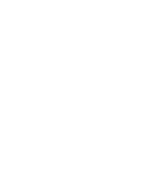
Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section: \_\_\_\_\_\_ Marked By: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**PROG 2500** - **Tech Check #1**

UWP – Contoso App Exploration & Enhancement



**Value:** 10% of overall course mark

15

**Time to complete:** 60 Minutes

**Setup Instructions:**

This Tech Check involves downloading a sample project and making modifications to it. The Contoso application is available here: <https://github.com/Microsoft/Windows-appsample-customers-orders-database>. Clone the repo (or download as zip) to your VM. Open the Contoso.sln file and build the entire solution. You will have to deploy the ContosoApp project prior to running the full application. To do this: Right-click the ContosoApp project and select Deploy. If prompted, don’t worry about connecting to Active Directory.

**Using the ContosoApp, answer all questions and complete all tasks listed below:**

1. Locate the View Code button {} in the appropriate markup file and modify it to provide a tooltip with text “View Code on GitHub.com”. Enter the file name, location (ie. Project and folder names) and beginning line number where you added the changes, followed by the code. **(2 pts)**

**Project:** Contoso.App

**Folder:** N/A

**File:** AppShell.xaml

**Starting Line Number:** 63

**Code:**

MUXC:NavigationView.PaneFooter>

<StackPanel>

<MUXC:NavigationViewItem

Content="View code"

ToolTipService.ToolTip="View code on GitHub.com"

Tapped="ViewCodeNavPaneButton\_Tapped">

<MUXC:NavigationViewItem.Icon>

<FontIcon FontFamily="Segoe MDL2 Assets" Glyph="&#xE943;"/>

</MUXC:NavigationViewItem.Icon>

</MUXC:NavigationViewItem>

<uc:AuthenticationControl/>

</StackPanel>

</MUXC:NavigationView.PaneFooter>

1. This project makes use of XAML user control elements. Find an example of a custom user control. Enter the XAML file name and location. **(1 pt)**

**Project:** Contoso.App

**Folder:** UserControls

**File:** AuthenticationControl.xaml

1. The CustomerDetailPage.xaml file has a save button on the Command bar that asynchronously saves any changes to the currently selected Customer to the database. Locate the code that makes a call to the Repository class to save the customer data to the database. Enter the file name, location and line number where you found the code, followed by the code. Hint: (Look for async and await keywords) **(2 pts)**

**Project:** Contoso.App

**Folder:** ViewModels

**File:** CustomerViewModel.cs

**Starting Line Number:** 235

**Code:**

public async Task SaveAsync()

{

IsInEdit = false;

IsModified = false;

if (IsNewCustomer)

{

IsNewCustomer = false;

App.ViewModel.Customers.Add(this);

}

await App.Repository.Customers.UpsertAsync(Model);

}

1. You’ve been asked to add code to set every button in the application to a common height of 30. Enter the name and location of the file where this could be achieved the most efficiently. **(1 pt)**

**Project:** Contoso.App

**Folder:** Styles

**File:** NavStyles.xaml

1. The OrderListPage.xaml file displays a list of orders in the system. Locate the code that makes a call to the Repository class to retrieve the order data from the database. Enter the file name, location and line number where you found the code, followed by the code. Hint: (Look for async and await keywords) **(2 pts)**

**Project:** Contoso.App

**Folder:** ViewModels

**File:** OrderListPageViewModel.cs

**Starting Line Number:** 123

**Code:**

/// <summary>

/// Retrieves orders from the data source.

/// </summary>

public async void LoadOrders()

{

await DispatcherHelper.ExecuteOnUIThreadAsync(() =>

{

IsLoading = true;

Orders.Clear();

MasterOrdersList.Clear();

});

var orders = await Task.Run(App.Repository.Orders.GetAsync);

await DispatcherHelper.ExecuteOnUIThreadAsync(() =>

{

foreach (var order in orders)

{

Orders.Add(order);

MasterOrdersList.Add(order);

}

IsLoading = false;

});

}

1. Find the Customer Model and add a public updateable property called MiddleName to it. Enter the file name, location and beginning line number where you added the changes, followed by the code. **(1 pt)**

**Project:** Contoso.Models

**Folder:** N/A

**File:** Customer.cs

**Starting Line Number:**  42 & 76

**Code:**

/// <summary>

/// Gets or sets the customer's middle name.

/// </summary>

public string MiddleName { get; set; }

public bool Equals(Customer other) =>

FirstName == other.FirstName &&

MiddleName == other.MiddleName &&

LastName == other.LastName &&

Company == other.Company &&

Email == other.Email &&

Phone == other.Phone &&

Address == other.Address;

}

1. Find the Customer View Model class (the intermediary class between the Model and the UI) and expose the MiddleName Property. Enter the file name, location and beginning line number where you added the changes, followed by the code. Hint: Look at the other properties for an example. **(2 pts)**

**Project:** Contoso.App

**Folder:** ViewModels

**File:** CustomerViewModel.cs

**Starting Line Number:** 87

**Code:**

/// <summary>

/// Gets or sets the customer's middle name.

/// </summary>

public string MiddleName

{

get => Model.MiddleName;

set

{

if (value != Model.MiddleName)

{

Model.FirstName = value;

IsModified = true;

OnPropertyChanged();

OnPropertyChanged(nameof(Name));

}

}

}

1. Find the Customer detail page and add a MiddleName text box to the General panel (between first name and last name). Two-way bind it to the MiddleName property you created in the previous question. Make sure the label reads “Middle Name” and that the previous first and last name fields are still visible. Enter the file name, location and beginning line number where you added the changes, followed by the code. **(3 pts)**

**Note:** The customer list in the starting view won’t show after you make your changes and run the app. That’s fine.

**Project:** Contoso.App

**Folder:** Views

**File:** CustomerDetailPage.xaml

**Starting Line Number:** 128

**Code:**

<TextBox

x:Name="MiddleName"

MinWidth="120"

Margin="0,8,16,8"

Header="Middle name"

IsReadOnly="{x:Bind vm:Converters.Not(ViewModel.IsInEdit),Mode=OneWay}"

RelativePanel.AlignLeftWithPanel="True"

Text="{x:Bind ViewModel.MiddleName, Mode=TwoWay}"

Visibility="{x:Bind ViewModel.IsInEdit, Mode=OneWay}" />

1. Make sure the app builds and deploys without errors. Do a Build Clean then zip and upload the entire project to the dropbox on Brightspace. (**1 pt** for running with no unexpected errors)
2. Double-check that you’re finished, then upload this Word document to the Brightspace dropbox.