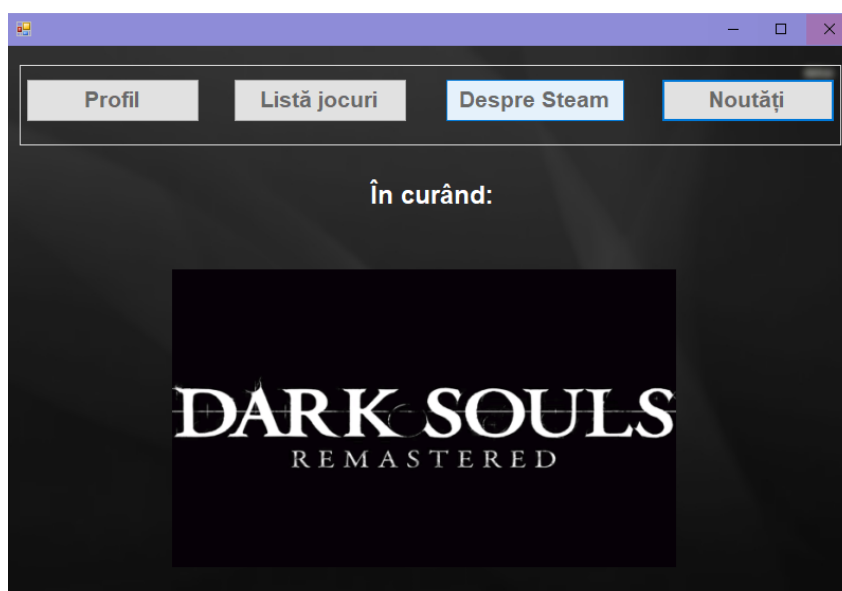
The screenshot shows a small error message dialog box with a close button (X) in the top right corner. The text inside the dialog reads: "Parolele nu sunt identice. Reintroduceti parola!". There is an "OK" button at the bottom right.

După ce crearea contului a fost realizată cu succes se va afișa mesajul următor:



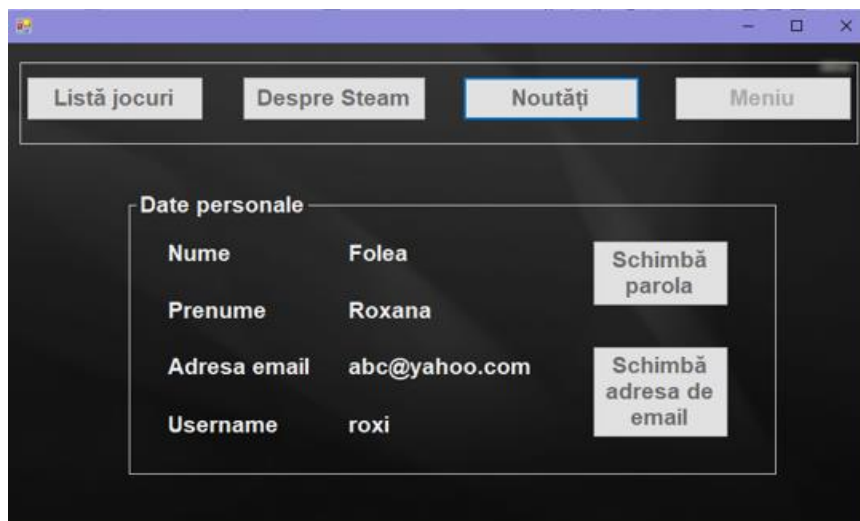
5.4. Al treilea formular

Al treilea formular este un meniu. Cele 4 butoane situate în partea superioară a formularului îi permit utilizatorului să desfășoare anumite acțiuni, cum ar fi: vizualizarea profilului, schimbarea datelor utilizatorului, cumpărarea unor jocuri aflate în oferta *Steam* șamd.



5.4.1. Formularul „Profil”

Acest formular afișează datele utilizatorului logat și de asemenea îi permite utilizatorului să-și schimbe parola și/sau adresa de email.



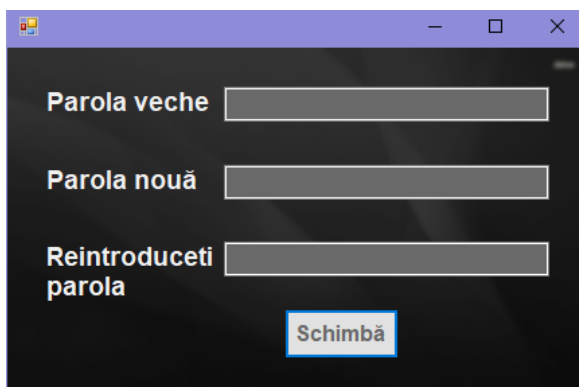
Date personale	
Nume	Folea
Prenume	Roxana
Adresa email	abc@yahoo.com
Username	roxi

Schimbă parola

Schimbă adresa de email

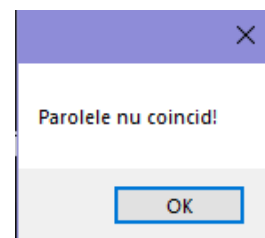
5.4.1.1. Formularul de schimbare a parolei

Acest formular se va deschide după apăsarea butonului „Schimbă parola” și îți va permite să îți modifice parola.



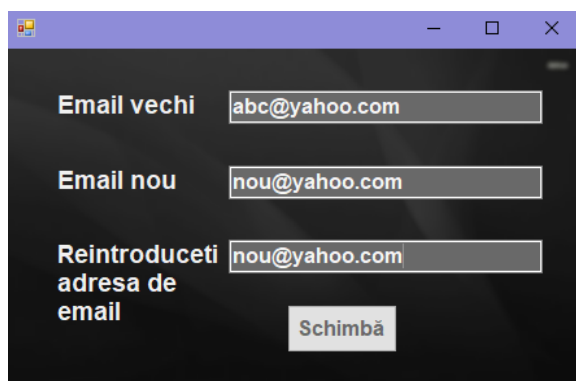
```
string s = "update Utilizatori  
set parola='" + textBox2.Text +  
' where id_utilizator=' + id +  
'";  
OleDbCommand cmd = new  
OleDbCommand(s, con);  
  
cmd.ExecuteNonQuery();
```

În cazul în care parola nouă din câmpul „Parola nouă” nu coincide cu parola din câmpul „Reintroduceți parola” se va afișa mesajul:



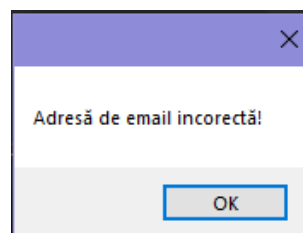
5.4.1.2. Formularul de schimbare a adresei de email

Acest formular se va deschide după apăsarea butonului „Schimbă adresa de email” și îți va permite să îți modifici adresa de email.

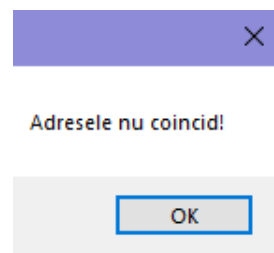


```
string s = "update Utilizatori  
set email='" + textBox2.Text +  
' where id_utilizator='" + id +  
'";  
  
OleDbCommand cmd = new  
OleDbCommand(s, con);  
  
cmd.ExecuteNonQuery();
```

În cazul în care adresa de email introdusă nu există se va afișa mesajul:

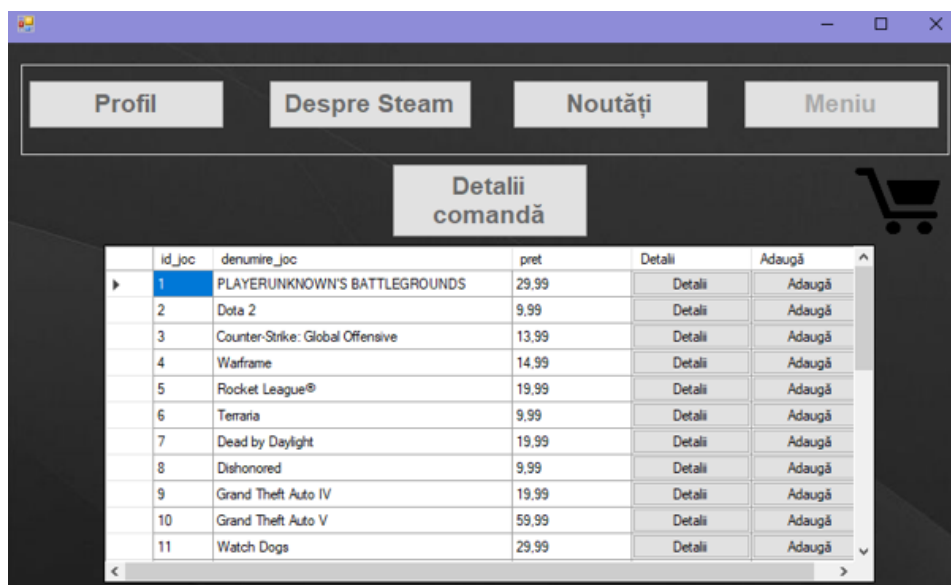


În cazul în care adresa nouă din câmpul „Email nou” nu coincide cu adresa din câmpul „Reintroduceți adresa de email” se va afișa mesajul:



5.4.2. Formularul „Listă_Jocuri”

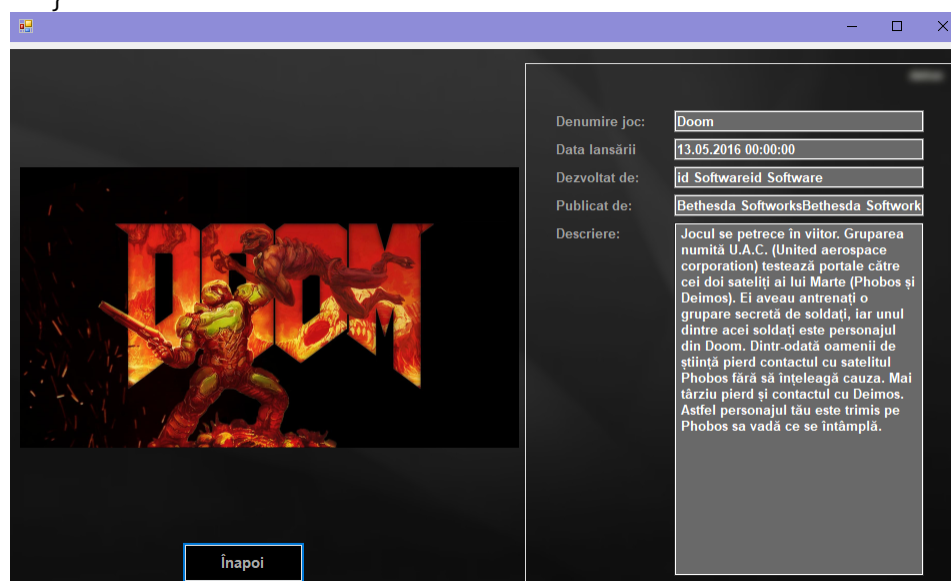
Acest formular afisează o listă cu toate jocurile disponibile și de asemenea îi oferă utilizatorului posibilitatea de a afla mai multe informații despre un joc(butonul „Detalii”) și de a-l cumpăra eventual(Adaugă).



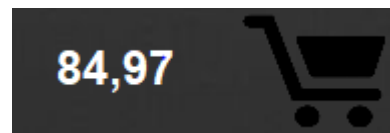
Butonul „Detalii”:

```
if (e.ColumnIndex == dataGridView1.Columns["detalii"].Index)
{
    DataGridViewRow r = dataGridView1.CurrentRow;
    int idj = int.Parse(r.Cells["id_joc"].Value.ToString());

    Form f = new Descriere(idj);
    f.ShowDialog();
}
```



Butonul „Adaugă”:



```

        if (e.ColumnIndex ==
dataGridView1.Columns["adauga"].Index)
        {
            DataGridViewRow r1 = dataGridView1.CurrentRow;
            double p= double.Parse(r1.Cells["pret"].Value.ToString());
            prettotal = prettotal + p;
            label1.Text = prettotal.ToString();
            con.Open();
            int idj = int.Parse(r1.Cells["id_joc"].Value.ToString());
            string sql = "insert into Subcomenzi (id_comanda,id_joc) values (" +
idcom + "," + idj + ")";
            OleDbCommand cmd = new OleDbCommand(sql, con);
            cmd.ExecuteNonQuery();
            ct++;
            con.Close();
        }
    
```

5.4.2.1. Formularul „Detalii comandă”

Elimină	id_joc	denumire_joc	pret
▶ Elimină	16	Metin2	2,99
▶ Elimină	18	Assassin's Creed Unity	29,99
▶ Elimină	22	Call of Duty: Modern Warfare 2	19,99
* Elimină			

Status comandă

Preț total

Butonul „Elimină”:

```

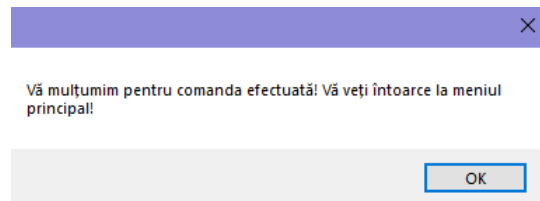
if (e.ColumnIndex == dataGridView1.Columns["elimina"].Index)
{
    float p = 0; //pentru preluare pret, cantit, kcal eliminate din comanda
    DataGridViewRow r = dataGridView1.CurrentRow.OwningRow;
    p = float.Parse(r.Cells["pret"].Value.ToString());

    con.Open();
    string sql = "select id_subcomanda " +
        "from Subcomenzi " +
        "where id_comanda=" + Lista_jocuri.idcom + " and " +
        "id_joc=" + r.Cells["id_joc"].Value + "";
    OleDbCommand cmd = new OleDbCommand(sql, con);
    OleDbDataReader rdr = cmd.ExecuteReader();
    while (rdr.Read())
    {
        idsub = int.Parse(rdr["id_subcomanda"].ToString());
    }
    string sql2 = "delete from Subcomenzi where id_subcomanda=" + idsub + "";
    cmd = new OleDbCommand(sql2, con);
    cmd.ExecuteNonQuery();
    con.Close();
    con.Open();
    string sql3 = @"select a.id_joc,b.denumire_joc,b.pret " +
        "from Subcomenzi a inner join Jocuri b " +
        "on a.id_joc=b.id_joc " +
        "where a.id_comanda=" + Lista_jocuri.idcom + "";
    OleDbCommand cmd2 = new OleDbCommand(sql3, con);
    OleDbDataAdapter da = new OleDbDataAdapter(cmd2);
    DataTable dt = new DataTable();
    da.Fill(dt);
    dataGridView1.DataSource = dt;
    textBox2.Text = (float.Parse(textBox2.Text) - p).ToString();

    con.Close();
}

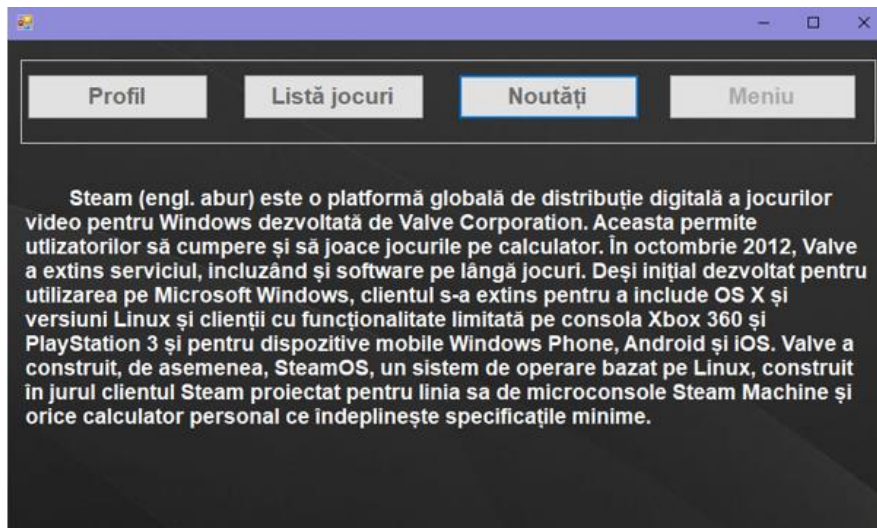
```

După ce comanda va fi finalizată se va afișa mesajul:



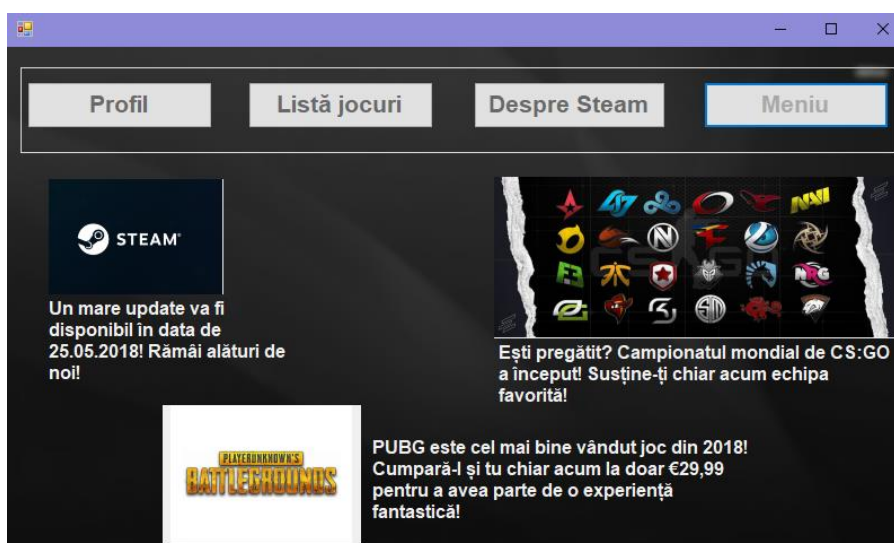
5.4.3. Formularul „Info”

Prin intermediul acestui formular utilizatorul poate afla informații despre platforma pe care o folosește.



5.4.4. Formularul „News”

Acest formular prezintă informații referitoare la următoarele update-uri ale clientului, evenimente legate de jocurile disponibile în oferta *Steam* și de asemenea despre *bestseller-uri*.



6. Bibliografie și webografie

<https://docs.microsoft.com/en-us/visualstudio/productinfo/vs2017-system-requirements-vs>

<https://store.steampowered.com/>

<https://www.google.ro/imghp?hl=ro&tab=wi>