.Net Project

Inventory Management System

Team members:

Maryam Naji
Zohreh Karim Zadeh

JohnAbbott College -IPD 15

November 13 , 2018

Background of inventory manage system

“A system to manage inventory and tracking operations”

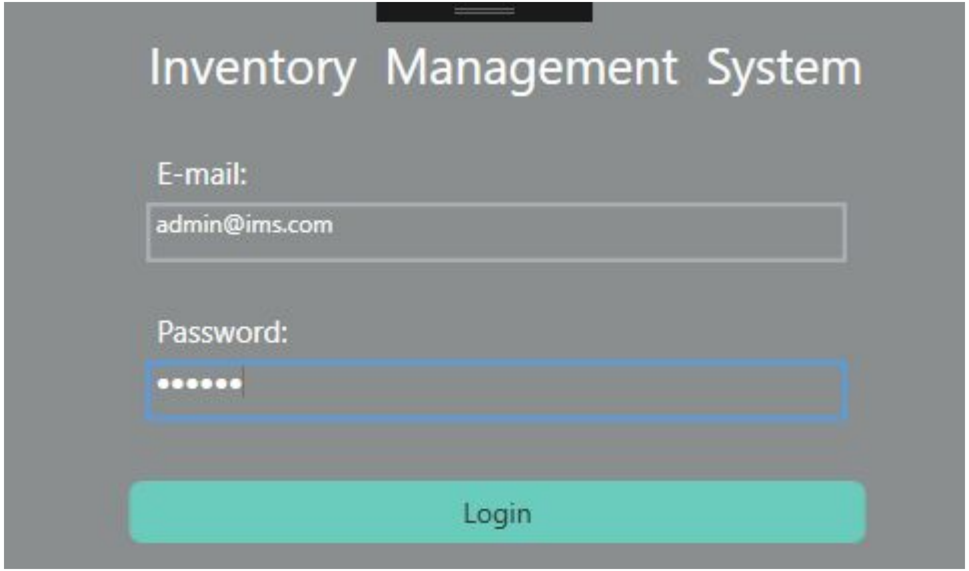
Through an inventory management system application ,Softwares and hardwares help user to manage operations automatically with more efficiency and less cost.

The application allows user :

- To record in/out orders information
- To keep stock in balance
- To update , add and delete information
- To Make report

Solution overview

Based on this multi-user application,many users can have access to the system by their own email and password to manage customers , products , in/out orders



Inventory Management System

E-mail:

admin@ims.com

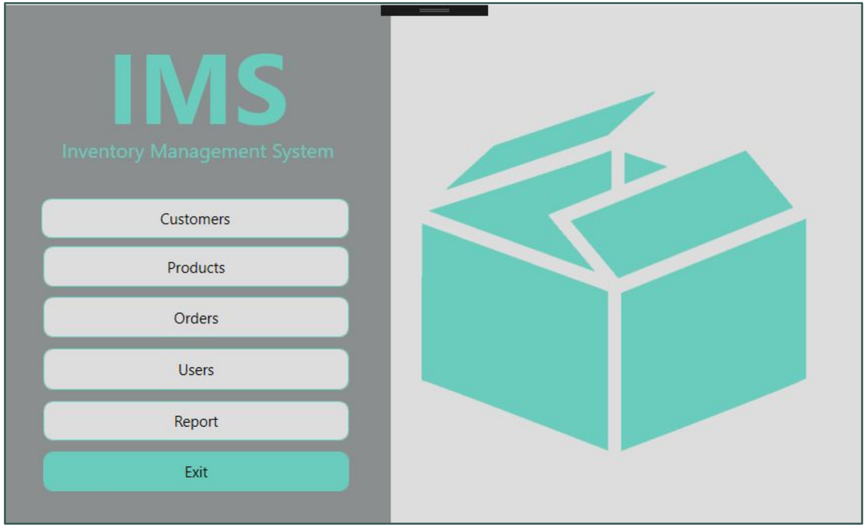
Password:

••••••

Functionalities:

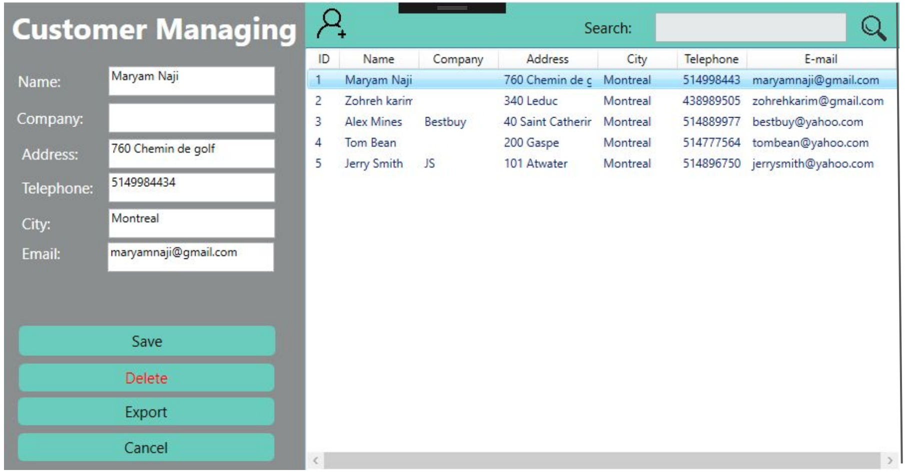
Adding image for products

Search for customers,products and



IMS

Inventory Management System



Customer Managing

Name: Maryam Naji

Company:

Address: 760 Chemin de golf

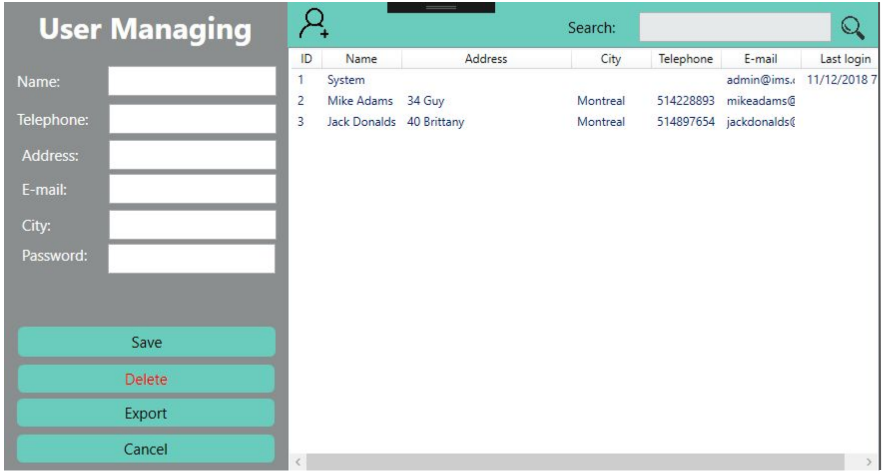
Telephone: 5149984434

City: Montreal

Email: maryamnaji@gmail.com

Search:

ID	Name	Company	Address	City	Telephone	E-mail
1	Maryam Naji		760 Chemin de g	Montreal	514998443	maryamnaji@gmail.com
2	Zohreh karim		340 Leduc	Montreal	438989505	zohrehkarim@gmail.com
3	Alex Mines	Bestbuy	40 Saint Catherir	Montreal	514889977	bestbuy@yahoo.com
4	Tom Bean		200 Gaspe	Montreal	514777564	tombean@yahoo.com
5	Jerry Smith	JS	101 Atwater	Montreal	514896750	jerrysmith@yahoo.com



User Managing

Name:

Telephone:

Address:

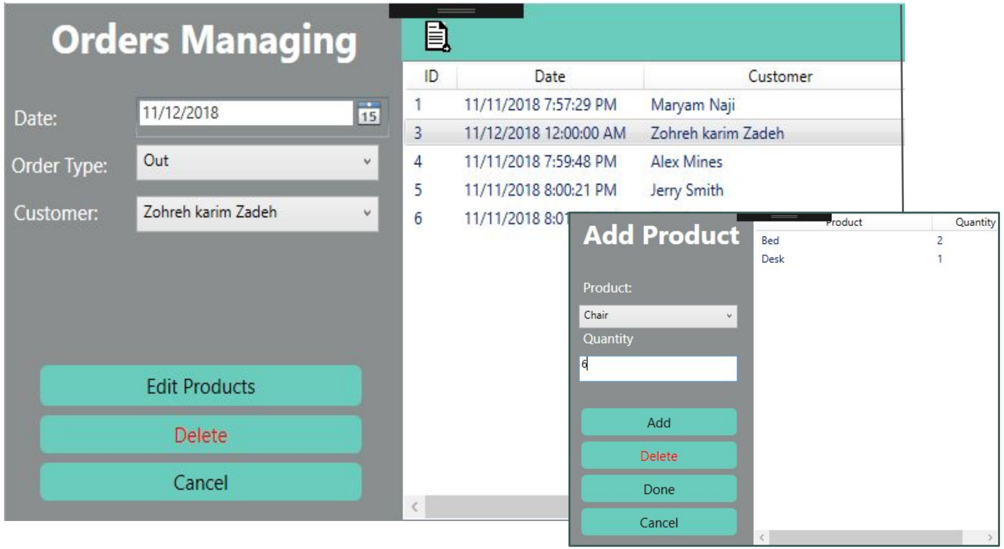
E-mail:

City:

Password:

Search:

ID	Name	Address	City	Telephone	E-mail	Last login
1	System				admin@ims.	11/12/2018 7
2	Mike Adams	34 Guy	Montreal	514228893	mikeadams@	
3	Jack Donalds	40 Brittany	Montreal	514897654	jackdonalds@	



Orders Managing

Date: 11/12/2018

Order Type: Out

Customer: Zohreh karim Zadeh

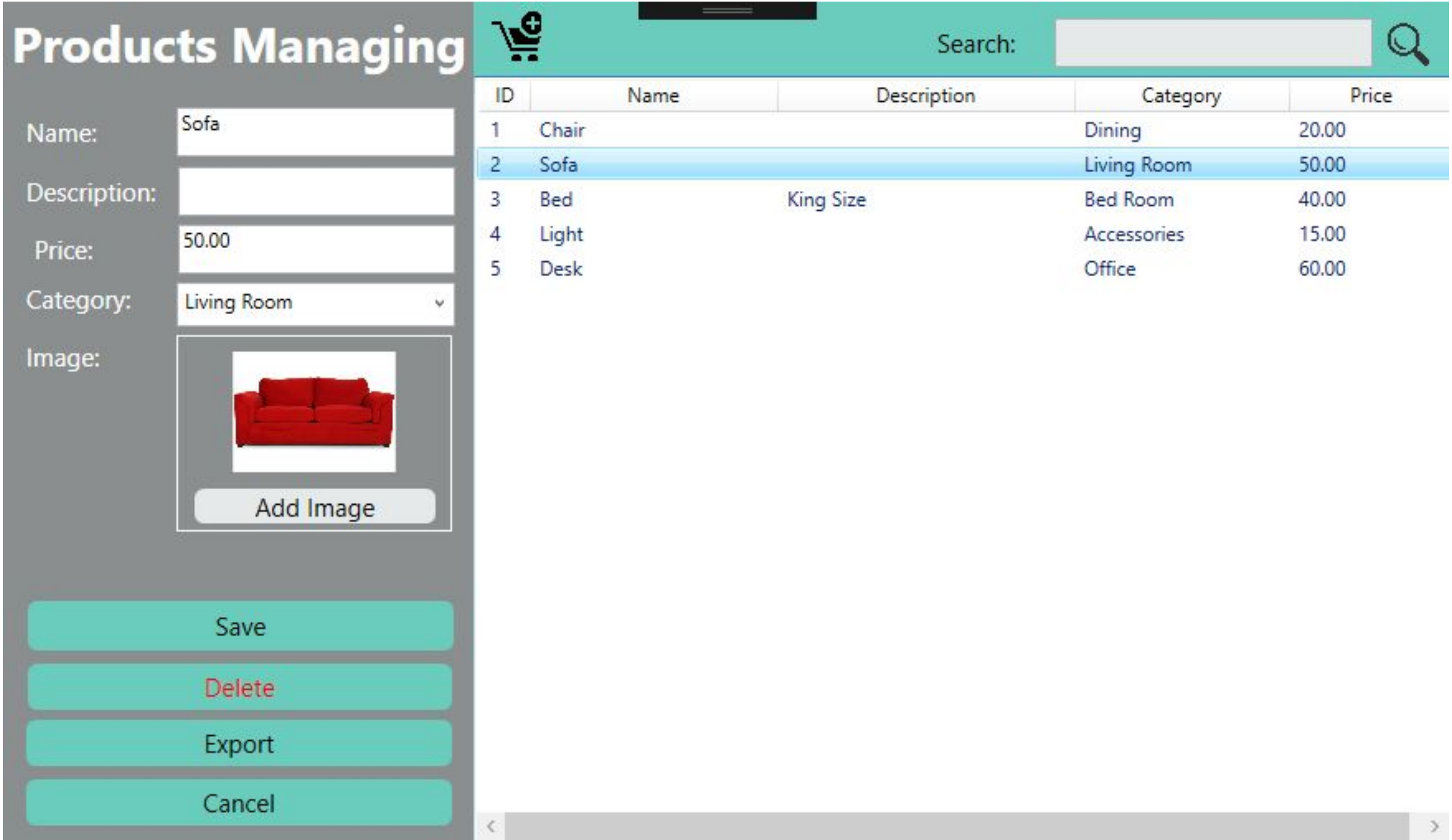
ID	Date	Customer
1	11/11/2018 7:57:29 PM	Maryam Naji
3	11/12/2018 12:00:00 AM	Zohreh karim Zadeh
4	11/11/2018 7:59:48 PM	Alex Mines
5	11/11/2018 8:00:21 PM	Jerry Smith
6	11/11/2018 8:0	

Add Product

Product: Chair

Quantity: 6

Product	Quantity
Bed	2
Desk	1



Products Managing

Name: Sofa

Description:

Price: 50.00

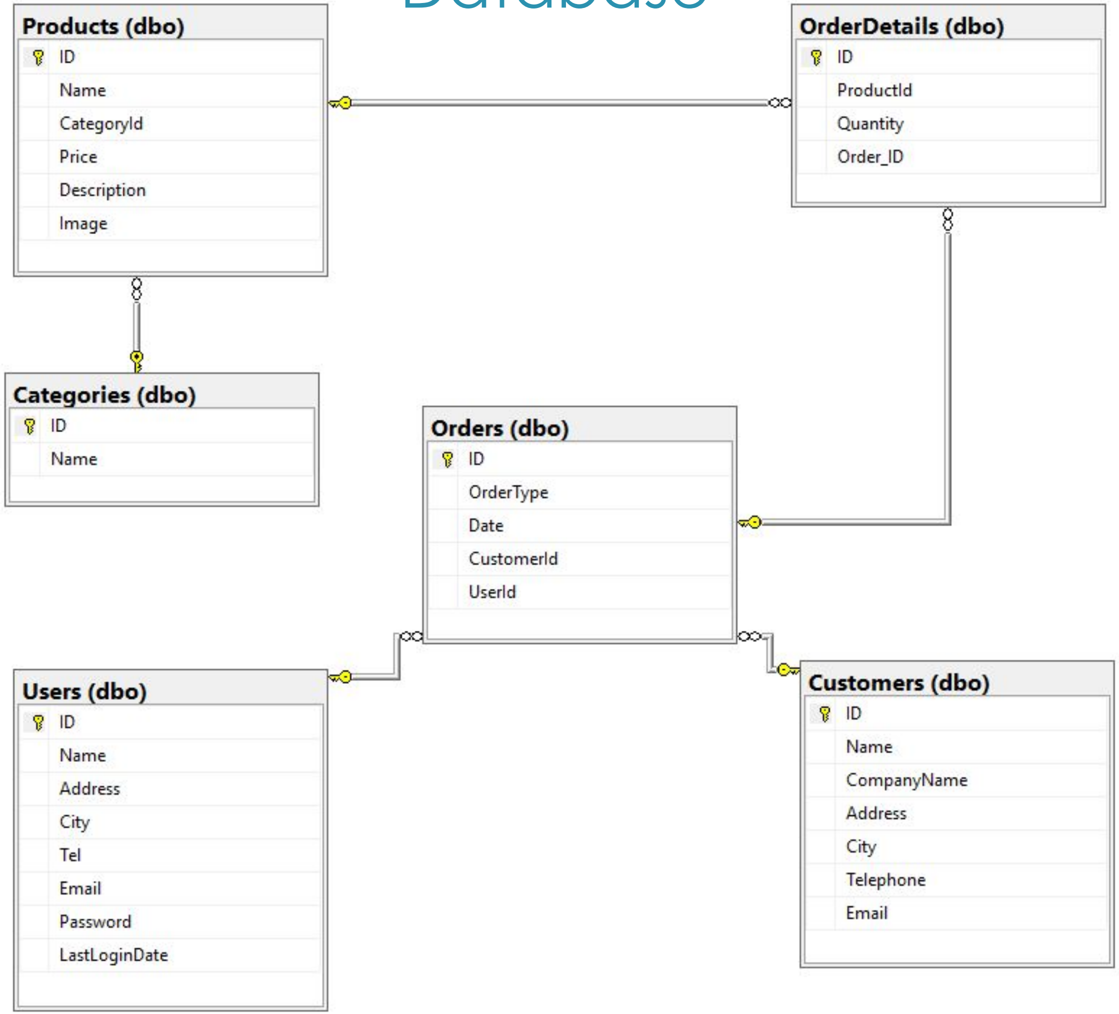
Category: Living Room

Image:

Search:

ID	Name	Description	Category	Price
1	Chair		Dining	20.00
2	Sofa		Living Room	50.00
3	Bed	King Size	Bed Room	40.00
4	Light		Accessories	15.00
5	Desk		Office	60.00

Database



Products (dbo)

ID
Name
CategoryId
Price
Description
Image

OrderDetails (dbo)

ID
ProductId
Quantity
Order_ID

Categories (dbo)

ID
Name

Orders (dbo)

ID
OrderType
Date
CustomerId
UserId

Users (dbo)

ID
Name
Address
City
Tel
Email
Password
LastLoginDate

Customers (dbo)

ID
Name
CompanyName
Address
City
Telephone
Email

Technologies

- WPF
- Database via Azure
- Unit tests
- EntityFramework
- Export to Excel
- LINQ
- MVVM

The biggest challenge

- Storing binary data in a BLOB

The three biggest challenges

- Moving event handler from code behind to ViewModel
- Binding Password box

  (For security reasons, the Password property is not implemented as a dependency property, which means that it can not be binded.)
- Implement Property Change Notification(Textbox and Listview)

1) Event → Command

Xaml:

```
<Button Command="{Binding SaveCommand}" />
```

ViewModel:

```
public ICommand SaveCommand { get; set; }

public ViewModel()
{
        this.SaveCommand = new
        RelayCommand(p  => this.Save(), p => true);
}
```

RelayCommand Class :

```
public class RelayCommand:ICommand
{
        Codes….
}
```

2) PaswordBox Binding

Xaml:

```
<PasswordBox  x:Name="PasswordBox"
ff:PasswordBoxAssistant.BindPassword="true"
ff:PasswordBoxAssistant.BoundPassword="{Binding
Path=CurrentUser.Password, Mode=TwoWay,
UpdateSourceTrigger=PropertyChanged}"/>
```

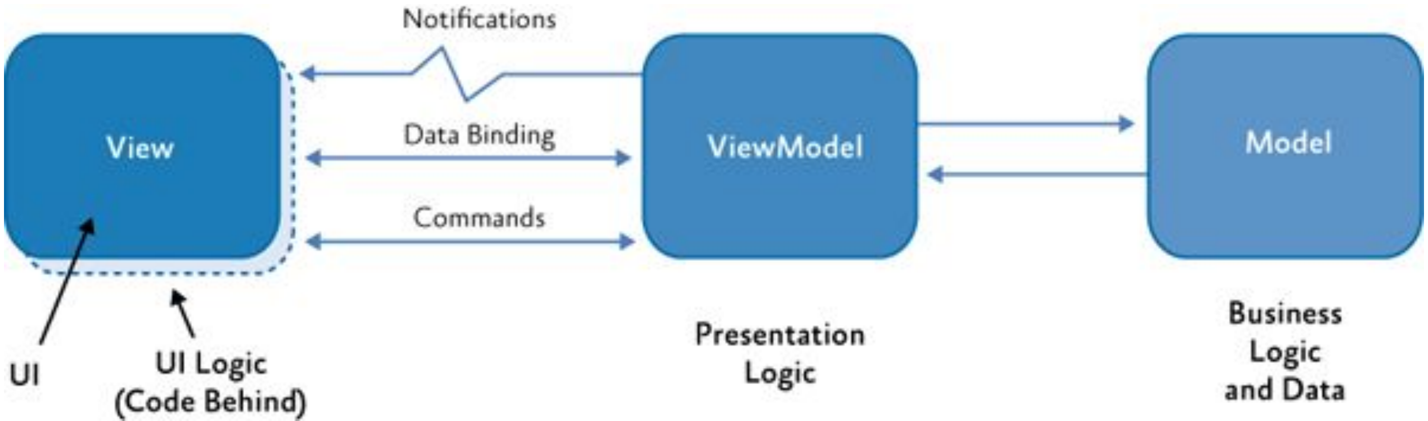
PasswordBoxAssistant class :

```
Public static class PasswordBoxAssistant
{
        Codes….
}
```

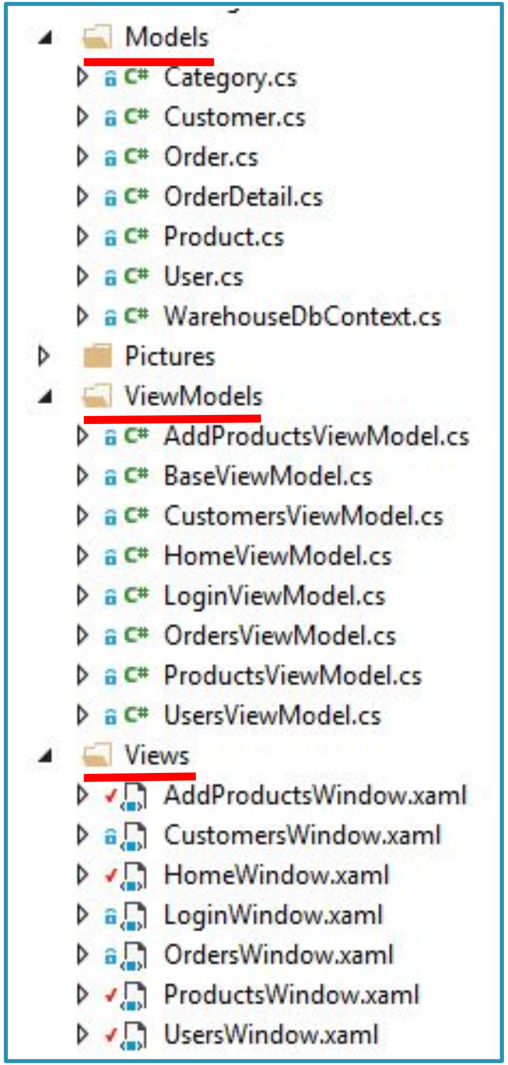
3) Implementation of INotifyPropertyChanged

```
public class BaseViewModel : INotifyPropertyChanged
{

    protected void OnPropertyChanged(string name)
    {
     codes...
    }

}
```

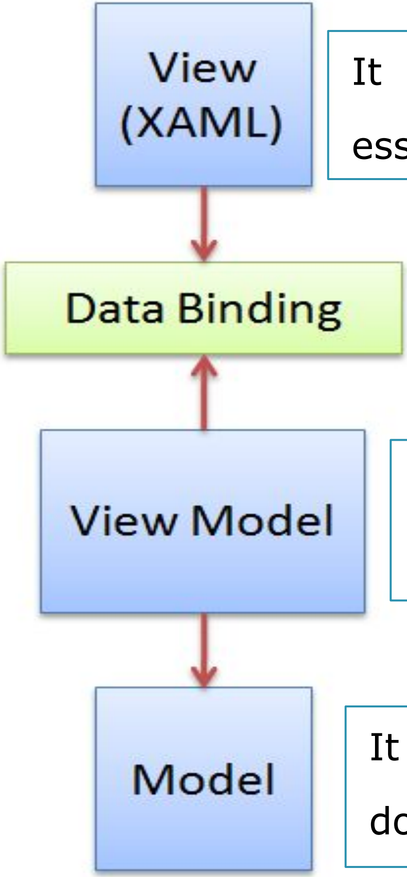
```
public class ViewModel: BaseViewModel
{
        private List<Customer> _customerList;
        public List<Customer> CustomerList
        {
            get { return _customerList; }
            set
            {
                _customerList = value;
               OnPropertyChanged("CustomerList");
            }
        }

}
```



What we learned



```
◢ Models
    ▷ C# Category.cs
    ▷ C# Customer.cs
    ▷ C# Order.cs
    ▷ C# OrderDetail.cs
    ▷ C# Product.cs
    ▷ C# User.cs
    ▷ C# WarehouseDbContext.cs
▷ Pictures
◢ ViewModels
    ▷ C# AddProductsViewModel.cs
    ▷ C# BaseViewModel.cs
    ▷ C# CustomersViewModel.cs
    ▷ C# HomeViewModel.cs
    ▷ C# LoginViewModel.cs
    ▷ C# OrdersViewModel.cs
    ▷ C# ProductsViewModel.cs
    ▷ C# UsersViewModel.cs
◢ Views
    ▷ AddProductsWindow.xaml
    ▷ CustomersWindow.xaml
    ▷ HomeWindow.xaml
    ▷ LoginWindow.xaml
    ▷ OrdersWindow.xaml
    ▷ ProductsWindow.xaml
    ▷ UsersWindow.xaml
```



It simply holds the formatted data and essentially delegates everything to the Model.

It acts as the link/connection between the Model and View and makes stuff look pretty.

It simply holds the data and has nothing to do with any of the business logic.

Future Work

- Localization
- Printing using WPF or additional library

Summary

By working on this project,we learnt about MVVM architecture that can help us to understand MVC pattern easier in future,because MVVM pattern and MVC pattern have the same goal :

providing a clear separation between domain logic and presentation layer,that leads to :

- Maintainability
- Testability
- Extensibility