ServoResto

PROIECT INFORMATICA INDUSTRIALA

|  |  |  |
| --- | --- | --- |
| Team leader:Lenghel Oana-Florina |  | Grupa:30135 |
| Developer:Chisăliță Denis-Marian |  | Grupa:30135 |
| Developer:Boca Tudor-Bogdan |  | Grupa:30135 |
| Tester:Ilea Raluca-Livia |  | Grupa:30135 |

**SINTEZA**

proiectului cu titlul:

«ServoResto»

* Cerinţele temei:

Crearea unei aplicații complexe, având o proiectare multinivel, cu nivelul de date în format .xml sau bază de date, cu nivelul logic implementat fie în interiorul programului, fie ca serviciu web și cu prezentare în format ASP.net sau Windows Form.

* Soluţii alese:

Implemetarea unei aplicații cu 3 nivele de proiectare:nivelul de date în format bază de date,nivelul logic implemetat ca serviciu web, iar prezentarea în format Windows Form.

* Rezultate obţinute:

Ca rezultate am obținut o aplicație a unui restaurant pentru comandă online cu o interfață intuitivă, ușor de utilizat și cu un aspect plăcut pentu un posibil client.

* Testări şi verificări:

Atât pe parcursul, cât și la finalul dezvoltării aplicației s-a efectuat un set de teste, pe baza anumitor scenarii uzuale. În urma acestor teste s-au realizat modificările aferente, ca mai apoi, la o ultimă verificare aplicația să treacă cu brio peste orice scenariu.

Cuprins

[Capitolul 1 INTRODUCERE 2](#_Toc41396866)

[1.1 Enuntul problemei 2](#_Toc41396867)

[1.2 Obiective 2](#_Toc41396868)

[1.3 Specificații 2](#_Toc41396869)

[Capitolul 2 ARHITECTURA APLICATIEI 3](#_Toc41396870)

[2.1. Diagrama claselor 3](#_Toc41396871)

[2.2. Diagrama use-case 3](#_Toc41396872)

[2.3. Diagrama bazei de date 4](#_Toc41396873)

[Capitolul 3 IMPLEMETAREA APLICATIEI 6](#_Toc41396874)

[Capitolul 4 TESTAREA APLICATIEI 12](#_Toc41396875)

[Anexa 1 (cod sursa) 13](#_Toc41396876)

# Capitolul 1 INTRODUCERE

## Enuntul problemei

Crearea unei aplicații complexe, având o proiectare pe mai multe niveluri.Primul nivel, cel al datelor să fie implemetat în format .xml sau bază de date, nivelul logic având o implemetare în interiorul programului sau ca serviciu web, iar ultimul nivel, cel de prezentare în format ASP.net sau Windows Form.Astfel, am creat aplicația ServoResto ce face posibilă comanda online a produselor și a serviciilor oferite de restaurant.

## Obiective

* Crearea datelor în formatul unei baze de date
* Implemetarea ca serviciu web
* Prezentarea în format Windows Form

## Specificații

Această aplicație este un Windows Form Application, scris în C#, cu o proiectare pe 3 niveluri, ce permite efectuarea unor operații tipice pe o bază de date relaționare(adăugare, ștergere).Nivelurile pe care este dezvoltat programul sunt următoarele:

* Primul nivel: datele sunt stocare într-o bază de date relaționale
* Al doilea nivel:logica programului este implemetată ca serviciu web
* Al treilea nivel:prezentarea aplicației este sub forma de Windows Form

Aplicația are 2 tipuri de utilizatori:utilizatorul normal, clientul și administratorul.Cel din urmă are o secțiune specială, unde poate intra doar el pentru a manageria produsele.Această secțiune nu poate fi accesată de către client.

# Capitolul 2 ARHITECTURA APLICATIEI

## 2.1. Diagrama claselor

A close up of text on a white background

Description automatically generated

Figura 1.Diagrama claselor

## 2.2. Diagrama use-case

În aplicația dezvoltată există două tipuri de utilizatori:managerul(cu acces complet la baza de date) și clientul(cu acces limitat).La utilizarea aplicației, utilizatorul va avea nevoie de autentificare pentru a putea face o comandă.Pentru autentificare orice utilizator trebuie mai întâi să se înregistreze, creând astfel un cont de client.Datele de autentificare pentru manager sunt deja introduse în partea de implemtare a aplicației, astfel orice nou utilizator, va putea crea un cont de client, nicidecum un cont de manager.

Clienții își pot crea un cont, unde au memorate datele personale, adresa de livrare, numărul de telefon.De asemenea pot adăuga/șterge produse din coș, pe care mai apoi le pot plasa într-o comadă.Totodată, își pot revedea/recomanda comenzile mai vechi.În cazul în care doresc să comande la o altă adresă, o pot schimba și pot să își urmărească comanda plasată, pentru care au primit confirmare.

Managerul poate face tot ce face un client, dar poate și manageria produsele.Astfel, din panoul de Management acesta poate adăuga un produs nou, sau poate șterge un anumit produs.

A close up of a map

Description automatically generated

Figura 2.Diagrama use-case

## 2.3. Diagrama bazei de date

Baza de date relațională utilizată la dezvoltarea acestei aplicații folosește patru tabele, dintre care un tabel de legatură.Fiecare tabel are cheie priamră care este cheie străină în alt tabel, compunând astfel relații one-to-one sau one-to-many.Tabelul de legătură este compus doar din chei străine, fiind un tabel în care se face legătura din două tabele.

Tabelul ‘Clients’ conține toate datele de autentificare ale utilizatorilor.În plus, are un câmp “isAdmin” pentru a specifica dacă utilizatorul are funcția de manager al aplicației și poate avea acces la panoul aferent, sau nu.

Tabelul ’Products’ conține datele relevante ale produselor existente.Acest tabel are un câmp special “Image” , reprezentând imaginea produsului respectiv, stocată ca șir de biți.

Tabelul ‘Orders’ conține datele aferente unei comenzi:număr de comandă,data la care a fost făcută,id-ul clientului care a făcut comanda respectivă, dar și un câmp unde sunt introduse anumite informații/specificații ale clientului.Între acest tabel și cel care conține clienții există o relație one-to-many, deoarece un client poate face mai multe comenzi.

Tabelul ‘CartItems’, folosit ca tabel de legătură, face conexiunea între comandă și produsele din cadrul comenzii.

A close up of text on a white background

Description automatically generated

Figure 3.Diagrama bazei de date

# Capitolul 3 IMPLEMETAREA APLICATIEI

Primul pas în dezvoltarea aplicației a fost crearea bazei de date în Microsoft Visual Studio. După crearea fiecărui tabel au fost setate cheile primare, apoi cheile străine, făcându-se astfel conexiunile aferente.

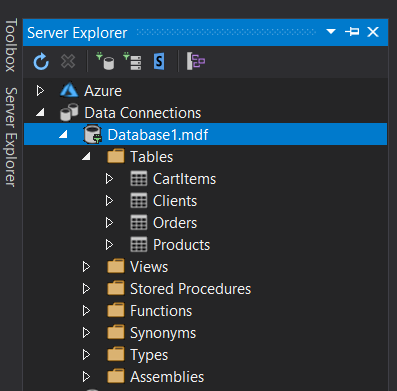


Figure 4.Conectarea la baza de date

Pentru început au fost create funcționalitățile aplicației în cadrul unui Web Service, unde a fost realizată și conexiunea cu baza de date. Astfel, au fost implementate următoarele metode:

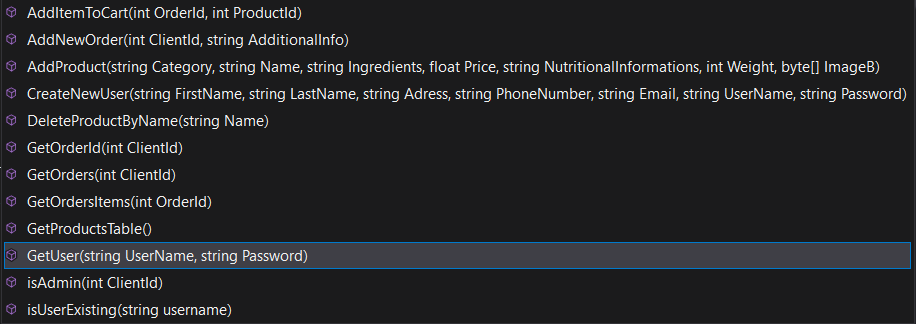


Figure 5.Funcționalități aplicație

După crearea celor de mai sus, s-a trecut la crearea interfeței aplicației. Pentru crearea acesteia am folosit formatul Windows Form, iar pentru fiecare meniu a fost creat câte un User Control, respectiv un Windows Forms.

Meniul “Home” este cel care apare prima dată la rularea aplicației. Acesta a fost creat cu ajutorul unui User Control și conține logo-ul aplicației, dar și o mică descriere.

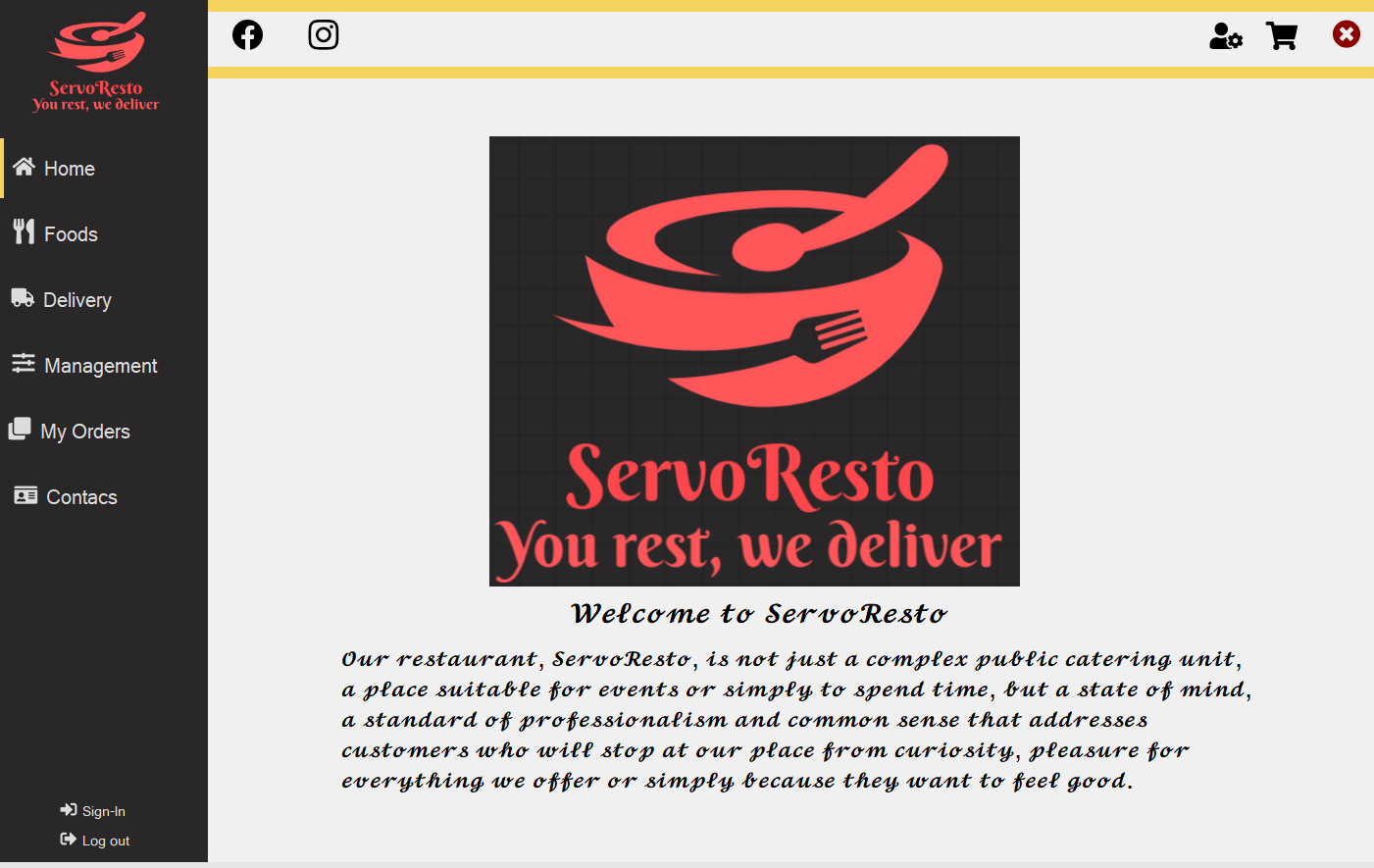


Figure 6.Meniul 'Home'

Meniul ”Foods” reprezintă partea principală a acestei aplicații, iar de aceea a fost creat în Windows Form-ul principal.Pentru adăugarea și afișarea produselor a fost folosit un Flow LayoutPanel, ce conține o listă de template-uri ale unui produs făcut într-un User Control separate, numit ProductItem.

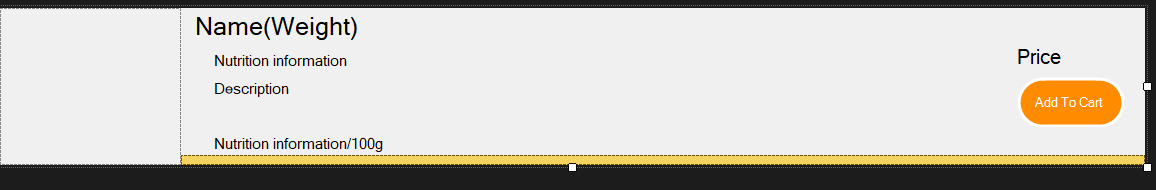


Figure 7.ProductItem

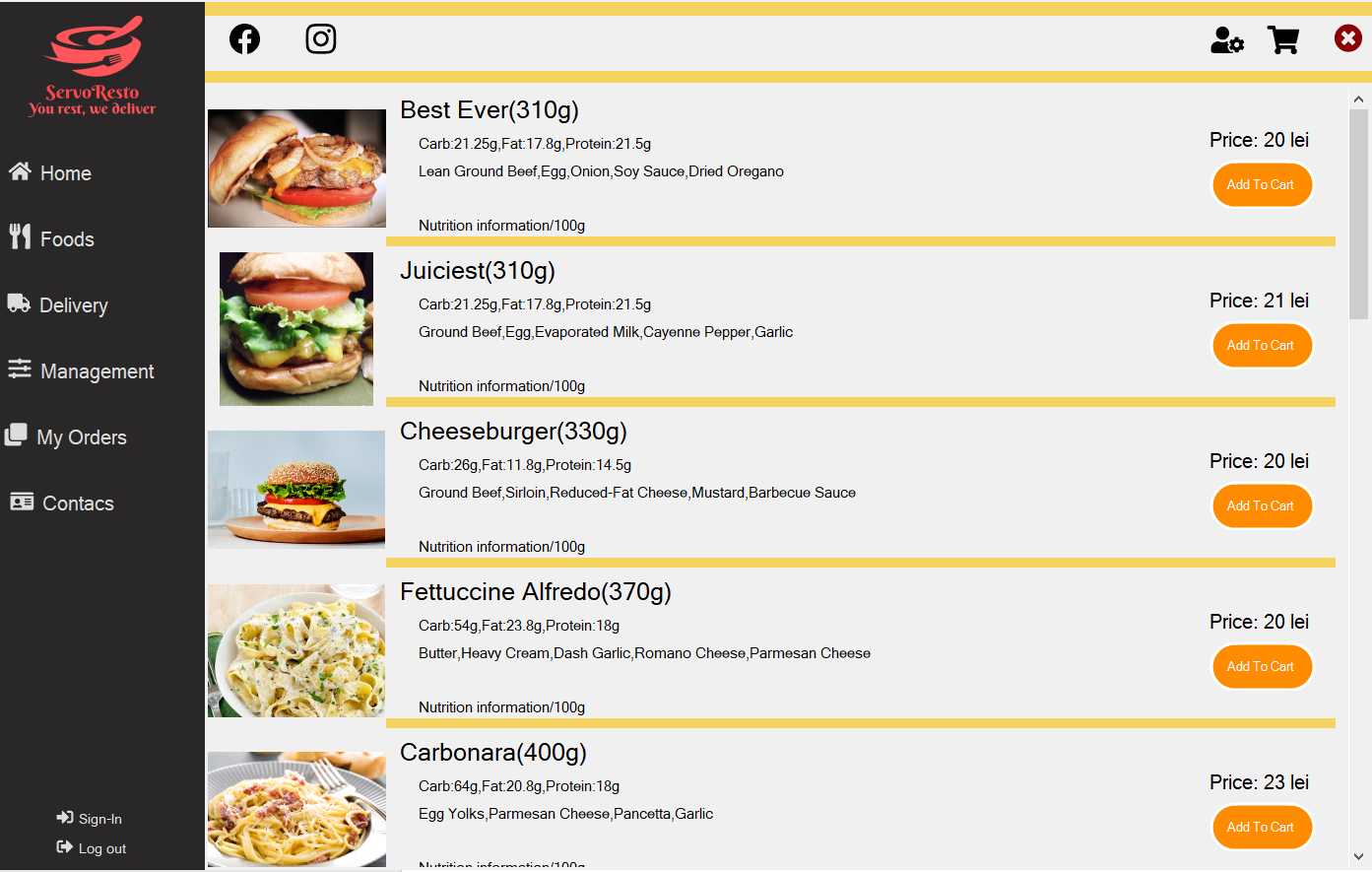


Figure 8.Meniul 'Foods'

Meniul “Delivery” a fost creat folosind User Control și are ca scop salvarea unei adrese temporare pentru o comandă curentă.

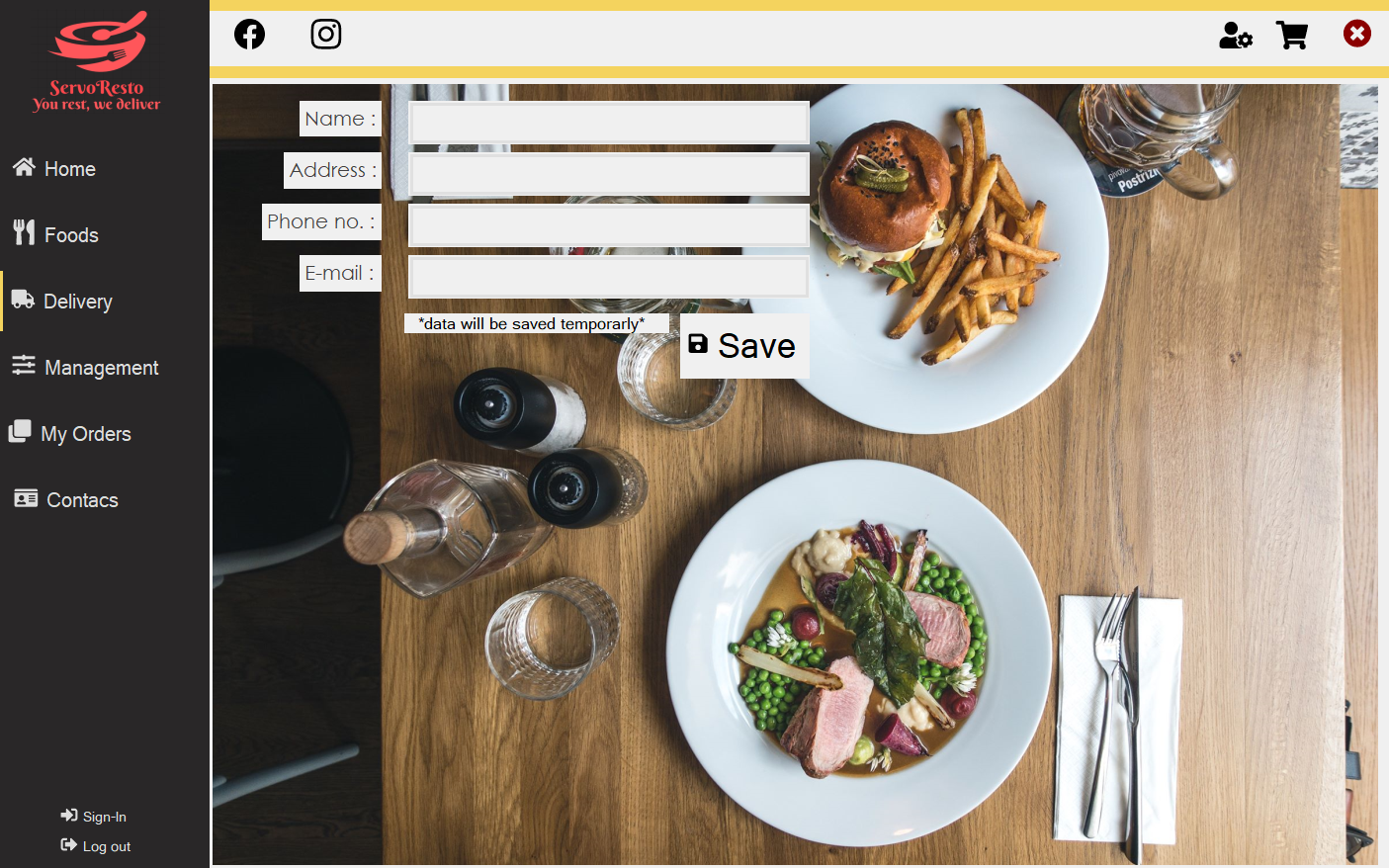


Figure 9.Meniul 'Delivery'

Meniul “Management” este creat la fel ca celelalte și are ca scop adăugarea/ștergerea unor produse din meniul Foods.Acesta poate fi accesat doar dacă ești logat cu un cont de admin, fiind inaccesibil pentru un client.

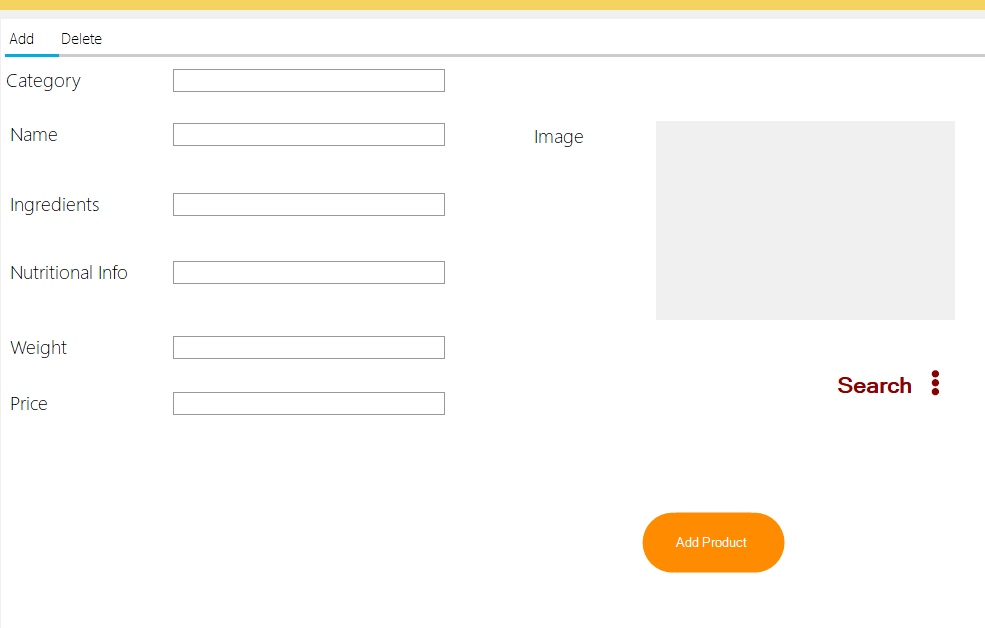


Figure 10.Add pentru meniul 'Management'

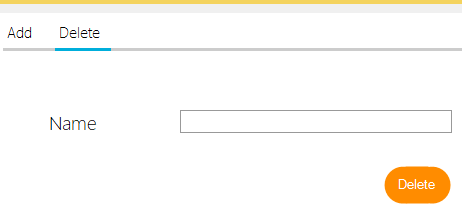


Figure 11.Delete pentru meniul 'Management'

Meniul ”My Orders” conține datele aferente unei comenzi ulterior plasate.Acest meniu poate fi accesat doar dacă clientul este logat în aplicație.De asemenea, clientul are posibilitatea de recomandare a unei comenzi mai vechi.La fel ca celelate meniuri si acesta este creat cu User Control.

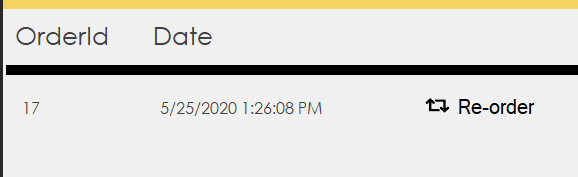


Figure 12.Meniul 'My Orders'

Meniul ”Contacts” ar trebui să conțină datele de contact ale restaurantului, dar în lipsa unor date reale, au fost adăugate datele de contact ale administratorilor aplicației.

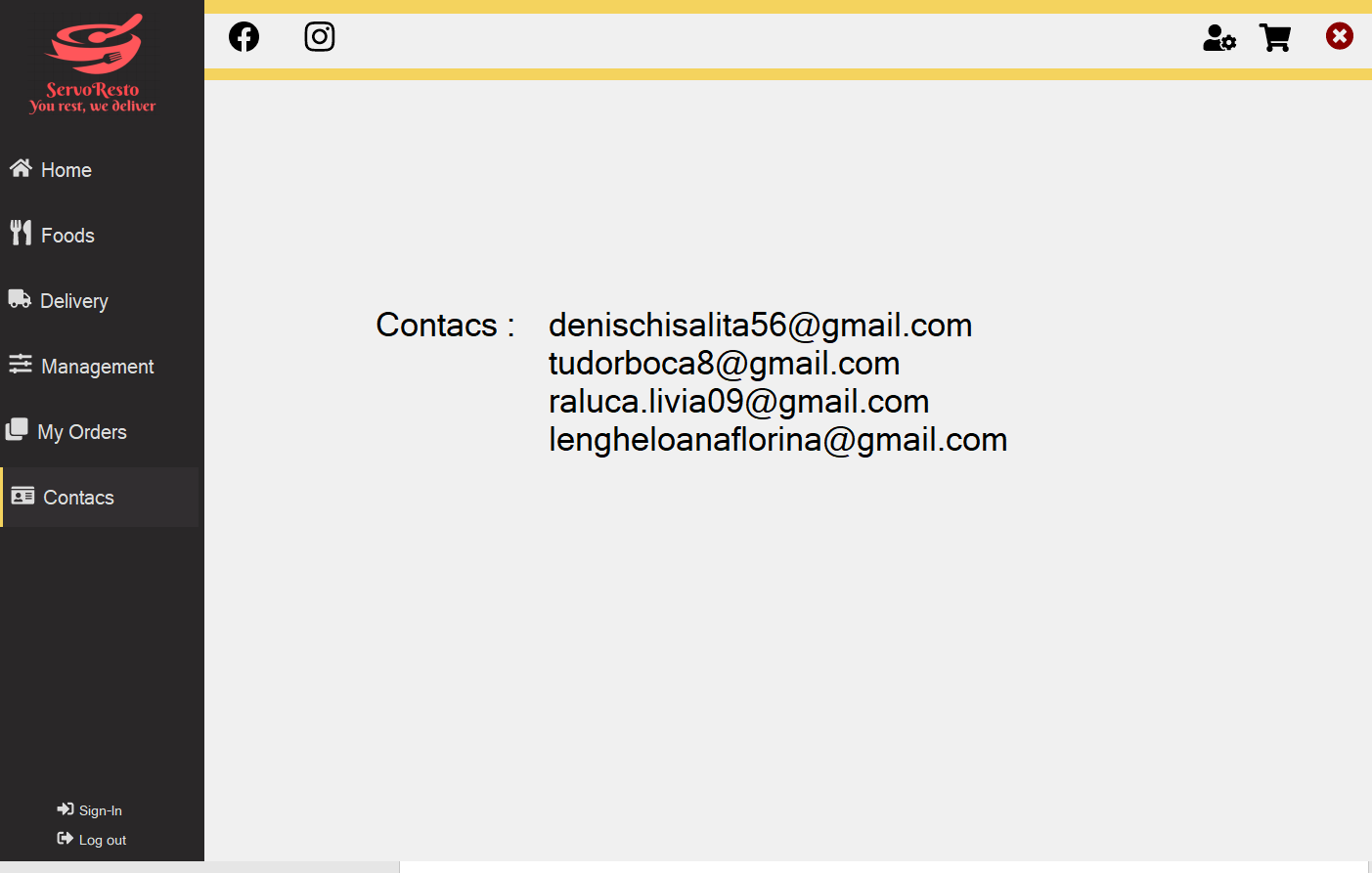


Figure 13.Meniul 'Contacts'

Pentru partea de Log-in a fost folosit un Windows Form, ce conține câmpuri necesare logării cu un cont, iar în cazul neexistenței acestuia se poate crea unul nou.Creacrea unui cont se realizează din meniul Register.

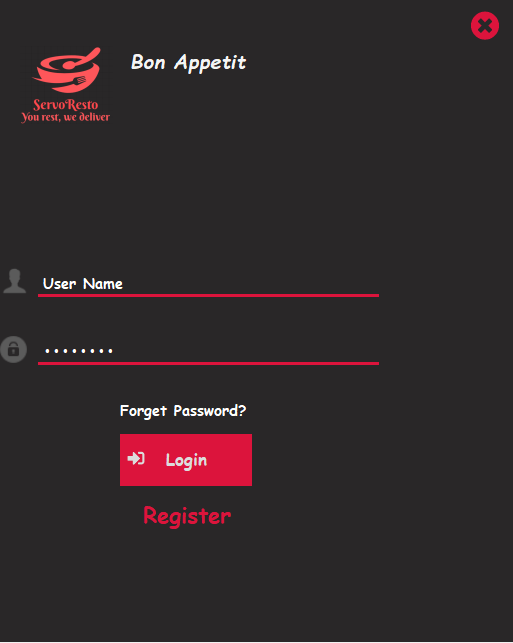


Figure 14.Meniul 'Log-in'

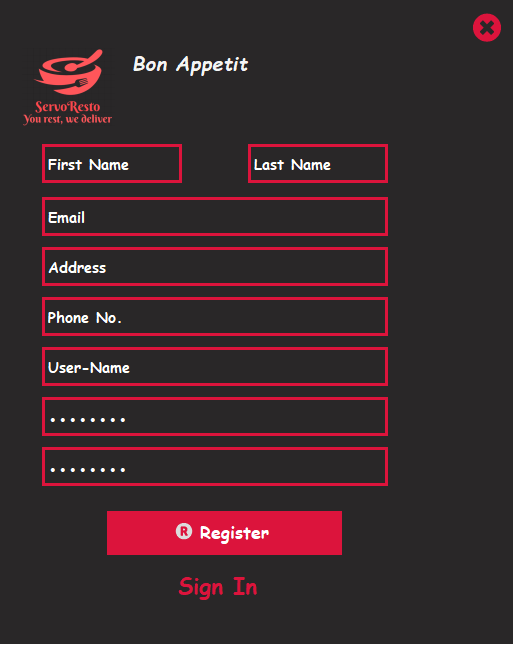


Figure 15.Register

Butonul de Log out a fost creat prin intermediul unui controler de tip button.La apăsarea acestuia, în cazul, în care clientul este logat, realizează delogarea cu succes, iar în caz contrat sugerează logarea.

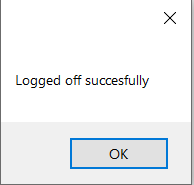


Figure 16.Mesaj 'Log-out' la conectarea cu cont

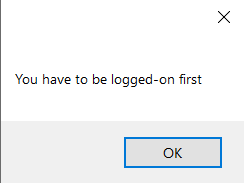


Figure 17.Mesaj 'Log-out' la conectarea fără cont

Meniul “Cart” reprezintă coșul propriu-zis, unde clientul are posibilitatea de a vizualiza produsele adăugate și poate plasa comanda.

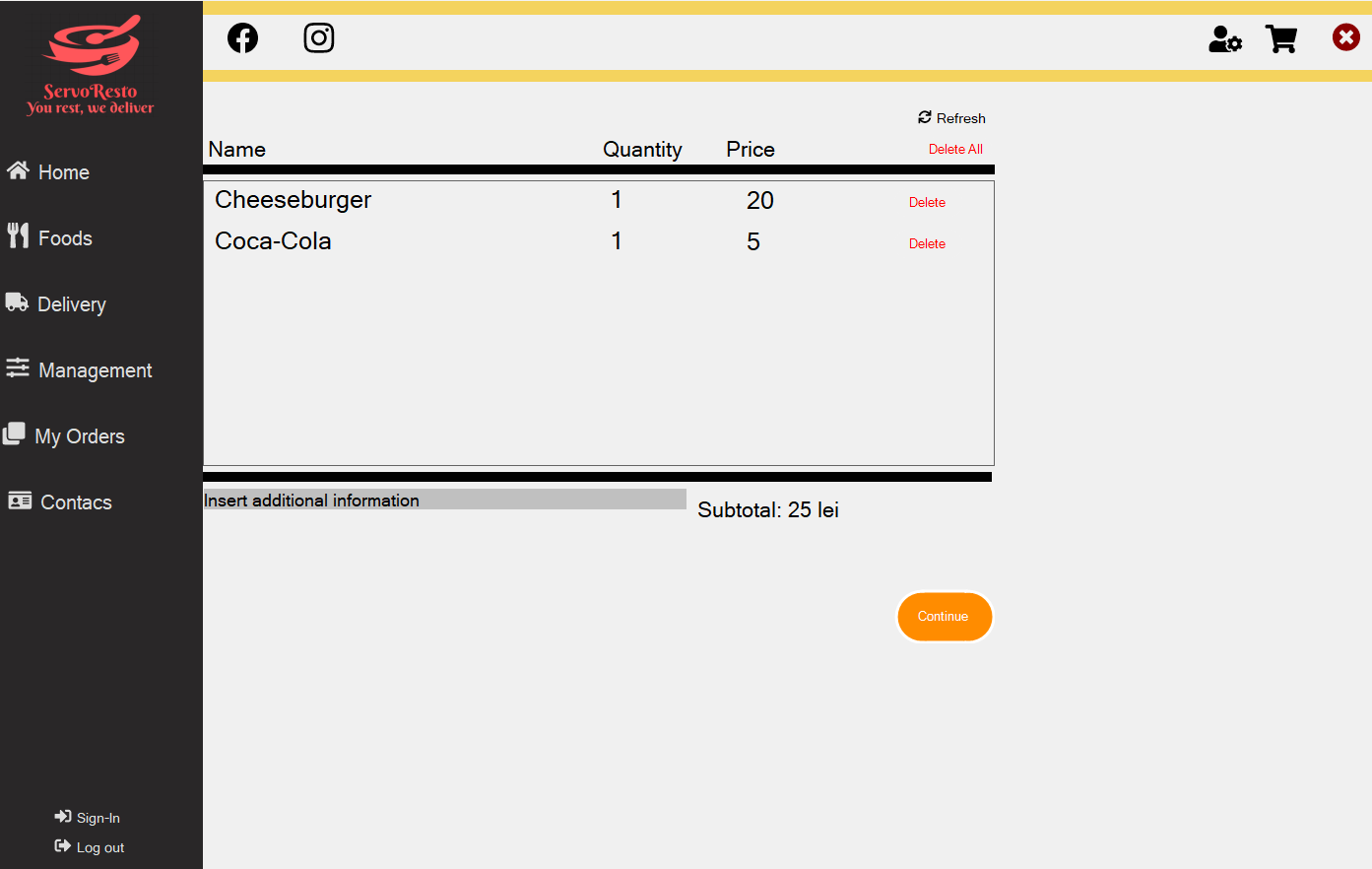


Figure 18.Meniu 'Cart'

# Capitolul 4 TESTAREA APLICATIEI

Testarea aplicației s-a efectuat în două etape: pe parcursul dezvoltării aplicației și după încheierea dezvoltării aplicației.În prima etapă a testării, scopul a fost să se ofere feedback dezvoltatorilor cu privire la anumite comportamente specifice, aștepându-se ulterioare modificări.A doua etapă a fost efectuată pentru a identifica erorile apărute în forma finală a aplicației.

Pentru forma finală a aplicației au fost analizate mai multe scenarii, având urmatoarele rezultate:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nr.crt | Scenariu | Rezultat așteptat | Rezultat primit | Rezultat test |
| 1. | Acces la serviciu web introducând adresa: localhost:60655/WebService1.asmx | Acees interzis | “This site can’t be reached” | Pass |
| 2. | Creearea unui cont fără a introduce date | Mesaj de eroare | "Account created succesfully | Fail |
| 3. | Comanadă fără cont | Mesaj de eroare | “You have to Log-in to place an order” | Pass |
| 4. | Intrare în partea de managemet cu un cont de client/fără cont | Mesaj de eroare | "Access denied.This account doesn't have the right to access this field"/"Log in with administrator account" | Pass |
| 5. | Ștergerea unui element din coș | Ștergerea elementului, dar și recalcularea sumei | Suma de plata rămâne aceeași | Fail |
| 6. | Ștergerea unui produs din meniu de către manager | Ștergerea produsului din meniu, dar și din baza de date | Eroare la rularea codului | Fail |

După realizarea testelor finale, dezvoltatorii au fost informați asupra scenariilor ale căror teste au căzut.Ulterior aceștia au rezolvat erorile apărute, iar produsul final este capabil să treacă orice test.

# Anexa 1 (cod sursa)

private void populateProducs()

{

flowLayoutPanel1.Controls.Clear();

DataTable productsTable = getWebService().GetProductsTable();

productItems = (from DataRow row in productsTable.Rows

select new ProductItem

{

ProductId = Convert.ToInt32(row["ProductId"]),

Name = row["Name"].ToString(),

Ingredients = row["Ingredients"].ToString(),

Price = (float)Convert.ToDouble(row["Price"]),

NutritionalInformations = row["NutritionalInformations"].ToString(),

Weight = Convert.ToInt32(row["Weight"]),

Imagec = (byte[])(row["Image"])

}).ToList();

foreach (ProductItem productItem in productItems)

{

flowLayoutPanel1.Controls.Add(productItem);

}

}

public void unlist()

{

flowLayoutPanel1.Visible = false;

deliveryAdd1.Visible = false;

homeUser1.Visible = false;

manageUser1.Visible = false;

contacsUser1.Visible = false;

cartControl1.Visible = false;

orders1.Visible = false;

}

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

{

CartItems cartItem = new CartItems();

cartItem.Name = this.Name;

cartItem.Quantity = "1";

cartItem.Price = this.Price;

cartItem.ProductId = this.ProductId;

cartItem.OrderId = Form1.orderId;

CartControl.cartList.Add(cartItem);

}

public void PopulateOrders()

{

ServiceReference1.WebService1SoapClient webService = Form1.getWebService();

flowLayoutPanel1.Controls.Clear();

DataTable orders = webService.GetOrders(Form1.user.ClientId);

OrderItems = (from DataRow row in orders.Rows

select new OrderItem

{

OrderId = Convert.ToInt32(row["OrderId"]),

Date = row["Date"].ToString()

}).ToList();

foreach (OrderItem orderItem in OrderItems)

{

flowLayoutPanel1.Controls.Add(orderItem);

}

}

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

{

ServiceReference1.WebService1SoapClient webService = Form1.getWebService();

DataTable productsTable = webService.GetOrdersItems(OrderId);

productItems = (from DataRow row in productsTable.Rows

select new ProductItem

{

ProductId = Convert.ToInt32(row["ProductId"]),

Name = row["Name"].ToString(),

Ingredients = row["Ingredients"].ToString(),

Price = (float)Convert.ToDouble(row["Price"]),

NutritionalInformations = row["NutritionalInformations"].ToString(),

Weight = Convert.ToInt32(row["Weight"]),

Imagec = (byte[])(row["Image"])

}).ToList();

foreach(ProductItem productItem in productItems)

{

CartItems cartItem = new CartItems();

cartItem.Name = productItem.Name;

cartItem.Quantity = "1";

cartItem.Price = productItem.Price;

cartItem.ProductId = productItem.ProductId;

cartItem.OrderId = Form1.orderId;

CartControl.cartList.Add(cartItem);

}

}

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

{

if (categoryAdd.Text != "")

{

if (nameAdd.Text != "")

{

if (ingredientsAdd.Text != "")

{

if (nutriAdd.Text != "")

{

if (weightAdd.Text != "")

{

if (priceAdd.Text != "")

{

if (imgAdd!=null)

{

byte[] images = ConvertImageToBinary(imgAdd.Image);

Form1.getWebService().AddProduct(categoryAdd.Text, nameAdd.Text, ingredientsAdd.Text, float.Parse(priceAdd.Text), nutriAdd.Text, Convert.ToInt32(weightAdd.Text), images);

categoryAdd.Text = "";

imgAdd.ImageLocation = "";

nameAdd.Text = "";

ingredientsAdd.Text = "";

priceAdd.Text = "";

nutriAdd.Text = "";

weightAdd.Text = "";

imLocation = "";

MessageBox.Show("Product added succesfully");

}

else

MessageBox.Show("Insert Image From Search Button");

}

else

MessageBox.Show("Insert Price");

}

else

MessageBox.Show("Insert Weight");

}

else

MessageBox.Show("Insert Nutritional Information");

}

else

MessageBox.Show("Insert Ingredients");

}

else

MessageBox.Show("Insert Name");

}

else

MessageBox.Show("Insert Category");

}

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

{

if (string.IsNullOrEmpty(nameDel.Text))

{

Form1.getWebService().DeleteProductByName(nameDel.Text);

nameDel.Text = "";

MessageBox.Show("Product(s) deleted succesfully");

}

else

MessageBox.Show("Insert a name");

}

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

{

if(string.IsNullOrWhiteSpace(lastName.Text)||string.IsNullOrWhiteSpace(firstName.Text)|| string.IsNullOrWhiteSpace(email.Text)|| string.IsNullOrWhiteSpace(address.Text)||string.IsNullOrWhiteSpace(phone.Text)|| string.IsNullOrWhiteSpace(userName.Text)|| string.IsNullOrWhiteSpace(passw.Text) || string.IsNullOrWhiteSpace(repass.Text))

{

MessageBox.Show("Enter a valid set of data");

}

else

{

webService = Form1.getWebService();

if (!webService.isUserExisting(userName.Text)) {

if (passw.Text != repass.Text)

MessageBox.Show("Re-enter same password");

else

{ webService.CreateNewUser(firstName.Text, lastName.Text, address.Text, phone.Text, email.Text, userName.Text, passw.Text);

MessageBox.Show("Account created succesfully");

lastName.Text = "Last Name";

firstName.Text = "First Name";

email.Text = "Email";

address.Text = "Address";

phone.Text = "Phone No.";

userName.Text = "User-Name";

passw.Text = "Password";

repass.Text = "Password";

}

}

else MessageBox.Show("User-name already exists");

}

}

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

{

//sign

if (userSign.Text != "" || pass.Text != "")

{

DataTable user = Form1.getWebService().GetUser(userSign.Text,pass.Text);

if (user != null)

{

List<UserClass> users = new List<UserClass>();

users = (from DataRow row in user.Rows

select new UserClass

{

ClientId = Convert.ToInt32(row["ClientId"]),

FirstName = row["FirstName"].ToString(),

LastName = row["LastName"].ToString(),

Address = row["Adress"].ToString(),

PhoneNumber = row["PhoneNumber"].ToString(),

Email = row["Email"].ToString(),

UserName = row["UserName"].ToString()

}).ToList();

if (users.Count > 0)

{

Form1.user = users[0];

MessageBox.Show("Log-in succesfully");

}

else

MessageBox.Show("Not found this specific User/Password");

}

else

MessageBox.Show("Not found this specific User/Password");

}

else

MessageBox.Show("Insert user-name and password");

}

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

{

if (nameC.Text != "")

{

if (addressC.Text!="")

{

if (phoneC.Text != "")

{

if (emailC.Text != "")

{

this.Name = nameC.Text;

this.Address = addressC.Text;

this.PhoneNr = phoneC.Text;

this.Email = emailC.Text;

nameC.Text = "";

addressC.Text = "";

phoneC.Text = "";

emailC.Text = "";

Form1.user.Address = this.Address;

Form1.user.PhoneNumber = this.PhoneNr;

Form1.user.Email = this.Email;

Form1.user.LastName = this.Name;

MessageBox.Show("Address Saved.");

}

else

MessageBox.Show("Insert E-mail");

}

else

MessageBox.Show("Insert Phone Number");

}

else

MessageBox.Show("Insert Address");

}

else

MessageBox.Show("Insert name");

}

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

{

CartControl.cartList.Remove(this);

this.Dispose();

}

internal void populateCart()

{

flowLayoutPanel1.Controls.Clear();

Subtotal = 0;

foreach (CartItems cartItem in cartList)

{

Subtotal += cartItem.Price;

flowLayoutPanel1.Controls.Add(cartItem);

}

}

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

{

cartList.Clear();

lblSubtotal.Text = "Subtotal:";

Subtotal = 0;

populateCart();

}

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

{

if (cartList.Count > 0)

{

if (Form1.user.ClientId > 0) {

ServiceReference1.WebService1SoapClient webService = Form1.getWebService();

webService.AddNewOrder(Form1.user.ClientId,additionalInfo.Text);

Form1.orderId = webService.GetOrderId(Form1.user.ClientId);

foreach (CartItems cartItem in cartList)

{

webService.AddItemToCart(Form1.orderId, cartItem.ProductId);

}

Form1.orderId = 0;

cartList.Clear();

populateCart();

if (additionalInfo.Text == "Insert additional information"||additionalInfo.Text=="")

additionalInfo.Text = "None";

MessageBox.Show("Order placed for:" + Form1.user.FirstName + " " + Form1.user.LastName + "\nAt:" + Form1.user.Address + "\nWith this phone number:" + Form1.user.PhoneNumber + "\nWith following additional informations given:"+additionalInfo.Text+"\nPaying with cash at delivery\nThank you!");

additionalInfo.Text = "Insert additional information";

Subtotal = 0;

}

else

MessageBox.Show("You have to Log-in to place a order");

}

else

MessageBox.Show("There is no item in cart");

}

[WebMethod]

public void AddProduct(String Category, String Name, String Ingredients, float Price, String NutritionalInformations, int Weight, byte[] ImageB)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("INSERT INTO [Products] (Category, Name, Ingredients, Price, NutritionalInformations, Weight, Image) VALUES (@Category, @Name, @Ingredients, @Price, @NutritionalInformations, @Weight, @Image)"))

{

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

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

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

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

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

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

cmd.Parameters.AddWithValue("@Image", ImageB);

cmd.Connection = myCon;

myCon.Open();

cmd.ExecuteNonQuery();

myCon.Close();

}

}

[WebMethod]

public void DeleteProductByName(string Name)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("UPDATE c1 SET c1.ProductId=22 FROM CartItems c1,Products p1 WHERE c1.ProductId=p1.ProductId AND p1.Name=@Name"))

{

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

cmd.Connection = myCon;

myCon.Open();

cmd.ExecuteNonQuery();

myCon.Close();

}

using (SqlCommand cmd = new SqlCommand("DELETE FROM [Products] WHERE Name=@Name"))

{

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

cmd.Connection = myCon;

myCon.Open();

cmd.ExecuteNonQuery();

myCon.Close();

}

}

[WebMethod]

public void CreateNewUser(String FirstName, String LastName, String Adress, String PhoneNumber, String Email, String UserName, String Password)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("INSERT INTO [Clients] (FirstName, LastName, Adress, PhoneNumber, Email, UserName, Password) VALUES (@FirstName, @LastName, @Adress, @PhoneNumber, @Email, @UserName, @Password)"))

{

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

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

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

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

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

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

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

cmd.Connection = myCon;

myCon.Open();

cmd.ExecuteNonQuery();

myCon.Close();

}

}

[WebMethod]

public DataTable GetUser(String UserName, String Password)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("SELECT \* FROM [Clients] WHERE UserName=@UserName AND Password=@Password"))

{

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

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

using (SqlDataAdapter sda = new SqlDataAdapter())

{

cmd.Connection = myCon;

sda.SelectCommand = cmd;

using (DataTable dt = new DataTable())

{

dt.TableName = "Table";

sda.Fill(dt);

myCon.Close();

return dt;

}

}

}

}

[WebMethod]

public void AddNewOrder(int ClientId, string AdditionalInfo)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("INSERT INTO [Orders] (Date,ClientId,AdditionalInfo) VALUES (@Date,@ClientId,@AdditionalInfo)"))

{

DateTime localDate = DateTime.Now;

cmd.Parameters.AddWithValue("@Date", localDate);

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

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

cmd.Connection = myCon;

myCon.Open();

cmd.ExecuteNonQuery();

myCon.Close();

}

}

[WebMethod]

public int GetOrderId(int ClientId)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("SELECT \* FROM [Orders] WHERE ClientId=@ClientId ORDER BY Date DESC"))

{

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

using (SqlDataAdapter sda = new SqlDataAdapter())

{

cmd.Connection = myCon;

sda.SelectCommand = cmd;

using (DataTable dt = new DataTable())

{

dt.TableName = "Table";

sda.Fill(dt);

myCon.Close();

int id = Convert.ToInt32(dt.Rows[0][0]);

return id;

}

}

}

}

[WebMethod]

public DataTable GetOrders(int ClientId)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("SELECT \* FROM [Orders] WHERE ClientId=@ClientId ORDER BY Date DESC"))

{

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

using (SqlDataAdapter sda = new SqlDataAdapter())

{

cmd.Connection = myCon;

sda.SelectCommand = cmd;

using (DataTable dt = new DataTable())

{

dt.TableName = "Table";

sda.Fill(dt);

myCon.Close();

return dt;

}

}

}

}

[WebMethod]

public DataTable GetOrdersItems(int OrderId)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("SELECT p1.ProductId,p1.Category,p1.Name,p1.Ingredients,p1.Price,p1.NutritionalInformations,p1.Weight,p1.Image FROM CartItems c1,Products p1 WHERE c1.OrderId=@OrderId AND c1.ProductId=p1.ProductId AND p1.ProductId <> 22 ORDER BY Category"))

{

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

using (SqlDataAdapter sda = new SqlDataAdapter())

{

cmd.Connection = myCon;

sda.SelectCommand = cmd;

using (DataTable dt = new DataTable())

{

dt.TableName = "Table";

sda.Fill(dt);

myCon.Close();

return dt;

}

}

}

}

[WebMethod]

public bool isUserExisting(String username)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("SELECT \* FROM [Clients] WHERE UserName=@UserName"))

{

cmd.Parameters.AddWithValue("@UserName", username);

using (SqlDataAdapter sda = new SqlDataAdapter())

{

cmd.Connection = myCon;

sda.SelectCommand = cmd;

DataTable dt = new DataTable();

dt.TableName = "Table";

sda.Fill(dt);

myCon.Close();

if (dt.Rows.Count > 0)

{

if (dt.Rows[0][6].ToString() == username)

return true;

}

return false;

}

}

}

[WebMethod]

public bool isAdmin(int ClientId)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("SELECT \* FROM [Clients] WHERE ClientId=@ClientId"))

{

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

using (SqlDataAdapter sda = new SqlDataAdapter())

{

cmd.Connection = myCon;

sda.SelectCommand = cmd;

using (DataTable dt = new DataTable())

{

dt.TableName = "Table";

sda.Fill(dt);

myCon.Close();

bool isAdmin = Convert.ToBoolean(dt.Rows[0][8]);

return isAdmin;

}

}

}

}

[WebMethod]

public void AddItemToCart(int OrderId, int ProductId)

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("INSERT INTO [CartItems] (OrderId,ProductId,Quantity) VALUES (@OrderId,@ProductId,@Quantity)"))

{

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

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

cmd.Parameters.AddWithValue("@Quantity", 1);

cmd.Connection = myCon;

myCon.Open();

cmd.ExecuteNonQuery();

myCon.Close();

}

}

[WebMethod]

public DataTable GetProductsTable()

{

SqlConnection myCon = new SqlConnection();

myCon.ConnectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Database1.mdf;Integrated Security=True";

using (SqlCommand cmd = new SqlCommand("SELECT \* FROM [Products] WHERE ProductId <> 22 ORDER BY Category"))

{

using (SqlDataAdapter sda = new SqlDataAdapter())

{

cmd.Connection = myCon;

sda.SelectCommand = cmd;

using (DataTable dt = new DataTable())

{

dt.TableName = "Table";

sda.Fill(dt);

myCon.Close();

return dt;

}

}

}

}

}