Universitatea Tehnica „Gheorghe Asachi” din IAȘI
Facultatea de Automatica si Calculatoare

Proiect Ingineria Programarii
Aplicatie de tip food delivery

Jardă Maria-Elisabeta, 1309B

Gîza David-Noel, 1309B

Moroșanu Radu-George, 1309B

Țiganaș Ștefan-Gabriel, 1309B

Curpins

1) Despre aplicatie

Aplicatia prezentata se numeste „Aplicatie de tip food delivery” prin intermediul careia clientii care o utilizeaza pot plasa comenzi de produse de la diferite restaurante. Totodata aceasta poate fi folosita si de curieri, care primesc informatii cu privire la comenzile pe care acesita trebuie sa le livreze.

Aplicatia noastra poate fi folosita de orice categorie de varste, avand un design prietenos si functionalitate simpla, astfel incat, chiar si cei fara experienta in plasarea de comenzi online, o pot utiliza fara probleme.

Credem ca „Aplicatia de tip food delivery” este importanta deoarece clientii care doresc sa plaseze comenzi de mancare, in mediul virtual, sunt scutiti de a consuma timp, energie sau combustibil. Help-ul asociat interfetei, vine in ajutorul celor care intampina probleme cand folosesc aplicatia.

La pornirea programului, se va deschide interfata, care intial va porni cu un panou de logre. Logarea se face in doua moduri:

1) pentru clienti, exista un fisier client.txt in care sunt memorate numele si parola acestora
2) pentru curieri, numele si parola pentru logare sunt „curier-curier”.

La apasarea butonului „Login” se va permite logarea, in functie de nume si parola. Daca acestea sunt gresite, se va afisa un mesaj de atentionare. De asemenea, in primul panou se afla si butonul de help care odata apasat, va deschide un meniu ajutator cu privire la aplicatie, respectiv butonul de exit, care, la apasare inchide aplicatia.

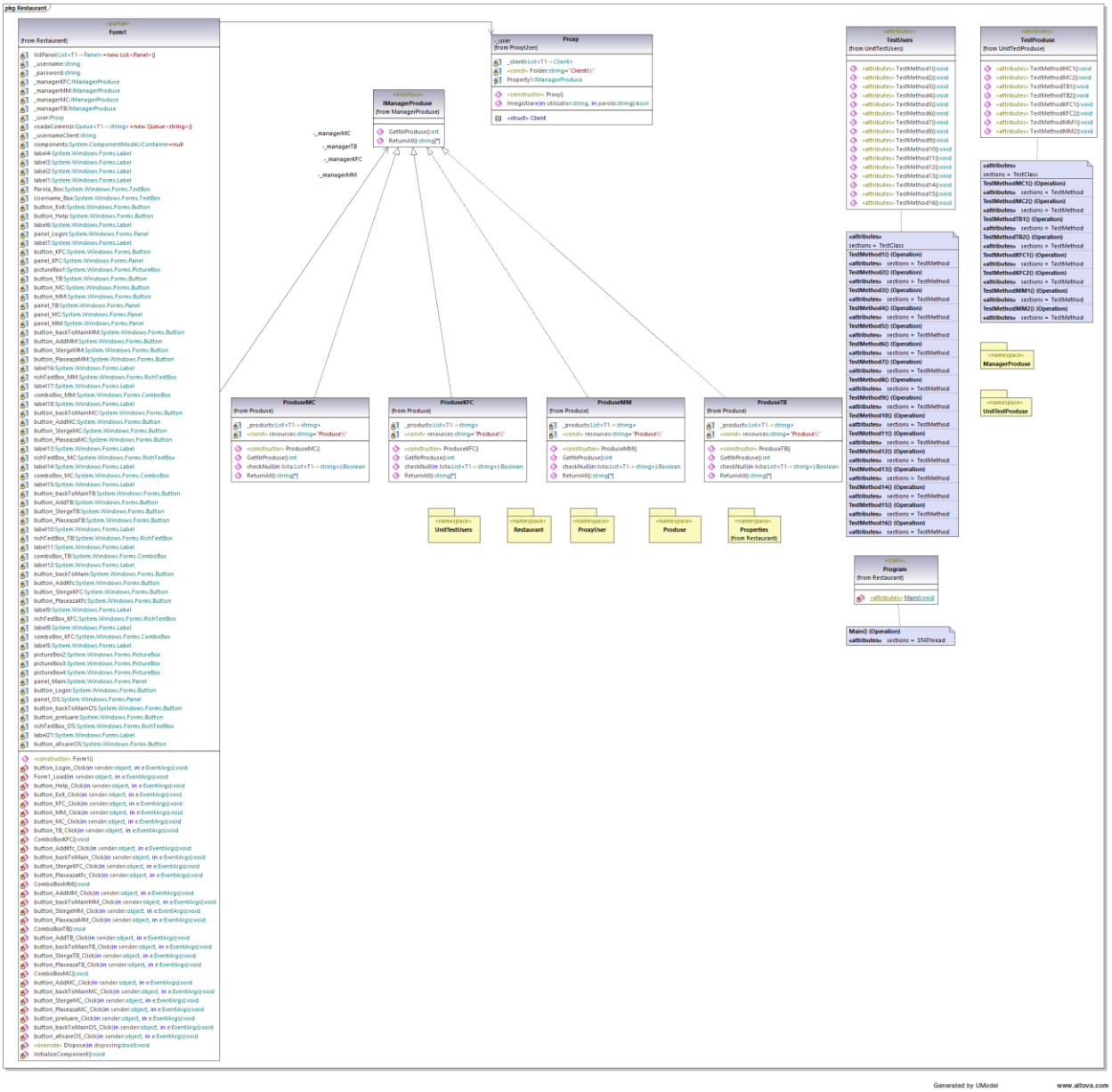
Dupa logarea clientului, va aparea un alt panou din care se va alege restaurantul de la care acesta doreste sa comande. Dupa selectarea restaurantului, indiferent de alegere, se va deschide un alt panou care permite plasarea de comenzi, de la restaurantul respectiv. Dintr-o lista, se vor alege unul sau mai multe produse care vor fi adaugate la comanda, plasate in cazul in care comanda este corecta, sau sterse daca s-a produs o greseala. Pe langa acestea, mai exista si un buton de revenire la pagina principala, care odata apasat, aduce utilizatorul inapoi la panoul initial. De asemenea, exista si un ultim panou valabil pentru curieri, in care acestia au afisata comanda pe care trebuie sa o livreze, si un buton de preluare comenzi.

2) Sablonul de proiectare utilizat

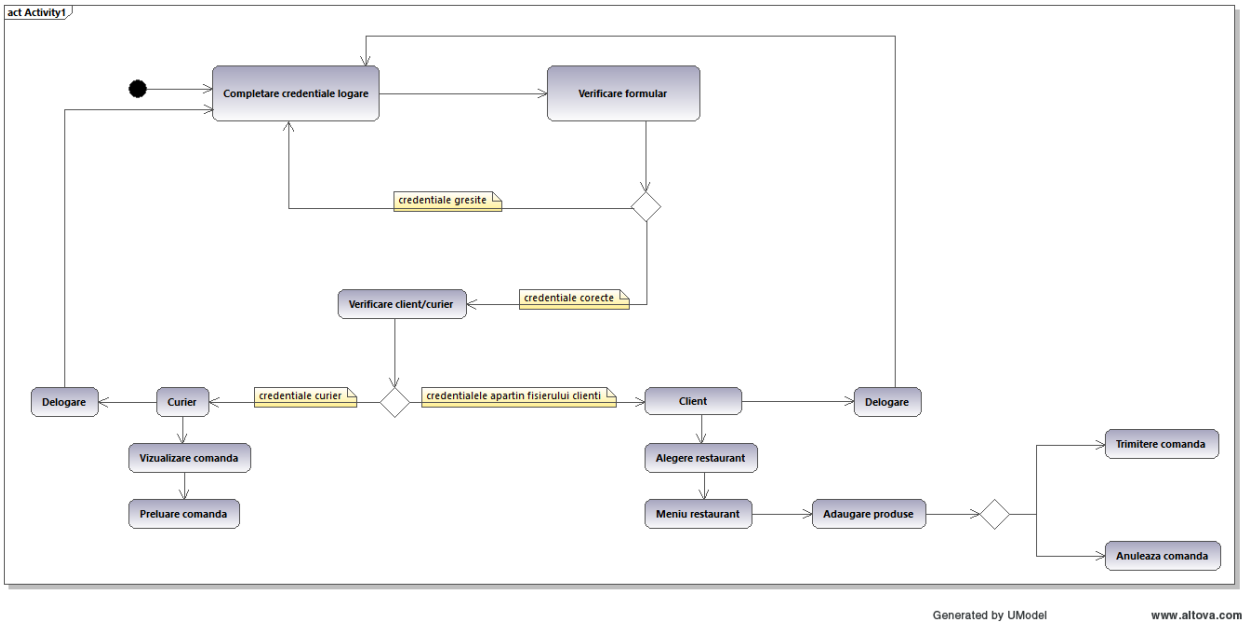
Ca sablon de proiectare am folosit proxy(protection proxy). Este sablonul care permite crearea unui inlocuitor pentru un obiect, inlocuitor care sa controleze accesul la obiectul respectiv**.** Am folosit acest sablon de proiectare, pentru a realiza logarea, astfel controland accesul utilizatorilor(prin username si parola). In proiect, exista un fiser numit “client.txt” care contine numele si parola utilizatorilor. Astfel, in fiserul “Proxy.cs” se verifica daca numele si parola introduse la logare sunt egale cu cele introduse in fiserul “client.txt”. Daca acestea sunt egale, se trece la urmatorul panou, iar in caz contrar, se va afisa un mesaj de eroare la introducerea numelui sau parolei.

3) Diagrame UML

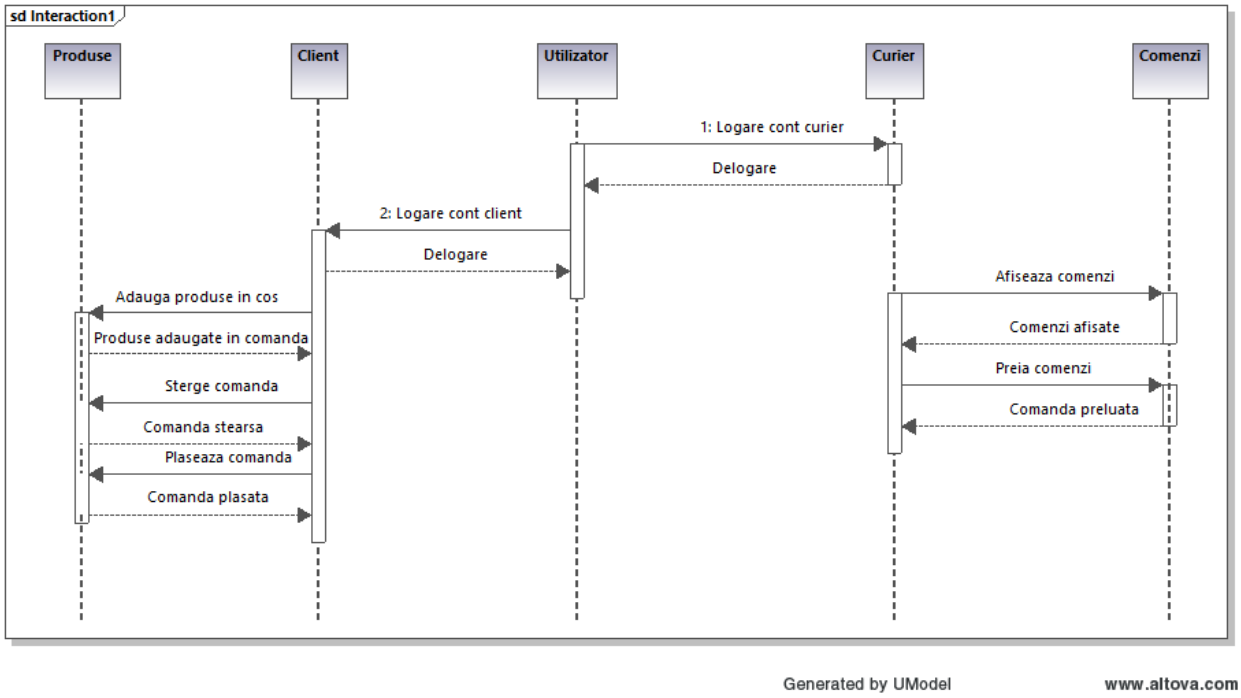
1) Diagrama de clase



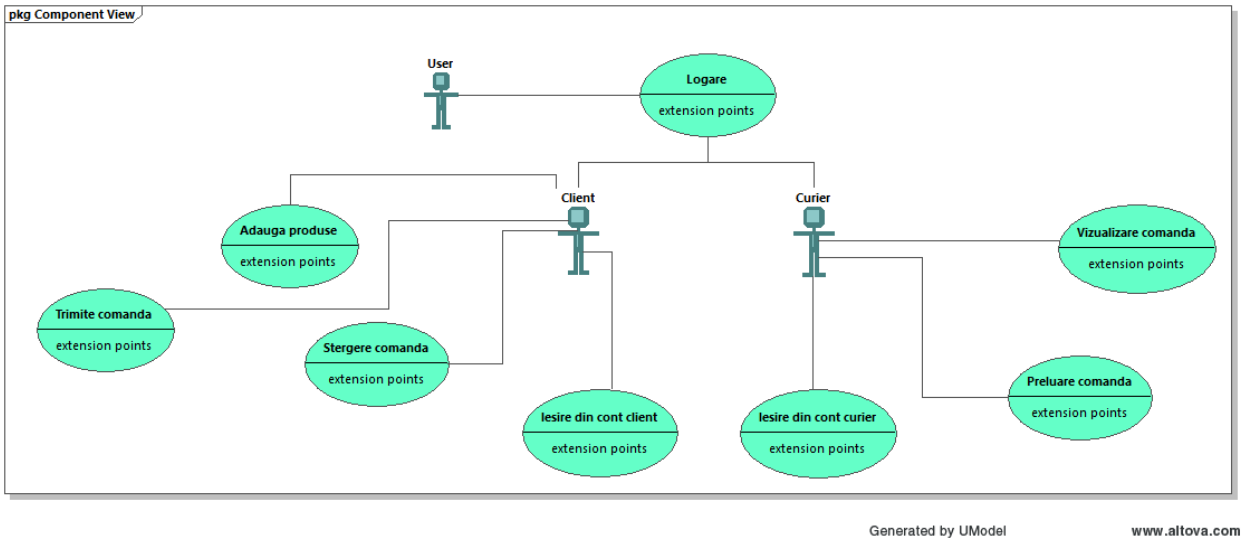
2) Diagrama de activitati



3) Diagrama de secvente



4) Diagrama de cazuri de utilizare

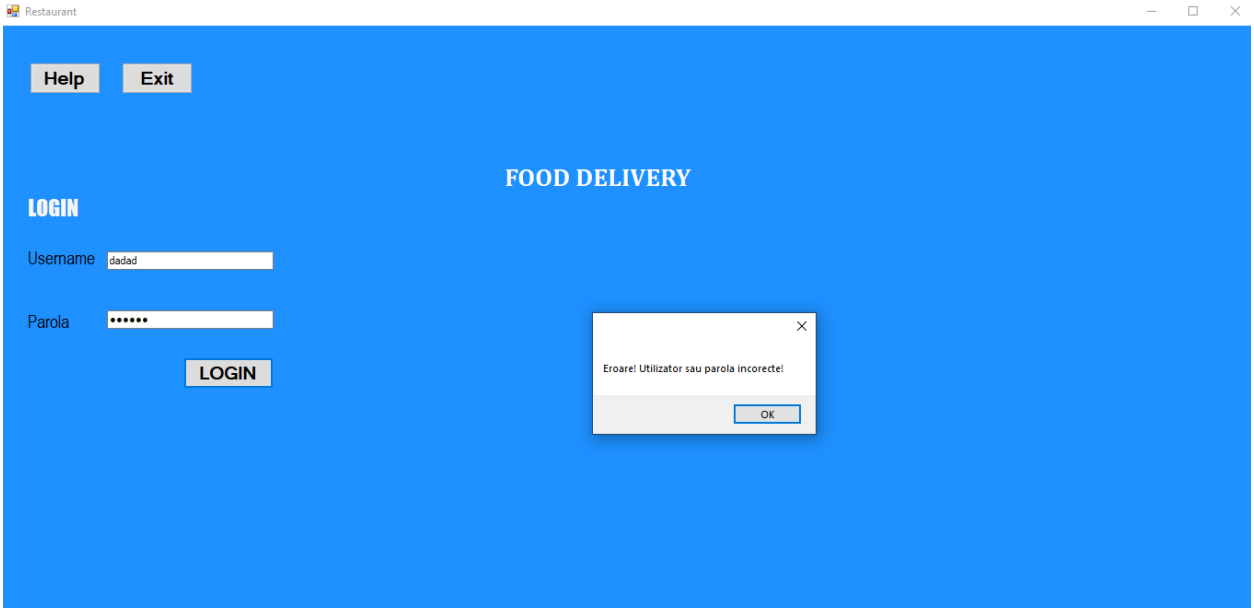


4) Capturi de ecran(cod + program in executie)

Capturi de ecarn la functionarea programului:

1) Login: captura de ecran cu primul panou al interfetei la rularea programului.

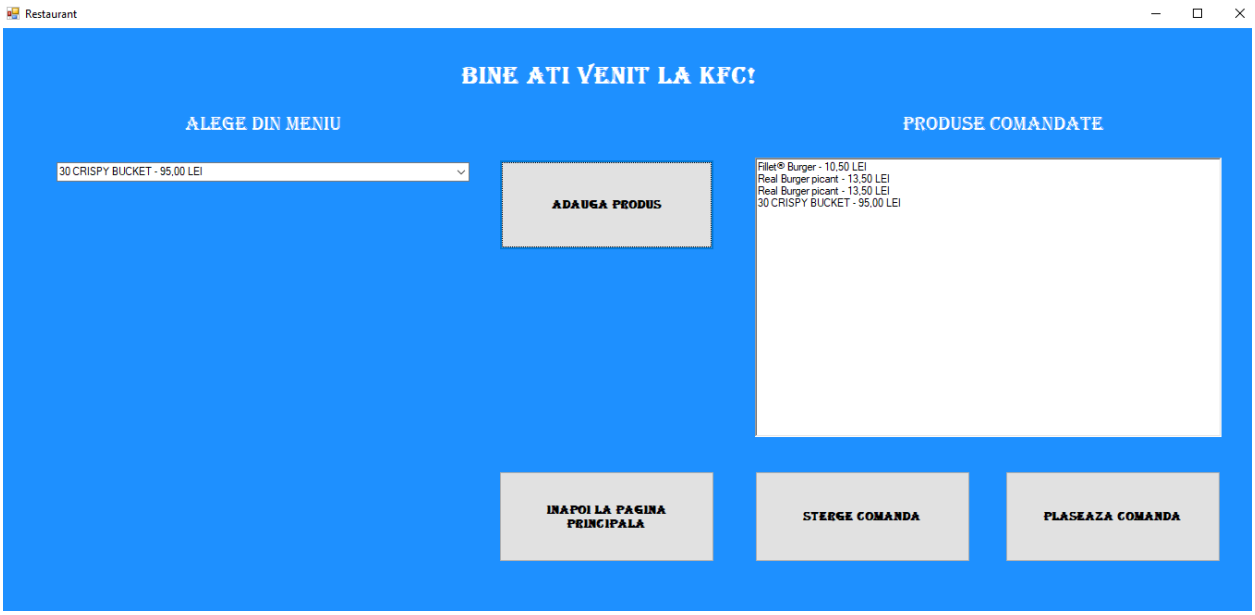
2) Eroare la logare: in cazul in care utilizatorul introduce date gresite pentru username sau parola, se va afisa un mesaj de atentionare.



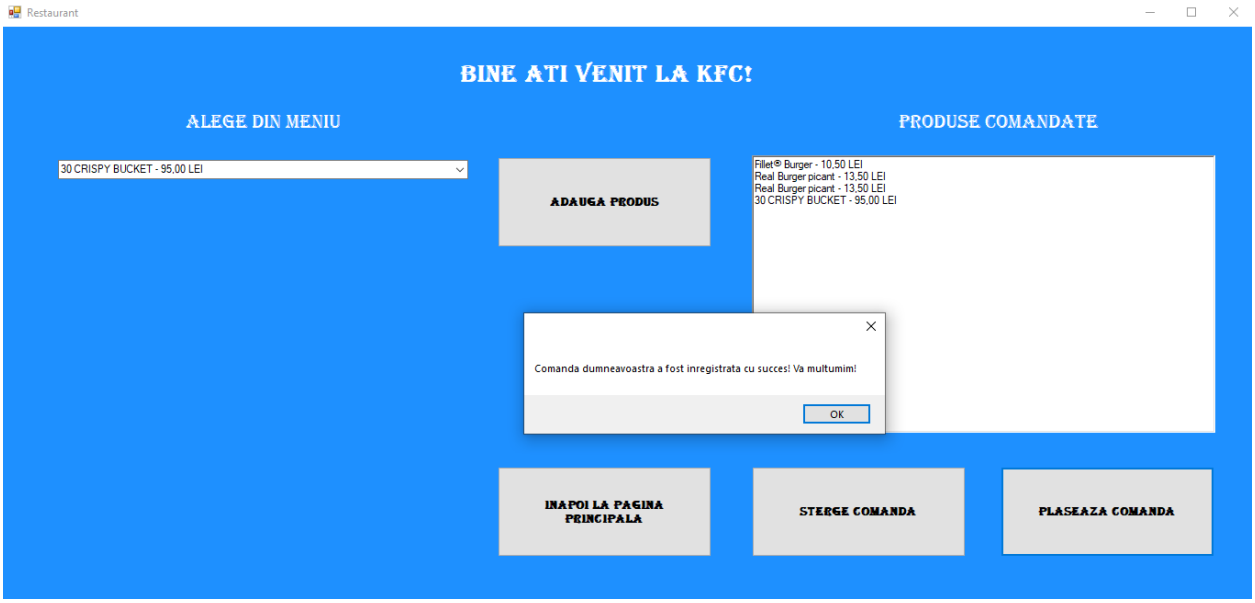
3) Afisarea restaurantelor valabile in aplicatie dupa logare cu succes a unui client.

4) Afisarea produselor si a panoului curent, dupa alegerea unui restaurant la care se va plasa comanda.

5) Afisarea produselor dupa adaugarea lor.

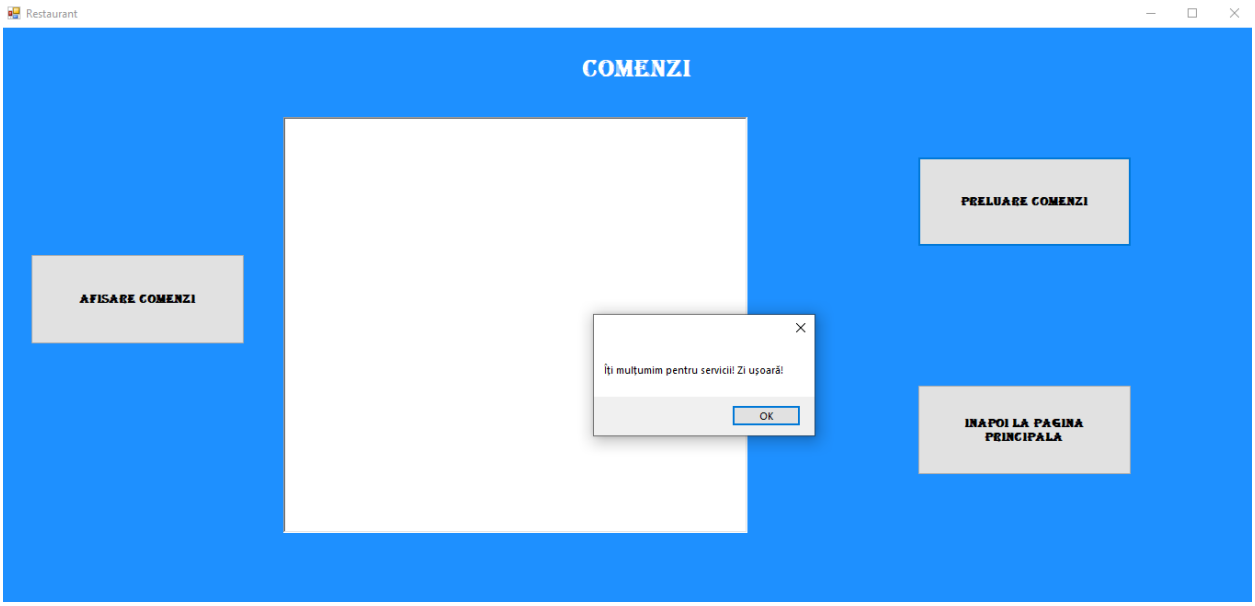


6) Plasarea comenzii dupa adaugare.



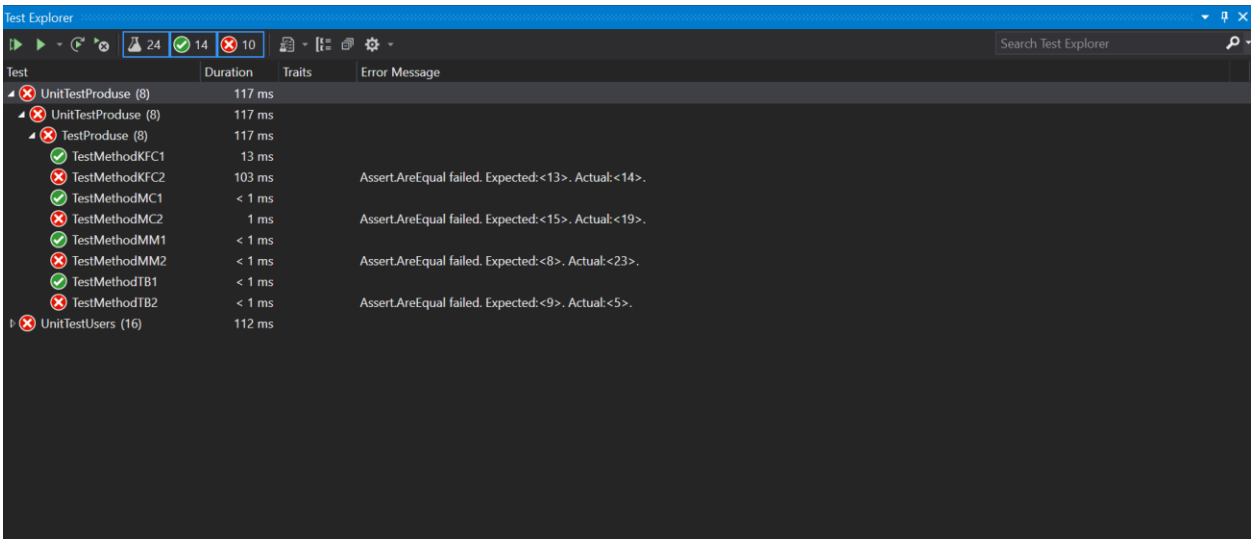
7) Afisarea comenzilor ce trebuiesc livrate de catre curieri, dupa logarea lor ca cu username-ul “curier” si parola “curier”.

8) Preluarea comenzii de catre curier.



5) Testarea unitatilor: se testeaza scenarii pentru user si pentru numarul de produse. De exemplu, se testeaza daca userul introdus este corect sau gresit, sau daca numarul de produse din meniul fiecarui restaurant corespunde cu numarul introdus la test.

1) Testarea produselor



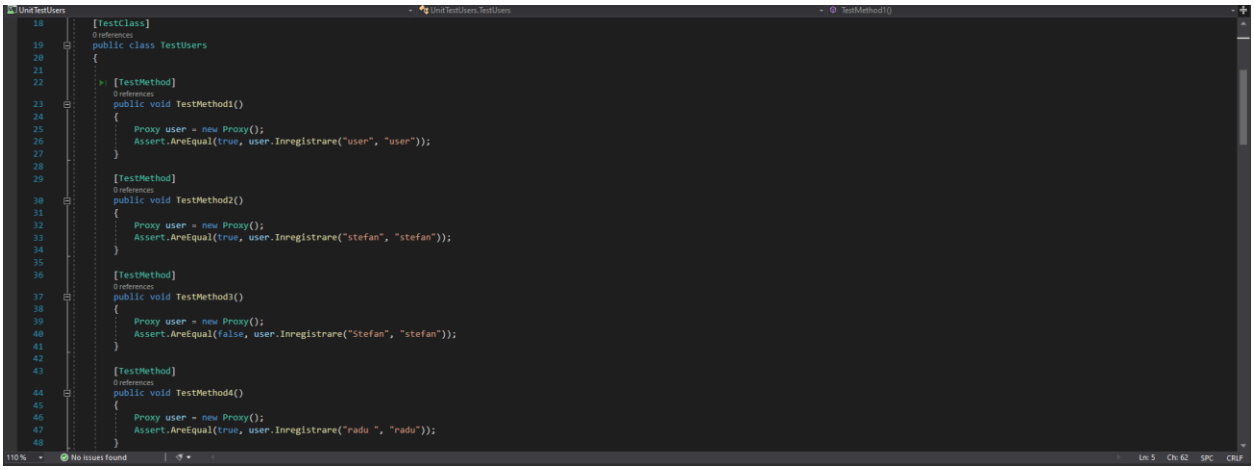
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using System;
using Produse;

namespace UnitTestProduse
{
    [TestClass]
    public class TestProduse
    {

        [TestMethod]
        public void TestMethodMC1()
        {
            ProduseMC produs = new ProduseMC();
            Assert.AreEqual(18, produs.GetNrProduse());
        }

        [TestMethod]
        public void TestMethodMC2()
        {
            ProduseMC produs = new ProduseMC();
            Assert.AreEqual(15, produs.GetNrProduse() + 1);
        }


        [TestMethod]
        public void TestMethodTB1()
        {
            ProduseTB produs = new ProduseTB();
            Assert.AreEqual(10, produs.GetNrProduse());
        }
```

2) Testarea user-ilor

Test Explorer

24 | 14 | 10 — Search Test Explorer

Test	Duration	Traits	Error Message
UnitTestProduse (8)	117 ms		
UnitTestUsers (16)	112 ms		
UnitTestUsers (16)	112 ms		
TestUsers (16)	112 ms		
TestMethod1	111 ms		Assert.AreEqual failed. Expected:<True>. Actual:<False>.
TestMethod10	< 1 ms		
TestMethod11	< 1 ms		
TestMethod12	1 ms		Assert.AreEqual failed. Expected:<False>. Actual:<True>.
TestMethod13	< 1 ms		
TestMethod14	< 1 ms		Assert.AreEqual failed. Expected:<True>. Actual:<False>.
TestMethod15	< 1 ms		
TestMethod16	< 1 ms		Assert.AreEqual failed. Expected:<False>. Actual:<True>.
TestMethod2	< 1 ms		
TestMethod3	< 1 ms		
TestMethod4	< 1 ms		Assert.AreEqual failed. Expected:<True>. Actual:<False>.
TestMethod5	< 1 ms		
TestMethod6	< 1 ms		
TestMethod7	< 1 ms		
TestMethod8	< 1 ms		Assert.AreEqual failed. Expected:<True>. Actual:<False>.
TestMethod9	< 1 ms		



```
[TestClass]
public class TestUsers
{

    [TestMethod]
    public void TestMethod1()
    {
        Proxy user = new Proxy();
        Assert.AreEqual(true, user.Inregistrare("user", "user"));
    }

    [TestMethod]
    public void TestMethod2()
    {
        Proxy user = new Proxy();
        Assert.AreEqual(true, user.Inregistrare("stefan", "stefan"));
    }

    [TestMethod]
    public void TestMethod3()
    {
        Proxy user = new Proxy();
        Assert.AreEqual(false, user.Inregistrare("Stefan", "stefan"));
    }

    [TestMethod]
    public void TestMethod4()
    {
        Proxy user = new Proxy();
        Assert.AreEqual(true, user.Inregistrare("radu ", "radu"));
    }
```

6) Anexa cod aplicatie

- Administrarea produselor unui restaurant:

```
public ProduseTB()
        {
            try
            {
                _products = new List<string>();
                StreamReader sr = new StreamReader("E:\\ELISA\\AN III\\SEM
II\\IP\\proiect_git\\V4\\Restaurant\\produseTB.txt");
                string line;
                line = sr.ReadLine();
                if (line == null)
                    Console.WriteLine("Fisier gol");
                else
                {
                    while (line != null)
                    {
                        _products.Add(line);
                        line = sr.ReadLine();
                    }
                }

            }
            catch (FileNotFoundException e)
            {
                Console.WriteLine(e.ToString());
            }
             }
```

- Metoda pentru returtnarea listei cu numele produselor

```
public string[] ReturnAll()
        {
            string[] lista_produse = new string[_products.Count];
            if (checkNull(_products) == true)
                return null;
            for (int i = 0; i < _products.Count; i++)
                lista_produse[i] = _products[i];

            return lista_produse;


        }
```

- Constructorul clasei Proxy care actualizeaza lista conturilor din fisierul client.txt

```
public Proxy()
        {
            try
            {
                _clienti = new List<Client>();
                StreamReader reader = new StreamReader("clienti.txt");
                string line;
                while ((line = reader.ReadLine()) != null)
                {
                    string[] clienti = line.Split(',');
                    Client user = new Client(clienti[0], clienti[1]);
                    _clienti.Add(user);
                }
                reader.Close();

            }
            catch (Exception exceptie)
            {
                Console.WriteLine(exceptie);
            }
        }
```

- Verificare cazuri de logare(returneaza true sau false)

```
public bool Inregistrare(string utilizator, string parola)
        {
            foreach (Client u in _clienti)
            {
                if (u.Utilizator.Equals(utilizator) &&
u.Parola.Equals(parola))
                {
```

```
                return true;
            }
        }
        return false;
    }
```

- Butonul login si functionalitatea sa

```
private void button_Login_Click(object sender, EventArgs e)
    {
        _username = Username_Box.Text;
        _password = Parola_Box.Text;
        bool ok = _user.Inregistrare(_username, _password);
        if (!ok)
        {
            MessageBox.Show("Eroare! Utilizator sau parola incorecte! ");
            return;
        }

        if (_username.Equals("curier"))
        {
            listPanel[6].BringToFront();
        }
        else
        {
            listPanel[1].BringToFront();
            _usernameClient = _username;
        }

    }
```

- Metoda prin care se revinde la panoul principal

```
private void button_backToMain_Click(object sender, EventArgs e)
    {
        Parola_Box.Text = "";
        Username_Box.Text = "";

        if (richTextBox_KFC.Text.Equals(""))
        {
            listPanel[0].BringToFront();

        }
        else
        {
            MessageBox.Show("Atentie, aveti produse in cos !");
            return;
        }
    }
```

7) Referinte

https://refactoring.guru/design-patterns/proxy/csharp/example

http://florinleon.byethost24.com/lab_ip.html

https://www.altova.com/umodel