Lab Assignment 4
Keira Wong

CODE (x2 files)

PART I: EXPENSEIT **********

ExpenseItHome.xaml

```
<Page x:Class="ExpenseIt.ExpenseItHome"
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
      xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
      xmlns:local="clr-namespace:ExpenseIt"
      mc:Ignorable="d"
      d:DesignHeight="350" d:DesignWidth="500"
      Title="ExpenseIt - Home">
    <Grid Margin="10, 0, 10, 10">

        <Grid.Resources>
            <!-- Expense Report Data -->
            <XmlDataProvider x:Key="ExpenseDataSource" XPath="Expenses">
                <x:XData>
                    <Expenses xmlns="">
                        <Person Name="Mike" Department="Legal">
                            <Expense ExpenseType="Lunch" ExpenseAmount="50" />
                            <Expense ExpenseType="Transportation" ExpenseAmount="50" />
                        </Person>
                        <Person Name="Lisa" Department="Marketing">
                            <Expense ExpenseType="Document printing"
         ExpenseAmount="50"/>
                            <Expense ExpenseType="Gift" ExpenseAmount="125" />
                        </Person>
                        <Person Name="John" Department="Engineering">
                            <Expense ExpenseType="Magazine subscription"
        ExpenseAmount="50"/>
                            <Expense ExpenseType="New machine" ExpenseAmount="600" />
                            <Expense ExpenseType="Software" ExpenseAmount="500" />
                        </Person>
                        <Person Name="Mary" Department="Finance">
                            <Expense ExpenseType="Dinner" ExpenseAmount="100" />
```

```
                </Person>
            </Expenses>
        </x:XData>
    </XmlDataProvider>

    <!-- Name item template -->
    <DataTemplate x:Key="nameItemTemplate">
        <Label Content="{Binding XPath=@Name}"/>
    </DataTemplate>
</Grid.Resources>

<Grid.Background>
    <ImageBrush ImageSource="watermark.png"  />
</Grid.Background>

<Grid.ColumnDefinitions>
    <ColumnDefinition Width="230" />
    <ColumnDefinition />
</Grid.ColumnDefinitions>

<Grid.RowDefinitions>
    <RowDefinition/>
    <RowDefinition Height="Auto"/>
    <RowDefinition />
    <RowDefinition Height="Auto"/>
</Grid.RowDefinitions>

<!-- People list -->

<Label Grid.Column="1" Style="{StaticResource headerTextStyle}" >
    View Expense Report
</Label>

<Border Grid.Column="1" Grid.Row="1" Style="{StaticResource listHeaderStyle}">
    <Label Style="{StaticResource listHeaderTextStyle}">Names</Label>
</Border>

<ListBox Name="peopleListBox" Grid.Column="1" Grid.Row="2"
 ItemsSource="{Binding Source={StaticResource ExpenseDataSource}, XPath=Person}"
 ItemTemplate="{StaticResource nameItemTemplate}">
</ListBox>

<!-- View report button -->
<Button Grid.Column="1" Grid.Row="3" Click="Button_Click" Style="{StaticResource
buttonStyle}">View</Button>

  </Grid>
</Page>
```

ExpenseItReportPage.xaml

```
<Page x:Class="ExpenseIt.ExpenseReportPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
```

```
  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
  xmlns:local="clr-namespace:ExpenseIt"
  mc:Ignorable="d"
  d:DesignHeight="350" d:DesignWidth="500"
  Title="ExpenseIt - View Expense">

<Grid>
    <Grid.Background>
        <ImageBrush ImageSource="watermark.png" />
    </Grid.Background>
    <Grid.ColumnDefinitions>
        <ColumnDefinition Width="230" />
        <ColumnDefinition />
    </Grid.ColumnDefinitions>
    <Grid.RowDefinitions>
        <RowDefinition Height="Auto" />
        <RowDefinition />
    </Grid.RowDefinitions>


    <Label Grid.Column="1" Style="{StaticResource headerTextStyle}">
        Expense Report For:
    </Label>
    <Grid Margin="10" Grid.Column="1" Grid.Row="1">

        <Grid.ColumnDefinitions>
            <ColumnDefinition />
            <ColumnDefinition />
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="Auto" />
            <RowDefinition Height="Auto" />
            <RowDefinition />
        </Grid.RowDefinitions>

        <!-- Name -->
        <StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="0" Orientation="Horizontal">
            <Label Style="{StaticResource labelStyle}">Name:</Label>
            <Label Style="{StaticResource labelStyle}" Content="{Binding XPath=@Name}"></Label>
        </StackPanel>

        <!-- Department -->
        <StackPanel Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="1" Orientation="Horizontal">
            <Label Style="{StaticResource labelStyle}">Department:</Label>
            <Label Style="{StaticResource labelStyle}" Content="{Binding XPath=@Department}"></Label>
        </StackPanel>

        <Grid Grid.Column="0" Grid.ColumnSpan="2" Grid.Row="2" VerticalAlignment="Top"
        HorizontalAlignment="Left">

            <!--Templates to display expense report data-->
            <Grid.Resources>
                <!-- Reason item template -->
                <DataTemplate x:Key="typeItemTemplate">
                    <Label Content="{Binding XPath=@ExpenseType}"/>
```

```
            </DataTemplate>
            <!-- Amount item template -->
            <DataTemplate x:Key="amountItemTemplate">
                <Label Content="{Binding XPath=@ExpenseAmount}"/>
            </DataTemplate>
        </Grid.Resources>

        <!-- Expense type and Amount table -->
        <DataGrid ItemsSource="{Binding XPath=Expense}" ColumnHeaderStyle="{StaticResource
columnHeaderStyle}" AutoGenerateColumns="False" RowHeaderWidth="0" >

            <DataGrid.Columns>
                <DataGridTemplateColumn Header="ExpenseType" CellTemplate="{StaticResource
typeItemTemplate}" />
                <DataGridTemplateColumn Header="Amount" CellTemplate="{StaticResource
amountItemTemplate}" />
            </DataGrid.Columns>

        </DataGrid>
      </Grid>
    </Grid>
  </Grid>
</Page>
```

MainWindow.xaml

```
<NavigationWindow x:Class="ExpenseIt.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:ExpenseIt"
    mc:Ignorable="d"
    Title="ExpenseIt" Height="350" Width="500" Source="ExpenseItHome.xaml">
</NavigationWindow>
```

PART II: CALCULATOR **********

MainWindow.xaml

```
<Window x:Class="Calculator.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:Calculator"
    mc:Ignorable="d"
    Title="Calculator" Height="430" Width="280">
```

```
    <Grid> <!-- This Window will be defined as a GRID of ... -->

        <!-- 4 COLUMNS -->
        <Grid.ColumnDefinitions>
            <ColumnDefinition/>
            <ColumnDefinition/>
            <ColumnDefinition/>
            <ColumnDefinition/>
        </Grid.ColumnDefinitions>

        <!-- 6 ROWS -->
        <Grid.RowDefinitions>
            <RowDefinition/>
            <RowDefinition/>
            <RowDefinition/>
            <RowDefinition/>
            <RowDefinition/>
        </Grid.RowDefinitions>

        <!-- Numbers -->
        <Button x:Name="seven" Grid.Row="1" Grid.Column="0" Click="seven_Click">7</Button> <!--
Designate "7" to the 3rd Row, 1st Column -->
        <Button x:Name="eight" Grid.Row="1" Grid.Column="1" Click="eight_Click">8</Button>
        <Button x:Name="nine" Grid.Row="1" Grid.Column="2" Click="nine_Click">9</Button>
        <Button x:Name="four" Grid.Row="2" Grid.Column="0" Click="four_Click">4</Button>
        <Button x:Name="five" Grid.Row="2" Grid.Column="1" Click="five_Click">5</Button>
        <Button x:Name="six" Grid.Row="2" Grid.Column="2" Click="six_Click">6</Button>
        <Button x:Name="one" Grid.Row="3" Grid.Column="0" Click="one_Click">1</Button>
        <Button x:Name="two" Grid.Row="3" Grid.Column="1" Click="two_Click">2</Button>
        <Button x:Name="three" Grid.Row="3" Grid.Column="2" Click="three_Click">3</Button>
        <Button x:Name="zero" Grid.Row="4" Grid.Column="1" Click="zero_Click">0</Button>

        <!-- Operators -->
        <Button x:Name="multiply" Grid.Row="1" Grid.Column="3" Click="multiply_Click">*</Button>
        <Button x:Name="divide" Grid.Row="2" Grid.Column="3" Click="divide_Click">/</Button>
        <Button x:Name="subtract" Grid.Row="3" Grid.Column="3" Click="subtract_Click">-</Button>
        <Button x:Name="add" Grid.Row="4" Grid.Column="3" Click="add_Click">+</Button>
        <Button x:Name="equals" Grid.Row="4" Grid.Column="2" Click="equals_Click">=</Button>

        <Button x:Name="clear" Grid.Row="4" Grid.Column="0" Click="clear_Click">C</Button>

        <!-- Entry Box -->
        <TextBox x:Name="entrybox"
                 Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="4"
                 IsReadOnly="True" FontSize="40" FontWeight="Bold"
                 BorderThickness="0" Padding="0, 0, 16, 0"
                 TextAlignment="Right" VerticalAlignment="Center" >0</TextBox>

    </Grid>
</Window>
```

MainWindow.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.Navigation;
using System.Windows.Shapes;

namespace Calculator
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        long firstEntry = 0;
        long secondEntry = 0;
        string operation = "";

        public MainWindow()
        {
            InitializeComponent();
        }

        private void zero_Click(object sender, RoutedEventArgs e) // IF "zero" is clicked ...
        {
            if (operation == "") // AND no operation has been selected,
            {
                // 1. "concatenate" a zero to the end of the firstEntry,
                firstEntry = (firstEntry * 10) + 0;
                // 2. display firstEntry.
                entrybox.Text = firstEntry.ToString();
            }
            else
            {
                // 1. "concatenate" a zero to the end of the secondEntry,
                secondEntry = (secondEntry * 10) + 0;
                // 2. display secondEntry.
                entrybox.Text = secondEntry.ToString();
            }
        }

        private void one_Click(object sender, RoutedEventArgs e) // IF "one" is clicked ...
        {
            if (operation == "") // AND no operation has been selected,
            {
                // 1. "concatenate" a one to the end of the firstEntry,
                firstEntry = (firstEntry * 10) + 1;
                // 2. display firstEntry.
```

```
        entrybox.Text = firstEntry.ToString();
    }
    else {
        // 1. "concatenate" a one to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 1;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void two_Click(object sender, RoutedEventArgs e) // IF "two" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a two to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 2;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
    {
        // 1. "concatenate" a two to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 2;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void three_Click(object sender, RoutedEventArgs e) // IF "three" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a three to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 3;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
    {
        // 1. "concatenate" a three to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 3;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void four_Click(object sender, RoutedEventArgs e) // IF "four" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a four to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 4;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
```

```
    {
        // 1. "concatenate" a four to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 4;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void five_Click(object sender, RoutedEventArgs e) // IF "five" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a five to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 5;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
    {
        // 1. "concatenate" a five to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 5;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void six_Click(object sender, RoutedEventArgs e) // IF "six" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a six to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 6;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
    {
        // 1. "concatenate" a six to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 6;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void seven_Click(object sender, RoutedEventArgs e) // IF "seven" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a seven to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 7;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
    {
        // 1. "concatenate" a seven to the end of the secondEntry,
```

```
        secondEntry = (secondEntry * 10) + 7;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void eight_Click(object sender, RoutedEventArgs e) // IF "eight" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a eight to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 8;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
    {
        // 1. "concatenate" a eight to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 8;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void nine_Click(object sender, RoutedEventArgs e) // IF "nine" is clicked ...
{
    if (operation == "") // AND no operation has been selected,
    {
        // 1. "concatenate" a eight to the end of the firstEntry,
        firstEntry = (firstEntry * 10) + 9;
        // 2. display firstEntry.
        entrybox.Text = firstEntry.ToString();
    }
    else
    {
        // 1. "concatenate" a nine to the end of the secondEntry,
        secondEntry = (secondEntry * 10) + 9;
        // 2. display secondEntry.
        entrybox.Text = secondEntry.ToString();
    }
}

private void multiply_Click(object sender, RoutedEventArgs e) // if "multiply" is clicked,
{
    // 1. set operation to "*"
    operation = "*";
    // 2. reset entrybox to display "0"
    entrybox.Text = "0";
}

private void divide_Click(object sender, RoutedEventArgs e) // if "divide" is clicked,
{
    // 1. set operation to "/"
    operation = "/";
    // 2. reset entrybox to display "0"
    entrybox.Text = "0";
```

```
}

private void subtract_Click(object sender, RoutedEventArgs e) // if "subtract" is clicked,
{
    // 1. set operation to "-"
    operation = "-";
    // 2. reset entrybox to display "0"
    entrybox.Text = "0";
}

private void add_Click(object sender, RoutedEventArgs e) // if "add" is clicked,
{
    // 1. set operation to "+"
    operation = "+";
    // 2. reset entrybox to display "0"
    entrybox.Text = "0";
}

private void equals_Click(object sender, RoutedEventArgs e) // if "equals" is clicked,
{
    long result = 0;

    // 1. determine the result
    if (operation == "*") {
        result = firstEntry * secondEntry;
    }
    else if (operation == "/") {
        result = firstEntry / secondEntry;
    }
    else if (operation == "-")
    {
        result = firstEntry - secondEntry;
    }
    else if (operation == "+")
    {
        result = firstEntry + secondEntry;
    }

    // 2. display result
    entrybox.Text = (result).ToString();

    // 3. reset firstEntry, secondEntry, and operation
    firstEntry = 0;
    secondEntry = 0;
    operation = "";
}

private void clear_Click(object sender, RoutedEventArgs e)
{
    // 1. reset firstEntry, secondEntry, and operation
    firstEntry = 0;
    secondEntry = 0;
    operation = "";

    // 2. "clear" the entrybox; display "0"
    entrybox.Text = "0";
```

```
    }
  }
}
```

PART III: WPF & WVMM APP **********

ReceiverView.xaml

```
<UserControl x:Class="HelloWorldWPFMVVMApp.View.ReceiverView"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:local="clr-namespace:HelloWorldWPFMVVMApp.View"
        mc:Ignorable="d"
        d:DesignHeight="300" d:DesignWidth="300" DataContext="{Binding Source={StaticResource
Locator}, Path=ReceiverViewModel}">
  <Grid>
    <Label Content="Receiver" Margin="90,34,0,232"/>
    <Label Content="{Binding ContentText}" FontSize="20" Margin="65,255,40,0"/>
  </Grid>
</UserControl>
```

SenderView.xaml

```
<UserControl x:Class="HelloWorldWPFMVVMApp.View.SenderView"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:i="http://schemas.microsoft.com/expression/2010/interactivity"
        xmlns:Command="http://www.galasoft.ch/mvvmlight"
        xmlns:local="clr-namespace:HelloWorldWPFMVVMApp.View"
        mc:Ignorable="d"
        d:DesignHeight="300" d:DesignWidth="300" DataContext="{Binding Source={StaticResource
Locator}, Path=SenderViewModel}">
  <Grid>
    <Label Content="Sender" Margin="90,34,0,232"/>
    <TextBox HorizontalAlignment="Left" Width="120" Height="20" Margin="50,266,0,11" Text="{Binding
TextBoxText}"/>
    <Button Content="Send" Width="50" Height="25" Margin="183,265,67,10">
      <i:Interaction.Triggers>
        <i:EventTrigger EventName="Click">
          <Command:EventToCommand Command="{Binding OnClickCommand}" />
        </i:EventTrigger>
      </i:Interaction.Triggers>
    </Button>
  </Grid>
</UserControl>
```

ReceiverViewModelWO.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using GalaSoft.MvvmLight; //For mvvmlight
using GalaSoft.MvvmLight.Command;
using GalaSoft.MvvmLight.Messaging;//for class Messenger
using HelloWorldWPFMVVMApp.Messages;

namespace HelloWorldWPFMVVMApp.ViewModel
{
    public class ReceiverViewModelWO : ViewModelBase
    {
        private string _contentText;

        public string ContentText
        {
            get { return _contentText; }
            set
            {
                _contentText = value;
                RaisePropertyChanged("ContentText");
            }
        }

        public ReceiverViewModelWO()
        {
            Messenger.Default.Register<ViewModelMessageWO>(this, OnReceiveMessageAction);
        }

        private void OnReceiveMessageAction(ViewModelMessageWO obj)
        {
            ContentText = obj.Text;
        }
    }
}
```

SenderViewModelWO.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using GalaSoft.MvvmLight; //For mvvmlight
using GalaSoft.MvvmLight.Command;
using GalaSoft.MvvmLight.Messaging;//for class Messenger
using HelloWorldWPFMVVMApp.Messages;
```

```
namespace HelloWorldWPFMVVMApp.ViewModel
{
    public class SenderViewModelWO : ViewModelBase
    {
        private String _textBoxText;
        public RelayCommand OnClickCommand { get; set; }

        public string TextBoxText
        {
            get
            { return _textBoxText; }

            set
            {
                _textBoxText = value;
                RaisePropertyChanged("TextBoxText");
            }
        }

        public SenderViewModelWO()
        {
            OnClickCommand = new RelayCommand(OnClickCommandAction, null);
        }

        private void OnClickCommandAction()
        {
            var viewModelMessage = new ViewModelMessageWO()
            {
                Text = TextBoxText
            };
            Messenger.Default.Send(viewModelMessage);
        }

    }
}
```

ViewModelLocatorWO.cs

```
/*
  In App.xaml:
  <Application.Resources>
      <vm:ViewModelLocator xmlns:vm="clr-namespace:wpfmvvm"
                           x:Key="Locator" />
  </Application.Resources>

  In the View:
  DataContext="{Binding Source={StaticResource Locator}, Path=ViewModelName}"

  You can also use Blend to do all this with the tool's support.
  See http://www.galasoft.ch/mvvm
*/

using GalaSoft.MvvmLight;
```

```
using GalaSoft.MvvmLight.Ioc;
//using Microsoft.Practices.ServiceLocation;
using CommonServiceLocator;

namespace HelloWorldWPFMVVMApp.ViewModel
{
    /// <summary>
    /// This class contains static references to all the view models in the
    /// application and provides an entry point for the bindings.
    /// </summary>
    public class ViewModelLocatorWO
    {
        /// <summary>
        /// Initializes a new instance of the ViewModelLocator class.
        /// </summary>
        public ViewModelLocatorWO()
        {
            ServiceLocator.SetLocatorProvider(() => SimpleIoc.Default);

            ////if (ViewModelBase.IsInDesignModeStatic)
            ////{
            ////    // Create design time view services and models
            ////    SimpleIoc.Default.Register<IDataService, DesignDataService>();
            ////}
            ////else
            ////{
            ////    // Create run time view services and models
            ////    SimpleIoc.Default.Register<IDataService, DataService>();
            ////}

            SimpleIoc.Default.Register<SenderViewModelWO>();
            SimpleIoc.Default.Register<ReceiverViewModelWO>();
        }

        public SenderViewModelWO SenderViewModel
        {
            get
            {
                return ServiceLocator.Current.GetInstance<SenderViewModelWO>();
            }
        }
        public ReceiverViewModelWO ReceiverViewModel
        {
            get
            {
                return ServiceLocator.Current.GetInstance<ReceiverViewModelWO>();
            }
        }


        public static void Cleanup()
        {
            // TODO Clear the ViewModels
        }
    }
}
```

ViewModelMessageWO.cs

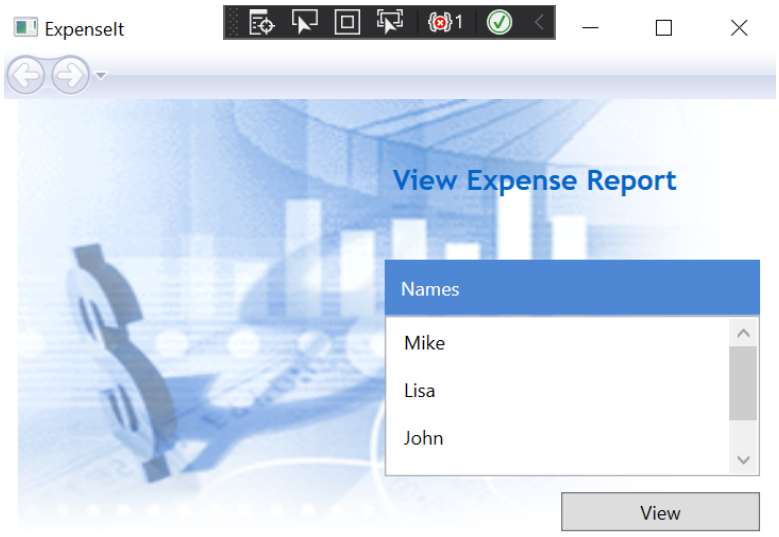
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using GalaSoft.MvvmLight.Messaging;

namespace HelloWorldWPFMVVMApp.Messages
{
    class ViewModelMessageWO : MessageBase
    {
        public string Text { get; set; }
    }
}
```

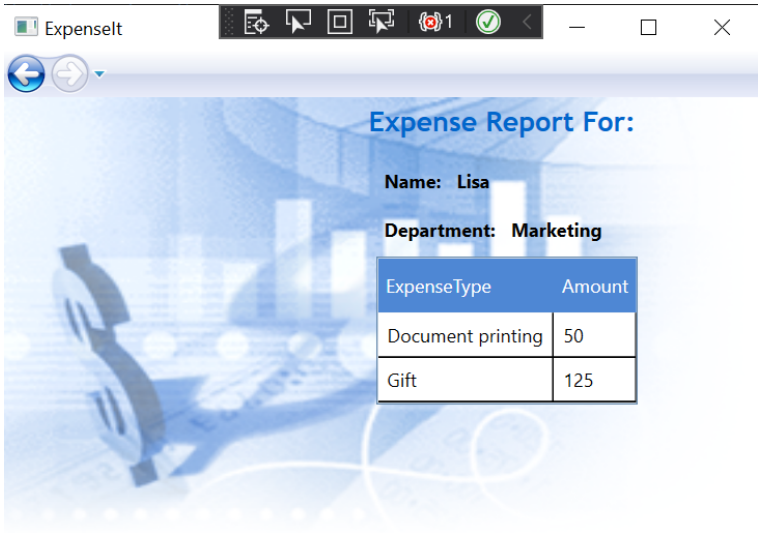
MainWindow.xaml

```
<Window x:Class="HelloWorldWPFMVVMApp.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:HelloWorldWPFMVVMApp"
        xmlns:view="clr-namespace:HelloWorldWPFMVVMApp.View"
        mc:Ignorable="d"
        Title="MainWindow" Height="350" Width="525">
    <Grid>
        <Grid>
            <Grid.RowDefinitions>
                <RowDefinition Height="Auto" />
            </Grid.RowDefinitions>
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width="Auto"/>
                <ColumnDefinition Width="Auto"/>
            </Grid.ColumnDefinitions>
            <view:ReceiverView Grid.Row="0" Grid.Column="0" />
            <view:SenderView Grid.Row="0" Grid.Column="1"  />
            <GridSplitter HorizontalAlignment="Left" Width="5" Height="320" Margin="245,0,0,-21"/>
        </Grid>
    </Grid>
</Window>
```

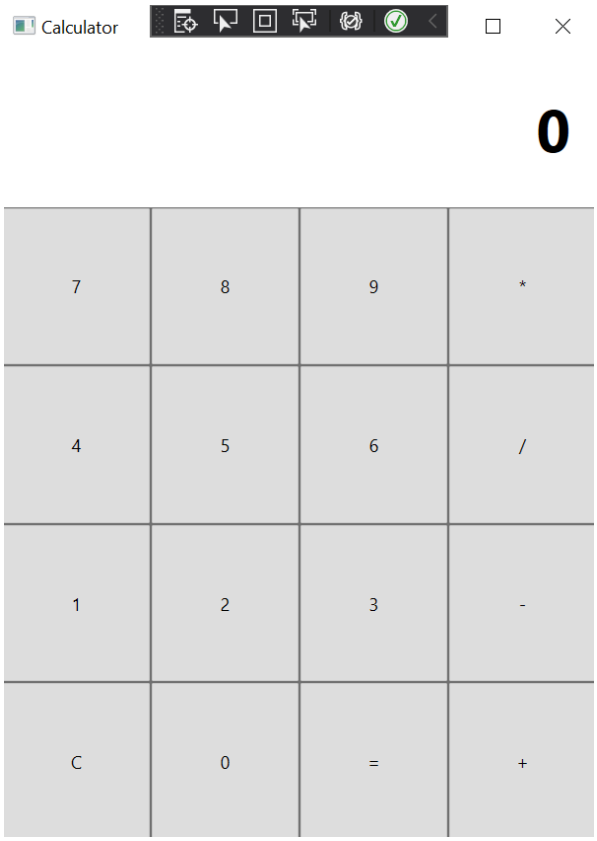
OUTPUT

Part I: ExpenseIt

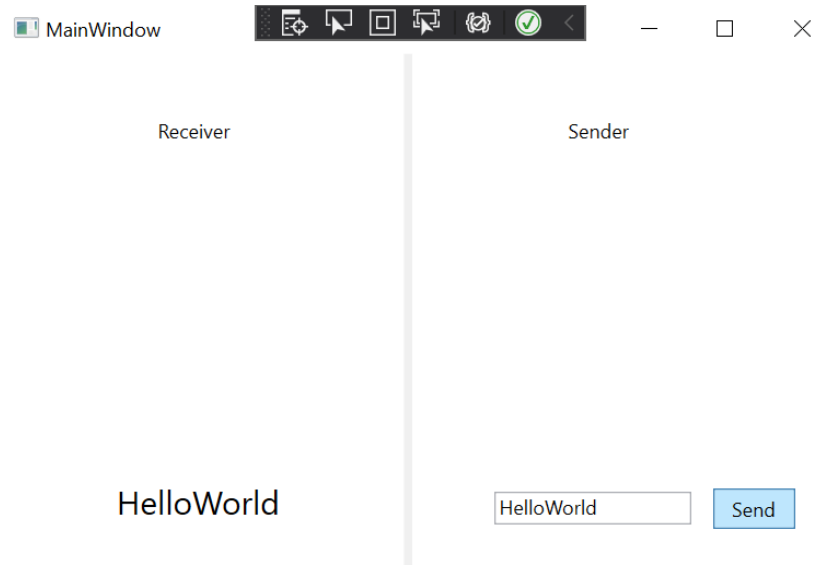




Part II: Calculator



Part III: WPF & WVMM App



TEAM MEMBER'S WORK

Keira Wong

- Part I: ExpenseIt (Code from Tutorial)
- Part II: Calculator
- Part III: WPF & WVMM App (Code from Tutorial)