Clothing Store Online project

1. About This Project

The project in introduce an interfaces for managing Orders, Customers and Products on Online Clothing Store.
The project system built in c# Programming with Entity Frame Work Core technology where the database is Sql Server.
The GUI bulit using Windows Forms.

Programming	Data Access	Database	GUI
C#	Entity Framework Core.	SQL Server	WinForms

In this project was used in Singelton design pattern.

2.Run the application :

- There is method name IsExist() that checks if the database is exist. If no, the method will create a database by the migrations are updated and call to DataSeeding() Method to initial the seeding data to database.
 - The connection strong defined in the Db Context at ClothingStoreDB class at On Model Creating Method that checks if the connection configure, if not the connection will configure.
For Login – there is users list in User Entity in page 7.

3. Entities, Relations :

Relationships:

Order – customer : one – to many
Order – Product : many – to many.

The ProductsQuantity property which is a dictionary, Represent the relationship between .ordered and products

.He is stored as Json in the Database

instead of where the key represents the product Id, and the value represents the quantity from that product).

I know that it is not the most acceptable way but for me that was the best practice and best way that i saw fit for introduce the details products of each order

.and implement the many to many realtion between orders and products

The project implements EF TPT inheritance model.

:Entities

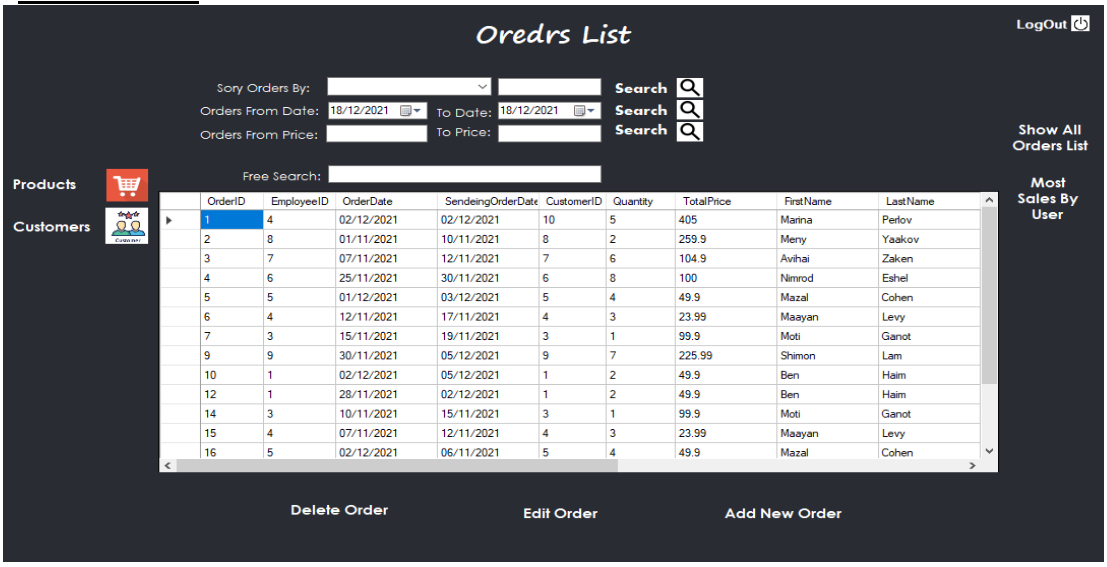
Orde :

public Class that represents an order with the properties as shown.

```
public int OrderID – praimry key. Represent the order number.
public int EmployeeID – represent the user that belong the order.
public DateTime OrderDate – represent the order date.
public DateTime SendeingOrderDate – represent the date of sending order.
public int CustomerID – represent the customer that the order belong to him.
 public Customer Customer – represent the customer in order.
public int Quantity – represent the total quantity in order.
public double TotalPrice – represent the total price of each order.
public enum ProductCategory – represent the category in order.
        {
            Men,
            Women,
            Children
        }
    public ProductCategory Category
    public Dictionary<int, int> ProductsQuantity public Dictionary<int, int>
```

ProductsQuantity – the ProductCategory property represent list of toal Products in each order by dictionary of <int,int>, when the key represented the prodctId and the value represent the number of products of same product.
He represent by Json like {"2":1},{"4":2} – meaning - 1 product from product 2 and 2 products from product 4.

Order screen:



Orders From Date: 18/12/2021 To Date: 18/12/2021

Orders From Price: To Price:

OrderID	EmployeeID	OrderDate	SendeingOrderDate	CustomerID	Quantity	TotalPrice	FirstName	LastName
1	4	02/12/2021	02/12/2021	10	5	405	Marina	Perlov
2	8	01/11/2021	10/11/2021	8	2	259.9	Meny	Yaakov
3	7	07/11/2021	12/11/2021	7	6	104.9	Avihai	Zaken
4	6	25/11/2021	30/11/2021	6	8	100	Nimrod	Eshel
5	5	01/12/2021	03/12/2021	5	4	49.9	Mazal	Cohen
6	4	12/11/2021	17/11/2021	4	3	23.99	Maayan	Levy
7	3	15/11/2021	19/11/2021	3	1	99.9	Moti	Ganot
9	9	30/11/2021	05/12/2021	9	7	225.99	Shimon	Lam
10	1	02/12/2021	05/12/2021	1	2	49.9	Ben	Haim
12	1	28/11/2021	02/12/2021	1	2	49.9	Ben	Haim
14	3	10/11/2021	15/11/2021	3	1	99.9	Moti	Ganot
15	4	07/11/2021	12/11/2021	4	3	23.99	Maayan	Levy
16	5	02/12/2021	06/11/2021	5	4	49.9	Mazal	Cohen

* The orders screen displays the list of all orders, implement the CRUD operation, When only users defined as Director can delete order.

* shows the ability of filter and ordering orders and filters by comboBox select Items.

* In addition you can filter orders from date to date, price to price and a button that displays .sales by users

- There is an ability for order search by free typing search In the content containing the search.

There is an ability for order search by free typing search In the content containing the search.

There is an ability for display the products details in each order by double clicking on the order table according to a specific orderId.

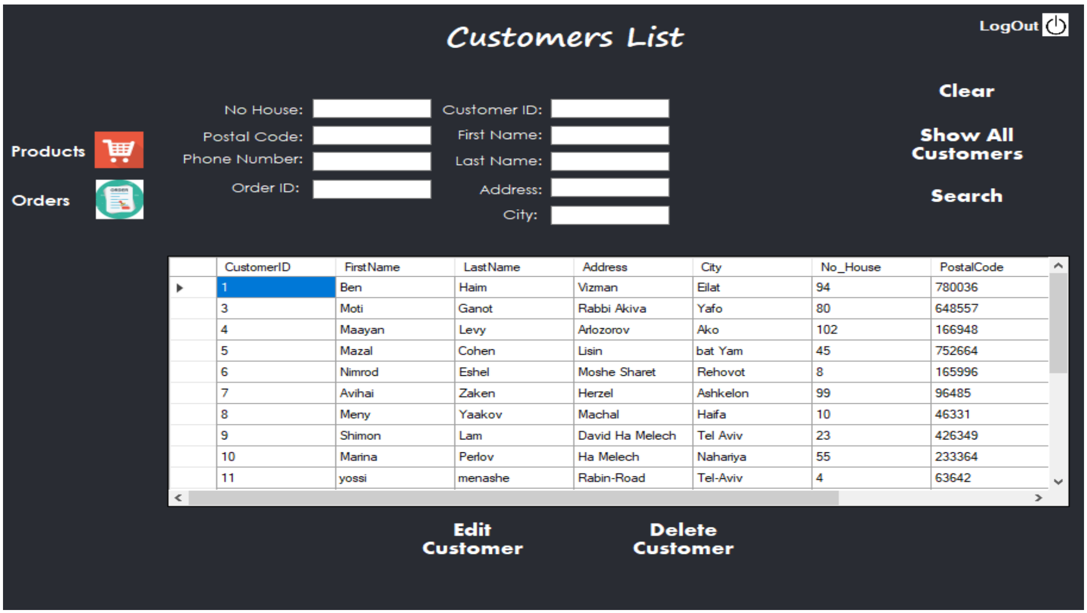
Customer:

public Class that represents an order with the fields as shown.

```
 public int CustomerID -primary key. represent the id customer.
 public string FirstName – represent the first name customer.
 public string LastName -represent the last name customer.
 public string FullName – represent the full name customer. Not mapped to database.
 public string Address – represent the address customer.
 public string City -represent the city customer.
 public int No_House – represent the number house customer.
 public int PostalCode – represent the postal code address customer.
 public string PhoneNumber – represent the phone number customer.
 public int OrderID - ForeignKey.represent the order id belong the customer.
```

implement the CRUD operation when a new customer can be added only after adding a new order.

Customer screen:



	CustomerID	FirstName	LastName	Address	City	No_House	PostalCode
▶	1	Ben	Haim	Vizman	Eilat	94	780036
	3	Moti	Ganot	Rabbi Akiva	Yafo	80	648557
	4	Maayan	Levy	Arlozorov	Ako	102	166948
	5	Mazal	Cohen	Lisin	bat Yam	45	752664
	6	Nimrod	Eshel	Moshe Sharet	Rehovot	8	165996
	7	Avihai	Zaken	Herzel	Ashkelon	99	96485
	8	Meny	Yaakov	Machal	Haifa	10	46331
	9	Shimon	Lam	David Ha Melech	Tel Aviv	23	426349
	10	Marina	Perlov	Ha Melech	Nahariya	55	233364
	11	yossi	menashe	Rabin-Road	Tel-Aviv	4	63642

Product:

public Class that represents an Product with the fields as shown.

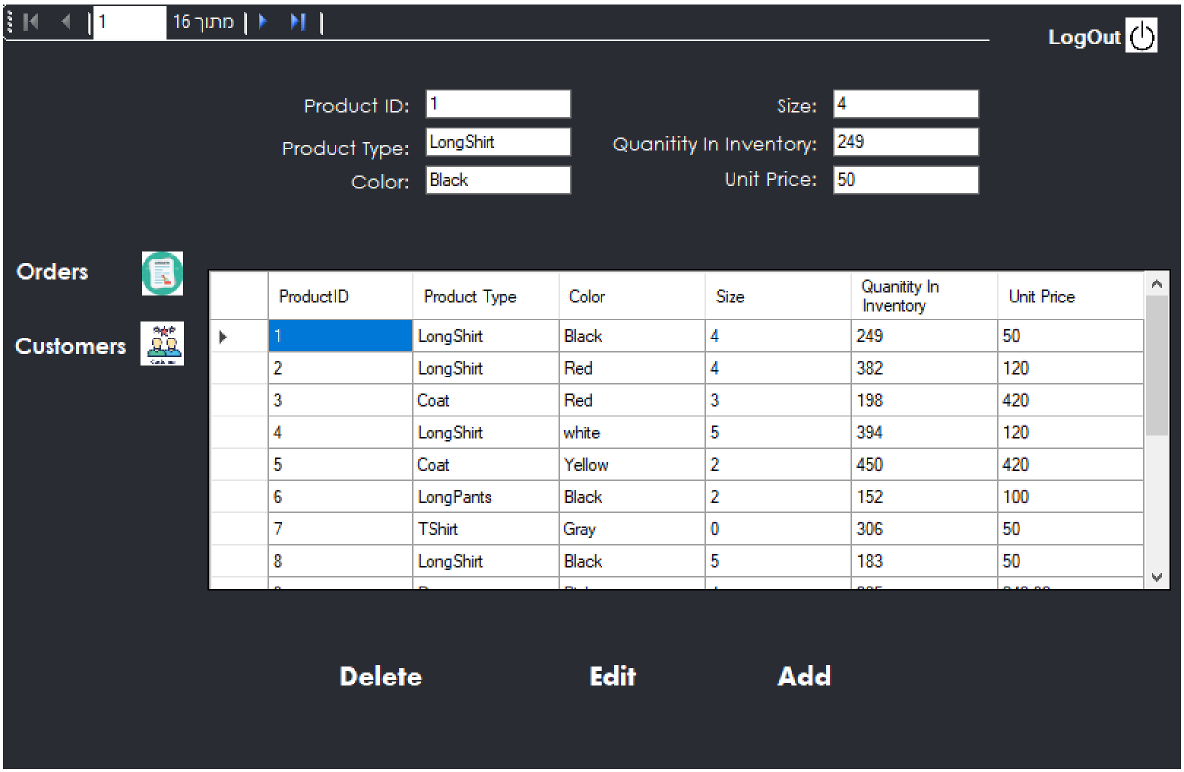
```
public class Product
public int ProductID – represent the productId.
public string ProductType - represent the product type.
public int QuanitityInInventory - represent quantity product in inventory.
public string Color- represent the color product.
public double UnitPrice - represent the unit price of product.
public enum ProductSize - represent the product size.

        {
            XS,
            S,
            M,
            L,
            XL,
            XXL
        };
        [Required]
    public ProductSize Size

    }
```

* implement the CRUD operation when a product is added only when the existing is in stock.

* When new order add i am checking if the product is in inventory. If so, the product will add to order and the quantity of product in stock decreases by 1, if no there is a message that the product is not in inventory.

Existing in stock- The assumption is that there is stock in supply.

Product screen:



Product ID: 1
Product Type: LongShirt
Color: Black
Size: 4
Quanitity In Inventory: 249
Unit Price: 50

	ProductID	Product Type	Color	Size	Quanitity In Inventory	Unit Price
▸	1	LongShirt	Black	4	249	50
	2	LongShirt	Red	4	382	120
	3	Coat	Red	3	198	420
	4	LongShirt	white	5	394	120
	5	Coat	Yellow	2	450	420
	6	LongPants	Black	2	152	100
	7	TShirt	Gray	0	306	50
	8	LongShirt	Black	5	183	50

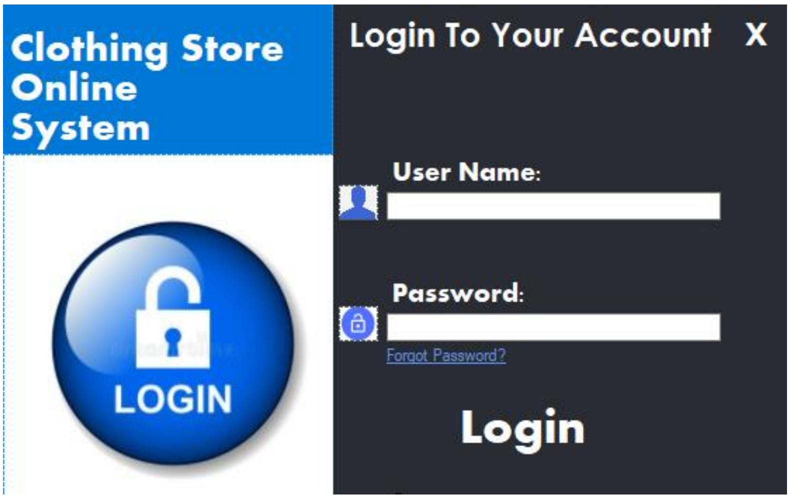
User:

public Class that represents an User with the fields as shown.

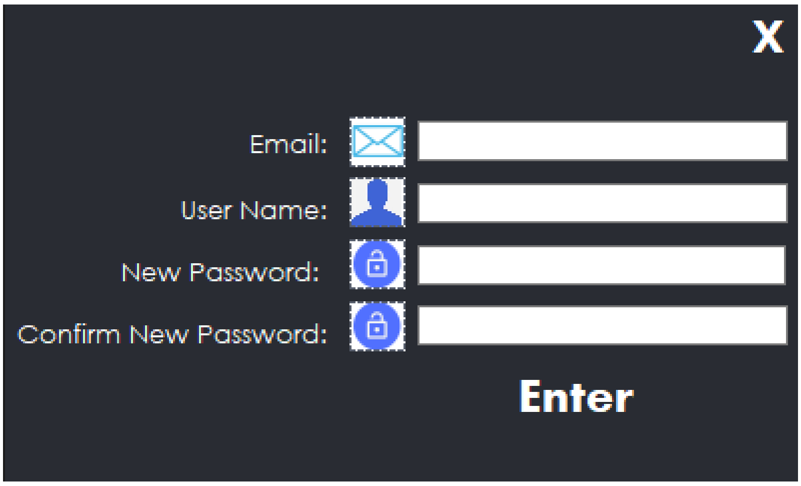
```
  public class User


 public int UserID – primary key. Represent the userId.
 public enum EnumUserType – represent the user type – director or official.
   {
      Director =0,
      Official =1
   }
 public EnumUserType UserType
 public string UserName - represent the user name of user.
 public string Password - represent the password of user .
 public string Email - represent the email of user.
```

* User logged in by username and password at Login Screen, When there is an option to change a new password in case the user forgets the password.
* Password change is done by entering an email and username.
* A user can add, edit and delete orders, products and customers from the system when The system assumption is that Only users that defined as Director can delete order, products and customers.

Login Screen:



Changing Password Screen:



(I could use the ability to recover a password by sending a temporary password to email but due to time I did not use and the assumption is that users will not exchange each other password.

List Of users for entering the system:

Email:	**UserName:**	**Password:**
Avi12@gmail.com	AviCohen	Avi123456
Daniel33@gmail.com	DanLevi	dani321
Rubi664@gmail.com	RubiKats	R966655

Yolam12@gamil.com	YossiLam	Yo123456
Meirt20@gamil.com	MeirTwito	S123963
Sima102@gamil.com	Sima Eliam	Elisi102
Noambl321@gamil.com	NoamBarLev	NoBar98764
Tami525@gamil.com	TamarEli	Tami5466