**Lab 01. Building an Product Management Application using LINQ and WPF**

# 1. Introduction

Imagine you're an employee of a store named **ProductStore**. Your manager has asked you to develop a WPF application for product management. The application has to support adding, viewing, modifying, and removing products—a standardized usage action verbs better known as Create, Read, Update, Delete (CRUD).

This lab explores creating an application using WPF with .NET Core, and C#. An "in-memory database" will be created to persist the car's data, so a collection is called List will be used for reading and managing product data.

# 2. Lab Objectives

In this lab, you will:

* Use the Visual Studio.NET to create WPF application and Class Library (.dll) project.
* Use the Visual Studio.NET to create Windows Forms and Class Library (.dll) project.
* Create a List of persisting products using LINQ to Object to find items.
* Apply passing data in WPF application
* Apply Repository pattern in a project.
* Add CRUD action methods to WPF application.
* Run the project and test the WPF application actions.

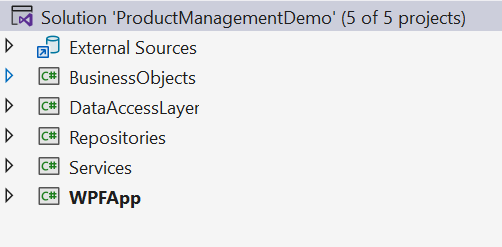
# Activity 01: Build a solution by Visual Studio.NET

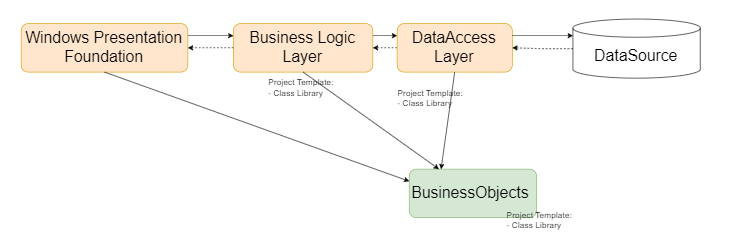
Create a Blank Solution named **ProductManagementDemo** then add new a **Class Library** project named **BusinessObjects, DataAccessObjects, Repositories, Services** and a WPF project named **ProductManagement**

## **Step 01**. Create a Blank solution.

## **Step 02.** Create 4 **Class Library** projects.

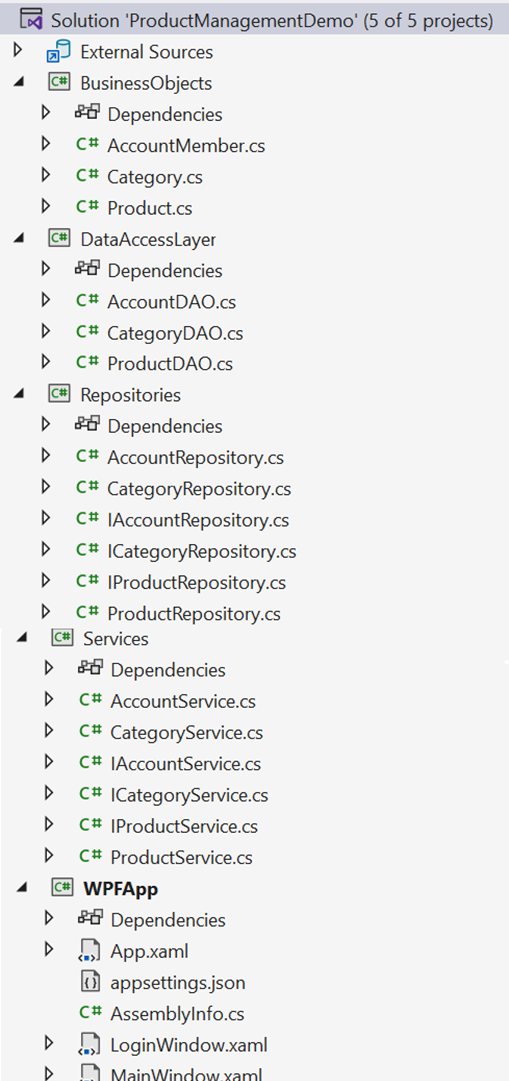
## **Step 03.** Create a WPF project.



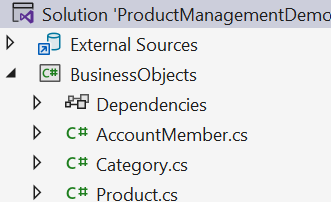
****

Note:

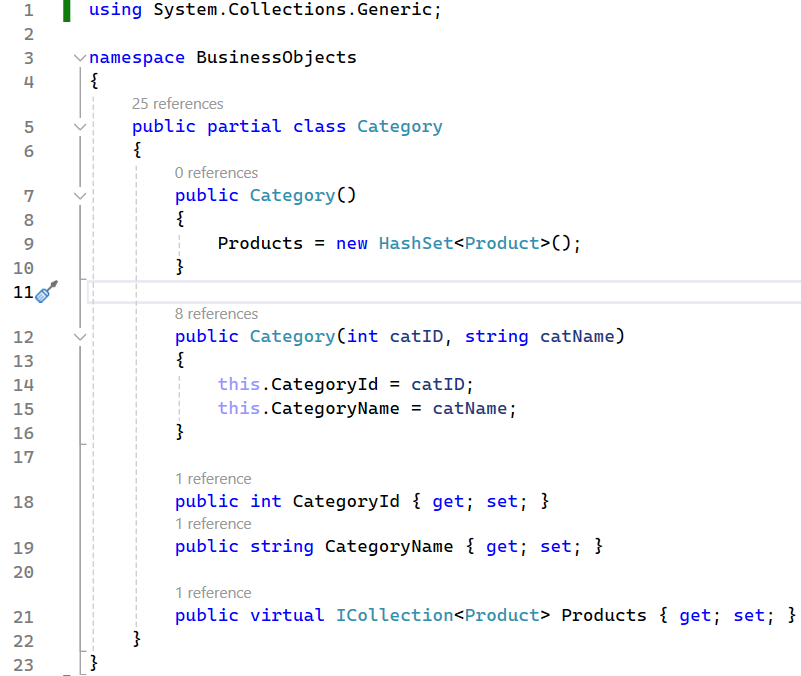
* **Data Source** in this case is the “**In memory Database”** (using List collection)
* **Services Project** – This project represents a layer or component responsible for implementing the business logic of an application.
* **Repository Project** – This project provides an abstraction layer between the application’s business logic and the underlying data source.
* **Data Access Layer Project** – This project used to abstract and encapsulate the logic for accessing data from a data source, such as a database.



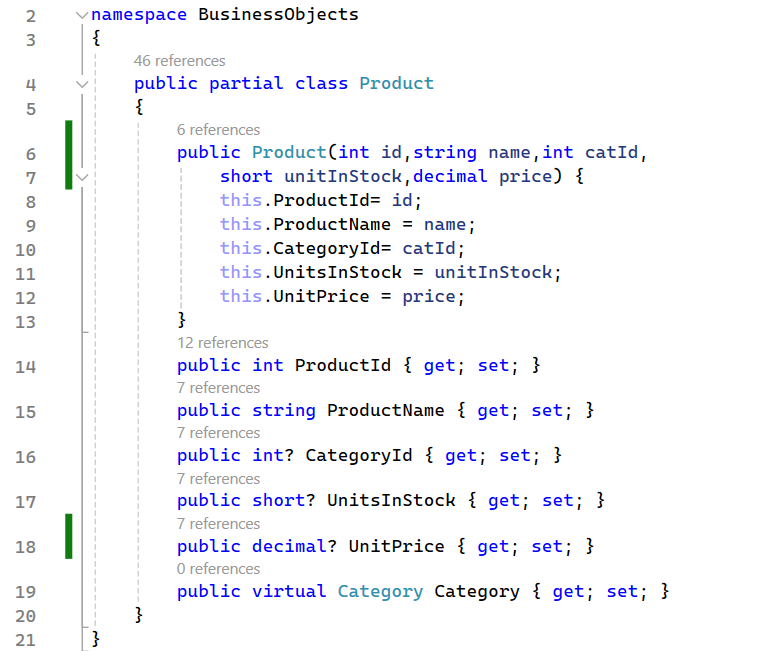
# Activity 02: Write codes for the BusinessObjects project



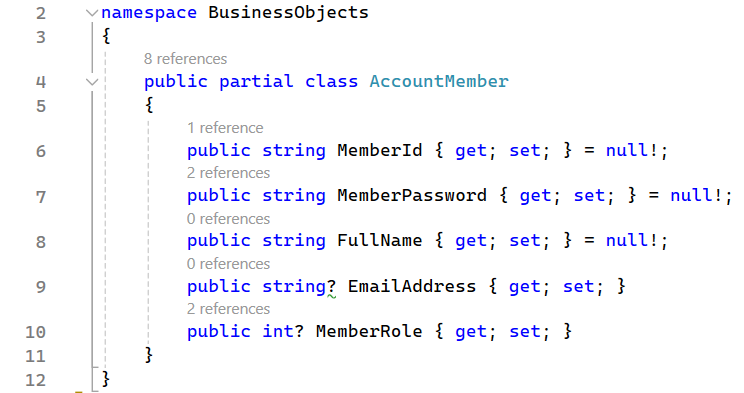
## **Step 01**. On the **BusinessObjects** project, add a class named **Product.cs** and write codes as follows:



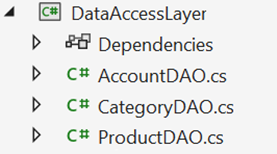
## **Step 02**. On the **BusinessObjects** project, add a class named **Category.cs** and write codes as follows:



## **Step 02**. On the **BusinessObjects** project, add a class named **Category.cs** and write codes as follows:



# Activity 03: Write codes for the DataAccessLayer project

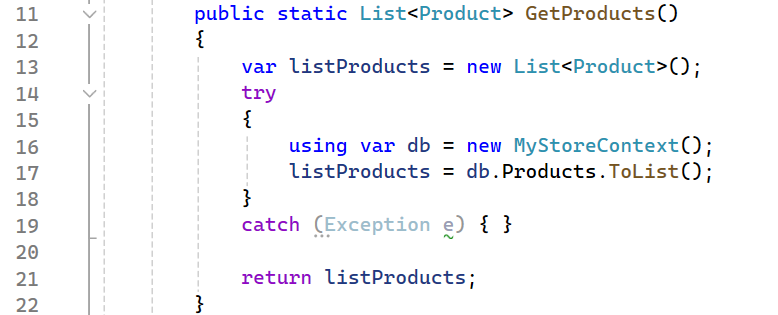
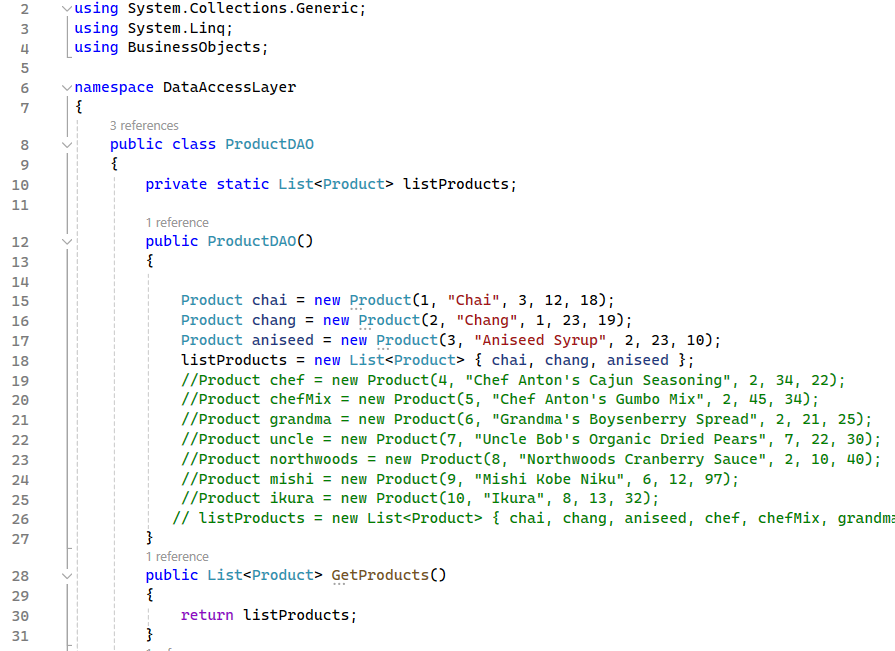


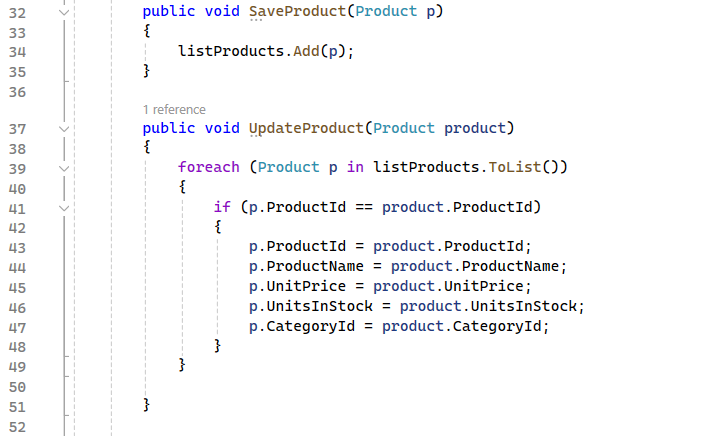
## **Step 01.** On the **DataAccessLayer** project, add a class named **CategoryDAO.cs** and write codes as follows:

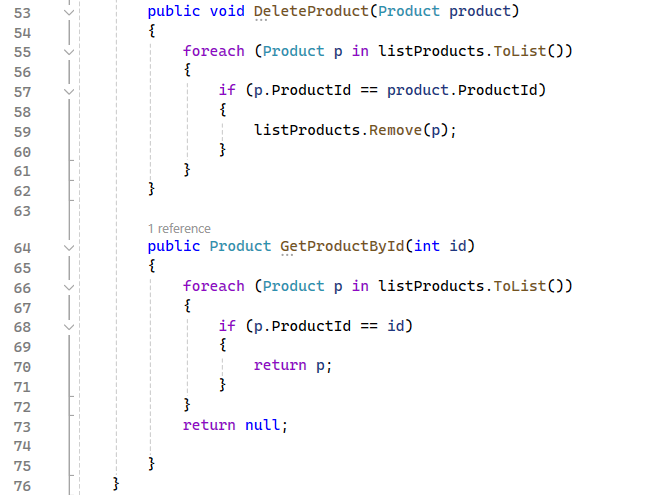


## **Step 02**. On the **DataAccessLayer** project, add a class named **ProductDAO.cs** and write codes as follows:

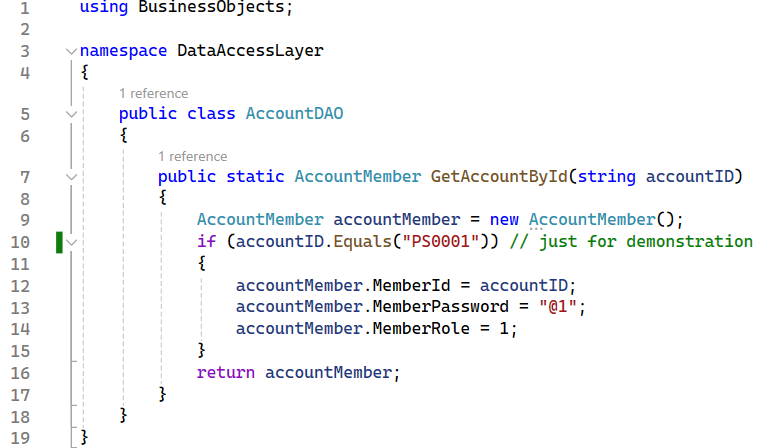
The details of functions in ProductDAO.cs:



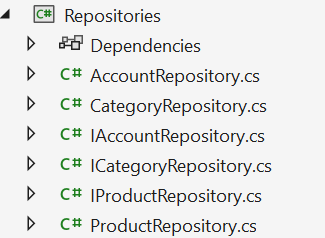




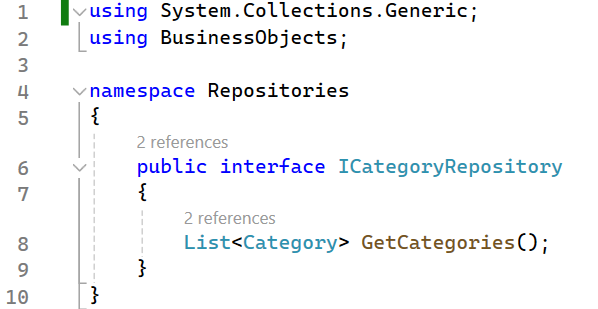
## **Step 03**. Write codes for **AccountDAO.cs** as follows:



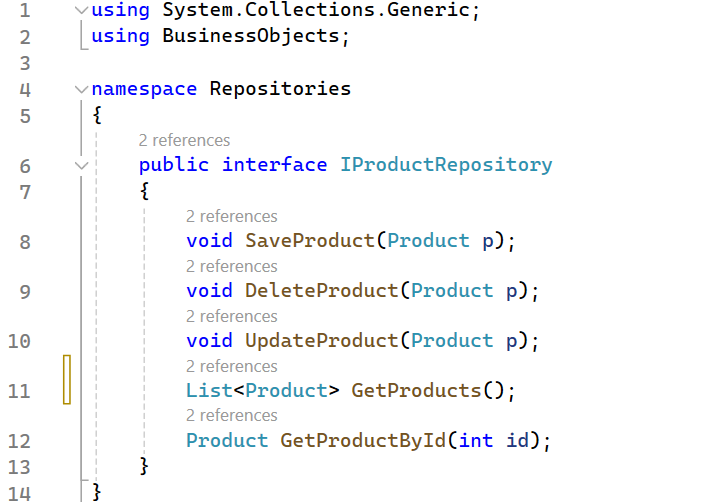
# Activity 04: Write codes for the Repositories project



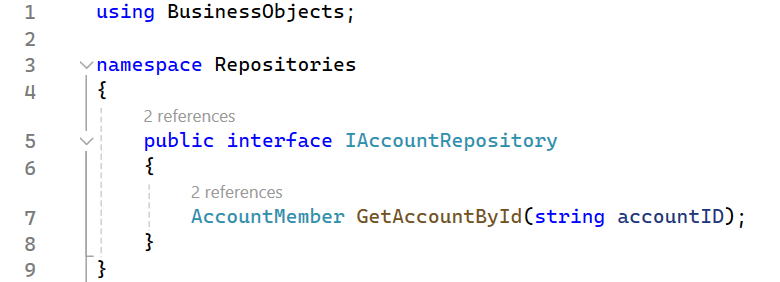
## **Step 01.** On the **Repositories** project, add an interface named **ICatergoryRepository.cs** and write codes as follows:



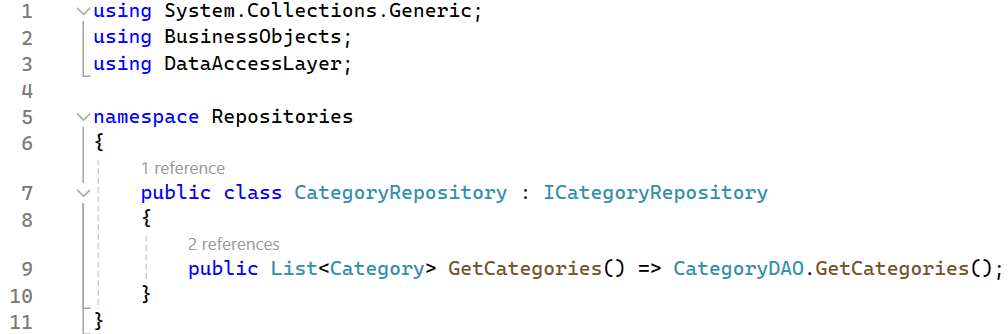
## **Step 02**. On the **Repositories** project, add an interface named **IProductRepository.cs** and write codes as follows:



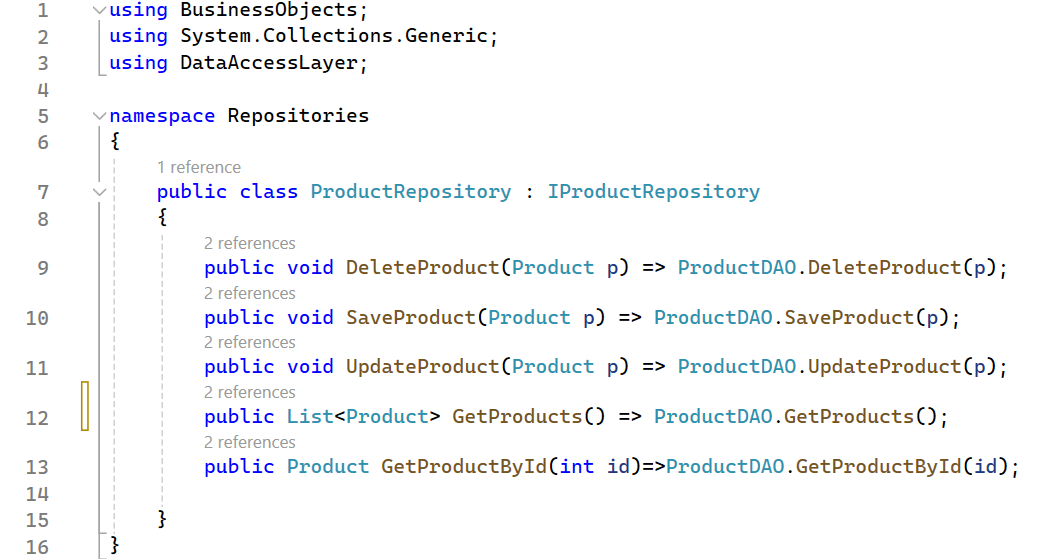
## **Step 03**. On the **Repositories** project, add an interface named **IAccountRepository.cs** and write codes as follows:



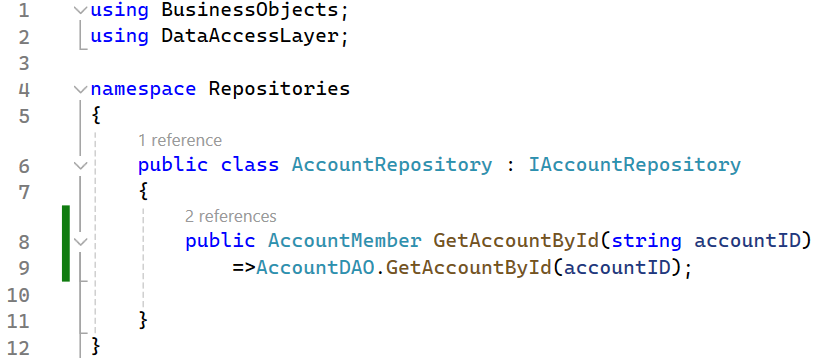
**Step 04**. Write codes for class **CategoryRepository.cs** as follows:



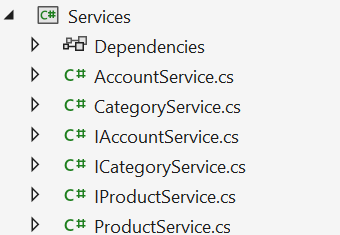
**Step 05**. Write codes for class **ProductRepository.cs** as follows:



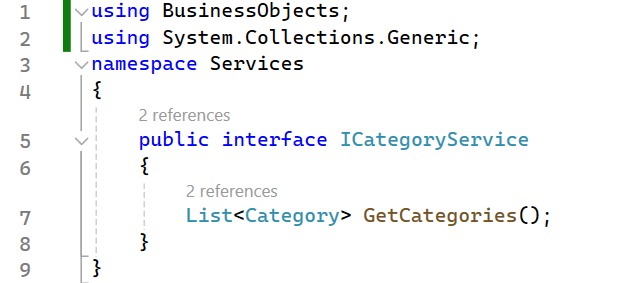
**Step 06**. Write codes for class **AccountRepository.cs** as follows:



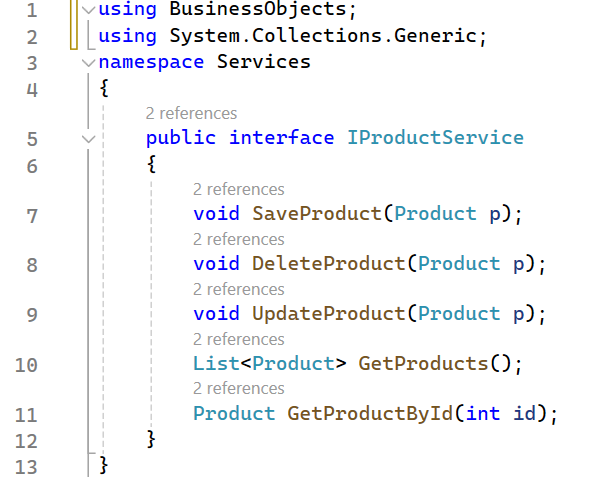
# Activity 05: Write codes for the Services project



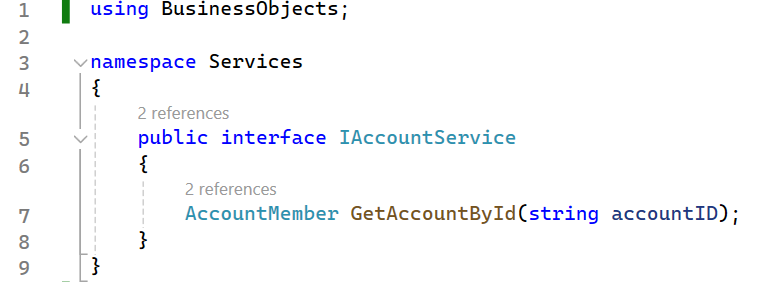
## **Step 01.** On the **Services** project, add an interface named **ICatergoryService.cs** and write codes as follows:



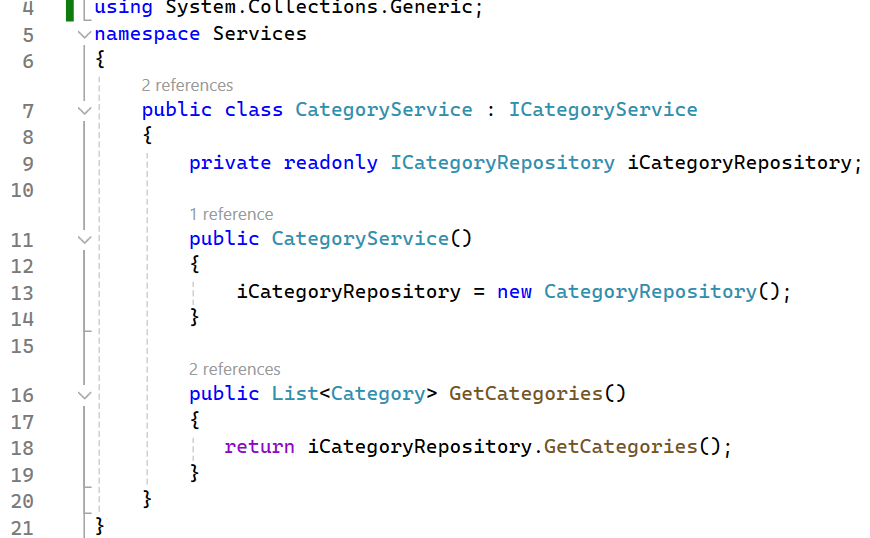
## **Step 02**. On the **Services** project, add an interface named **IProductService.cs** and write codes as follows:



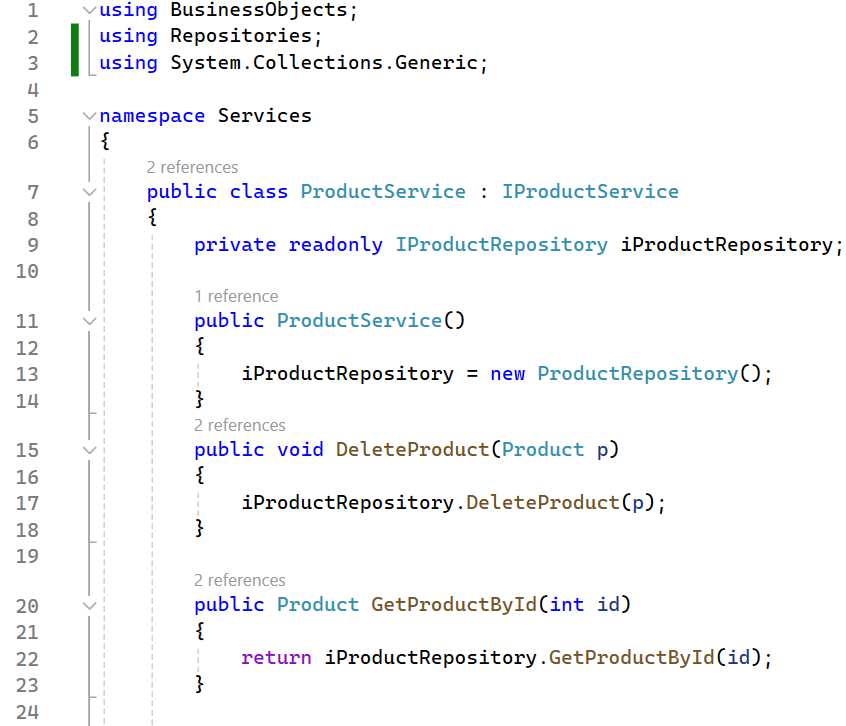
## **Step 03**. On the **Services** project, add an interface named **IAccountService.cs** and write codes as follows:

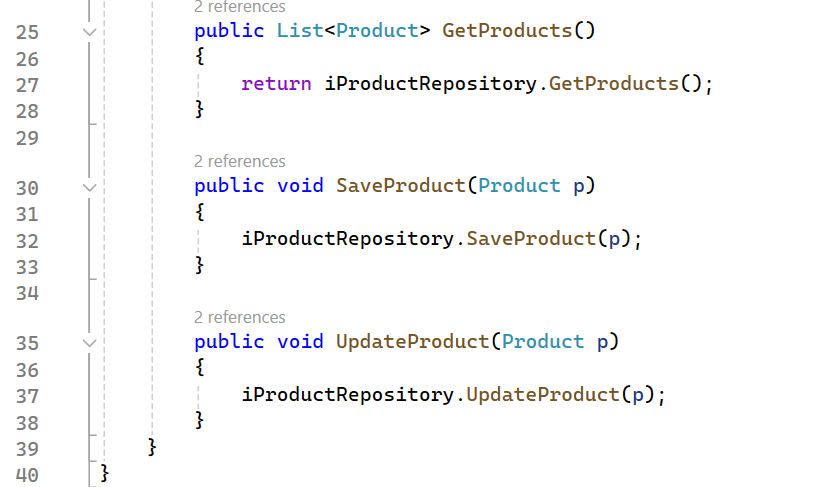


**Step 04**. Write codes for class **CategoryService.cs** as follows:

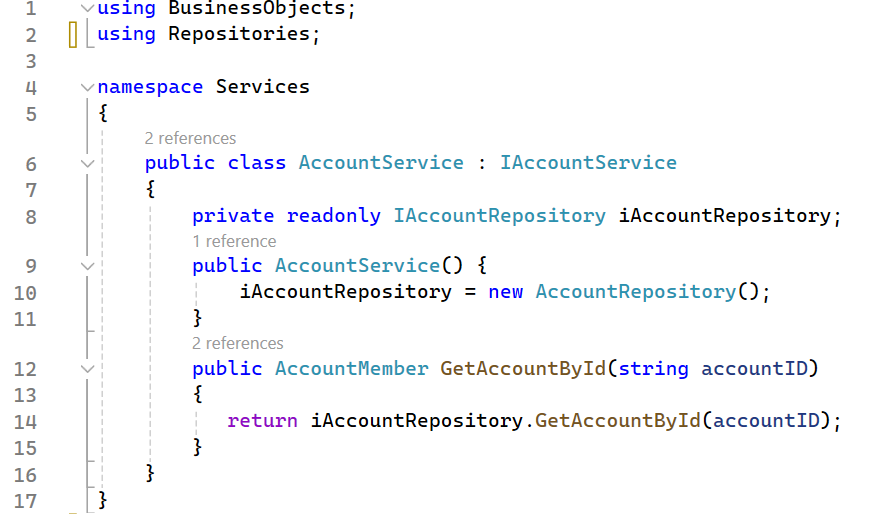


**Step 05**. Write codes for class **ProductService.cs** as follows:

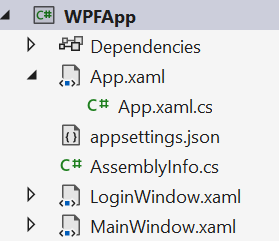




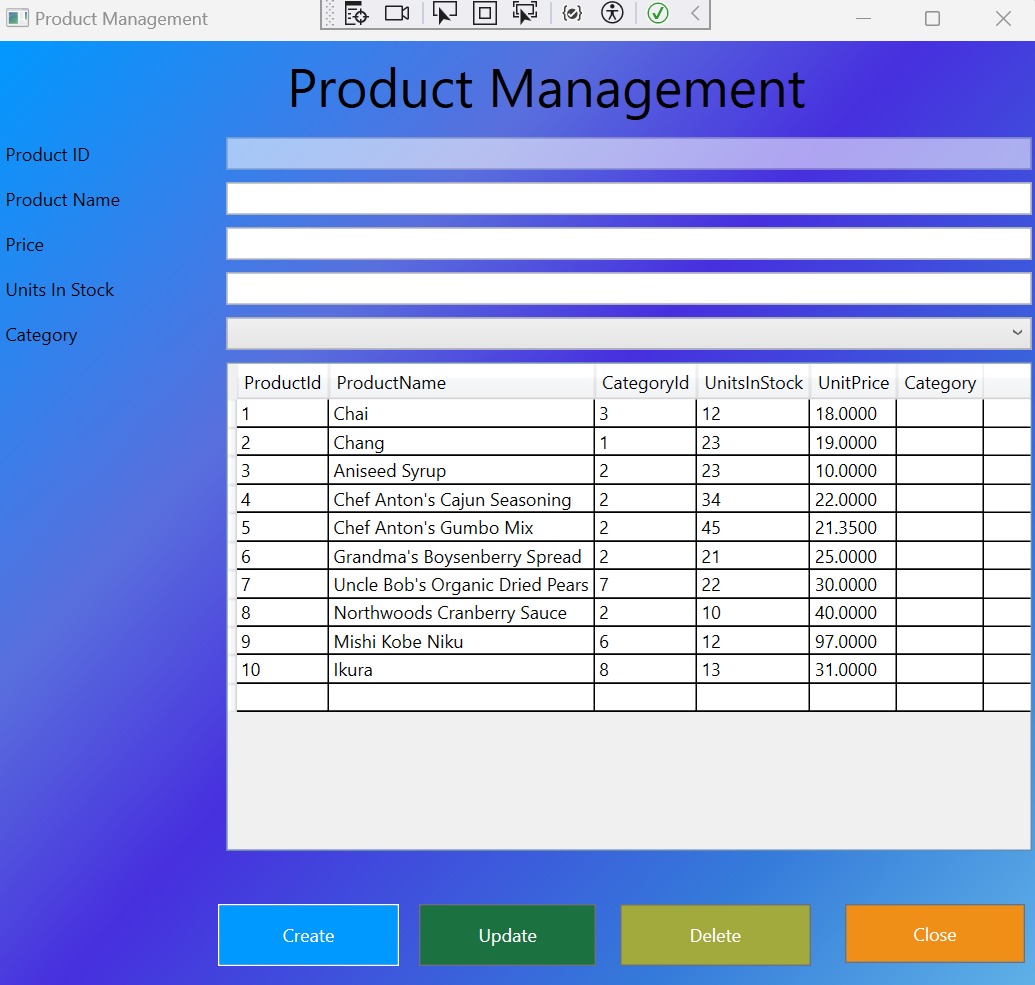
**Step 06**. Write codes for class **AccountService.cs** as follows:



# Activity 04: Design UI and write codes for WPF project



## **Step 01.** On the **WPF** project, design UI as follows**:**

****

* **XAML** code for MainWindow.xaml

<Window x:Class="WPFApp.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:WPFApp"

mc:Ignorable="d"

Loaded="Window\_Loaded"

WindowStartupLocation="CenterScreen"

Title="Product Management" Height="670" Width="710">

<Grid>

<Grid>

<Grid.Background>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<GradientStop Color="#0099FF" Offset="0"/>

<GradientStop Color="#FF347BDA" Offset="0.794"/>

<GradientStop Color="#FF60B1E7" Offset="1"/>

<GradientStop Color="#FF596FDD" Offset="0.309"/>

<GradientStop Color="#FF472FDE" Offset="0.484"/>

</LinearGradientBrush>

</Grid.Background>

<Grid.RowDefinitions>

<RowDefinition Height="60"/>

<RowDefinition Height="30"/>

<RowDefinition Height="30"/>

<RowDefinition Height="30"/>

<RowDefinition Height="30"/>

<RowDefinition Height="30"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="30"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="119.415"/>

<ColumnDefinition Width="30.585"/>

<ColumnDefinition Width="47\*"/>

<ColumnDefinition Width="513\*"/>

</Grid.ColumnDefinitions>

<Label x:Name="label" Content="Product Management" Grid.Column="2" Grid.Row="0" FontSize="36" Grid.ColumnSpan="2" HorizontalAlignment="Center" Width="466"/>

<Label x:Name="label1" Margin ="2,2,2,2" Content="Product ID" Grid.Column="0" Grid.Row="1" Grid.ColumnSpan="2"/>

<TextBox x:Name="txtProductID" Margin ="4,4,4,4" Grid.Column="2" Grid.Row="1" Text="" TextWrapping="Wrap" Grid.ColumnSpan="2" IsEnabled="False" />

<Label x:Name="label2" Margin ="2,2,2,2" Grid.Column="0" Grid.Row="2" Content="Product Name" Grid.ColumnSpan="2" />

<TextBox x:Name="txtProductName" Margin ="4,4,4,4" Grid.Column="2" Grid.Row="2" Text="" TextWrapping="Wrap" Grid.ColumnSpan="2" />

<Label x:Name="label8" Margin ="2,2,2,2" Content="Price" Grid.Column="0" Grid.Row="3" Grid.ColumnSpan="2"/>

<TextBox x:Name="txtPrice" Margin ="4,4,4,4" Grid.Column="2" Grid.Row="3" Text="" TextWrapping="Wrap" Grid.ColumnSpan="2" />

<Label x:Name="label3" Margin ="2,2,2,2" Content="Units In Stock" Grid.Column="0" Grid.Row="4" Grid.ColumnSpan="2"/>

<TextBox x:Name="txtUnitsInStock" Margin ="4,4,4,4" Grid.Column="2" Grid.Row="4" Text="" TextWrapping="Wrap" Grid.ColumnSpan="2" />

<Label x:Name="label4" Margin ="2,2,2,2" Content="Category" Grid.Column="0" Grid.Row="5" Grid.ColumnSpan="2"/>

<ComboBox x:Name="cboCategory" Margin ="4,4,4,4" Grid.Column="2" Grid.Row="5" Grid.ColumnSpan="2" />

<DataGrid x:Name="dgData" Margin ="4,4,4,63" Grid.Column="2" Grid.Row="6" Grid.ColumnSpan="2" SelectionChanged="dgData\_SelectionChanged" />

<Button x:Name="btnCreate" Content="Create" HorizontalAlignment="Left" Margin="29,365,0,16" Grid.Row="6" Grid.RowSpan="2" Width="121" Background="#FF0099FF" BorderBrush="White" Foreground="White" Grid.ColumnSpan="3" Grid.Column="1" Click="btnCreate\_Click"/>

<Button x:Name="btnUpdate" Content="Update" Grid.Column="3" HorizontalAlignment="Left" Margin="87,365,0,16" Grid.Row="6" Width="118" Background="#FF1B7140" Foreground="White" Grid.RowSpan="2" RenderTransformOrigin="0.37,0.2" Click="btnUpdate\_Click"/>

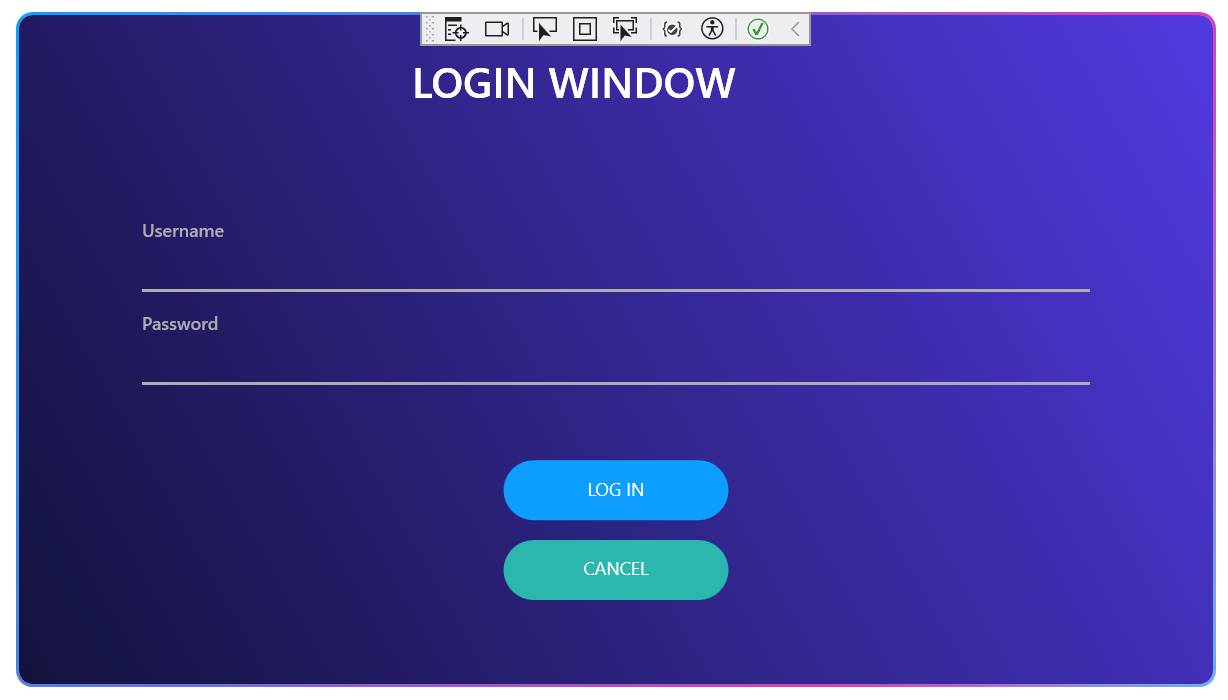
<Button x:Name="btnDelete" Content="Delete" Grid.Column="3" HorizontalAlignment="Left" Margin="221,365,0,16" Grid.Row="6" Grid.RowSpan="2" Width="127" Foreground="White" Background="#FFA2AA3D" Click="btnDelete\_Click"/>

<Button x:Name="btnClose" Content="Close" Grid.Column="3" HorizontalAlignment="Left" Margin="371,365,0,18" Grid.Row="6" Grid.RowSpan="2" Width="120" Background="#FFEF8F18" Foreground="White" Click="btnClose\_Click"/>

</Grid>

</Grid>

</Window>



<Window x:Class="WPFApp.LoginWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:WPFApp"

mc:Ignorable="d"

Title="LoginWindow" Height="450" Width="800"

WindowStartupLocation="CenterScreen"

WindowStyle="None"

Background="Transparent"

AllowsTransparency="True">

<Grid>

<Border CornerRadius="10"

BorderThickness="2"

Opacity="0.95">

<Border.BorderBrush>

<LinearGradientBrush StartPoint="0,0" EndPoint="1,1">

<GradientStop Color="#0099FF" Offset="0"/>

<GradientStop Color="#DA34AE" Offset="0.75"/>

<GradientStop Color="#FF60B1E7" Offset="1"/>

<GradientStop Color="#FF596FDD" Offset="0.309"/>

<GradientStop Color="#FF8C57CA" Offset="0.484"/>

</LinearGradientBrush>

</Border.BorderBrush>

<Border.Background>

<LinearGradientBrush StartPoint="0,1" EndPoint="1,0">

<GradientStop Color="#060531" Offset="0"/>

<GradientStop Color="#FF472FDE" Offset="1"/>

</LinearGradientBrush>

</Border.Background>

<Grid>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center"

Height="82" VerticalAlignment="Top" Width="632">

<TextBlock Text="LOGIN WINDOW"

Foreground="White"

FontSize="28"

FontWeight="Medium"

FontFamily="Montserrat"

Cursor="Hand"

Margin="180,30,0,0" Width="377"

/>

</StackPanel>

<StackPanel

Orientation="Vertical"

Margin="82,102,82,68">

<TextBlock Text="Username"

Foreground="DarkGray"

FontSize="12"

FontWeight="Medium"

FontFamily="Montserrat"

Margin="0,35,0,0"/>

<TextBox x:Name="txtUser"

FontSize="13"

FontWeight="Medium"

FontFamily="Montserrat"

Foreground="White"

CaretBrush="LightGray"

BorderBrush="DarkGray"

BorderThickness="0,0,0,2"

Height="28"

VerticalContentAlignment="Center"

Margin="0,5,0,0"

>

<TextBox.Background>

<LinearGradientBrush></LinearGradientBrush>

</TextBox.Background>

</TextBox>

<TextBlock Text="Password"

Foreground="DarkGray"

FontSize="12"

FontWeight="Medium"

FontFamily="Montserrat"

Margin="0,15,0,0"/>

<PasswordBox x:Name="txtPass"

FontSize="13"

FontWeight="Medium"

FontFamily="Montserrat"

Foreground="White"

CaretBrush="LightGray"

BorderBrush="DarkGray"

BorderThickness="0,0,0,2"

Height="28"

VerticalContentAlignment="Center"

Margin="0,5,0,0">

<PasswordBox.Background>

<LinearGradientBrush></LinearGradientBrush>

</PasswordBox.Background>

</PasswordBox>

<Button x:Name="btnLogin"

BorderThickness="0"

Content="LOG IN"

Foreground="White"

FontSize="12"

FontFamily="Montserrat"

Cursor="Hand"

Margin="0,50,0,0"

Click="btnLogin\_Click">

<Button.Style>

<Style TargetType="Button">

<Setter Property="Background" Value="#0099FF"/>

<Style.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter Property="Background" Value="#28AEED"/>

</Trigger>

</Style.Triggers>

</Style>

</Button.Style>

<Button.Template>

<ControlTemplate TargetType="Button">

<Border Width="150" Height="40"

CornerRadius="20"

Background="{TemplateBinding Background}">

<ContentPresenter VerticalAlignment="Center"

HorizontalAlignment="Center"/>

</Border>

</ControlTemplate>

</Button.Template>

</Button>

</StackPanel>

<StackPanel>

<Button x:Name="btnCancel"

BorderThickness="0"

Content="CANCEL"

Foreground="White"

FontSize="12"

FontFamily="Montserrat"

Cursor="Hand"

Margin="20,350,20,0"

Click="btnCancel\_Click">

<Button.Style>

<Style TargetType="Button">

<Setter Property="Background" Value="LightSeaGreen"/>

<Style.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter Property="Background" Value="SeaGreen"/>

</Trigger>

</Style.Triggers>

</Style>

</Button.Style>

<Button.Template>

<ControlTemplate TargetType="Button">

<Border Width="150" Height="40"

CornerRadius="20"

Background="{TemplateBinding Background}">

<ContentPresenter VerticalAlignment="Center"

HorizontalAlignment="Center"/>

</Border>

</ControlTemplate>

</Button.Template>

</Button>

</StackPanel>

</Grid>

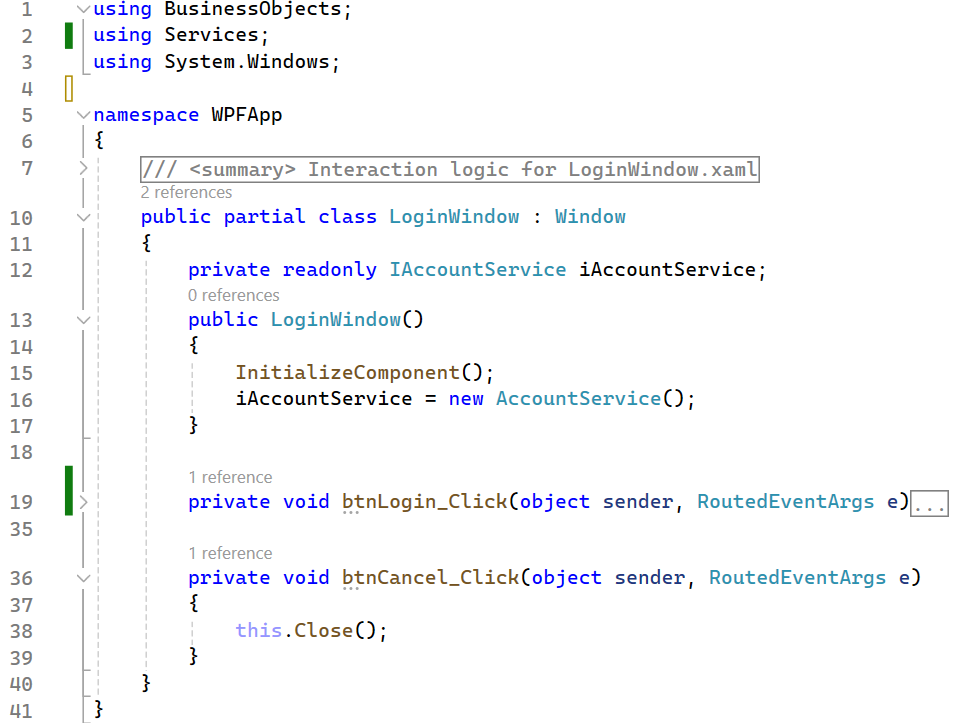
</Border>

</Grid>

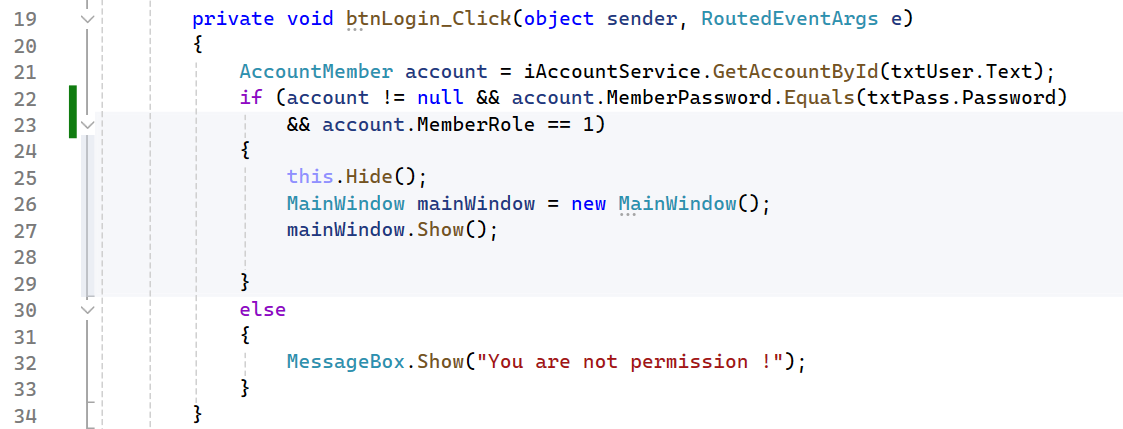
</Window>

**Step 03.** Add a reference to the WPF project to Services Project

## **Step 04.** Write codes for LoginWindow.xaml.cs:

****

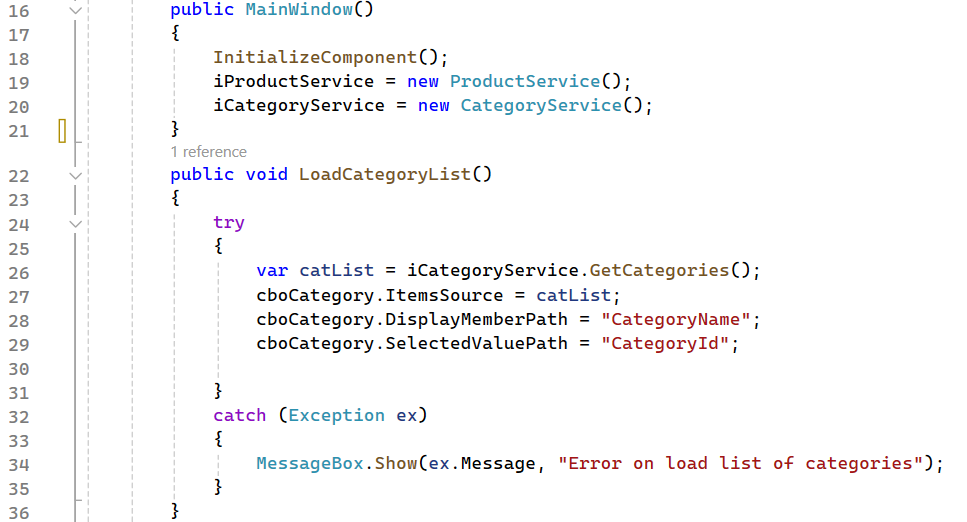
The details for btnLogin\_Click() function:

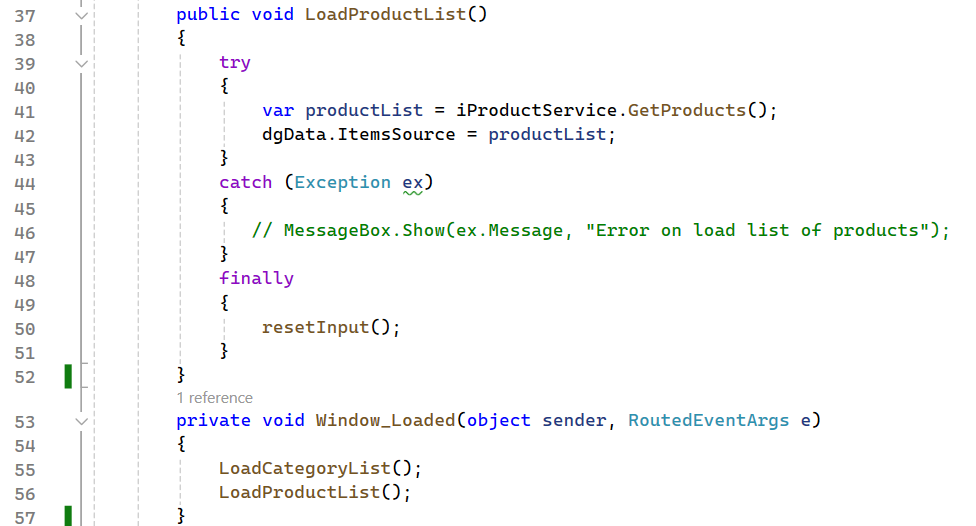


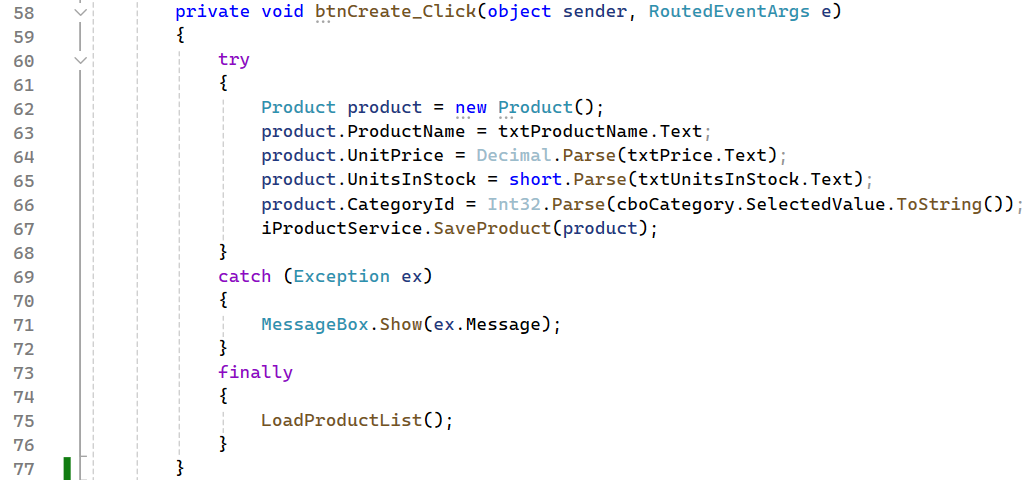
## **Step 05.** Write codes for MainWindow.xaml.cs:

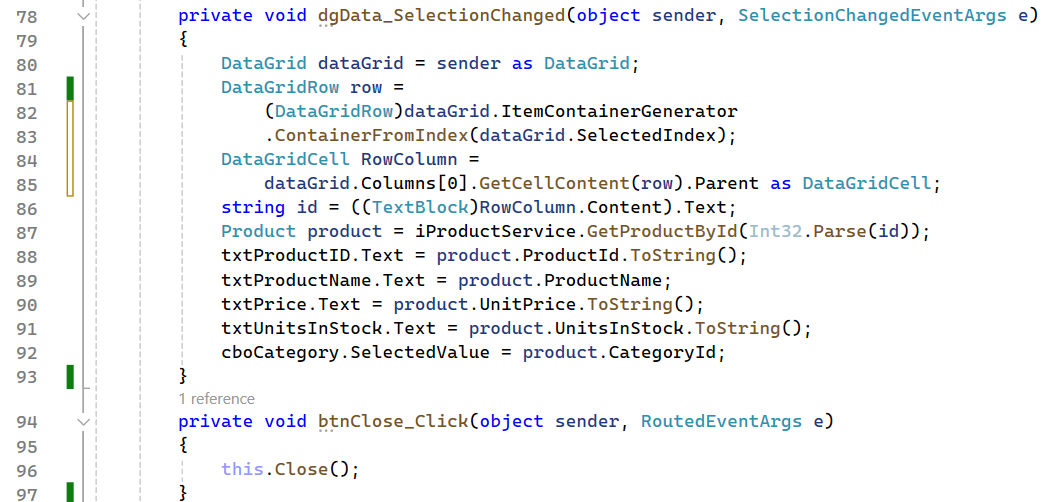
****

The functions in details:



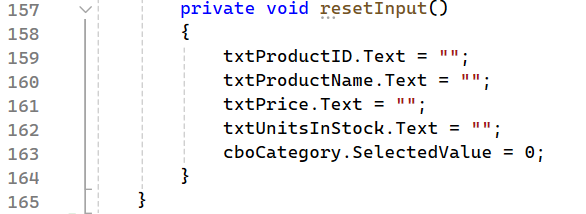












## **Step 06.** Open App.xaml and then update XAML code as follows:

<Application x:Class="WPFApp.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:WPFApp"

StartupUri="LoginWindow.xaml">

<Application.Resources>

</Application.Resources>

</Application>

# Activity 05: Run the WPFApp project and test all actions