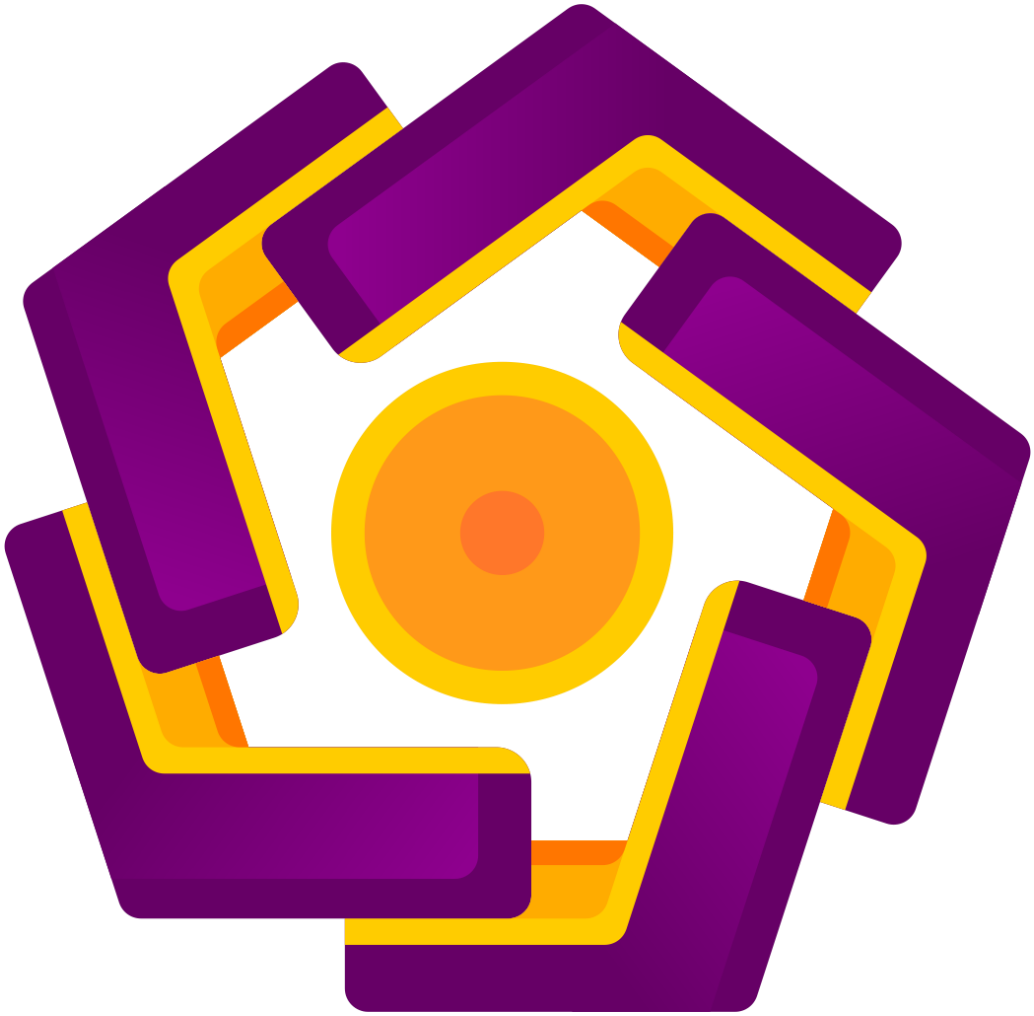
LAPORAN FINAL PROJECT UJIAN AKHIR

SEMESTER 3



Disusun Oleh **:**

MUHAMMAD ABDUL RAAFI (21.11.3867)

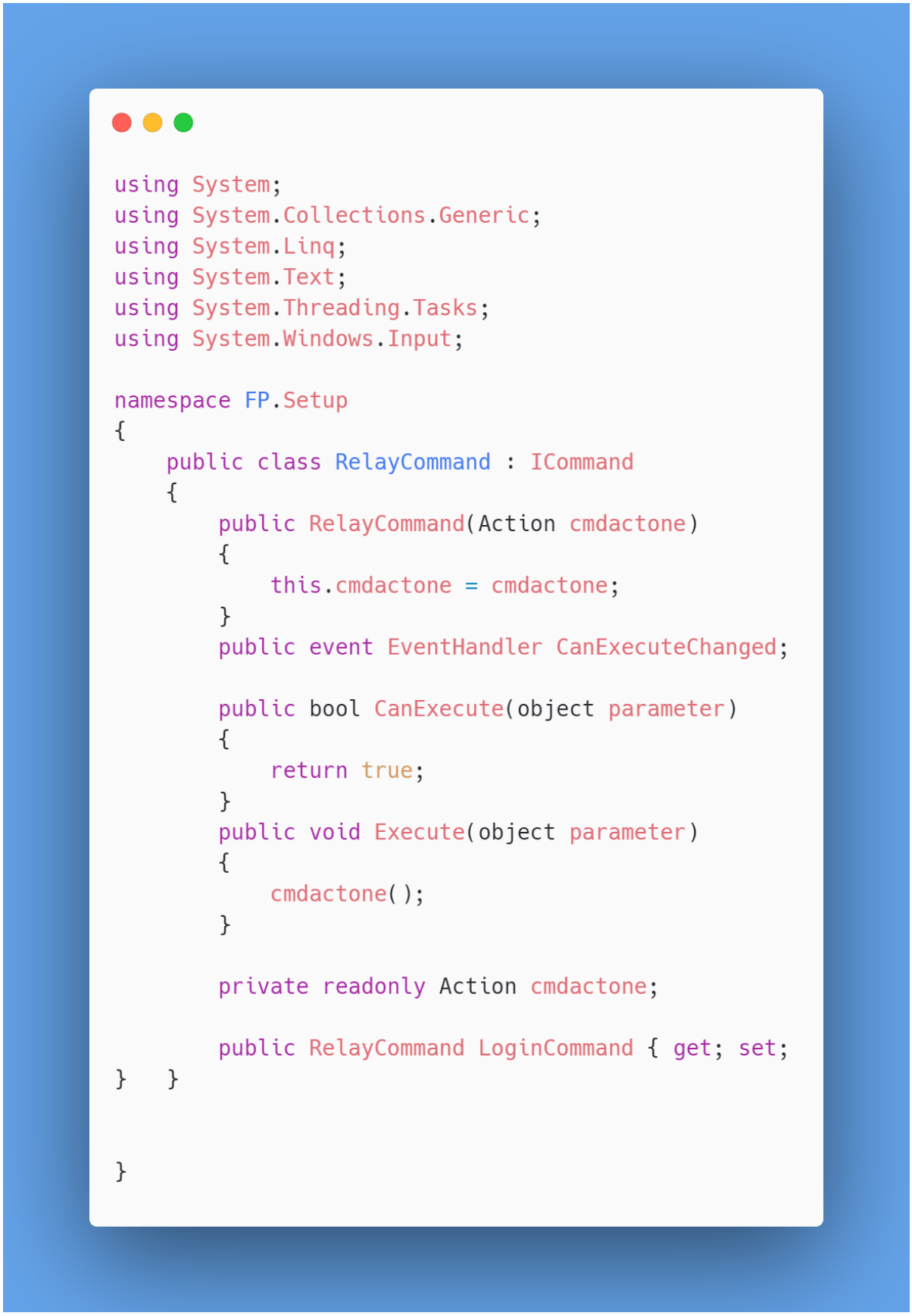
AHMAD MABRURI (21.11.3872)

A. Setup

1. Db_Connection

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.SqlClient;

namespace FP.Setup
{
    public class Db_Connection
    {
        public SqlConnection SqlConnection;
        public string SqlNotice;

        public Db_Connection()
        {
            SqlConnection = new SqlConnection();
            dbserver = @"LAPTOP-B8L8P8B9";
            dbname = "Retailer_DB";
        }

        public bool OpenConnection()
        {
            SqlConnection.ConnectionString =
                $"Server={dbserver};Database={dbname};Trusted_Connection=yes";
            SqlConnection.InfoMessage += Notice_Handler;
            SqlConnection.FireInfoMessageEventOnUserErrors = true;
            SqlConnection.Open();
            return true;

        }
        public bool CloseConnection()
        {
            SqlConnection.InfoMessage -= Notice_Handler;
            SqlConnection.Close();
            return true;
        }
        private readonly string dbserver;
        private readonly string dbname;

        private void Notice_Handler(object sender, SqlInfoMessageEventArgs e)
        {
            SqlNotice = e.Message;
        }
    }
}
```

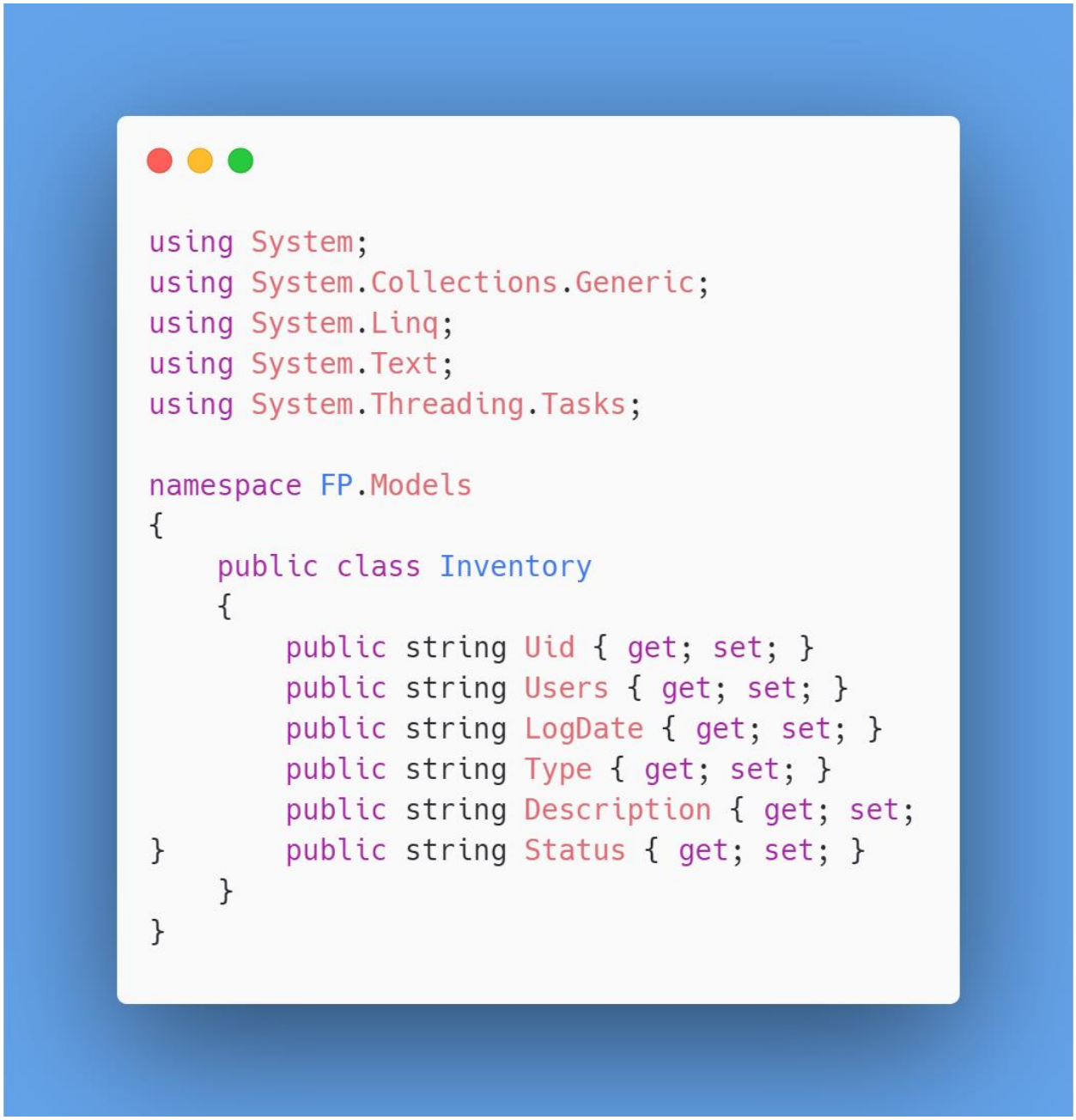
2. RelayCommand



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Input;

namespace FP.Setup
{
    public class RelayCommand : ICommand
    {
        public RelayCommand(Action cmdactone)
        {
            this.cmdactone = cmdactone;
        }
        public event EventHandler CanExecuteChanged;

        public bool CanExecute(object parameter)
        {
            return true;
        }
        public void Execute(object parameter)
        {
            cmdactone();
        }

        private readonly Action cmdactone;

        public RelayCommand LoginCommand { get; set;
}   }


}
```

B. Models

1. Inventory.cs

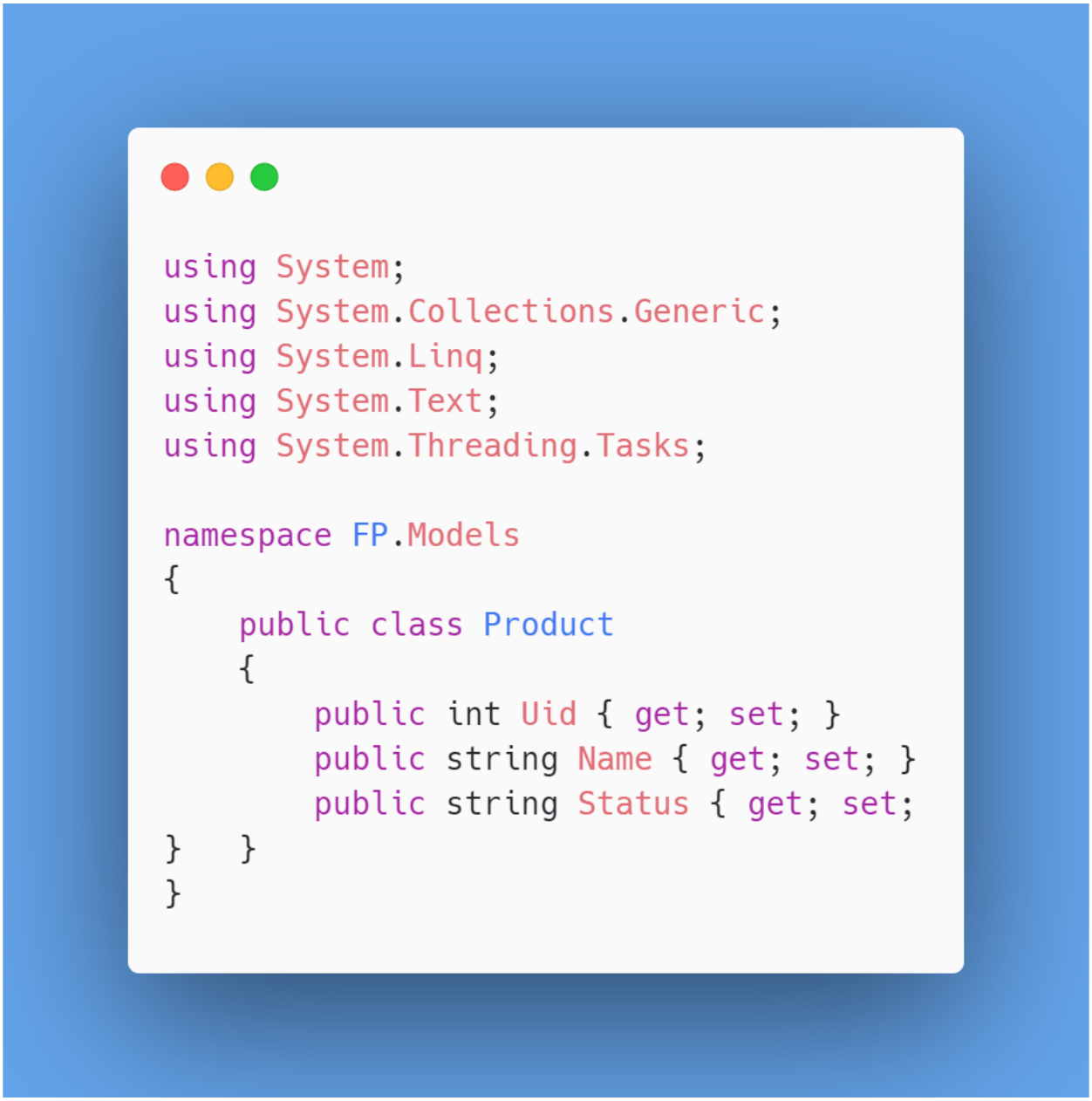


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FP.Models
{
    public class Inventory
    {
        public string Uid { get; set; }
        public string Users { get; set; }
        public string LogDate { get; set; }
        public string Type { get; set; }
        public string Description { get; set;
}       public string Status { get; set; }
    }
}
```

2. InventoryLog.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FP.Models
{
    class InventoryLog
    {
        public int Uid { get; set; }
        public string inventories { get; set;
}       public string Products { get; set; }
        public int Qty { get; set; }
    }
}
```

3. Product.cs

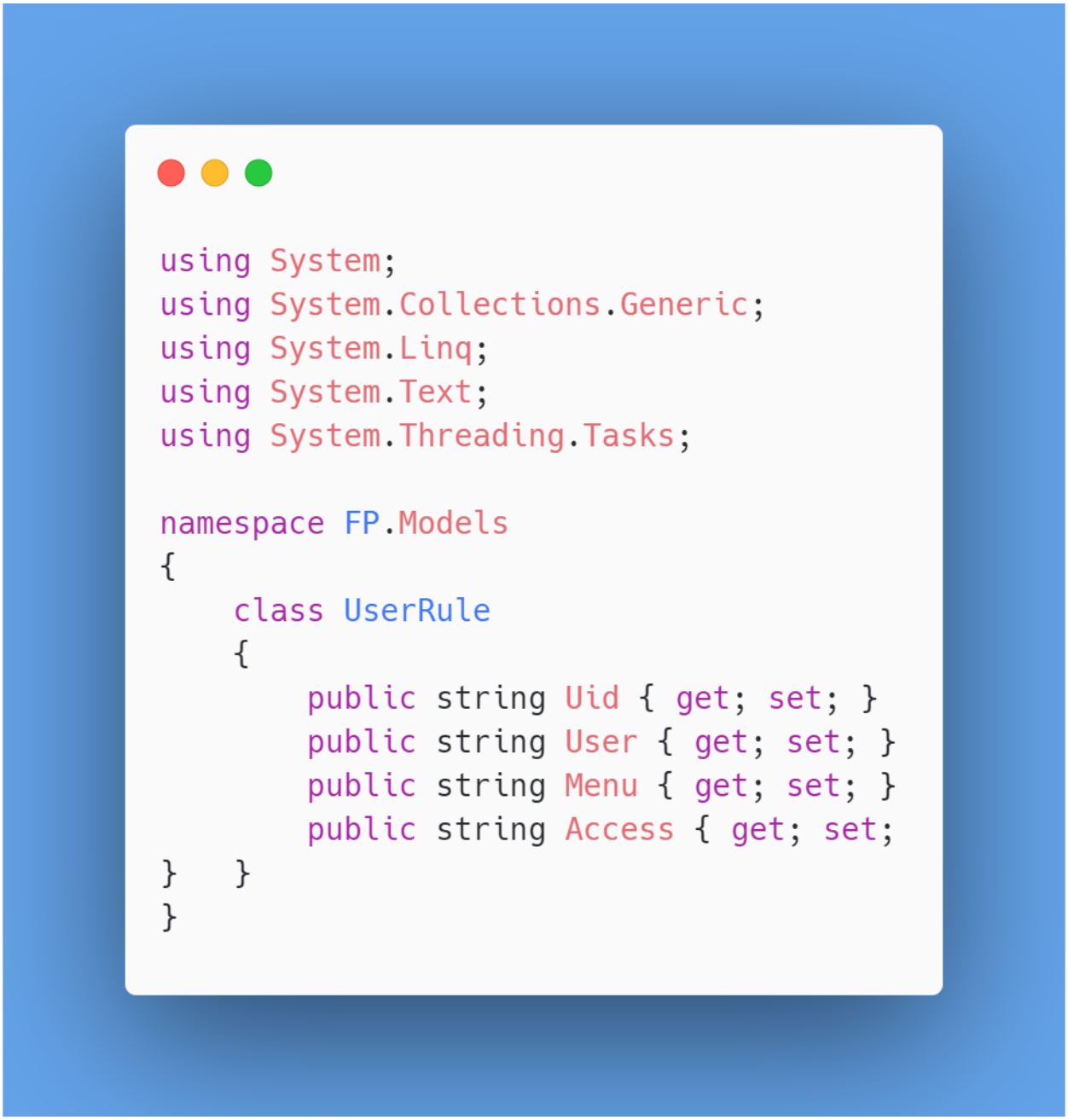


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FP.Models
{
    public class Product
    {
        public int Uid { get; set; }
        public string Name { get; set; }
        public string Status { get; set;
}   }
}
```

4. User.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FP.Models
{
    public class User
    {
        public int Uid { get; set; }
        public string Name { get; set; }
        public string UserName { get; set;
}       public string Keypass { get; set; }
        public string Status { get; set; }
    }
}
```

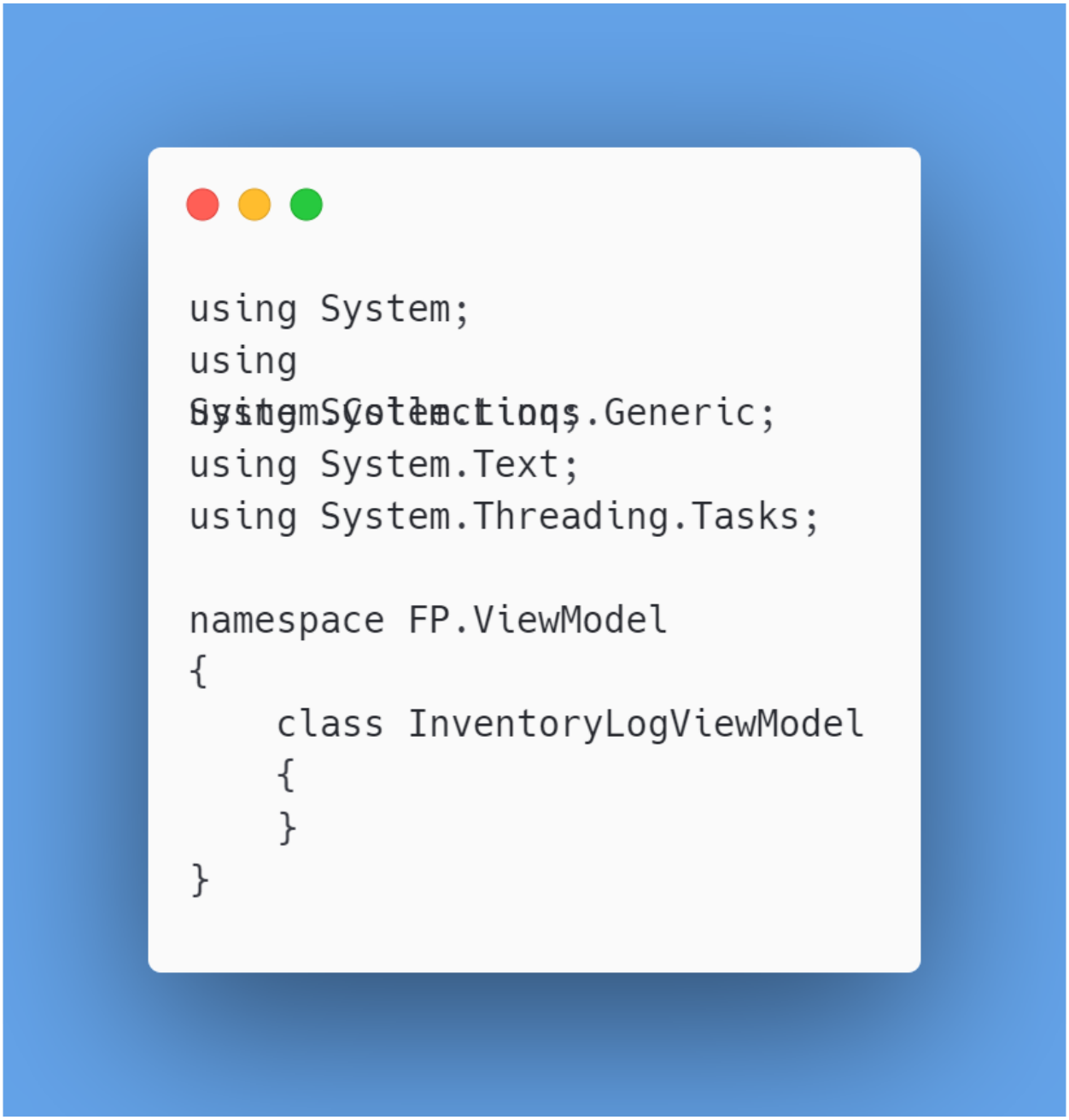
5. UserRule.cs



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FP.Models
{
    class UserRule
    {
        public string Uid { get; set; }
        public string User { get; set; }
        public string Menu { get; set; }
        public string Access { get; set;
}   }
}
```

C. ViewModel

1. InventoryLogView.cs



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FP.ViewModel
{
    class InventoryLogViewModel
    {
    }
}
```

2. InventoryViewModel.cs

```
using System;
using System.Threading.Tasks;
using System.ComponentModel;
using System.Collections.ObjectModel;
using System.Data.SqlClient;
using System.Windows;
using FP.Setup;
using FP.Models;


namespace FP.ViewModel
{
    public class InventoryViewModel : INotifyPropertyChanged
    {
        public InventoryViewModel()
        {
            collection = new ObservableCollection<Inventory>();
            dbconn = new Db_Connection();
            model = new Inventory();


            SelectCommand = new RelayCommand(async () => await ReadDataAsync());
            UpdateCommand = new RelayCommand(async () => await
UpdateDataAsync());
            DeleteCommand = new RelayCommand(async () => await
DeleteDataAsync());
            SelectCommand.Execute(null);

        }
        public RelayCommand SelectCommand { get; set; }
        public RelayCommand UpdateCommand { get; set; }
        public RelayCommand DeleteCommand { get; set; }
        public ObservableCollection<Inventory> Collection
        {
            get
            {
                return collection;
            }
            set
            {
                collection = value;
                PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));
            }

        }

        public Inventory Model
        {
            get
            {
                return model;
            }
            set
            {
                if (value != null)
                {
                    IsChecked = (value.Status == "Active") ? true : false;
                }
```

```
            model = value;
            PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));
        }
    }
    public bool IsChecked
    {
        get
        {
            return check;
        }
        set
        {
            check = value;
            PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));

        }
    }



    public event PropertyChangedEventHandler PropertyChanged;
    public event Action OnCallBack;

    private readonly Db_Connection dbconn;
    private ObservableCollection<Inventory> collection;
    private Inventory model;
    private bool check;

    private async Task ReadDataAsync()
    {
        dbconn.OpenConnection();

        await Task.Delay(0);
        var query = "SELECT * FROM vinventories";
        var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
        var sqlresult = sqlcmd.ExecuteReader();

        if (sqlresult.HasRows)
        {
            Collection.Clear();
            while (sqlresult.Read())
            {
                Collection.Add(new Inventory
                {
                    Uid = sqlresult[0].ToString(),
                    LogDate = sqlresult[1].ToString(),
                    Users = sqlresult[2].ToString(),
                    Type = sqlresult[2].ToString(),
                    Status = (sqlresult[3].ToString() == "1") ?
                    "Active" : "Not Active",
                });
            }
        }
        dbconn.CloseConnection();
        OnCallBack?.Invoke();
    }

    private async Task UpdateDataAsync()
    {
        if (isValidating())
        {
            dbconn.OpenConnection();
            var state = check ? "1" : "0";
            var query = $"UPDATE vinventories SET " +
```

```
                            $"Code = '{model.Uid}'," +
                            $"LogDate = '{model.LogDate}'," +
                            $"Users = '{model.Users}'," +
                            $"Type = '{model.Type}'," +
                            $"status = '{model.Status}'" +
                            $"WHERE Code = '{model.Uid}'";
                var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
                sqlcmd.ExecuteNonQuery();
                dbconn.CloseConnection();
                await ReadDataAsync();
            }
        }

        private async Task DeleteDataAsync()
        {
            if (isValidating())
            {
                var msg = MessageBox.Show("Are you sure?", "Question",
                          MessageBoxButton.YesNo, MessageBoxImage.Question);
                if (msg == MessageBoxResult.Yes)
                {
                    dbconn.OpenConnection();
                    var query = $"DELETE vinventory " +
                                $"WHERE Code = '{model.Uid}'";
                    var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
                    sqlcmd.ExecuteNonQuery();
                    dbconn.CloseConnection();
                }

                await ReadDataAsync();

            }
        }


        private bool isValidating()
        {
            var flag = true;

            return flag;
        }
    }
}
```

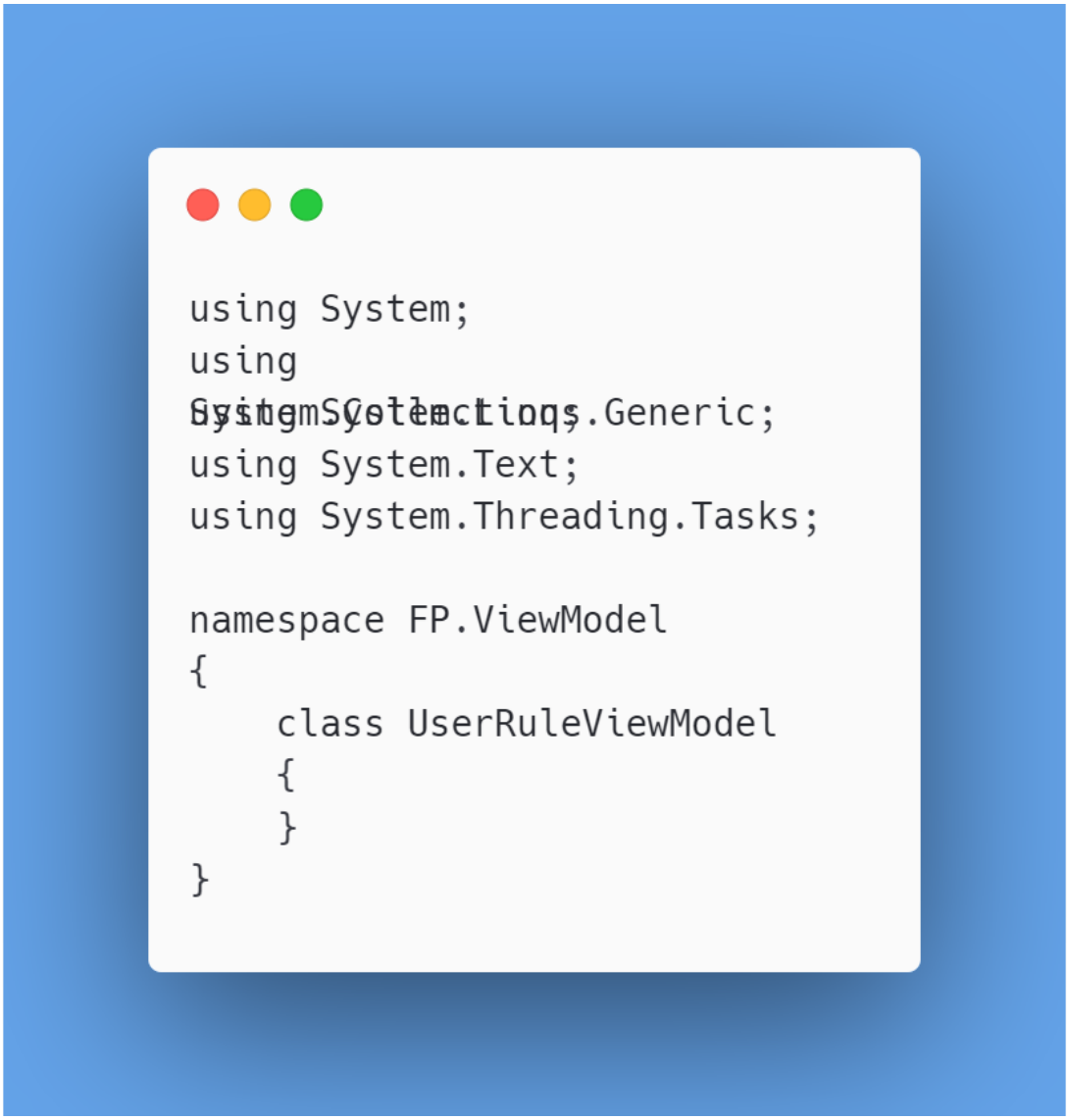
3. ProductViewModel.cs

```
using System;
using System.Threading.Tasks;
using System.ComponentModel;
using System.Collections.ObjectModel;
using System.Data.SqlClient;
using System.Windows;
using FP.Setup;
using FP.Models;
using FP.View.Inventories;


namespace FP.ViewModel
{
    public class ProductViewModel : INotifyPropertyChanged
    {
        public ProductViewModel()
        {
            collection = new ObservableCollection<Product>();
            dbconn = new Db_Connection();
            model = new Product();
            //tproduct = new ProductView();


            SelectCommand = new RelayCommand(async () => await ReadDataAsync());
            UpdateCommand = new RelayCommand(async () => await
UpdateDataAsync());
            DeleteCommand = new RelayCommand(async () => await
DeleteDataAsync());
            BackupCommand = new RelayCommand(async () => await Backupdata());
            SelectCommand.Execute(null);

        }
        public RelayCommand SelectCommand { get; set; }
        public RelayCommand UpdateCommand { get; set; }
        public RelayCommand DeleteCommand { get; set; }
        public RelayCommand BackupCommand { get; set; }
        public ObservableCollection<Product> Collection
        {
            get
            {
                return collection;
            }
            set
            {
                collection = value;
                PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));
            }

        }

        public Product Model
        {
            get
            {
                return model;
            }
```

```
            set
            {
                if (value != null)
                {
                    IsChecked = (value.Status == "Active") ? true : false;
                }
                model = value;
                PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));
            }
        }
        public bool IsChecked
        {
            get
            {
                return check;
            }
            set
            {
                check = value;
                PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));

            }
        }



        public event PropertyChangedEventHandler PropertyChanged;
        public event Action OnCallBack;

        //public ProductView tproduct;
        private readonly Db_Connection dbconn;
        private ObservableCollection<Product> collection;
        private Product model;
        private bool check;

        private async Task ReadDataAsync()
        {
            dbconn.OpenConnection();

            await Task.Delay(0);
            var query = "SELECT * FROM products";
            var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
            var sqlresult = sqlcmd.ExecuteReader();

            if (sqlresult.HasRows)
            {
                Collection.Clear();
                while (sqlresult.Read())
                {
                    Collection.Add(new Product
                    {
                        Uid = sqlresult[0] as int? ?? 0,
                        Name = sqlresult[1].ToString(),
                        Status = (sqlresult[2].ToString() == "1") ?
                        "Active" : "Not Active",
                    });
                }
            }
            dbconn.CloseConnection();
            OnCallBack?.Invoke();
        }

        private async Task UpdateDataAsync()
        {
            if (isValidating())
            {
                dbconn.OpenConnection();
                var state = check ? "1" : "0";
                var query = $"UPDATE products SET " +
```

```
                            $"name = '{model.Name}'," +
                            $"status = '{state}'" +
                            $"WHERE uid = '{model.Uid}'";
                var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
                sqlcmd.ExecuteNonQuery();
                dbconn.CloseConnection();
                await ReadDataAsync();
            }
        }

        private async Task Backupdata()
        {
            if (isValidating())
            {
                dbconn.OpenConnection();
                var query = $"EXECUTE sp_backupdatabase";
                var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
                sqlcmd.ExecuteNonQuery();
                dbconn.CloseConnection();
                await ReadDataAsync();
            }
        }

        private async Task DeleteDataAsync()
        {
            if (isValidating())
            {
                var msg = MessageBox.Show("Are you sure?", "Question",
                          MessageBoxButton.YesNo, MessageBoxImage.Question);
                if (msg == MessageBoxResult.Yes)
                {
                    dbconn.OpenConnection();
                    var query = $"DELETE products " +
                                $"WHERE uid = '{model.Uid}'";
                    var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
                    sqlcmd.ExecuteNonQuery();
                    dbconn.CloseConnection();
                }
                await ReadDataAsync();
            }
        }


        private bool isValidating()
        {
            var flag = true;
            if (Model.Name == null)
            {
                MessageBox.Show("Name Cannot Empty", "Warning",
                    MessageBoxButton.OK,
                    MessageBoxImage.Exclamation);
            }
            return flag;
        }
    }
}
```

4. UserRuleViewModel.cs



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FP.ViewModel
{
    class UserRuleViewModel
    {
    }
}
```

5. UserViewModel.cs

```
using System;
using System.Threading.Tasks;
using System.ComponentModel;
using System.Collections.ObjectModel;
using System.Data.SqlClient;
using System.Windows;
using FP.Setup;
using FP.Models;


namespace FP.ViewModel
{
    public class UserViewModel : INotifyPropertyChanged
    {
        public UserViewModel()
        {
            collection = new ObservableCollection<User>();
            dbconn = new Db_Connection();
            model = new User();


            SelectCommand = new RelayCommand(async () => await ReadDataAsync());
            UpdateCommand = new RelayCommand(async () => await UpdateDataAsync());
            DeleteCommand = new RelayCommand(async () => await DeleteDataAsync());

            SelectCommand.Execute(null);

        }
        public RelayCommand SelectCommand { get; set; }
        public RelayCommand UpdateCommand { get; set; }
        public RelayCommand DeleteCommand { get; set; }

        public ObservableCollection<User> Collection
        {
            get
            {
                return collection;
            }
            set
            {
                collection = value;
                PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));
            }

        }

        public User Model
        {
            get
            {
                return model;
            }
            set
            {
                if (value != null)
                {
                    IsChecked = (value.Status == "Active") ? true : false;
                }
                model = value;
                PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));
            }
        }
        public bool IsChecked
        {
            get
            {
                return check;
            }
            set
            {
                check = value;
                PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(null));

            }
        }
```

```
        public event PropertyChangedEventHandler PropertyChanged;
        public event Action OnCallBack;

        private readonly Db_Connection dbconn;
        private ObservableCollection<User> collection;
        private User model;
        private bool check;

        private async Task ReadDataAsync()
        {
            dbconn.OpenConnection();

            await Task.Delay(0);
            var query = "SELECT * FROM Users";
            var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
            var sqlresult = sqlcmd.ExecuteReader();

            if (sqlresult.HasRows)
            {
                Collection.Clear();
                while (sqlresult.Read())
                {
                    Collection.Add(new User
                    {
                        Uid = sqlresult[0] as int? ?? 0,
                        Name = sqlresult[1].ToString(),
                        UserName = sqlresult[2].ToString(),
                        Keypass = sqlresult[3].ToString(),
                        Status = (sqlresult[4].ToString() == "1") ?
                        "Active" : "Not Active",
                    });
                }
            }


            dbconn.CloseConnection();
            OnCallBack?.Invoke();
        }

        public bool Login()
        {
            bool status;
            if (Collection.Count > 0)
            {
                Collection.Clear();
            }
            dbconn.OpenConnection();
            var query = $"select * from Users where username='{model.UserName}' and
keypass='{model.Keypass}'";
            var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
            var sqlresult = sqlcmd.ExecuteReader();
            if (sqlresult.HasRows)
            {
                status = true;
            }
            else
            {
                status = false;
            }
            dbconn.CloseConnection();
            return status;
        }
        private async Task UpdateDataAsync()
        {
            if (isValidating())
            {
                dbconn.OpenConnection();
                var state = check ? "1" : "0";
                var query = $"UPDATE users SET " +
                            $"name = '{model.Name}'," +
                            $"username = '{model.UserName}'," +
                            $"keypass = '{model.Keypass}'," +
                            $"status = '{state}'" +
                            $"WHERE uid = '{model.Uid}'";
                var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
                sqlcmd.ExecuteNonQuery();
                dbconn.CloseConnection();
                await ReadDataAsync();
            }
        }
```

```
        private async Task DeleteDataAsync()
        {
            if (isValidating())
            {
                var msg = MessageBox.Show("Are you sure?", "Question",
                          MessageBoxButton.YesNo, MessageBoxImage.Question);
                if (msg == MessageBoxResult.Yes)
                {
                    dbconn.OpenConnection();
                    var query = $"DELETE users " +
                                $"WHERE uid = '{model.Uid}'";
                    var sqlcmd = new SqlCommand(query, dbconn.SqlConnection);
                    sqlcmd.ExecuteNonQuery();
                    dbconn.CloseConnection();
                }

                await ReadDataAsync();

            }
        }


        private bool isValidating()
        {
            var flag = true;
            if (Model.Name == null)
            {
                MessageBox.Show("Name Cannot Empty", "Warning",
                    MessageBoxButton.OK,
                    MessageBoxImage.Exclamation);
            }
            return flag;
        }


    }
}
```

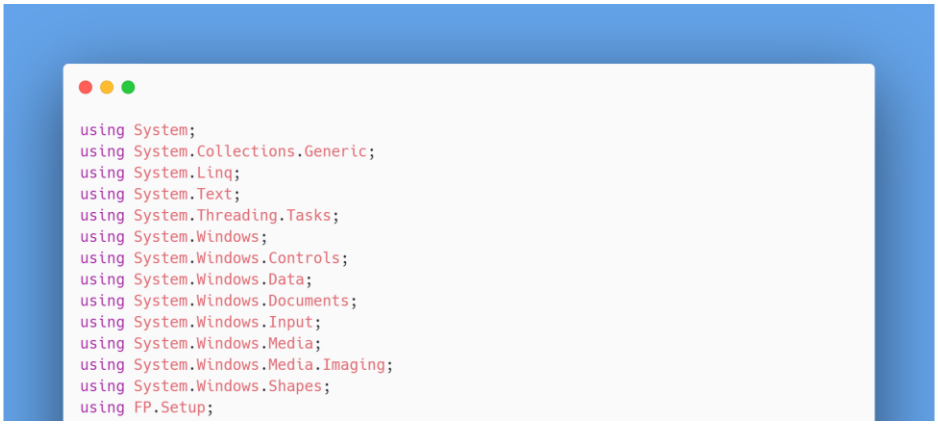
D. View

- **Home**

1. Dashboard.xaml

```
<Window x:Class="FP.View.Home.Dashboard"
        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:FP.View.Home"
        mc:Ignorable="d"
        Title="Dashboard" Height="450" Width="800">
    <Window.Background>
        <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BG1.jpeg"></ImageBrush>
    </Window.Background>
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="24"/>
            <RowDefinition Height="*"/>
        </Grid.RowDefinitions>
        <Menu Grid.Row="0">
            <MenuItem Header="_Home">
                <MenuItem Header="Relogin" x:Name="MnuRelogin" Click="MnuRelogin_Click"/>
                <Separator/>
                <MenuItem Header="Task">
                    <MenuItem Header="Backup Database" Command="{Binding BackupCommand}"
x:Name="MnuBackup" Click="MnuBackup_Click"/>
                </MenuItem>
                <MenuItem Header="Users" x:Name="MnuUsers" Click="MnuUsers_Click"/>
                <Separator/>
                <MenuItem Header="Exit" x:Name="MnuExit" Click="MnuExit_Click"/>
            </MenuItem>
            <MenuItem Header="_Inventories">
                <MenuItem Header="Inventory" x:Name="MnuInventory" Click="MnuInventory_Click"/>
                <MenuItem Header="Stock" x:Name="MnuStock" Click="MnuStock_Click"/>
                <Separator/>
                <MenuItem Header="Product" x:Name="MnuProduct" Click="MnuProduct_Click_1"/>
            </MenuItem>
        </Menu>
        <DockPanel x:Name="PnlContent" Grid.Row="1"/>
    </Grid>
</Window>
```

➢ Dashboard.xaml.cs



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using FP.Setup;
```

```
namespace FP.View.Home
{
    /// <summary>
    /// Interaction logic for Dashboard.xaml
    /// </summary>
    public partial class Dashboard : Window
    {
        public Dashboard()
        {
            InitializeComponent();
        }


        private void MnuUsers_Click(object sender, RoutedEventArgs e)
        {
            Home.UserView user = new Home.UserView();
            user.Show();
            this.Close();

        }

        private void MnuBackup_Click(object sender, RoutedEventArgs e)
        {

        }

        private void MnuInventory_Click(object sender, RoutedEventArgs e)
        {
            Inventories.InventoriesView inventories = new Inventories.InventoriesView();
            inventories.Show();
            this.Close();
        }

        private void MnuStock_Click(object sender, RoutedEventArgs e)
        {
            Inventories.InventoriesView inventories = new Inventories.InventoriesView();
            inventories.Show();
            this.Close();
        }

        private void MnuProduct_Click(object sender, RoutedEventArgs e)
        {
            Inventories.ProductView product = new Inventories.ProductView();
            product.Show();
            this.Close();
        }

        private void MnuRelogin_Click(object sender, RoutedEventArgs e)
        {
            Home.LoginDialog login = new Home.LoginDialog();
            login.Show();
            this.Close();
        }

        private void MnuExit_Click(object sender, RoutedEventArgs e)
        {
            this.Close();
        }

        private void MnuProduct_Click_1(object sender, RoutedEventArgs e)
        {
            Inventories.ProductView product = new Inventories.ProductView();
            product.Show();
            this.Close();
        }
    }
}
```

2. LoginDialog.xaml

```
<Window x:Class="FP.View.Home.LoginDialog"
        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:FP.View.Home"
        mc:Ignorable="d"
        Title="Form Login" Height="300" Width="500">
    <Window.Background>
        <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BG1.jpeg"></ImageBrush>
    </Window.Background>
    <Grid>
        <Label Content="User ID" HorizontalAlignment="Left" Height="29" Margin="70,63,0,0"
VerticalAlignment="Top" Width="68"/>
        <Label Content="Password" HorizontalAlignment="Left" Height="29" Margin="70,97,0,0"
VerticalAlignment="Top" Width="68"/>
        <Button Content="" HorizontalAlignment="Left" Height="45" Margin="70,150,0,0"
VerticalAlignment="Top" Width="131" Click="Button_Click_1">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\button_login.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="btnClose" Click="btnClose_Click" Content="" HorizontalAlignment="Left"
Height="45" Margin="271,150,0,0" VerticalAlignment="Top" Width="131">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_CLOSE.png"/>
            </Button.Background>
        </Button>
        <TextBox x:Name="TxtUser" HorizontalAlignment="Left" Height="29" Margin="201,63,0,0"
TextWrapping="Wrap" VerticalAlignment="Top" Width="201" TextChanged="TextBox_TextChanged"/>
        <PasswordBox x:Name="TxtPassword" PasswordChanged="TxtPassword_PasswordChanged"
HorizontalAlignment="Left" Height="29" Margin="201,97,0,0" VerticalAlignment="Top" Width="201"/>
        <TextBlock x:Name="Status" HorizontalAlignment="Left" Margin="165,200,0,0" TextWrapping="Wrap"
Text="Username Atau Password Salah!" VerticalAlignment="Top" FontWeight="Bold" Background="Red"
Visibility="Hidden"/>

    </Grid>
</Window>
```

- LoginDialog.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using FP.ViewModel;
using FP.Models;
```

```
namespace FP.View.Home
{
    /// <summary>
    /// Interaction logic for LoginDialog.xaml
    /// </summary>
    public partial class LoginDialog : Window
    {
        public string username;
        public string password;
        private Dashboard Dashboard;

        public LoginDialog()
        {
            vm = new UserViewModel();
            InitializeComponent();
            vm.OnCallBack += Close;
            DataContext = vm;
        }

        private readonly UserViewModel vm;

        //public RelayCommand LoginCommand { get; set; }
        // LoginCommand = new RelayCommand(async () => await LoginDataAsync());

        private void TextBox_TextChanged(object sender, TextChangedEventArgs e)
        {

        }

        private void Button_Click_1(object sender, RoutedEventArgs e)
        {
            vm.Model = new User
            {
                UserName = TxtUser.Text,
                Keypass = TxtPassword.Password
            };
            if (vm.Login())
            {
                Dashboard = new Dashboard();
                Dashboard.Show();
                this.Close();
            }
            else
            {
                Status.Visibility = Visibility.Visible;
            }
        }

        private void btnClose_Click(object sender, RoutedEventArgs e)
        {
            Home.Dashboard dashboard = new Home.Dashboard();
            dashboard.Show();
            this.Close();
        }

        private void TxtPassword_PasswordChanged(object sender, RoutedEventArgs e)
        {

        }
    }
}
```

3. RulesDialog.xaml

```xaml
<Window x:Class="FP.View.Home.RulesDialog"
        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:FP.View.Home"
        mc:Ignorable="d"
        Title="RulesDialog" Height="450" Width="800">
    <Grid>

    </Grid>
</Window>
```

➢ RulesDialog.xaml.cs

```cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace FP.View.Home
{
    /// <summary>
    /// Interaction logic for
RulesDialog.xaml
    /// </summary>
    public partial class RulesDialog : Window
    {
        public RulesDialog()
        {
            InitializeComponent();
        }
    }
}
```

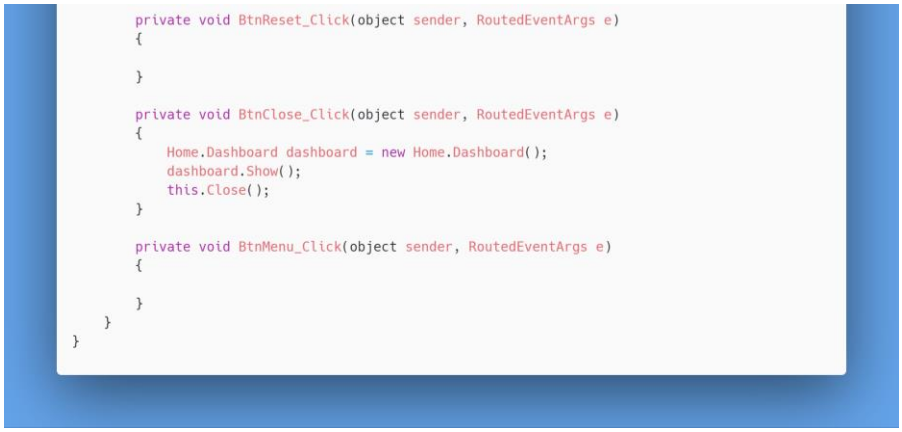
4. UserView.xaml

```
<Window x:Class="FP.View.Home.UserView"
        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:FP.View.Home"
        mc:Ignorable="d"
        Title="UserView" Height="450" Width="800">
    <Window.Background>
        <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BG1.jpeg"></ImageBrush>
    </Window.Background>
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
            <ColumnDefinition Width="200"/>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition/>
            <RowDefinition Height="40"/>
            <RowDefinition Height="76*"/>
            <RowDefinition Height="123"/>
            <RowDefinition Height="80"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>
        <Grid Grid.RowSpan="5" Grid.ColumnSpan="2">
            <TextBlock TextWrapping="Wrap" Margin="10,10,498,383" FontSize="16" FontWeight="Bold"><Run
Text="USER"/><LineBreak/><Run/></TextBlock>
        </Grid>
        <DataGrid x:Name="TblUser" Grid.Row="2" Grid.Column="0" Margin="10,3,8,0"
                  HeadersVisibility="Column" EnableRowVirtualization="False" CanUserAddRows="False"
CanUserDeleteRows="False" AutoGenerateColumns="False"
                  AlternatingRowBackground="AliceBlue"
SelectedCellsChanged="TblUser_SelectedCellsChanged"
                  ItemsSource="{Binding Collection}" SelectedItem="{Binding Model}"
SelectionChanged="TblUser_SelectionChanged">
            <DataGrid.Columns>
                <DataGridTextColumn Header="CODE" Binding="{Binding Uid}" Width="auto"/>
                <DataGridTextColumn Header="NAME" Binding="{Binding Name}" Width="*"/>
                <DataGridTextColumn Header="USER" Binding="{Binding UserName}" Width="*"/>
                <DataGridTextColumn Header="STATUS" Binding="{Binding Status}" Width="auto"/>
            </DataGrid.Columns>

        </DataGrid>
        <TextBox x:Name="TxtName" Text="{Binding Model.Name, Mode=TwoWay}" TextWrapping="Wrap"
Margin="254,20,8,86" Grid.Row="3"/>
        <TextBox x:Name="TxtUser" Text="{Binding Model.UserName, Mode=TwoWay}" TextWrapping="Wrap"
Margin="254,57,179,49" Grid.Row="3"/>
        <Label Content="Name" Margin="188,15,351,80" Grid.Row="3"/>
        <Label Content="UserId" Margin="188,52,351,43" Grid.Row="3"/>
        <Label Content="Status" Margin="196,0,343,52" Grid.Row="4" RenderTransformOrigin="0.384,-1.211"/>
        <CheckBox x:Name="IsChecked" IsChecked="{Binding IsChecked, Mode=TwoWay}" Content="Active"
Margin="254,10,266,54" Grid.Row="4"/>
        <Button x:Name="BtnUpdate" Click="BtnUpdate_Click" Content="UPDATE" Grid.Column="2" Command="
{Binding UpdateCommand}" HorizontalAlignment="Left" Margin="10,77,0,0" Grid.Row="2"
VerticalAlignment="Top" Width="180" Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_UPDATE.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnEdit" Click="BtnEdit_Click" Content="EDIT" Grid.Column="2"
HorizontalAlignment="Left" Margin="10,10,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_EDIT.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnDelete" Click="BtnDelete_Click" Content="DELETE" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,145,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" Command="{Binding DeleteCommand}" Grid.RowSpan="2">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_DELETE.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnMenu" Click="BtnMenu_Click" Content="MENU" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,41,0,0" Grid.Row="3" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_MENU.png"/>
            </Button.Background>
        </Button>
```

```
        <Button x:Name="BtnClose" Click="BtnClose_Click" Content="CLOSE" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,110,0,0" Grid.Row="3" VerticalAlignment="Top" Width="180"
Height="47" Grid.RowSpan="2">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_CLOSE.png"/>
            </Button.Background>
        </Button>
        <TextBox x:Name="TxtPassword" Text="{Binding Model.Keypass, Mode=TwoWay}" TextWrapping="Wrap"
Margin="254,96,8,10" Grid.Row="3"/>
        <Label Content="Password" Margin="174,92,351,3" Grid.Row="3"/>
        <Label Content="Code" Margin="441,52,98,43" Grid.Row="3"/>
        <TextBox x:Name="TxtCode" Text="{Binding Model.Uid, Mode=TwoWay}" TextWrapping="Wrap"
Margin="499,57,8,49" Grid.Row="3"/>

    </Grid>
</Window>
```

- UserView.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using FP.ViewModel;

namespace FP.View.Home
{
    /// <summary>
    /// Interaction logic for UserView.xaml
    /// </summary>
    public partial class UserView : Window
    {
        public UserView()
        {
            InitializeComponent();
            vm = new UserViewModel();
            vm.OnCallBack += Close;
            DataContext = vm;
        }

        private readonly UserViewModel vm;

        private void TblData_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {

        }

        private void TblUser_SelectedCellsChanged(object sender, SelectedCellsChangedEventArgs
e)      {

        }

        private void TblUser_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {

        }

        private void BtnUpdate_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnEdit_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnDelete_Click(object sender, RoutedEventArgs e)
        {

        }
```



```
        private void BtnReset_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnClose_Click(object sender, RoutedEventArgs e)
        {
            Home.Dashboard dashboard = new Home.Dashboard();
            dashboard.Show();
            this.Close();
        }

        private void BtnMenu_Click(object sender, RoutedEventArgs e)
        {

        }
    }
}
```

- **Inventories**

1. EditorView.xaml

```
<Window x:Class="FP.View.Inventories.EditorView"
        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:FP.View.Inventories"
        mc:Ignorable="d"
        Title="EditorView" Height="450" Width="800">
    <Grid>

    </Grid>
</Window>
```

➢ EditorView.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace FP.View.Inventories
{
    /// <summary>
    /// Interaction logic for
EditorView.xaml
    /// </summary>
    public partial class EditorView : Window
```

```
    public partial class EditorView : Window
    {
        public EditorView()
        {
            InitializeComponent();
        }
    }
}
```

2. InventoriesView.xaml

```
<Window x:Class="FP.View.Inventories.InventoriesView"
        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:FP.View.Inventories"
        mc:Ignorable="d"
        Title="InventoriesView" Height="450" Width="800">
    <Window.Background>
        <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BG1.jpeg"></ImageBrush>
    </Window.Background>
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
            <ColumnDefinition Width="200"/>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition/>
            <RowDefinition Height="40"/>
            <RowDefinition Height="76*"/>
            <RowDefinition Height="123"/>
            <RowDefinition Height="80"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>
        <Grid Grid.RowSpan="5" Grid.ColumnSpan="2">
            <TextBlock TextWrapping="Wrap" Margin="10,10,498,358" FontSize="16" FontWeight="Bold"
Text="DATA INVENTORIES"/>
            <ComboBox HorizontalAlignment="Left" Margin="57,48,0,0" VerticalAlignment="Top" Width="120">
                <ComboBoxItem Name="_1">Guest</ComboBoxItem>
                <ComboBoxItem Name="_2">Super Admin</ComboBoxItem>
            </ComboBox>
            <Label Content="Type" HorizontalAlignment="Left" Margin="17,45,0,0" VerticalAlignment="Top"
Height="28"/>
            <DatePicker HorizontalAlignment="Left" Height="26" Margin="277,46,0,0"
VerticalAlignment="Top" Width="153"/>
            <DatePicker HorizontalAlignment="Left" Height="26" Margin="455,46,0,0"
VerticalAlignment="Top" Width="116"/>
            <Label Content="/" HorizontalAlignment="Left" Margin="435,45,0,0" VerticalAlignment="Top"
Height="28" RenderTransformOrigin="2.711,0.619"/>
            <Label Content="Show Date" HorizontalAlignment="Left" Margin="191,48,0,0"
VerticalAlignment="Top" Height="28"/>
        </Grid>
        <DataGrid x:Name="TblDataInv" Grid.Row="2" Grid.Column="0" Margin="18,56,0,50"
                  HeadersVisibility="Column" EnableRowVirtualization="False" CanUserAddRows="False"
CanUserDeleteRows="False" AutoGenerateColumns="False"
                  AlternatingRowBackground="AliceBlue"
SelectedCellsChanged="TblDataInv_SelectedCellsChanged_1"
                  ItemsSource="{Binding Collection}" SelectedItem="{Binding Model}"
SelectionChanged="TblDataInv_SelectionChanged_1" Grid.RowSpan="3">
```

```
            <DataGrid.Columns>
                <DataGridTextColumn Header="CODE" Binding="{Binding Uid}" Width="auto"/>
                <DataGridTextColumn Header="DATE" Binding="{Binding LogDate}" Width="*"/>
                <DataGridTextColumn Header="USER" Binding="{Binding Users}" Width="*"/>
                <DataGridTextColumn Header="TYPE" Binding="{Binding Type}" Width="auto"/>
                <DataGridTextColumn Header="STATUS" Binding="{Binding Status}" Width="auto"/>
            </DataGrid.Columns>

        </DataGrid>
        <Button x:Name="BtnUpdate" Click="BtnUpdate_Click" Content="SHOW" Grid.Column="2"
HorizontalAlignment="Left" Margin="10,10,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_SHOW.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnEdit" Click="BtnEdit_Click" Content="NEW" Grid.Column="2"
HorizontalAlignment="Left" Margin="10,77,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_NEW.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnDelete" Click="BtnDelete_Click" Content="EDIT" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,145,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" Command="{Binding DeleteCommand}" Grid.RowSpan="2">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_EDIT.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnReset" Click="BtnReset_Click" Content="RESET" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,41,0,0" Grid.Row="3" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_RESET.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnClose" Click="BtnClose_Click" Content="CLOSE" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,110,0,0" Grid.Row="3" VerticalAlignment="Top" Width="180"
Height="47" Grid.RowSpan="2">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_CLOSE.png"/>
            </Button.Background>
        </Button>

    </Grid>
</Window>
```

- InventoriesView.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using FP.ViewModel;
namespace FP.View.Inventories
{
    /// <summary>
    /// Interaction logic for InventoriesView.xaml
    /// </summary>
    public partial class InventoriesView : Window
    {
```

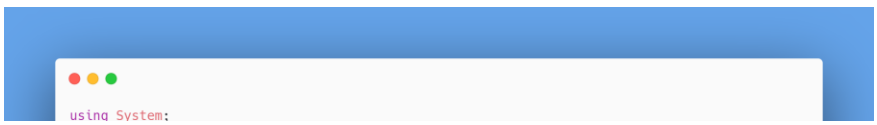
```
        public InventoriesView()
        {
            InitializeComponent();
            vm = new InventoryViewModel();
            vm.OnCallBack += Close;
            DataContext = vm;
        }
        private readonly InventoryViewModel vm;
        private void TblData_SelectedCellsChanged(object sender, SelectedCellsChangedEventArgs e)
        {

        }

        private void TblData_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {

        }

        private void BtnUpdate_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnEdit_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnDelete_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnReset_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnClose_Click(object sender, RoutedEventArgs e)
        {
            Home.Dashboard dashboard = new Home.Dashboard();
            dashboard.Show();
            this.Close();
        }

        private void TblDataInv_SelectedCellsChanged(object sender, SelectedCellsChangedEventArgs e)
        {

        }

        private void TblDataInv_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {

        }

        private void TblDataInv_SelectedCellsChanged_1(object sender, SelectedCellsChangedEventArgs e)
        {

        }

        private void TblDataInv_SelectionChanged_1(object sender, SelectionChangedEventArgs e)
        {

        }
    }
}
```

3. ProductView.xaml

```
<Window x:Class="FP.View.Inventories.ProductView"
        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:FP.View.Inventories"
        mc:Ignorable="d"
        Title="ProductView" Height="450" Width="800">
    <Window.Background>
        <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BG1.jpeg"></ImageBrush>
    </Window.Background>
    <Grid>
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
```

```
            <ColumnDefinition Width="200"/>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition/>
            <RowDefinition Height="40"/>
            <RowDefinition Height="76*"/>
            <RowDefinition Height="123"/>
            <RowDefinition Height="80"/>
            <RowDefinition Height="Auto"/>
        </Grid.RowDefinitions>
        <Grid Grid.RowSpan="5" Grid.ColumnSpan="2">
            <TextBlock TextWrapping="Wrap" Margin="10,10,498,344" FontSize="16" FontWeight="Bold"><Run
Text="Product"/><LineBreak/><Run/></TextBlock>
        </Grid>
        <DataGrid x:Name="TblData" Grid.Row="2" Grid.Column="0" Margin="10,3,8,0"
                  HeadersVisibility="Column" EnableRowVirtualization="False" CanUserAddRows="False"
CanUserDeleteRows="False" AutoGenerateColumns="False"
                  AlternatingRowBackground="AliceBlue"
SelectedCellsChanged="TblData_SelectedCellsChanged"
                  ItemsSource="{Binding Collection}" SelectedItem="{Binding Model}"
SelectionChanged="TblData_SelectionChanged">
            <DataGrid.Columns>
                <DataGridTextColumn Header="CODE" Binding="{Binding Uid}" Width="auto"/>
                <DataGridTextColumn Header="NAME" Binding="{Binding Name}" Width="*"/>
                <DataGridTextColumn Header="STATUS" Binding="{Binding Status}" Width="auto"/>
            </DataGrid.Columns>

        </DataGrid>
        <TextBox x:Name="TxtUid" Text="{Binding Model.Uid, Mode=TwoWay}" TextWrapping="Wrap"
Margin="254,20,8,86" Grid.Row="3"/>
        <TextBox x:Name="TxtName" Text="{Binding Model.Name, Mode=TwoWay}" TextWrapping="Wrap"
Margin="254,57,8,49" Grid.Row="3"/>
        <Label Content="Code" Margin="188,15,351,80" Grid.Row="3"/>
        <Label Content="Name" Margin="188,52,351,43" Grid.Row="3"/>
        <Label Content="Status" Margin="188,89,351,6" Grid.Row="3" RenderTransformOrigin="0.384,-1.211"/>
        <CheckBox x:Name="IsChecked" IsChecked="{Binding IsChecked, Mode=TwoWay}" Content="Active"
Margin="254,95,266,12" Grid.Row="3"/>
        <Button x:Name="BtnUpdate" Command="{Binding UpdateCommand}" Content="UPDATE" Grid.Column="2"
HorizontalAlignment="Left" Margin="10,10,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_UPDATE.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnEdit" Click="BtnEdit_Click" Content="EDIT" Grid.Column="2"
HorizontalAlignment="Left" Margin="10,77,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_EDIT.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnDelete" Click="BtnDelete_Click" Content="DELETE" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,145,0,0" Grid.Row="2" VerticalAlignment="Top" Width="180"
Height="47" Command="{Binding DeleteCommand}" Grid.RowSpan="2">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_DELETE.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnReset" Click="BtnReset_Click" Content="RESET" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,41,0,0" Grid.Row="3" VerticalAlignment="Top" Width="180"
Height="47" >
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_RESET.png"/>
            </Button.Background>
        </Button>
        <Button x:Name="BtnClose" Click="BtnClose_Click" Content="CLOSE" Grid.Column="1"
HorizontalAlignment="Left" Margin="10,110,0,0" Grid.Row="3" VerticalAlignment="Top" Width="180"
Height="47" Grid.RowSpan="2">
            <Button.Background>
                <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON_CLOSE.png"/>
            </Button.Background>
        </Button>

    </Grid>

</Window>
```

- ProductView.xaml.cs



```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using FP.ViewModel;

namespace FP.View.Inventories
{
    /// <summary>
    /// Interaction logic for ProductView.xaml
    /// </summary>
    public partial class ProductView : Window
    {
        public ProductView()
        {
            InitializeComponent();
            vm = new ProductViewModel();
            vm.OnCallBack += Close;
            DataContext = vm;
        }
        private readonly ProductViewModel vm;
        private void BtnUpdate_Click(object sender, RoutedEventArgs e)
        {
            TxtUid.IsEnabled = true;
            TxtName.IsEnabled = true;

            vm.Model = new Models.Product();
            vm.IsChecked = true;
            TxtName.Focus();
        }

        private void BtnEdit_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnDelete_Click(object sender, RoutedEventArgs e)
        {
            Home.Dashboard dashboard = new Home.Dashboard();
            dashboard.Show();
            this.Close();
        }

        private void BtnReset_Click(object sender, RoutedEventArgs e)
        {

        }

        private void BtnClose_Click(object sender, RoutedEventArgs e)
        {
            Home.Dashboard dashboard = new Home.Dashboard();
            dashboard.Show();
            this.Close();
        }

        private void TblData_SelectedCellsChanged(object sender, SelectedCellsChangedEventArgs
e)      {

        }

        private void TblData_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {

        }
    }
}
```

4. StockDialog.xaml

```xaml
<Window x:Class="FP.View.Inventories.StockDialog"
        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:FP.View.Inventories"
        mc:Ignorable="d"
        Title="StockDialog" Height="450" Width="800">
    <Grid>

    </Grid>
</Window>
```

- StockDialog.xaml.cs

```cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace FP.View.Inventories
{
    /// <summary>
    /// Interaction logic for StockDialog.xaml
    /// </summary>
    public partial class StockDialog : Window
    {
        public StockDialog()
        {
            InitializeComponent();
        }
    }
```

```
}
```

E. Welcome

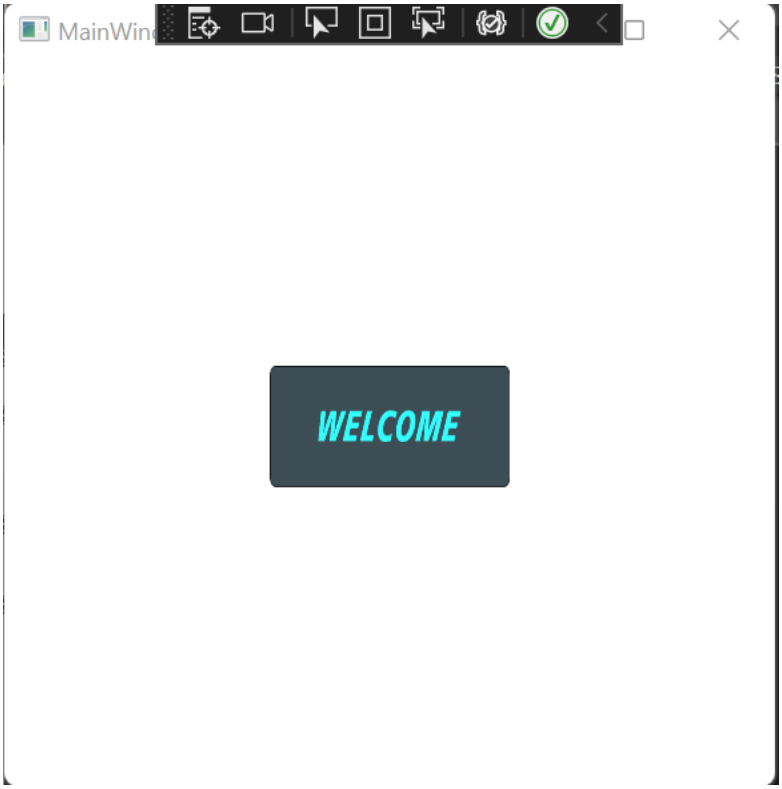
1. Welcome.xaml

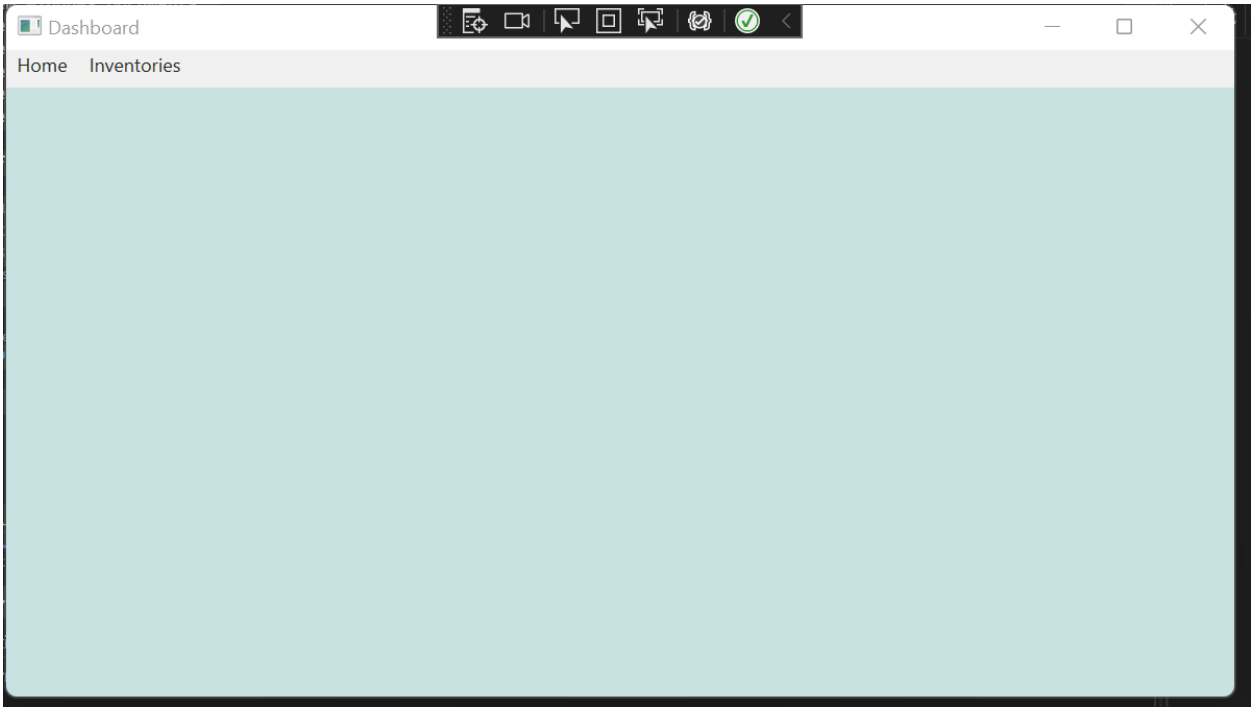
```
<Window x:Class="FP.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:FP"
        mc:Ignorable="d"
        Title="MainWindow" Height="400" Width="400">

    <Window.Resources>
        <ResourceDictionary>
            <Style TargetType="Button" x:Key="Button_Form1">
                <Setter Property="Background">
                    <Setter.Value>
                        <ImageBrush ImageSource="D:\semster 3\pmrog\FP\FP\Asset\BUTTON WELCOME.png"/>
                    </Setter.Value>
                </Setter>

                <Setter Property="Template">
                    <Setter.Value>
                        <ControlTemplate TargetType="{x:Type Button}">
                            <Border Background="{TemplateBinding Background}">
                                <ContentPresenter HorizontalAlignment="Center"
VerticalAlignment="Center"/>
                            </Border>
                        </ControlTemplate>
                    </Setter.Value>
                </Setter>
                <Style.Triggers>
                    <Trigger Property="IsPressed" Value="true">
                        <Setter Property="Content">
                            <Setter.Value>
                                <Image Source="D:\semster 3\pmrog\FP\FP\Asset\BUTTON WELCOME.png"
Stretch="Fill"/>
                            </Setter.Value>
                        </Setter>
                    </Trigger>
                </Style.Triggers>
            </Style>

        </ResourceDictionary>

    </Window.Resources>

    <Grid>
        <Button Margin="20,20,20,20" Style="{StaticResource Button_Form1}" Click="Button_Click_1"/>

    </Grid>
</Window>
```

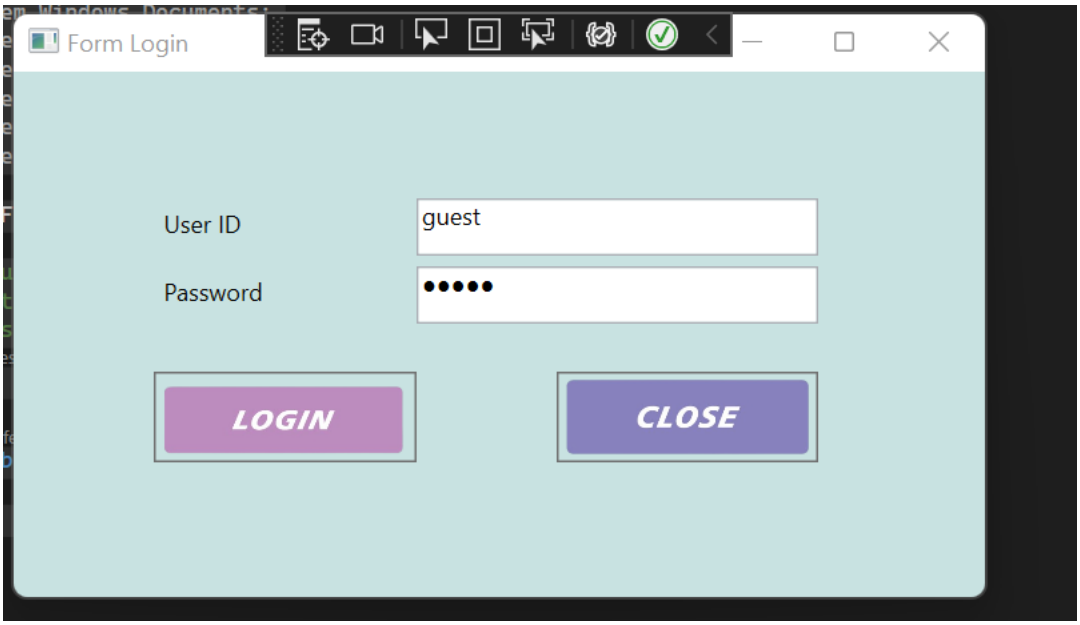
a. Welcome.xaml.cs

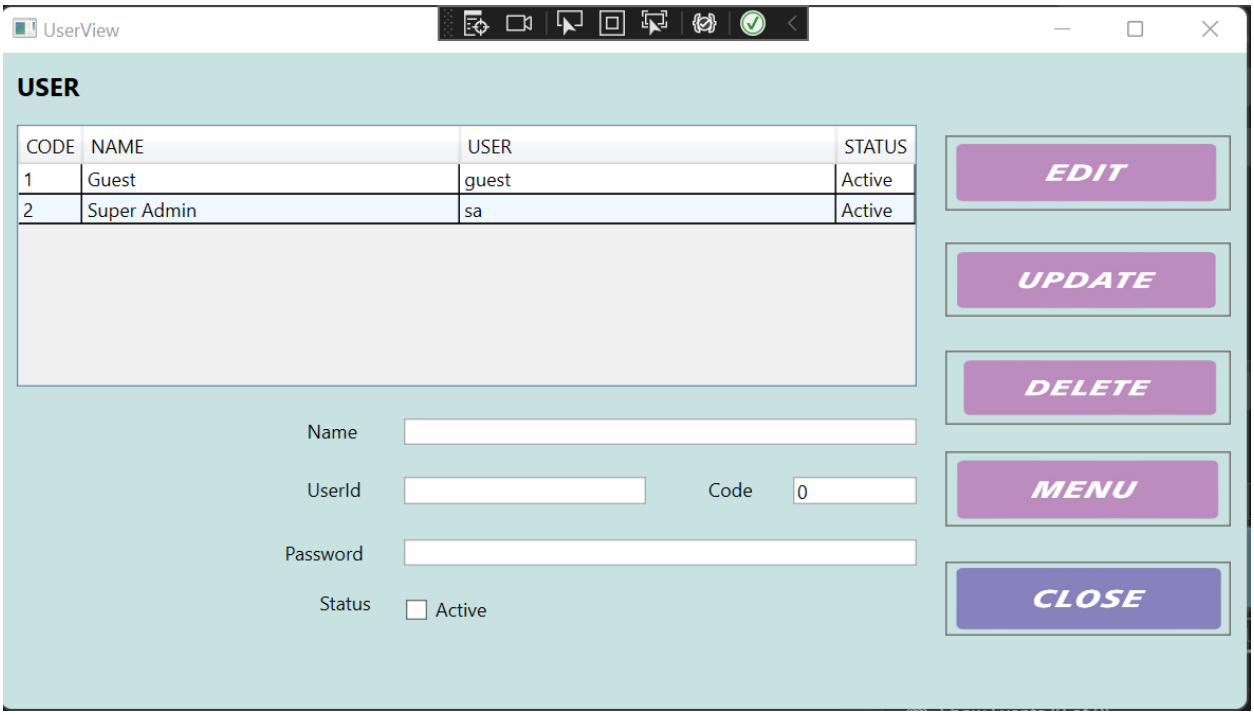
```cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace FP
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }



        private void Button_Click_1(object sender, RoutedEventArgs e)
        {
            View.Home.Dashboard dashboard = new View.Home.Dashboard();
            dashboard.Show();
            this.Close();
        }
    }

}
```

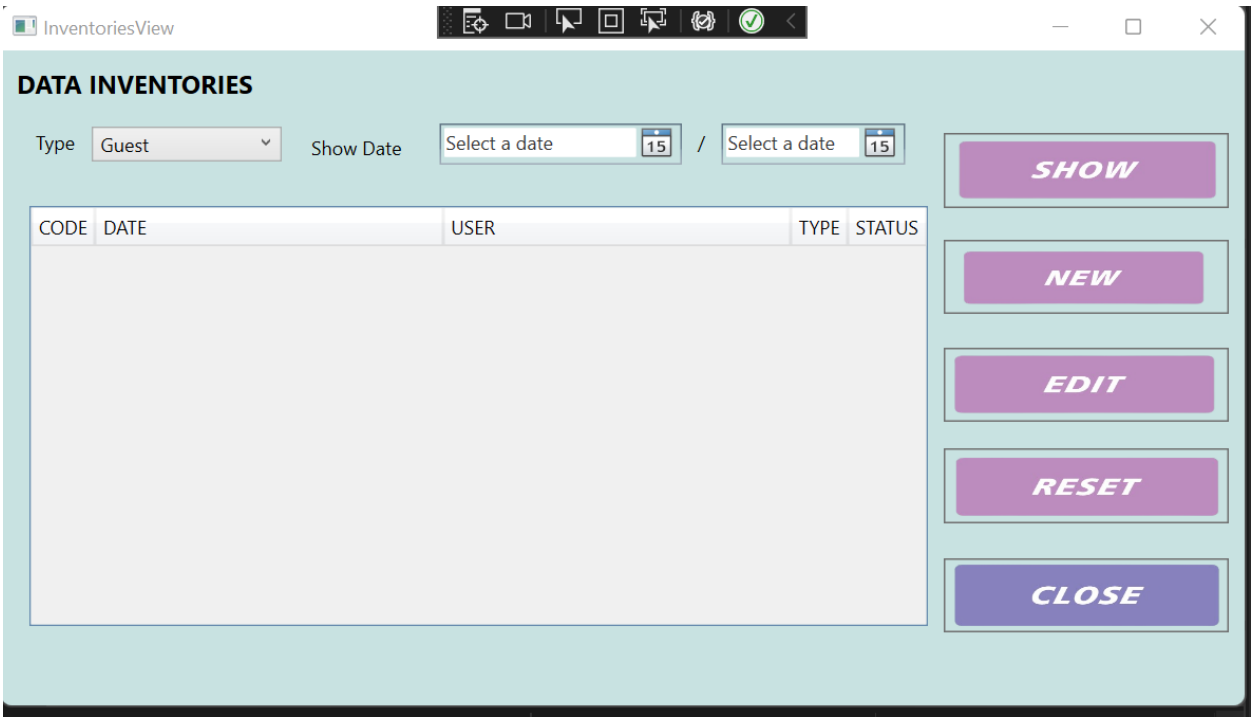
F. Hasil Run

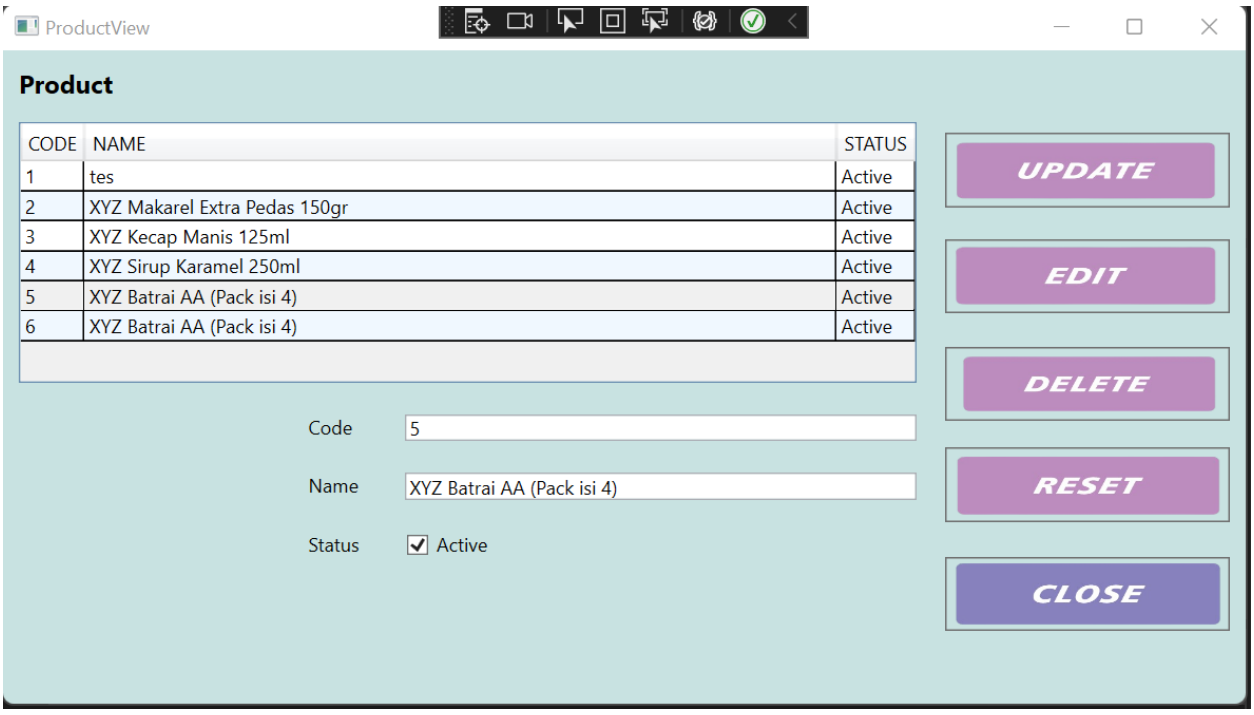





Form Login
User ID
guest
Password
LOGIN
CLOSE


UserView
USER
CODE
NAME
USER
STATUS
1
Guest
guest
Active
2
Super Admin
sa
Active
EDIT
UPDATE
DELETE
MENU
CLOSE
Name
UserId
Code
0
Password
Status
Active


InventoriesView
DATA INVENTORIES
Type
Guest
Show Date
Select a date
/
Select a date
CODE
DATE
USER
TYPE
STATUS
SHOW
NEW
EDIT
RESET
CLOSE


ProductView
Product
CODE
NAME
STATUS
1
tes
Active
2
XYZ Makarel Extra Pedas 150gr
Active
3
XYZ Kecap Manis 125ml
Active
4
XYZ Sirup Karamel 250ml
Active
5
XYZ Batrai AA (Pack isi 4)
Active
6
XYZ Batrai AA (Pack isi 4)
Active
Code
5
Name
XYZ Batrai AA (Pack isi 4)
Status
Active
UPDATE
EDIT
DELETE
RESET
CLOSE