Navigation Techniques in .NET MAUI Labs

Perform these labs on your own computer using Visual Studio 2022 or later to ensure you understand the lessons presented in the corresponding videos and lectures.

Lab 1: Create User Detail Screen

Stop the application.

Right mouse-click on the project and add a new folder named **Views**.

Right mouse-click on the **Views** folder and select **Add | New Item | .NET MAUI | .NET MAUI ContentPage (XAML)…** from the context-sensitive menu.

Set the name of the page to **UserDetailView**.

Change the **Title** property to "**User Information**".

Delete the <VerticalStackLayout> and all XAML within it.

Cut the <Grid> from the **MainPage.xaml** and put it in this content page.

Update the Main Page

Where the <Grid> was in the **MainPage.xaml**, put back the <Label> you had when you first created this project.

```
<Label Text="{StaticResource ApplicationTitle}"
       FontSize="Large"
       VerticalOptions="Center"
       HorizontalOptions="Center" />
```

Lab 2: TabBar Navigation

Open the **AppShell.xaml** file and add a new XML namespace.

```
xmlns:views="clr-namespace:AdventureWorks.MAUI.Views"
```

Replace the existing <ShellContent> element with the following XAML.

```
<TabBar>
  <ShellContent Title="Home"
                ContentTemplate="{DataTemplate
local:MainPage}"
                Route="MainPage" />

  <ShellContent Title="Users"
                ContentTemplate="{DataTemplate
views:UserDetailView}" />
</TabBar>
```

Try It Out

Run the application on **Windows**. Click on the **Users** menu to navigate to the User Detail page.

Run the application on the **Android** Emulator. Click on the **Users** menu at the bottom of the page to navigate to the User Detail page.

Lab 3: Create More Pages

Right mouse-click on the **Views** folder and add a new content page named **LoginView**.

Set the **Title** attribute to "Login".

Replace the <VerticalStackLayout> and all the XAML within it with the following <Label>.

```
<Label Text="Login"
       FontSize="Large"
       VerticalOptions="Center"
       HorizontalOptions="Center" />
```

Create a Product Page

Right mouse-click on the Views folder and add a new content page named **ProductDetailView.xaml**.

Set the **Title** attribute to "Product Information".

Replace the <VerticalStackLayout> and all the XAML within it with the following <Label>.

```
<Label Text="Product Information"
       FontSize="Large"
       VerticalOptions="Center"
       HorizontalOptions="Center" />
```

Create a Customer Page

Right mouse-click on the Views folder and add a new content page named **CustomerDetailView.xaml**.

Set the **Title** attribute to "Customer Information".

Replace the <VerticalStackLayout> and all the XAML within it with the following <Label>.

```
<Label Text="Customer Information"
       FontSize="Large"
       VerticalOptions="Center"
       HorizontalOptions="Center" />
```

Create a Color List Page

Right mouse-click on the Views folder and add a new content page named **ColorListView.xaml**.

Set the **Title** attribute to "Color List".

Replace the <VerticalStackLayout> and all the XAML within it with the following <Label>.

```
<Label Text="Color List"
       FontSize="Large"
       VerticalOptions="Center"
       HorizontalOptions="Center" />
```

Create a Phone Types List Page

Right mouse-click on the Views folder and add a new content page named **PhoneTypesListView.xaml**.

Set the **Title** attribute to "Phone Types List ".

Replace the <VerticalStackLayout> and all the XAML within it with the following <Label>.

```
<Label Text="Phone Types List"
       FontSize="Large"
       VerticalOptions="Center"
       HorizontalOptions="Center" />
```

Modify AppShell

Open the **AppShell.xaml** file and add two more <ShellContent> elements immediately after the Users element.

```
<ShellContent Title="Products"
              ContentTemplate="{DataTemplate
views:ProductDetailView}" />

<ShellContent Title="Customers"
              ContentTemplate="{DataTemplate
views:CustomerDetailView}" />
```

Try It Out

Run the application and click on the **Users**, **Products**, and **Customers** menus to navigate to each page.

Lab 4: Create a Sub Menu

Open the **AppShell.xaml** file and immediately after the Customers menu, add the following XAML.

```
<Tab Title="Maintenance">
  <ShellContent Title="Colors"
                ContentTemplate="{DataTemplate
views:ColorListView}" />

  <ShellContent Title="Phone Types"
                ContentTemplate="{DataTemplate
views:PhoneTypesListView}" />
</Tab>

<ShellContent Title="Login"
              ContentTemplate="{DataTemplate
views:LoginView}" />
```

Try It Out

You must **STOP** the application for **Shell** changes.

Run the application on **Windows**.

Show the **Workflow** and **Maintenance** drop-down menus.

Run the application on the **Android** Emulator.

Click on **More …** item to see the **Maintenance** and Login sub-menus appear.

Click on **Maintenance** and you should see **Colors** and **Phone Types** menus appear at the top of the screen.

Lab 5: Create Routes and Navigate from Page to Page

Right mouse-click on the Views folder and add a new content page named **UserListView.xaml**.

Set the **Title** attribute to "User List".

Replace the <VerticalStackLayout> with the following XAML.

```
<VerticalStackLayout VerticalOptions="Center"
                     HorizontalOptions="Center"
                     Spacing="10">
  <Label Text="User List"
         FontSize="Header"
         HorizontalOptions="Center" />
  <Button Text="Navigate to Detail"
          Clicked="NavigateToDetail_Clicked" />
</VerticalStackLayout>
```

Create the **Clicked** event procedure.

```
private async void NavigateToDetail_Clicked(object
sender, EventArgs e)
{
  await
Shell.Current.GoToAsync(nameof(Views.UserDetailView));
}
```

Open the **AppShell.xaml** file and modify the <ShellContent> element that used to call the UserDetailView page and replace it with a call to the User List view.

```
<ShellContent Title="Users"
  ContentTemplate="{DataTemplate views:UserListView}" />
```

Register the Route

Now that the <ShellContent> no longer navigates to the **UserDetailView** page, you need to somehow register that route so you can navigate to it using the GoToAsync() method.

Open the **AppShell.xaml.cs** file and in the constructor register the route.

```
public AppShell()
{
 InitializeComponent();

 // Register a route
 Routing.RegisterRoute(nameof(Views.UserDetailView),
typeof(Views.UserDetailView));
}
```

Try It Out

Run the application and click on the Users menu.

Click on the **Navigate to Detail** button and you should see the **User Information** page appear.