Milestone 01 - Master data

Yêu cầu

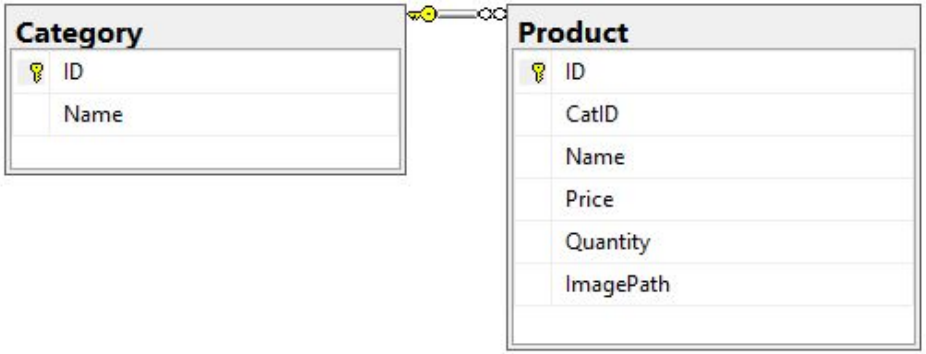
Nội dung chính

Dữ liệu master gồm: Loại sản phẩm & Sản phẩm

- **Category**: **ID** (int), Name (text)
- **Product**: **ID** (int), CatID (int), Name (text), Price(text), Quantity (int), ImagePath (text)

4 Thao tác chính

- Xem: Xem danh sách, xem chi tiết, sắp xếp, lọc, tìm kiếm
- \+ Thêm mới, Xóa, Sửa
- \+ Backup, restore
- \+ Đóng gói, viết hướng dẫn sử dụng



Script tạo bảng có thể tải tại đây

Lưu ảnh ở đâu? CSDL hay trên ổ đĩa?

- Nếu ảnh < 256KB: sử dụng cột kiểu VarBinary thì hiệu năng tốt hơn
- Nếu ảnh > 1MB: Lưu trên ổ đĩa thì hiệu năng tốt hơn
- Ở giữa 256KB và 1MB: thích lưu đâu thì lưu

Nguồn tham khảo: **To BLOB or Not To BLOB: Large Object Storage in a Database or a Filesystem**
http://research.microsoft.com/apps/pubs/default.aspx?id=64525

Chi tiết các chức năng của milestone

1. Import dữ liệu gốc (Category & Product) từ file excel (2đ)
2. Xem danh sách loại sản phẩm (0.5 đ)
    a. Chỉnh sửa tên loại sản phẩm (0.5 đ)
    b. Xóa một loại sản phẩm (0.5 đ)
3. Xem danh sách sản phẩm theo loại sản phẩm (3.5 đ)
    a. Xem chi tiết sản phẩm: Cập nhật (1.5 đ), Xóa (0.5 đ)
    b. Thêm một sản phẩm (1 đ)

Cách tổ chức bài nộp

Tên project: **MyShop**

1. Thư mục **Release**: Chứa **tập tin thực thi** chương trình (exe) đã được biên dịch từ mã nguồn (Có thể đóng gói thành file cài đặt càng tốt)
2. Thư mục **Source**: Chứa **mã nguồn** của chương trình (đã xóa đi các tập tin trung gian của quá trình biên dịch dùng menu Build > Clean)
3. Tập tin **Readme.txt**:
   a. Chứa thông tin sinh viên (họ tên, MSSV). Phân công + tỉ lệ điểm
   b. Sinh viên cần quay video demo, upload lên youtube ở chế độ Unlisted và nộp lại link này. Video không quá 5 phút, không lồng nhạc, không lồng tiếng. Gõ nội dung muốn nói vào file text / powerpoint hoặc làm phụ đề nếu được
   c. Liệt kê danh sách các chức năng đã làm được và không làm được
   d. Có thể nêu thêm điểm đề nghị trên thang 10

Phần mềm gợi ý để quay video

- **ShareX (miễn phí)**
- Camtasia

Các icon có thể tải tại: flaticon.com hoặc đã tổng hợp tại:

https://drive.google.com/drive/folders/1clGmMKTaXfHV6c_VzmC0SOFhFumwedGm?usp=sharing

Các tài nguyên

uifaces.co

flaticon.com

pixabay.com

unsplash.com

Login

Vấn đề placeholder

1. Tạo một Label mới với màu Foreground=”Gray”, đặt bên dưới TextBox, Chỉnh FontStyle=”Italic” (in nghiêng)
2. Cho màu Foreground TextBox là Transparent (trong suốt)
3. Xử lí sự kiện GotFocus và LostFocus để ẩn hiện tương ứng
   - ❏ Gán thuộc tính Visibility = Visibility.Hidden để ẩn
   - ❏ Visibility.Visible để hiện lại

Vấn đề phím tắt binding label và textbox

```
<Label Content="_Username" Target="{Binding ElementName=usernameTextBox}"
```

Settings

Đọc thông tin từ tập tin cấu hình App.config

```
var server = ConfigurationManager.AppSettings["server"];

var db = ConfigurationManager.AppSettings["database"];
```

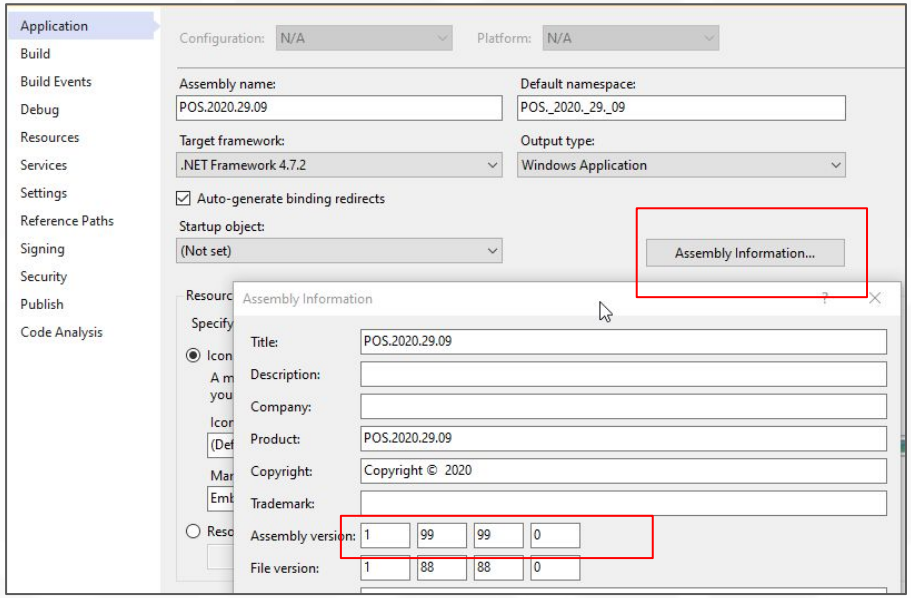
```
<configuration>
    <startup>
        <supportedRuntime version="v4.0" sk
    </startup>
  <appSettings>
    <add key="server" value="localhost"/>
    <add key="database" value="MyShop"/>
  </appSettings>
</configuration>
```

Lưu thông tin từ tập tin cấu hình

```
var config = ConfigurationManager.OpenExeConfiguration(ConfigurationUserLevel.None);

config.AppSettings.Settings["server"].Value = “localhost”;

config.AppSettings.Settings["database"].Value = “MyShop”;

config.Save(ConfigurationSaveMode.Minimal);
```

Lấy thông tin phiên bản

```
var version = Assembly.GetExecutingAssembly().GetName().Version;

versionLabel.Content = $"v{version}";
```



Tạo ra dialog

Hai button chính của Dialog

- OK Button

  IsDefault = True để mặc định là nút chính

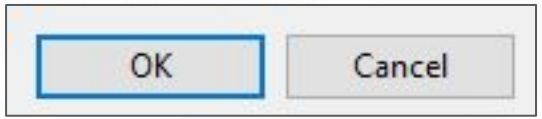
  Cần xử lí sự kiện Click

  Gán các giá trị

  Bên trong gán DialogResult = true

- Cancel Button

  Gán IsCancel = True, kích hoạt bằng phím **Esc**



Kĩ thuật truyền - nhận giá trị giữa hai màn hình

Truyền bằng đối số hàm tạo

```
var server = ConfigurationManager.AppSettings["server"];
var db = ConfigurationManager.AppSettings["database"];

Debug.WriteLine($"Server: {server}, db: {db}");

var screen = new SettingsWindow(server, db);

if (screen.ShowDialog() == true)
{
```

Trả lại bằng public attributes / properties

```
public string Server = "";
public string DB = "";

1 reference
private void okButton_Click(object sender, RoutedEventArgs e)
{
    Server = serverTextBox.Text;
    DB = dbTextBox.Text;

    DialogResult = true;
}
```

Dashboard

Cần tạo dashboard cho 2 người

AdminDashboard

SaleDashboard

Tổng quan

1. Tạo giao diện Ribbon
2. Thêm tính năng Import cho loại sản phẩm và sản phẩm
3. Quản lí loại sản phẩm
4. Quản lí sản phẩm

Bước 1: Tạo giao diện Ribbon

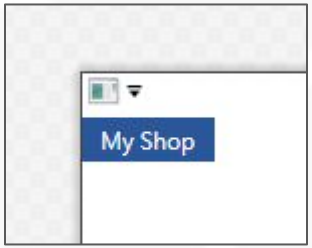
1. Tạo mới dự án dạng WPF
2. Thêm thư viện Fluent.Ribbon
3. Thêm khai báo xmlns:Fluent="clr-namespace:Fluent;assembly=Fluent"
4. Sửa tên thẻ Window thành Fluent:RibbonWindow
5. Thêm khai báo resource trong App.xaml

```
<Application.Resources>
    <ResourceDictionary>
      <ResourceDictionary.MergedDictionaries>
        <ResourceDictionary Source="pack://application:,,,/Fluent;Component/Themes/Generic.xaml" />
      </ResourceDictionary.MergedDictionaries>
    </ResourceDictionary>
</Application.Resources>
```

Bước 2 - Chuẩn bị Backstage

1. Thêm backstage

```
<Fluent:Ribbon>
  <Fluent:Ribbon.Menu>
    <Fluent:Backstage Header="My Shop" />
  </Fluent:Ribbon.Menu>
</Fluent:Ribbon>
```



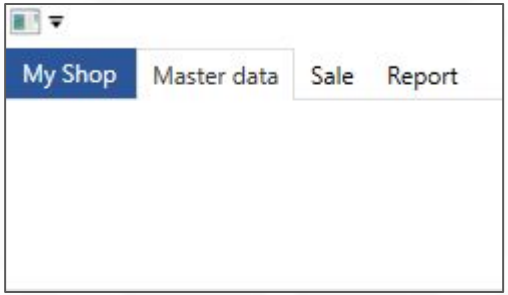
2. Nội dung cho backstage

```
<Fluent:Backstage Header="My Shop" >
        <Fluent:BackstageTabControl>
          <Fluent:BackstageTabItem Header="Info">
            <Label Content="MSSV - Họ và tên"/>
          </Fluent:BackstageTabItem>
          <Fluent:BackstageTabItem Header="Exit"
            MouseLeftButtonDown="exitMenu_MouseLeftButtonDown"/>
        </Fluent:BackstageTabControl>
 </Fluent:Backstage>
```


```
Application.Current.Shutdown();
```

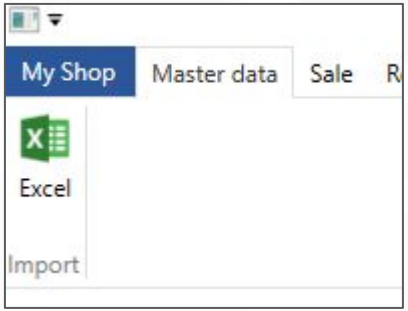
Bước 3 - Tạo 3 tab chính: Master Data, Sale, Report

```
<Fluent:RibbonTabItem Header="Master data">
</Fluent:RibbonTabItem>

<Fluent:RibbonTabItem Header="Sale">
</Fluent:RibbonTabItem>

<Fluent:RibbonTabItem Header="Report">
</Fluent:RibbonTabItem>
```



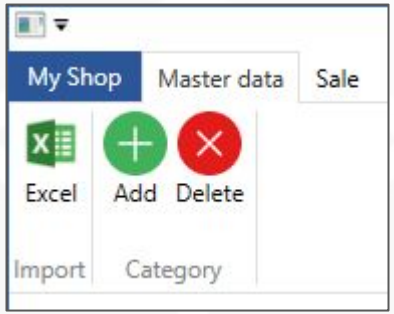
Bước 4 - Thêm các hành động trên RibbonTab

```
<Fluent:RibbonTabItem Header="Master data">
    <Fluent:RibbonGroupBox Header="Import">
        <Fluent:Button Header="Excel" Click="excelImportButton_Clicked">
            <Fluent:Button.LargeIcon >
                <Image Source="/Images/Excel.png" RenderOptions.BitmapScalingMode="HighQuality"/>
            </Fluent:Button.LargeIcon>
        </Fluent:Button>
    </Fluent:RibbonGroupBox>
</Fluent:RibbonTabItem>
```

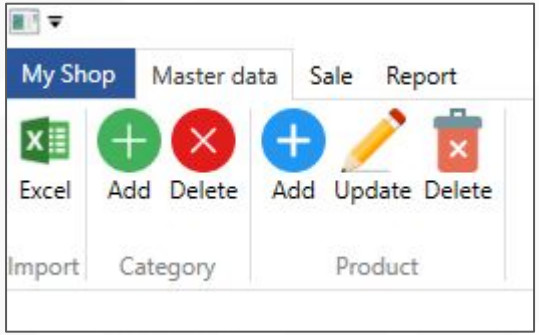


Tạo các lệnh cần thiết cho Category

```
<Fluent:RibbonGroupBox Header="Category">
        <Fluent:Button Header="Add" Click="addCategorytButton_Clicked">
                <Fluent:Button.LargeIcon>
                        <Image Source="/Images/plus.png" RenderOptions.BitmapScalingMode="HighQuality"/>
                </Fluent:Button.LargeIcon>
        </Fluent:Button>
        <Fluent:Button Header="Delete" Click="deleteCategorytButton_Clicked">
                <Fluent:Button.LargeIcon>
                        <Image Source="/Images/delete.png"
                                RenderOptions.BitmapScalingMode="HighQuality"/>
                </Fluent:Button.LargeIcon>
         </Fluent:Button>
</Fluent:RibbonGroupBox>
```



Tạo các lệnh cần thiết cho Product (tự làm)



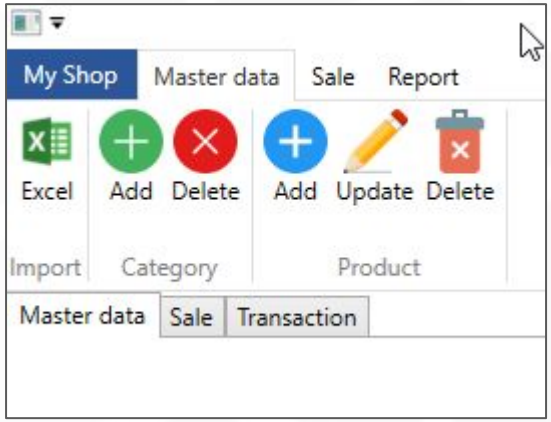
Bước 5: Tạo các tab chứa nội dung thật sự

1. Sửa control chứa nội dung gốc từ Grid thành DockPanel

```
<DockPanel LastChildFill="True">
    <Fluent:Ribbon DockPanel.Dock="Top">
    ...
    </Fluent:Ribbon>
```

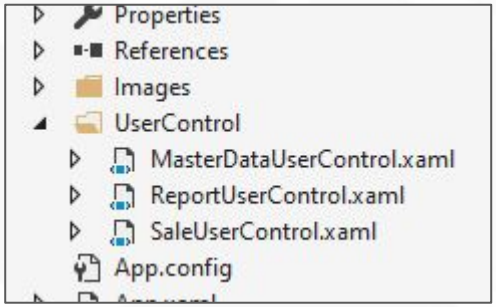
2. Thêm vào TabControl với tên tabs

```
<TabControl Name="tabs">
    <TabItem Header="Master data"></TabItem>
    <TabItem Header="Sale"></TabItem>
    <TabItem Header="Transaction"></TabItem>
</TabControl>
```



Bước 6 - Nạp động nội dung dùng UserControl

1. Tạo 3 UserControl: MasterDataUserControl, SaleUserControl, ReportUserControl
2. Khởi tạo các control




```
private void RibbonWindow_Loaded(object sender, RoutedEventArgs e)
{
        var screens = new ObservableCollection<TabItem>()
        {
                new TabItem() { Content = new MasterDataUserControl()},
                new TabItem() { Content = new SaleUserControl()},
                new TabItem() { Content = new ReportUserControl()}
        };
        tabs.ItemsSource = screens;
}
```

Thực hiện binding thuộc tính SelectedIndex

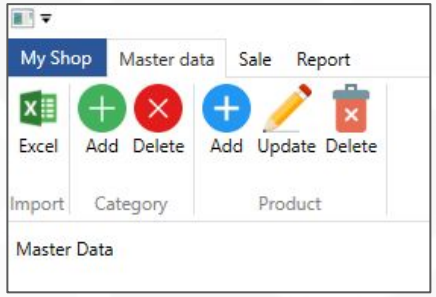
```
<TabControl Name="tabs"
      SelectedIndex="{Binding ElementName=ribbon, Path=SelectedTabIndex}">
 </TabControl>
```

Checkpoint: mỗi khi chuyển tab của Ribbon thì nội dung của tab control bên dưới thay đổi tương ứng.

Loại bỏ đi phần header thừa của TabControl

Ý tưởng: tạo ControlTemplate cho TabControl nhưng rỗng

```
<TabControl Name="tabs" BorderThickness="0"
            SelectedIndex="{Binding ElementName=ribbon, Path=SelectedTabIndex}">
    <TabControl.Resources>
        <Style TargetType="TabItem">
            <Setter Property="Template">
                <Setter.Value>
                    <ControlTemplate TargetType="TabItem">
                    </ControlTemplate>
                </Setter.Value>
            </Setter>
        </Style>
    </TabControl.Resources>
</TabControl>
```



Bước 6: Chuẩn bị tập tin Excel để import

Sinh viên được yêu cầu chuẩn bị dữ liệu mẫu gồm

- Tối thiểu 3 loại sản phẩm
- Tối thiểu 12 sản phẩm mỗi loại (tổng cộng là 36 sản phẩm) có hình minh họa

Nếu sử dụng lại dữ liệu mẫu được cung cấp cần bổ sung thêm 3 loại sản phẩm khác và 36 sản phẩm để có tổng cộng 6 loại sản phẩm và 72 sản phẩm **khác nhau**.

File excel mẫu và hình trong thư mục **products** tải tại đây

Cấu trúc tab Categories & Products

Order	Name
1	Asus
2	Microsoft
3	Apple

Order	Category	Name	Price	Quantity	ImageName
1	Asus	ASUS ZenBook Edition 30 UX334FL -30- A4057T I7 8565U MX250 RAM 8GB SSD 512GB 13.3" FHD	15,000,000	40	asus01.jpg
2	Asus	ASUS A412DA - EK346T AMD Ryzen 3 3200U RAM 4GB SSD 512GB 14" FHD WINDOW 10	67,000,000	31	asus02.jpg
3	Asus	ASUS A412FA - EK153T I5 8265U RAM 8GB HDD 1TB 14" FHD WINDOW 10	82,000,000	29	asus03.jpg
4	Asus	ASUS A412FA - EK155T I3 8145U RAM 4GB HDD 1TB 14" FHD WINDOW 10	67,000,000	24	asus04.jpg
5	Asus	ASUS F560UD-BQ327T I5 8250U GTX 1050 RAM 8GB 1TB HDD 15.6" FHD	27,000,000	87	asus05.jpg

Bước 7: Thực hiện import vào CSDL

1. Tạo ánh xạ Entity từ CSDL, đặt tên MyShopModel
   Các entity được lưu với tên MyShopEntities
2. Đọc danh sách các Loại sản phẩm, lưu vào CSDL lấy id
3. Tạo ra từ điển ngược để tra ra id từ tên loại sản phẩm
4. Đọc danh sách các sản phẩm
5. Lưu các hình ảnh vào thư mục chỉ định

(Chú ý backup database để test nhiều lần)

Mở file

```
var screen = new OpenFileDialog();

if (screen.ShowDialog() == true)

{



}
```

Đọc danh sách các loại sản phẩm

```
var filename = screen.FileName;
var workbook = new Workbook(filename);
var sheet = workbook.Worksheets[0];

// Đọc danh sách các category
var row = 3;
var cell = sheet.Cells[$"C{row}"];

do  {
        var name = cell.StringValue;
        Debug.WriteLine(name);

        // Đi qua dòng kế
        row++;
        cell = sheet.Cells[$"C{row}"];
}
while (cell.Value != null);
```

Chèn các loại sản phẩm vào CSDL

```
var db = new MyShopEntities(); // Mở kết nối tới CSDL
var cell = sheet.Cells[$"C{row}"];
var categories = new List<Category>();

do {
        var name = cell.StringValue;
        categories.Add(new Category() { Name = name });

        // Đi qua dòng kế
        row++;
        cell = categorySheet.Cells[$"C{row}"];
}
while (cell.Value != null);

db.Categories.AddRange(categories);
db.SaveChanges();
```

Tạo ra từ điển tra ngược ID từ tên Loại sản phẩm

```
// Tạo ra từ điển tra ngược từ tên Loại sản phẩm ra ID
var dictionary = new Dictionary<string, int>();
foreach(var category in categories)
{
    dictionary.Add(category.Name, category.ID);
}
```

Đọc các sản phẩm và đưa vào CSDL

```
// Đọc danh sách các sản phẩm
row = 3;
sheet = workbook.Worksheets[1];
cell = sheet.Cells[$"C{row}"];
var productCount = 0;

do
{
var catName = cell.StringValue;
var productName = sheet.Cells[$"D{row}"].StringValue;
var price = sheet.Cells[$"E{row}"].FloatValue;
var quantity = sheet.Cells[$"F{row}"].IntValue;
var imagePath = sheet.Cells[$"G{row}"].StringValue;

db.Products.Add(new Product()
{
CatID = dictionary[catName], Name = productName,
Price = price, Quantity = quantity,ImagePath = imagePath
});

// Đi qua dòng kế
row++; productCount++;
cell = sheet.Cells[$"C{row}"];
}
while (cell.Value != null);

db.SaveChanges();

MessageBox.Show($"Đã thêm vào hệ thống
{categories.Count} loại sản phẩm và {productCount} sản
phẩm", "Thành công", MessageBoxButton.OK,
MessageBoxImage.Information);
```

Chú ý import xong

Cần ra lệnh nạp lại dữ liệu của màn hình

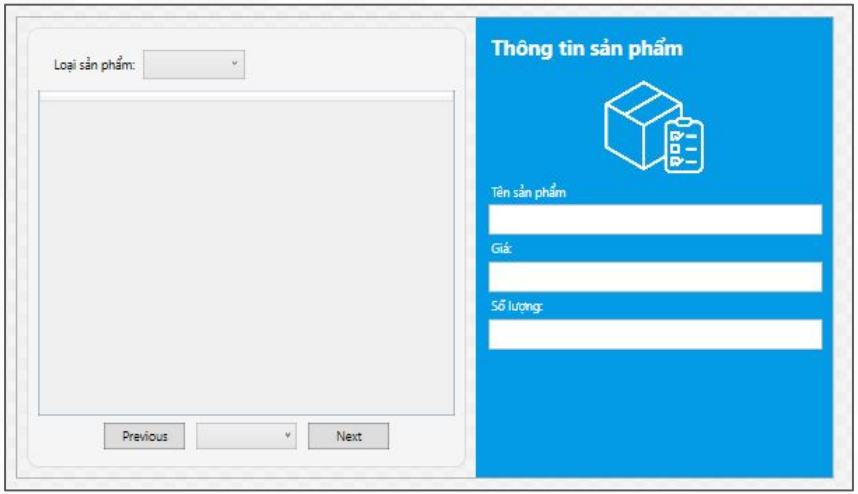
Copy ảnh vào thư mục ảnh, mặc định là Images

```
// Xác định thư mục chứa file excel
var excelFileInfo = new FileInfo(filename);

// Lấy thư mục chứa hình ảnh đi kèm file excel
var productImagesFolder = excelFileInfo.Directory +
@"\products";

// Nếu có thì mới import ảnh
if (Directory.Exists(productImagesFolder))
{
// Lấy thư mục hiện hành
var exeFolder = AppDomain.CurrentDomain.BaseDirectory;
var imgSubFolder = exeFolder + "Images";

// Tạo thư mục Images để chứa ảnh
 if (!Directory.Exists(imgSubFolder)) {
        Directory.CreateDirectory(imgSubFolder);
}
```

```
var source = productImagesFolder + @"\" + imagePath;
var sourceInfo = new FileInfo(source);
var extension = sourceInfo.Extension; // Trích xuất phần đuôi
var newName = Guid.NewGuid() + extension; // Tự phát sinh id
duy nhất toàn hệ thống
var destination = $"{imgSubFolder}\\{newName}";

File.Copy(source, destination);

// Cập nhật tên mới để lưu vào CSDL
imagePath = newName;
}
```

Bước 8: Hiển thị danh sách các loại sản phẩm

- Chuẩn bị giao diện gồm hai phần (DockPanel)



Phần bên phải có độ rộng 350, bên trái là phần còn lại

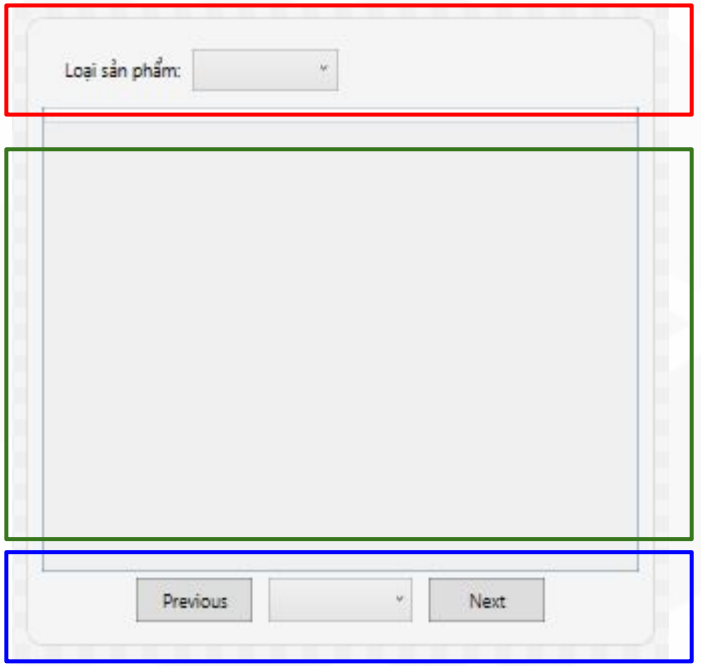
Phần nội dung chính bên trái

Có cấu trúc 3 phần, lại sử dụng DockPanel

StackPanel theo chiều ngang

DataGrid

StackPanel theo chiều ngang



Tạo border với viền cong

```
<Border Margin="10" Padding="10" CornerRadius="10"
    BorderThickness="1" BorderBrush="LightGray"
    Background="#F5F5F5">
```

Sự kiện Initialized của user control

```
MyShopEntities _db = new MyShopEntities();
private void UserControl_Initialized(object sender, EventArgs e)
{
        categoriesComboBox.ItemsSource = _db.Categories.ToList();
        categoriesComboBox.SelectedIndex = 0;
}
```

Khi lựa chọn loại sản phẩm thay đổi, ta sẽ nạp danh sách sản phẩm tương ứng lên, do có phân trang nên sẽ chỉ hiện thị trang đầu tiên

Bước 9: Hiển thị danh sách sản phẩm

Trước hết tính toán thông tin phân trang

```
class PagingRow
{
        public int Page { get; set; }
        public int TotalPages { get; set; }
}
```

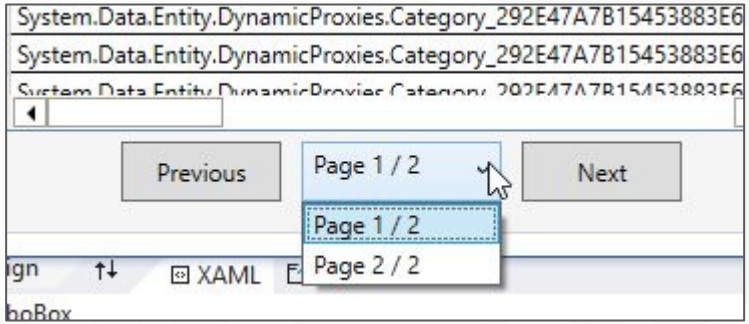
```
class PagingInfo
    {
        public List<PagingRow> Items { get; set; }
        public PagingInfo(int totalPages)
        {
            Items = new List<PagingRow>();

            for (int i = 1; i <= totalPages; i++)
            {
                Items.Add(new PagingRow()
                {
                    Page = i,
                    TotalPages = totalPages
                });
            }
        }
    }
```

Tính toán thông tin phân trang

```
private void categoriesComboBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
        var category = categoriesComboBox.SelectedItem as Category;

        // Tính toán các thông tin phân trang
        _totalProducts = category.Products.Count; // Tổng số sản phẩm
        _totalPages = _totalProducts / _rowsPerPage; // Tổng số trang, chia lấy phần nguyên
        if (_totalProducts % _rowsPerPage != 0) // Nếu còn dư thì thêm một trang
        {
           _totalPages++;
        }
        _currentPage = 1; // Trang hiện tại là trang 1

        var pagingInfo = new PagingInfo(_totalPages);
        pagesComboBox.ItemsSource = pagingInfo.Items;
        pagesComboBox.SelectedIndex = 0;
}
```

Khuôn mẫu hiển thị phân trang

```
<ComboBox.ItemTemplate>
    <DataTemplate>
        <StackPanel Orientation="Horizontal" HorizontalAlignment="Center">
            <TextBlock Text="Page "/>
            <TextBlock Text="{Binding Page}"/>
            <TextBlock Text=" / "/>
            <TextBlock Text="{Binding TotalPages}"/>
        </StackPanel>
    </DataTemplate>
</ComboBox.ItemTemplate>
```



Khi lựa chọn trang hiện tại thay đổi

Ta sẽ hiển thị danh sách sản phẩm tương ứng ở đây

```
private void pagesComboBox_SelectionChanged(object sender, SelectionChangedEventArgs e)
 {
        var category = categoriesComboBox.SelectedItem as Category;

        var next = pagesComboBox.SelectedItem as PagingRow;
        _currentPage = next.Page;

        // Chỉ lấy các sản phẩm của trang hiện tại
        productsDataGrid.ItemsSource = category.Products
            .Skip((_currentPage - 1) * _rowsPerPage)
            .Take(_rowsPerPage);
}
```

Di chuyển giữa các trang

```
private void nextPageButton_Click(object sender, RoutedEventArgs e)
{
        var currentIndex = pagesComboBox.SelectedIndex;
        if (currentIndex < pagesComboBox.Items.Count - 1)
        {
            pagesComboBox.SelectedIndex = currentIndex + 1;
        }
 }
```

Hiển thị chi tiết khi click vào một sản phẩm

IsReadOnly="True" AutoGenerateColumns="False"

DataContext="{Binding ElementName=productsDataGrid, Path=SelectedItem}"

Tạo converter chuyển đổi đường dẫn ảnh

Từ tương đối sang tuyệt đối

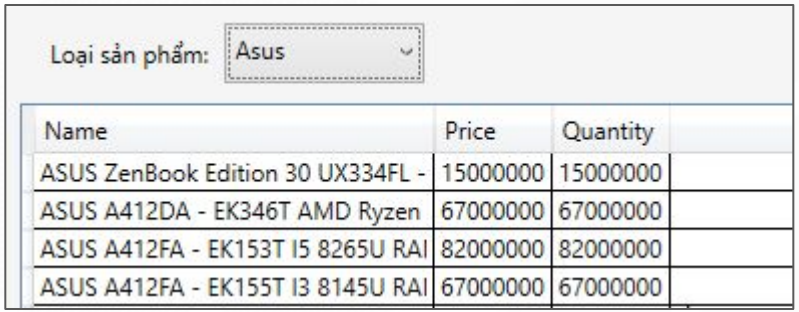
```
class RelativeToAbsolutePathConverter : IValueConverter
{
    public object Convert(object value, Type targetType, object parameter, CultureInfo culture)
    {
        var imageFile = (string)value;
        var exeFolder = AppDomain.CurrentDomain.BaseDirectory;
        return $"{exeFolder}Images\\{imageFile}";
    }

    public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)
    {
        throw new NotImplementedException();
    }
}
```

Khai báo và sử dụng converter

```
<UserControl.Resources>
    <local:RelativeToAbsolutePathConverter x:Key="absoluteConverter" />
</UserControl.Resources>


<Image Source="{Binding ImagePath, Converter={StaticResource
                                                absoluteConverter}}" />
```

Cấu hình các cột của DataGrid

```
<DataGrid.Columns >
<DataGridTextColumn Header="Name" Binding="{Binding Name}" Width="200" />
<DataGridTextColumn Header="Price" Binding="{Binding Price}" />
<DataGridTextColumn Header="Quantity" Binding="{Binding Price}" />
</DataGrid.Columns>
```



Làm sao từ nút bấm ribbon ra lệnh cho Usercontrol?

Bên trong UserControl tạo enum chứa các lệnh nó hiểu và có thể xử lí

```
public partial class MasterDataUserControl : UserControl{
    public enum MasterDataAction
    {
        AddNewCategory,             // Thêm mới một Loại sản phẩm
        DeleteSelectedCategory,     // Xóa Loại sản phẩm đang được chọn
        AddNewProduct,              // Thêm mới một Sản phẩm
        UpdateSelectedProduct,      // Cập nhật Sản phẩm đang được chọn
        DeleteSelectedProduct       // Xóa Sản phẩm đang được chọn
    };
```

Bên trong UserControl, chừa sẵn hàm xử lí

```
public void HandleParentEvent(MasterDataAction action)
        {
            switch (action)
            {
                case MasterDataAction.AddNewCategory:
                    addNewCategory();
                break;
                case MasterDataAction.DeleteSelectedCategory:
                    deleteSelectedCategory();
                break;

                    .....
```

Ở màn hình main, nút bấm Ribbon sẽ gọi hàm

```
private void addCategorytButton_Clicked(object sender, RoutedEventArgs e)
        {
            var userControl = tabs.Items[0] as MasterDataUserControl;
            userControl.HandleParentEvent(
                MasterDataUserControl.MasterDataAction.AddNewCategory
            );
        }
```

Tương tự cho các nút bấm còn lại