```
 1  using GalaSoft.MvvmLight;
 2  using CustomerMaintenance.Model;
 3  using System.Windows.Input;
 4  using System.Data;
 5  using GalaSoft.MvvmLight.Messaging;
 6  using GalaSoft.MvvmLight.Threading;
 7  using GalaSoft.MvvmLight.Command;
 8  using CustomerMaintenance.Views;
 9  using System.Windows;
10  using System.Linq;
11  using System;
12  using CustomerMaintenance.Messages;
13  using System.Collections.Generic;
14  using System.Collections.ObjectModel;
15  using System.Data.Entity;
16
17  namespace CustomerMaintenance.ViewModel
18  {
19      public class AddViewModel : ViewModelBase
20      {
21          private ObservableCollection<string> stateList;
22          public ICommand AddCustomerCommand { get; private set; }
23          public ICommand CancelCommand { get; private set; }
24
25          private int id;
26          private string name;
27          private string address;
28          private string city;
29          private string state;
30          private string zipCode;
31
32          public AddViewModel()
33          {
34              stateList = new ObservableCollection<string>();
35              var query = from states in MMABookEntity.BookEntity.States select
                  states.StateName;
36              var result = query.ToList();
37              foreach(string r in result)
38              {
39                  stateList.Add(r);
40              }
41              RaisePropertyChanged("StateList");
42              AddCustomerCommand = new RelayCommand<IClosable>
                  (this.AddCustomerMethod);
43              CancelCommand = new RelayCommand<IClosable>(this.CancelMethod);;
44          }
45          public ObservableCollection<string> StateList
46          {
47              get{ return stateList; }
```

```
48              set
49              {
50                  stateList = value;
51                  RaisePropertyChanged("StateList");
52              }
53          }
54
55          public int ID
56          {
57              get { return id; }
58              set
59              {
60                  id = value;
61                  RaisePropertyChanged("ID");
62              }
63          }
64
65          public string Name
66          {
67              get { return name; }
68              set
69              {
70                  name = value;
71                  RaisePropertyChanged("Name");
72              }
73          }
74
75          public string Address
76          {
77              get { return address; }
78              set
79              {
80                  address = value;
81                  RaisePropertyChanged("Address");
82              }
83          }
84
85          public string City
86          {
87              get { return city; }
88              set
89              {
90                  city = value;
91                  RaisePropertyChanged("City");
92              }
93          }
94
95          public string State
96          {
```

```
97              get { return state; }
98              set
99              {
100                 state = value;
101                 RaisePropertyChanged("State");
102             }
103         }
104
105         public string ZipCode
106         {
107             get { return zipCode; }
108             set
109             {
110                 zipCode = value;
111                 RaisePropertyChanged("ZipCode");
112             }
113         }
114
115         public void AddCustomerMethod(IClosable window)
116         {
117             if (
118                 Validator.IsPresent("Name", name)
119                 && Validator.IsPresent("Address", address)
120                 && Validator.IsPresent("City", city)
121                 && Validator.IsPresent("State", state)
122                 && Validator.IsPresent("ZipCode", zipCode)
123                 && Validator.IsValidZipCode(zipCode)
124                 )
125             {
126                 var stateQuery = from states in MMABookEntity.BookEntity.States
                       where states.StateName == state select states.StateCode;
127                 string stateCode = stateQuery.FirstOrDefault();
128                 var idQuery = from cust in MMABookEntity.BookEntity.Customers
                      orderby cust.CustomerID descending select cust.CustomerID;
129                 id = idQuery.FirstOrDefault();
130                 Customer c = new Customer
131                 {
132                     CustomerID = id + 1,
133                     Name = name,
134                     Address = address,
135                     City = city,
136                     State = stateCode,
137                     ZipCode = zipCode,
138                 };
139                 MMABookEntity.BookEntity.Customers.Add(c);
140                 //MMABookEntity.BookEntity.Entry(c).State = EntityState.Added;
141                 try
142                 {
143                     MMABookEntity.BookEntity.SaveChanges();
```

```
144                     string customerString = String.Format("ID: {0}\nName: {1}
                        \nAddress: {2}\nCity: {3}\nState: {4}\nZipCode: {5}",
                        c.CustomerID, c.Name, c.Address, c.City, state, c.ZipCode);
145                     MessageBox.Show(customerString, "Customer Added",
                        MessageBoxButton.OK, MessageBoxImage.Asterisk);
146                 }
147                 catch (Exception e)
148                 {
149                     MessageBox.Show(e.ToString(), "Error", MessageBoxButton.OK,
                         MessageBoxImage.Exclamation);
150                 }
151                 if (window != null)
152                 {
153                     ResetValues();
154                     window.Close();
155                 }
156             }
157         }
158 
159         public void CancelMethod(IClosable window)
160         {
161             if (window != null)
162             {
163                 ResetValues();
164                 window.Close();
165             }
166         }
167 
168         public void ResetValues()
169         {
170             id = 0;
171             name = null;
172             address = null;
173             city = null;
174             state = null;
175             zipCode = null;
176             RaisePropertyChanged("ID");
177             RaisePropertyChanged("Name");
178             RaisePropertyChanged("Address");
179             RaisePropertyChanged("City");
180             RaisePropertyChanged("State");
181             RaisePropertyChanged("ZipCode");
182         }
183     }
184 }
185 
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using System.Windows;
7  using System.Windows.Controls;
8  using System.Windows.Data;
9  using System.Windows.Documents;
10 using System.Windows.Input;
11 using System.Windows.Media;
12 using System.Windows.Media.Imaging;
13 using System.Windows.Shapes;
14 using CustomerMaintenance.ViewModel;
15
16 namespace CustomerMaintenance.Views
17 {
18     /// <summary>
19     /// Interaction logic for AddView.xaml
20     /// </summary>
21     public partial class AddView : Window, IClosable
22     {
23         public AddView()
24         {
25             InitializeComponent();
26         }
27     }
28 }
29
```

```
<Window x:Class="CustomerMaintenance.Views.AddView"
        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:CustomerMaintenance.Views"
        x:Name="AddViewWindow"
        mc:Ignorable="d"
        Title="Add Customer" Height="350" Width="600">
    <Grid DataContext="{Binding AddViewModel, Source = {StaticResource
      ViewModelLocator}}">
        <Label Content="Name:" HorizontalAlignment="Left" Margin="69,49,0,0"
          VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="150,53,0,0"
          TextWrapping="Wrap" Text="{Binding Name}" VerticalAlignment="Top"
          Width="372"/>
        <Label Content="Address:" HorizontalAlignment="Left" Margin="69,108,0,0"
           VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="150,112,0,0"
          TextWrapping="Wrap" Text="{Binding Address}" VerticalAlignment="Top"
          Width="372"/>
        <Label Content="City:" HorizontalAlignment="Left" Margin="69,172,0,0"
          VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="150,176,0,0"
          TextWrapping="Wrap" Text="{Binding City}" VerticalAlignment="Top"
          Width="372"/>
        <Label Content="State:" HorizontalAlignment="Left" Margin="69,234,0,0"
          VerticalAlignment="Top"/>
        <ComboBox ItemsSource="{Binding StateList}" SelectedItem="{Binding
          State}" HorizontalAlignment="Left" Margin="129,236,0,0"
          VerticalAlignment="Top" Width="163"/>
        <Label Content="Zip Code:" HorizontalAlignment="Left"
          Margin="318,237,0,0" VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="384,239,0,0"
          TextWrapping="Wrap" Text="{Binding ZipCode}" VerticalAlignment="Top"
          Width="138"/>
        <Button Content="Accept" Command="{Binding AddCustomerCommand}"
          CommandParameter="{Binding ElementName=AddViewWindow}"
          HorizontalAlignment="Left" Margin="48,283,0,0" VerticalAlignment="Top"
           Width="75"/>
        <Button Content="Cancel" Command="{Binding CancelCommand}"
          CommandParameter="{Binding ElementName=AddViewWindow}"
          HorizontalAlignment="Left" Margin="466,283,0,0"
          VerticalAlignment="Top" Width="75"/>

    </Grid>
</Window>

```

```
<Application x:Class="CustomerMaintenance.App" xmlns="http://
  schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://
  schemas.microsoft.com/winfx/2006/xaml" xmlns:local="clr-
  namespace:CustomerMaintenance" 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:CustomerMaintenance.ViewModel" />
    </ResourceDictionary>
  </Application.Resources>
</Application>
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using GalaSoft.MvvmLight;
7  using GalaSoft.MvvmLight.Messaging;
8
9  namespace CustomerMaintenance.Messages
10 {
11     public class DataMessage : MessageBase
12     {
13         public string CommandText { get; set; }
14         public Customer CustomerMsg { get; set; }
15         public int? IDText { get; set; }
16         public string NameText { get; set; }
17         public string AddressText { get; set; }
18         public string CityText { get; set; }
19         public string StateText { get; set; }
20         public string ZipCodeText { get; set; }
21     }
22 }
23
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace CustomerMaintenance.ViewModel
 8  {
 9      public interface IClosable
10      {
11          void Close();
12      }
13  }
14
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6  using GalaSoft.MvvmLight;
 7  using System.Data.Entity;
 8  namespace CustomerMaintenance.Model
 9  {
10      public static class MMABookEntity
11      {
12          public static MMABooksEntities BookEntity = new MMABooksEntities();
13      }
14  }
15
```

```
 1  using GalaSoft.MvvmLight;
 2  using CustomerMaintenance.Model;
 3  using System.Windows.Input;
 4  using System.Data;
 5  using GalaSoft.MvvmLight.Messaging;
 6  using GalaSoft.MvvmLight.Threading;
 7  using GalaSoft.MvvmLight.Command;
 8  using CustomerMaintenance.Views;
 9  using System.Windows;
10  using System.Linq;
11  using System;
12  using CustomerMaintenance.Messages;
13  using System.Data.Entity.Infrastructure;
14  using System.Data.Entity;
15
16  namespace CustomerMaintenance.ViewModel
17  {
18      public class MainViewModel : ViewModelBase
19      {
20
21          private string name;
22          private string address;
23          private string city;
24          private string state;
25          private string zip;
26          private string id;
27          static Customer selectedCustomer;
28          public ICommand ShowModifyCommand { get; private set; }
29          public ICommand ShowAddCommand { get; private set; }
30          public ICommand DeleteCustomerCommand { get; private set; }
31          public ICommand ExitCommand { get; private set; }
32          public ICommand GetCustomerCommand { get; private set; }
33          /// <summary>
34          /// Initializes a new instance of the MainViewModel class.
35          /// </summary>
36          public MainViewModel()
37          {
38              ShowModifyCommand = new RelayCommand(ShowModifyMethod);
39              ShowAddCommand = new RelayCommand(ShowAddMethod);
40              DeleteCustomerCommand = new RelayCommand(DeleteCustomerMethod);
41              ExitCommand = new RelayCommand(ExitMethod);
42              GetCustomerCommand = new RelayCommand(GetCustomerMethod);
43              Messenger.Default.Register<DataMessage>(this,
                  OnReceiveMessageAction);
44          }
45
46          public string Name
47          {
48              get { return name; }
```

```
49              set
50              {
51                  name = value;
52                  RaisePropertyChanged("Name");
53              }
54          }
55
56          public string Address
57          {
58              get { return address; }
59              set
60              {
61                  address = value;
62                  RaisePropertyChanged("Address");
63              }
64          }
65
66          public string City
67          {
68              get { return city; }
69              set
70              {
71                  city = value;
72                  RaisePropertyChanged("City");
73              }
74          }
75
76          public string State
77          {
78              get { return state; }
79              set
80              {
81                  state = value;
82                  RaisePropertyChanged("State");
83              }
84          }
85
86          public string ZipCode
87          {
88              get { return zip; }
89              set
90              {
91                  zip = value;
92                  RaisePropertyChanged("ZipCode");
93              }
94          }
95
96          public string ID
97          {
```

```
            get { return id; }
            set
            {
                id = value;
                RaisePropertyChanged("ID");
            }
        }

        public void OnReceiveMessageAction(DataMessage m)
        {
            if (m.CommandText == "Modifications Complete")
            {
                var query = from cust in MMABookEntity.BookEntity.Customers
                  where cust.CustomerID == m.IDText select cust;
                selectedCustomer = query.FirstOrDefault();
                id = selectedCustomer.CustomerID.ToString();
                name = selectedCustomer.Name;
                address = selectedCustomer.Address;
                city = selectedCustomer.City;
                state = selectedCustomer.State;
                zip = selectedCustomer.ZipCode;
            }
            else if (m.CommandText == "Deleted")
            {
                id = m.IDText.ToString();
                name = m.NameText;
                address = m.AddressText;
                city = m.CityText;
                state = m.StateText;
                zip = m.ZipCodeText;
            }
            RaisePropertyChanged("Name");
            RaisePropertyChanged("Address");
            RaisePropertyChanged("City");
            RaisePropertyChanged("State");
            RaisePropertyChanged("ID");
            RaisePropertyChanged("ZipCode");
        }

        public void ShowAddMethod()
        {
            var addView = new AddView();
            addView.Show();
        }

        public void ShowModifyMethod()
        {
            if(selectedCustomer == null)
            {
```

```
146                 MessageBox.Show("Select a customer first");
147             }
148             else
149             {
150                 DataMessage m = new DataMessage()
151                 {
152                     CommandText = "Modify",
153                     CustomerMsg = selectedCustomer
154                 };
155 
156                 var modifyView = new ModifyView();
157                 modifyView.Show();
158                 Messenger.Default.Send(m);
159             }
160         }
161 
162         public void ResetTextBoxValues()
163         {
164             selectedCustomer = null;
165             id = null;
166             name = null;
167             address = null;
168             city = null;
169             state = null;
170             zip = null;
171             RaisePropertyChanged("Name");
172             RaisePropertyChanged("Address");
173             RaisePropertyChanged("City");
174             RaisePropertyChanged("State");
175             RaisePropertyChanged("ID");
176             RaisePropertyChanged("ZipCode");
177         }
178 
179         public void SendDeleteMessage()
180         {
181             DataMessage m = new DataMessage()
182             {
183                 CommandText = "Deleted",
184                 IDText = null,
185                 NameText = null,
186                 AddressText = null,
187                 CityText = null,
188                 StateText = null,
189                 ZipCodeText = null
190             };
191             Messenger.Default.Send(m);
192         }
193 
194         public void DeleteCustomerMethod()
```

```
195         {
196             if (id == null || address == null || city == null || state == null
                  || zip == null)
197             {
198                 MessageBox.Show("Select a customer first");
199             }
200             else
201             {
202                 var query = from cust in MMABookEntity.BookEntity.Customers
                      where cust.CustomerID.ToString() == id select cust;
203                 selectedCustomer = query.FirstOrDefault();
204                 try
205                 {
206                     if (MessageBox.Show("Confirm delete: " +
                        selectedCustomer.Name + "?", "Confirm Delete",
                        MessageBoxButton.OKCancel, MessageBoxImage.Question) ==
                        MessageBoxResult.OK)
207                     {
208                         MMABookEntity.BookEntity.Customers.Remove
                        (selectedCustomer);
209                         MMABookEntity.BookEntity.SaveChanges();
210                         MMABookEntity.BookEntity.Entry(selectedCustomer).State
                        = EntityState.Detached;
211                         SendDeleteMessage();
212                     }
213                 }
214                 catch (DbUpdateConcurrencyException dbe)
215                 {
216                     dbe.Entries.Single().Reload();
217                     if (MMABookEntity.BookEntity.Entry(selectedCustomer).State
                        == EntityState.Detached)
218                     {
219                         MessageBox.Show("Another user has deleted that
                        customer.", "Concurrency Error", MessageBoxButton.OK,
                        MessageBoxImage.Error);
220                         SendDeleteMessage();
221                     }
222                     else
223                     {
224                         MessageBox.Show("Another user has updated that
                        customer.", "Concurrency Error", MessageBoxButton.OK,
                        MessageBoxImage.Error);
225                         DataMessage m = new DataMessage()
226                         {
227                             CommandText = "Modifications Complete",
228                             IDText = selectedCustomer.CustomerID,
229                             NameText = null,
230                             AddressText = null,
231                             CityText = null,
```

```
232                                 StateText = null,
233                                 ZipCodeText = null
234                             };
235                             Messenger.Default.Send(m);
236                         }
237                     }
238                     catch (Exception ex)
239                     {
240                         MessageBox.Show(ex.Message, ex.GetType().ToString());
241                     }
242                 }
243             }
244 
245             public void ExitMethod()
246             {
247                 MMABookEntity.BookEntity.SaveChanges();
248                 System.Windows.Application.Current.Shutdown();
249             }
250 
251             public void GetCustomerMethod()
252             {
253                 if (!Validator.IsInt32(id))
254                 {
255                     ResetTextBoxValues();
256                 }
257                 else
258                 {
259                     var query = from cust in MMABookEntity.BookEntity.Customers
260                                 where cust.CustomerID.ToString() == id
261                                 select cust;
262                     selectedCustomer = query.FirstOrDefault();
263                     if (selectedCustomer == null)
264                     {
265                         MessageBox.Show("A customer with that ID does not exist.");
266                         ResetTextBoxValues();
267                     }
268                     else
269                     {
270                         try
271                         {
272                             name = selectedCustomer.Name;
273                             city = selectedCustomer.City;
274                             address = selectedCustomer.Address;
275                             state = selectedCustomer.State;
276                             zip = selectedCustomer.ZipCode;
277                             RaisePropertyChanged("Name");
278                             RaisePropertyChanged("City");
279                             RaisePropertyChanged("Address");
280                             RaisePropertyChanged("State");
```

```
281                         RaisePropertyChanged("ZipCode");
282                     }
283                     catch (DbUpdateConcurrencyException dbe)
284                     {
285                         dbe.Entries.Single().Reload();
286                     }
287                     catch (Exception ex)
288                     {
289                         MessageBox.Show(ex.Message, ex.GetType().ToString());
290                     }
291                 }
292             }
293         }
294     }
295 }
```

```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 CustomerMaintenance
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
    }
}

```

```
<Window x:Class="CustomerMaintenance.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:CustomerMaintenance"
        mc:Ignorable="d"
        Title="MainWindow" Height="400" Width="600">
    <Grid DataContext="{Binding MainViewModel, Source={StaticResource
      ViewModelLocator}}">
        <Label Content="Customer ID:" HorizontalAlignment="Left"
          Margin="66,24,0,0" VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Text="{Binding ID}" Height="23"
          Margin="157,24,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
          Width="120"/>
        <Button Content="Get Customer" Command="{Binding GetCustomerCommand}"
          HorizontalAlignment="Left" Margin="302,27,0,0" VerticalAlignment="Top"
           Width="105"/>

        <Label Content="Name:" HorizontalAlignment="Left" Margin="66,69,0,0"
          VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="157,69,0,0"
          IsReadOnly="True" TextWrapping="Wrap" Text="{Binding Name}"
          VerticalAlignment="Top" Width="360"/>

        <Label Content="Address:" HorizontalAlignment="Left" Margin="66,129,0,0"
           VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="157,133,0,0"
          IsReadOnly="True" TextWrapping="Wrap" Text="{Binding Address}"
          VerticalAlignment="Top" Width="360"/>

        <Label Content="City:" HorizontalAlignment="Left" Margin="72,188,0,0"
          VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="157,188,0,0"
          IsReadOnly="True" TextWrapping="Wrap" Text="{Binding City}"
          VerticalAlignment="Top" Width="360"/>

        <Label Content="State:" HorizontalAlignment="Left" Margin="72,247,0,0"
          VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="157,251,0,0"
          IsReadOnly="True" TextWrapping="Wrap" Text="{Binding State}"
          VerticalAlignment="Top" Width="120"/>

        <Label Content="Zip Code:" HorizontalAlignment="Left"
          Margin="291,247,0,0" VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="368,251,0,0"
          IsReadOnly="True" TextWrapping="Wrap" Text="{Binding ZipCode}"
          VerticalAlignment="Top" Width="149"/>

```

```
29          <Button Content="Add" Command="{Binding ShowAddCommand}"
              HorizontalAlignment="Left" Margin="45,312,0,0" VerticalAlignment="Top"
               Width="75"/>
30          <Button Content="Modify" Command="{Binding ShowModifyCommand}"
              HorizontalAlignment="Left" Margin="171,312,0,0"
              VerticalAlignment="Top" Width="75"/>
31          <Button Content="Delete" Command="{Binding DeleteCustomerCommand}"
              HorizontalAlignment="Left" Margin="291,312,0,0"
              VerticalAlignment="Top" Width="75"/>
32          <Button Content="Exit" Command="{Binding ExitCommand}"
              HorizontalAlignment="Left" Margin="460,312,0,0"
              VerticalAlignment="Top" Width="75"/>
33
34      </Grid>
35  </Window>
36
```

```
using GalaSoft.MvvmLight;
using CustomerMaintenance.Model;
using System.Windows.Input;
using System.Data;
using GalaSoft.MvvmLight.Messaging;
using GalaSoft.MvvmLight.Threading;
using GalaSoft.MvvmLight.Command;
using CustomerMaintenance.Views;
using System.Windows;
using System.Linq;
using System;
using CustomerMaintenance.Messages;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Data.Entity;
using System.Data.Entity.Infrastructure;

namespace CustomerMaintenance.ViewModel
{
    public class ModifyViewModel : ViewModelBase
    {
        private int? id;
        private string name;
        private string address;
        private string city;
        private string state;
        private string zipCode;
        static Customer selectedCustomer;
        private ObservableCollection<string> stateList;
        public ICommand SaveChangesCommand { get; private set; }
        public ICommand CancelCommand { get; private set; }
        public ModifyViewModel()
        {

            stateList = new ObservableCollection<string>();
            var query = from states in MMABookEntity.BookEntity.States select
              states.StateName;
            var result = query.ToList();
            foreach (string r in result)
            {
                stateList.Add(r);
            }
            RaisePropertyChanged("StateList");

            Messenger.Default.Register<DataMessage>(this,
              OnReceiveMessageAction);
            SaveChangesCommand = new RelayCommand<IClosable>
              (this.SaveChangesMethod);
            CancelCommand = new RelayCommand<IClosable>(this.CancelMethod);
```

```
47          }
48
49          public void OnReceiveMessageAction(DataMessage m)
50          {
51              if (m.CommandText == "Modify")
52              {
53                  selectedCustomer = m.CustomerMsg;
54                  id = m.CustomerMsg.CustomerID;
55                  name = m.CustomerMsg.Name;
56                  address = m.CustomerMsg.Address;
57                  city = m.CustomerMsg.City;
58                  var stateQuery = from states in MMABookEntity.BookEntity.States ↵
                        where states.StateCode == m.CustomerMsg.State select        ↵
                      states.StateName;
59                  state = stateQuery.FirstOrDefault().ToString();
60                  zipCode = m.CustomerMsg.ZipCode;
61              }
62              RaiseChanges();
63          }
64
65          public void RaiseChanges()
66          {
67              RaisePropertyChanged("ID");
68              RaisePropertyChanged("Name");
69              RaisePropertyChanged("Address");
70              RaisePropertyChanged("City");
71              RaisePropertyChanged("State");
72              RaisePropertyChanged("ZipCode");
73          }
74
75          public int? ID
76          {
77              get { return id; }
78              set
79              {
80                  id = value;
81                  RaisePropertyChanged("ID");
82              }
83          }
84
85          public string Name
86          {
87              get { return name; }
88              set
89              {
90                  name = value;
91                  RaisePropertyChanged("Name");
92              }
93          }
```

```
94
95          public string Address
96          {
97              get { return address; }
98              set
99              {
100                 address = value;
101                 RaisePropertyChanged("Address");
102             }
103         }
104
105         public string City
106         {
107             get { return city; }
108             set
109             {
110                 city = value;
111                 RaisePropertyChanged("City");
112             }
113         }
114
115         public string State
116         {
117             get { return state; }
118             set
119             {
120                 state = value;
121                 RaisePropertyChanged("State");
122             }
123         }
124
125         public string ZipCode
126         {
127             get { return zipCode; }
128             set
129             {
130                 zipCode = value;
131                 RaisePropertyChanged("ZipCode");
132             }
133         }
134
135         public ObservableCollection<string> StateList
136         {
137             get { return stateList; }
138             set
139             {
140                 stateList = value;
141                 RaisePropertyChanged("StateList");
142             }
```

```
143         }
144
145         public void SaveChangesMethod(IClosable window)
146         {
147             RaiseChanges();
148             if (
149                 Validator.IsPresent("Name", name)
150                 && Validator.IsPresent("Address", address)
151                 && Validator.IsPresent("City", city)
152                 && Validator.IsPresent("State", state)
153                 && Validator.IsPresent("Zip Code", zipCode)
154                 && Validator.IsValidZipCode(zipCode)
155                 )
156             {
157                 var stateQuery = from states in MMABookEntity.BookEntity.States
                       where states.StateName == state select states.StateCode;
158                 string stateCode = stateQuery.FirstOrDefault().ToString();
159                 selectedCustomer.CustomerID = Convert.ToInt32(id);
160                 selectedCustomer.Name = name;
161                 selectedCustomer.Address = address;
162                 selectedCustomer.City = city;
163                 selectedCustomer.State = stateCode;
164                 selectedCustomer.ZipCode = zipCode;
165                 try
166                 {
167                     MMABookEntity.BookEntity.SaveChanges();
168                     MessageBox.Show("Customer modified", "Changes Saved",
                        MessageBoxButton.OK, MessageBoxImage.Asterisk);
169                     DataMessage m = new DataMessage()
170                     {
171                         CommandText = "Modifications Complete",
172                         IDText = selectedCustomer.CustomerID,
173                         NameText = selectedCustomer.Name,
174                         AddressText = selectedCustomer.Address,
175                         CityText = selectedCustomer.City,
176                         StateText = selectedCustomer.State,
177                         ZipCodeText = selectedCustomer.ZipCode
178                     };
179                     Messenger.Default.Send(m);
180                     window.Close();
181                 }
182                 catch (DbUpdateConcurrencyException dbe)
183                 {
184                     dbe.Entries.Single().Reload();
185                     if (MMABookEntity.BookEntity.Entry(selectedCustomer).State
                        == EntityState.Detached)
186                     {
187                         MessageBox.Show("Another user has deleted that
                        customer.", "Concurrency Error", MessageBoxButton.OK,
```

```
                    MessageBoxImage.Error);
188                     DataMessage m = new DataMessage()
189                     {
190                         CommandText = "Deleted",
191                         IDText = null,
192                         NameText = null,
193                         AddressText = null,
194                         CityText = null,
195                         StateText = null,
196                         ZipCodeText = null
197                     };
198                     Messenger.Default.Send(m);
199                 }
200                 else
201                 {
202                     MessageBox.Show("Another user has updated that customer.", "Concurrency Error", MessageBoxButton.OK, MessageBoxImage.Error);
203                     DataMessage m = new DataMessage()
204                     {
205                         CommandText = "Modifications Complete",
206                         IDText = selectedCustomer.CustomerID,
207                         NameText = null,
208                         AddressText = null,
209                         CityText = null,
210                         StateText = null,
211                         ZipCodeText = null
212                     };
213                     Messenger.Default.Send(m);
214                 }
215                 window.Close();
216             }
217             catch (Exception ex)
218             {
219                 MessageBox.Show(ex.Message, ex.GetType().ToString());
220                 window.Close();
221             }
222         }
223     }
224
225     public void CancelMethod(IClosable window)
226     {
227         if(window != null)
228         {
229             window.Close();
230         }
231     }
232 }
233 }
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using System.Windows;
7  using System.Windows.Controls;
8  using System.Windows.Data;
9  using System.Windows.Documents;
10 using System.Windows.Input;
11 using System.Windows.Media;
12 using System.Windows.Media.Imaging;
13 using System.Windows.Shapes;
14 using CustomerMaintenance.ViewModel;
15
16 namespace CustomerMaintenance.Views
17 {
18     /// <summary>
19     /// Interaction logic for ModifyView.xaml
20     /// </summary>
21     public partial class ModifyView : Window, IClosable
22     {
23         public ModifyView()
24         {
25             InitializeComponent();
26         }
27     }
28 }
29
```

```
<Window x:Class="CustomerMaintenance.Views.ModifyView"
        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:CustomerMaintenance.Views"
        mc:Ignorable="d"
        x:Name="ModifyViewWindow"
        Title="ModifyView" Height="350" Width="600">
    <Grid DataContext="{Binding ModifyViewModel, Source={StaticResource
      ViewModelLocator}}">
        <Label Content="Name:" HorizontalAlignment="Left" Margin="69,49,0,0"
          VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="150,53,0,0"
          TextWrapping="Wrap" Text="{Binding Name}" VerticalAlignment="Top"
          Width="372"/>
        <Label Content="Address:" HorizontalAlignment="Left" Margin="69,108,0,0"
           VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="150,112,0,0"
          TextWrapping="Wrap" Text="{Binding Address}" VerticalAlignment="Top"
          Width="372"/>
        <Label Content="City:" HorizontalAlignment="Left" Margin="69,172,0,0"
          VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="150,176,0,0"
          TextWrapping="Wrap" Text="{Binding City}" VerticalAlignment="Top"
          Width="372"/>
        <Label Content="State:" HorizontalAlignment="Left" Margin="69,234,0,0"
          VerticalAlignment="Top"/>
        <ComboBox ItemsSource="{Binding StateList}" SelectedItem="{Binding
          State}" Text="{Binding State}" HorizontalAlignment="Left"
          Margin="129,236,0,0" VerticalAlignment="Top" Width="163"/>
        <Label Content="Zip Code:" HorizontalAlignment="Left"
          Margin="318,237,0,0" VerticalAlignment="Top"/>
        <TextBox HorizontalAlignment="Left" Height="23" Margin="384,239,0,0"
          TextWrapping="Wrap" Text="{Binding ZipCode}" VerticalAlignment="Top"
          Width="138"/>
        <Button Content="Accept" Command="{Binding SaveChangesCommand}"
          CommandParameter="{Binding ElementName=ModifyViewWindow}"
          HorizontalAlignment="Left" Margin="48,283,0,0" VerticalAlignment="Top"
           Width="75"/>
        <Button Content="Cancel" Command="{Binding CancelCommand}"
          CommandParameter="{Binding ElementName=ModifyViewWindow}"
          HorizontalAlignment="Left" Margin="466,283,0,0"
          VerticalAlignment="Top" Width="75"/>
    </Grid>
</Window>

```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6  using GalaSoft.MvvmLight;
 7  using GalaSoft.MvvmLight.Command;
 8  using System.Windows;
 9  using System.Windows.Input;
10  using CustomerMaintenance.ViewModel;
11
12  namespace CustomerMaintenance.Model
13  {
14      public class Validator
15      {
16          private static string title = "Entity Error";
17
18          public static string Title
19          {
20              get
21              {
22                  return title;
23              }
24              set
25              {
26                  title = value;
27              }
28          }
29
30          public static bool IsValidZipCode(string zip)
31          {
32              zip = zip.Replace(" ", String.Empty);
33              if (zip.Length == 5)
34              {
35                  return true;
36              }
37              else if (zip.Length == 10 && zip.ElementAt(5) == '-')
38              {
39                  return true;
40              }
41              MessageBox.Show(zip + " is not a valid zip code.", Title);
42              return false;
43          }
44
45          public static bool IsPresent(string textBox, string text)
46          {
47              if (String.IsNullOrEmpty(text))
48              {
49                  MessageBox.Show(textBox + " is required", Title);
```

```
50                  //textBox.Focus();
51                  return false;
52              }
53              return true;
54          }
55
56          public static bool IsDecimal(string textBox)
57          {
58              decimal number = 0m;
59              if (Decimal.TryParse(textBox, out number))
60              {
61                  return true;
62              }
63              else
64              {
65                  MessageBox.Show(textBox + " must be a decimal value.", Title);
66                  //textBox.Focus();
67                  return false;
68              }
69          }
70
71          public static bool IsInt32(string textBox)
72          {
73              int number = 0;
74              if (Int32.TryParse(textBox, out number))
75              {
76                  return true;
77              }
78              else
79              {
80                  MessageBox.Show(textBox + " must be an integer.", Title);
81                  //textBox.Focus();
82                  return false;
83              }
84          }
85
86          public static bool IsWithinRange(string textBox, decimal min, decimal
              max)
87          {
88              int number = textBox.Length;
89              if (number < min || number > max)
90              {
91                  MessageBox.Show(textBox + " must be between " + min + " and " +
                      max + " characters.", Title);
92                  //textBox.Focus();
93                  return false;
94              }
95              return true;
96          }
```

```
 97
 98          public static bool IsValidEmail(string textBox)
 99          {
100              if (textBox.IndexOf("@") == -1 || textBox.IndexOf(".") == -1)
101              {
102                  MessageBox.Show(textBox + " must be a valid email address",
                       Title);
103                  //textBox.Focus()
104                  return false;
105              }
106              return true;
107          }
108      }
109  }
110
```

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

using CustomerMaintenance.Model;

namespace CustomerMaintenance.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);

            SimpleIoc.Default.Register<MainViewModel>();
            SimpleIoc.Default.Register<AddViewModel>();
            SimpleIoc.Default.Register<ModifyViewModel>();
            Messenger.Default.Register<NotificationMessage>(this,
              NotifyUserMethod);
        }

        public MainViewModel MainViewModel
        {
            get
```

```
49              {
50                  return ServiceLocator.Current.GetInstance<MainViewModel>();
51              }
52          }
53
54          public AddViewModel AddViewModel
55          {
56              get
57              {
58                  return ServiceLocator.Current.GetInstance<AddViewModel>();
59              }
60          }
61
62          public ModifyViewModel ModifyViewModel
63          {
64              get
65              {
66                  return ServiceLocator.Current.GetInstance<ModifyViewModel>();
67              }
68          }
69
70          private void NotifyUserMethod(NotificationMessage message)
71          {
72              MessageBox.Show(message.Notification);
73          }
74
75          public static void Cleanup()
76          {
77              // TODO Clear the ViewModels
78          }
79      }
80  }
```