```xml
<Application x:Class="Assignment3Mvvm.App"
             xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
             xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
             xmlns:local="clr-namespace:Assignment3Mvvm"
             StartupUri="MainWindow.xaml"
             xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
             d1p1:Ignorable="d"
             xmlns:d1p1="http://schemas.openxmlformats.org/markup-compatibility/2006">
  <Application.Resources>
    <ResourceDictionary>
            <vm:ViewModelLocator x:Key="ViewModelLocator" d:IsDataSource="True" xmlns:vm="clr-namespace:Assignment3Mvvm.ViewModel" />
        </ResourceDictionary>
  </Application.Resources>
</Application>
```

```
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Ioc;
using GalaSoft.MvvmLight.Messaging;
using GalaSoft.MvvmLight.Command;
using System.Collections.ObjectModel;
using GalaSoft.MvvmLight.Views;
using Assignment3Mvvm.Model;
using System.Windows.Input;
using Assignment3Mvvm.Views;
using Assignment3Mvvm.Messages;
using System.Windows;

namespace Assignment3Mvvm.ViewModel
{
    /// <summary>
    /// This class contains properties that the main View can data bind to.
    /// <para>
    /// Use the <strong>mvvminpc</strong> snippet to add bindable properties to this ViewModel.
    /// </para>
    /// <para>
    /// You can also use Blend to data bind with the tool's support.
    /// </para>
    /// <para>
    /// See http://www.galasoft.ch/mvvm
    /// </para>
    /// </summary>
    public class MainViewModel : ViewModelBase
    {
        /// <summary>
        /// Initializes a new instance of the MainViewModel class.
        /// </summary>
        private MembershipList members;
        private AddMemberView addMemberView;
        private Member selectedMember;
        public ICommand ShowAddCommand { get; private set; }
        public ICommand DeleteMemberCommand { get; private set; }
        public ICommand ExitAppCommand { get; private set; }

        public ICommand UpdateMemberCommand { get; private set; }
        public MainViewModel()
        {
            ////if (IsInDesignMode)
            ////{
            ////    // Code runs in Blend --> create design time data.
            ////}
            ////else
            ////{
            ////    // Code runs "for real"
            ////}
            members = new MembershipList();
            members.Write();
            ShowAddCommand = new RelayCommand(ShowAddMethod);
            DeleteMemberCommand = new RelayCommand(DeleteMemberMethod);
            ExitAppCommand = new RelayCommand(ExitAppMethod);
            Messenger.Default.Register<ViewModelMessage>(this, OnReceiveMessageAction);

        }

        public ObservableCollection<Member> MemberList
        {
            get
            {
                return members.Members;
            }
        }

        public Member SelectedMember
        {
            get { return selectedMember; }
            set
            {
                selectedMember = value;
                RaisePropertyChanged("SelectedMember");
            }
        }

        public AddMemberView AddMemberViewWindow { get => addMemberView; set => addMemberView = value; }

        public void ShowAddMethod()
        {
            addMemberView = new AddMemberView();
            addMemberView.Show();
        }

        public void DeleteMemberMethod()
        {
            members.Remove(selectedMember);
            members.Save();
            this.RaisePropertyChanged(() => this.MemberList);

        }

        private void OnReceiveMessageAction(ViewModelMessage m)
        {
            if (m.WindowCommandText != null && m.WindowCommandText.Equals
              ("Close"))
            {
                addMemberView.Close();
            }
            else
            {
                members.Add(new Member(m.FirstNameText, m.LastNameText, m.EmailText));
                addMemberView.Close();
                members.Save();
                this.RaisePropertyChanged(() => this.MemberList);
            }
        }

        public void ExitAppMethod()
        {
            System.Windows.Application.Current.Shutdown();
        }
    }
}
```

```xml
<Window x:Class="Assignment3Mvvm.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:Assignment3Mvvm"
        mc:Ignorable="d"
        Title="Membership Maintenance" Height="500" Width="800" ResizeMode="NoResize">
    <DockPanel LastChildFill="True" DataContext="{Binding MainViewModel, Source={StaticResource ViewModelLocator}}">
        <TextBlock Text="Customers: " FontWeight="Bold" DockPanel.Dock="Top" Padding="0,20,0,0" HorizontalAlignment="Center"/>
        <ListBox DockPanel.Dock="Left" BorderBrush="Black" BorderThickness="2" HorizontalAlignment="Center" VerticalAlignment="Center" ItemsSource="{Binding MemberList}"
                 DisplayMemberPath="GetDisplayText" SelectedItem="{Binding SelectedMember}" Width="500" Height="400">
            <ListBox.InputBindings>
                <!-- Here is where double click for update member would be -->
            </ListBox.InputBindings>
        </ListBox>
        <DockPanel DockPanel.Dock="Right">
            <StackPanel Orientation="Vertical" VerticalAlignment="Center">
                <Button Content="Add" Width="150" Height="25" Command="{Binding ShowAddCommand}" />
                <Button Content="Delete" Width="150" Height="25" Command="{Binding DeleteMemberCommand}"/>
                <Button Content="Exit" Width="150" Height="25" Command="{Binding ExitAppCommand}" />
            </StackPanel>
        </DockPanel>
    </DockPanel>
</Window>

```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Messaging;
using GalaSoft.MvvmLight.Command;
using System.Windows.Input;
using System.Collections.ObjectModel;

namespace Assignment3Mvvm.Model
{
    public class Member : ObservableObject
    {
        private string firstName;
        private string lastName;
        private string email;

        public Member()
        {

        }

        public Member(string _firstName, string _lastName, string _email)
        {
            firstName = _firstName;
            lastName = _lastName;
            email = _email;
        }

        public string FirstName
        {
            get
            {
                return firstName;
            }
            set
            {
                Set<string>(() => this.FirstName, ref firstName, value);
            }
        }

        public string LastName
        {
            get
            {
                return lastName;
            }
            set
            {
                Set<string>(() => this.LastName, ref lastName, value);
            }
        }

        public string Email
        {
            get
            {
                return email;
            }
            set
            {
                Set<string>(() => this.Email, ref email, value);
            }
        }

        public string GetDisplayText
        {
            get { return firstName + " " + lastName + " - " + email; }
        }
    }
}

```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.Threading.Tasks;
using System.Windows.Input;
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Messaging;
using System.Collections.ObjectModel;

namespace Assignment3Mvvm.Model
{
    public class MembershipData
    {
        private string filename;
        public MembershipData()
        {
            filename = "Memberships.txt";
        }

        public ObservableCollection<Member> GetMemberships()
        {
            ObservableCollection<Member> m = new ObservableCollection<Member>();
            string currentLine;
            try
            {
                System.IO.StreamReader file = new System.IO.StreamReader(filename);
                while ((currentLine = file.ReadLine()) != null)
                {
                    string[] memberFields = currentLine.Split(',');
                    m.Add(new Member(memberFields[0], memberFields[1], memberFields[2]));
                }
                file.Close();
                return m;
            }
            catch (FileNotFoundException e)
            {

            }
            return null;
        }

        public void SaveMemberships(ObservableCollection<Member> members)
        {
            string filename = "Memberships.txt";
            File.WriteAllText(filename, String.Empty);
            foreach (Member m in members)
            {
                string text = m.FirstName + "," + m.LastName + "," + m.Email + Environment.NewLine;
                File.AppendAllText(filename, text);
            }
        }
    }
}

```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Input;
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using System.Collections.ObjectModel;

namespace Assignment3Mvvm.Model
{
    public class MembershipList : ObservableObject
    {
        public event ChangeHandler Changed;
        private ObservableCollection<Member> members;
        private int propertyCount;
        public MembershipList()
        {
            members = new ObservableCollection<Member>();
            propertyCount = 0;
        }

        public int PropertyCount
        {
            get
            {
                return propertyCount;
            }
            set
            {
                Set<int>(() => this.PropertyCount, ref propertyCount, value);
            }
        }

        public void Add(Member m)
        {
            members.Add(m);
            propertyCount++;
            OnChange(EventArgs.Empty);
        }

        public void Remove(Member m)
        {
            members.Remove(m);
            propertyCount--;
            OnChange(EventArgs.Empty);
        }

        public void Write()
        {
            MembershipData data = new MembershipData();
            ObservableCollection<Member> tempM = data.GetMemberships();
            if (tempM != null)
            {
                foreach (Member m in tempM)
                {
                    members.Add(m);
                }
            }
        }

        public void Save()
        {
            MembershipData data = new MembershipData();
            data.SaveMemberships(members);
        }

        public ObservableCollection<Member> Members
        {
            get
            {
                return members;
            }
            set
            {
                Set<ObservableCollection<Member>>(() => this.Members, ref members, value);
            }
        }

        protected virtual void OnChange(EventArgs e)
        {
            Changed?.Invoke(this);
        }


    }

    public delegate void ChangeHandler(MembershipList m);
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using Assignment3Mvvm.ViewModel;
using System.Windows;
using System.Windows.Input;

namespace Assignment3Mvvm.Model
{
    public class Validator
    {
        private static string title = "Entity Error";

        public static string Title
        {
            get
            {
                return title;
            }
            set
            {
                title = value;
            }
        }

        public static bool IsPresent(string textBox, string text)
        {
            if (text == null)
            {
                MessageBox.Show(textBox + " is required", Title);
                //textBox.Focus();
                return false;
            }
            return true;
        }

        public static bool IsDecimal(string textBox)
        {
            decimal number = 0m;
            if (Decimal.TryParse(textBox, out number))
            {
                return true;
            }
            else
            {
                MessageBox.Show(textBox + " must be a decimal value.", Title);
                //textBox.Focus();
                return false;
            }
        }

        public static bool IsInt32(string textBox)
        {
            int number = 0;
            if (Int32.TryParse(textBox, out number))
            {
                return true;
            }
            else
            {
                MessageBox.Show(textBox + " must be an integer.", Title);
                //textBox.Focus();
                return false;
            }
        }

        public static bool IsWithinRange(string textBox, decimal min, decimal max)
        {
            int number = textBox.Length;
            if (number < min || number > max)
            {
                MessageBox.Show(textBox + " must be between " + min + " and " + max + " characters.", Title);
                //textBox.Focus();
                return false;
            }
            return true;
        }

        public static bool IsValidEmail(string textBox)
        {
            if (textBox.IndexOf("@") == -1 || textBox.IndexOf(".") == -1)
            {
                MessageBox.Show(textBox + " must be a valid email address", Title);
                //textBox.Focus()
                return false;
            }
            return true;
        }
    }
}
```

```
/*
  In App.xaml:
  <Application.Resources>
      <vm:ViewModelLocator xmlns:vm="clr-namespace:Assignment3Mvvm"
                           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 System;
using System.Windows;
using Assignment3Mvvm.ViewModel;
using GalaSoft.MvvmLight.Messaging;

namespace Assignment3Mvvm.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 ViewModelLocator
    {
        /// <summary>
        /// Initializes a new instance of the ViewModelLocator class.
        /// </summary>
        public ViewModelLocator()
        {
            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<MainViewModel>();
            Messenger.Default.Register<NotificationMessage>(this, NotifyUserMethod);
            SimpleIoc.Default.Register<AddViewModel>();
        }

        public MainViewModel MainViewModel
        {
            get
            {
                return ServiceLocator.Current.GetInstance<MainViewModel>();
            }
        }

        public AddViewModel AddViewModel
        {
            get
            {
                return ServiceLocator.Current.GetInstance<AddViewModel>();
            }
        }

        private void NotifyUserMethod(NotificationMessage message)
        {
            MessageBox.Show(message.Notification);
        }

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

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

namespace Assignment3Mvvm.Messages
{
    public class ViewModelMessage : MessageBase
    {
        public string FirstNameText { get; set; }
        public string LastNameText { get; set; }
        public string EmailText { get; set; }
        public string WindowCommandText { get; set; }
    }
}

```

```xml
<Window x:Class="Assignment3Mvvm.Views.AddMemberView"
        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:Assignment3Mvvm.Views"
        mc:Ignorable="d"
        Title="Add Member" Height="400" Width="500" ResizeMode="NoResize">
    <Grid DataContext="{Binding AddViewModel, Source={StaticResource ViewModelLocator}}">
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="150" />
            <ColumnDefinition Width="350"/>
        </Grid.ColumnDefinitions>
        <Grid.RowDefinitions>
            <RowDefinition Height="80"/>
            <RowDefinition Height="80"/>
            <RowDefinition Height="80"/>
            <RowDefinition Height="80"/>
        </Grid.RowDefinitions>
        <TextBlock Grid.Row="0" Grid.Column="0" Text="First Name:" VerticalAlignment="Center" HorizontalAlignment="Center" FontWeight="Bold"/>
        <TextBox Grid.Row="0" Grid.Column="1" Height="30" Width="300" HorizontalAlignment="Right" Margin="0,0,30,0" BorderThickness="1" BorderBrush="Black" Text="{Binding FirstNameTextBox}" />
        <TextBlock Grid.Row="1" Grid.Column="0" Text="Last Name:" VerticalAlignment="Center" HorizontalAlignment="Center" FontWeight="Bold"/>
        <TextBox Grid.Row="1" Grid.Column="1" Height="30" Width="300" HorizontalAlignment="Right" Margin="0,0,30,0" BorderBrush="Black" BorderThickness="1" Text="{Binding LastNameTextBox}" />
        <TextBlock Grid.Row="2" Grid.Column="0" Text="Email:" HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold" />
        <TextBox Grid.Row="2" Grid.Column="1"  Height="30" Width="300" HorizontalAlignment="Right" Margin="0,0,30,0" BorderBrush="Black" BorderThickness="1" Text="{Binding EmailTextBox}"/>
        <Button Grid.ColumnSpan="2" Content="Save" HorizontalAlignment="Left" Margin="106,34.6,0,0" Grid.Row="3" VerticalAlignment="Top" Width="75" Command="{Binding AddCommand}"/>
        <Button Content="Cancel" Grid.Column="1" HorizontalAlignment="Left" Margin="165,34.6,0,0" Grid.Row="3" VerticalAlignment="Top" Width="75" Command="{Binding CancelCommand}"/>
    </Grid>
</Window>
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Input;
using Assignment3Mvvm.Views;
using Assignment3Mvvm.Messages;
using GalaSoft.MvvmLight.Messaging;
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using Assignment3Mvvm.Model;

namespace Assignment3Mvvm.ViewModel
{
    public class AddViewModel : ViewModelBase
    {
        public ICommand AddCommand { get; private set; }
        public ICommand CancelCommand { get; private set; }

        private string firstNameText;
        private string lastNameText;
        private string emailText;

        public AddViewModel()
        {
            AddCommand = new RelayCommand(OnClickAddMember, null);
            CancelCommand = new RelayCommand(OnClickCloseWindow, null);

        }

        public string FirstNameTextBox
        {
            get { return firstNameText; }
            set
            {
                firstNameText = value;
                RaisePropertyChanged("FirstNameTextBox");
            }
        }

        public string LastNameTextBox
        {
            get { return lastNameText; }
            set
            {
                lastNameText = value;
                RaisePropertyChanged("LastNameTextBox");
            }
        }

        public string EmailTextBox
        {
            get { return emailText; }
            set
            {
                emailText = value;
                RaisePropertyChanged("EmailTextBox");
            }
        }

        private void OnClickAddMember()
        {
            if (Validator.IsPresent("First name", firstNameText)
                && Validator.IsPresent("Last name", lastNameText)
                && Validator.IsPresent("Email", emailText)
                && Validator.IsValidEmail(emailText)
                && Validator.IsWithinRange(firstNameText, 1, 25)
                && Validator.IsWithinRange(lastNameText, 1, 25)
                && Validator.IsWithinRange(emailText, 1, 25))
            {
                var memberMessage = new ViewModelMessage()
                {
                    FirstNameText = FirstNameTextBox,
                    LastNameText = LastNameTextBox,
                    EmailText = EmailTextBox
                };
                string notificationText = String.Format("New member added: {0} {1} - {2}", firstNameText, lastNameText, emailText);
                firstNameText = null;
                lastNameText = null;
                emailText = null;
                Messenger.Default.Send<NotificationMessage>(new NotificationMessage(notificationText));
                Messenger.Default.Send(memberMessage);
            }
        }

        private void OnClickCloseWindow()
        {
            var closeWindowMessage = new ViewModelMessage()
            {
                WindowCommandText = "Close",
                FirstNameText = null,
                LastNameText = null,
                EmailText = null
            };
            Messenger.Default.Send(closeWindowMessage);
        }
    }
}

```