```
using GalaSoft.MvvmLight;
using GalaSoft.MvvmLight.Command;
using System.Net;
using System.Threading.Tasks;
using System.Windows.Input;
using System;
using System.Windows;
using System.Collections.ObjectModel;
using LabAssignment7A.Model;
using System.Windows.Controls;
using System.Xml.Linq;
using System.Linq;
using System.Collections.Generic;
using System.IO;
using System.Windows.Media;
using System.Windows.Media.Imaging;

namespace LabAssignment7A.ViewModel
{
    public class MainViewModel : ViewModelBase
    {
        private const string KEY = "c3eb740edc2fd819ec14eaa0b66ab00e";
        WebClient fClient = new WebClient();
        Task<string> fTask;
        public Image pBox;

        private  string title;
        private string url;
        private ObservableCollection<FlickerImage> fList=new
          ObservableCollection<FlickerImage>();
        private List<string> titleName=new List<string>();
        FlickerImage fi = new FlickerImage();
        private FlickerImage selectedItem;

        /// <summary>
        /// Constructor for MainViewModel Class
        /// </summary>
        /// <param name="image"></param>
        public MainViewModel(Image image)
        {
            pBox = image;
            SearchEngine = new RelayCommand(SeachTag);
            SelectedItemCommand = new RelayCommand(SelectedPhoto);
        }

        /// <summary>
        /// Displays the image based on the selected item
        /// </summary>
        private async void SelectedPhoto()
```

```
49          {
50              if (SelectedItem != null)
51              {
52                  var selectedURL =
53                     ((FlickerImage)SelectedItem).Url;
54                  WebClient client = new WebClient();
55                  Task<byte[]> downloadTask = client.DownloadDataTaskAsync(new
                      Uri(selectedURL));
56                  byte[] imageBytes = await downloadTask;
57                  MemoryStream stream = new MemoryStream(imageBytes);
58                  var imagesource = new BitmapImage();
59                  imagesource.BeginInit();
60                  imagesource.StreamSource = stream;
61                  imagesource.EndInit();
62                  pBox.Source = imagesource;
63              }
64          }
65
66          /// <summary>
67          /// Displays the first image on searching for the keyword in the search
                box
68          /// </summary>
69          /// <param name="selectedURL"></param>
70          private async void DisplayPicture(string selectedURL)
71          {
72              if (selectedURL != null)
73              {
74                  try
75                  {
76                      WebClient client = new WebClient();
77                      Task<byte[]> downloadTask = client.DownloadDataTaskAsync
                        (new Uri(selectedURL));
78                      byte[] imageBytes = await downloadTask;
79                      MemoryStream stream = new MemoryStream(imageBytes);
80                      var imagesource = new BitmapImage();
81                      imagesource.BeginInit();
82                      imagesource.StreamSource = stream;
83                      imagesource.EndInit();
84                      pBox.Source = imagesource;
85                  }
86                  catch(Exception ex)
87                  {
88                      MessageBox.Show(ex.Message, "No Result Found!",
                        MessageBoxButton.OK, MessageBoxImage.Error);
89                  }
90              }
91              else
92              {
93                  MessageBox.Show("No Result Found!");
```

```
94                  pBox.Source = null;
95              }
96          }
97
98          /// <summary>
99          /// Search for the keyword in the application
100         /// </summary>
101         private async void SeachTag()
102         {
103             if (fTask != null &&
104                fTask.Status != TaskStatus.RanToCompletion)
105             {
106                 var result = MessageBox.Show("Cancel the Flicker Search?", "Are
                       you sure?",
107                                  MessageBoxButton.YesNo, MessageBoxImage.Question);
108                 if (result == MessageBoxResult.No)
109                     return;
110                 else
111                 {
112                     fClient.CancelAsync();
113                 }
114             }
115             var flickrUrl = string.Format("https://api.flickr.com/services/
                rest/?method=flickr.photos.search&tags={0}&api_key={1}
                &privacy_filter=1",Title.Replace(" ",","),KEY);
116             FlickerList.Clear();
117             try
118             {
119                 fTask = fClient.DownloadStringTaskAsync(new Uri(flickrUrl));
120                 XDocument doc = XDocument.Parse(await fTask);
121                 var flickrPhotos =
122                        from photo in doc.Descendants("photo")
123                        let id = photo.Attribute("id").Value.ToString()
124                        let title = photo.Attribute("title").Value.ToString()
125                        let secret = photo.Attribute("secret").Value.ToString()
126                        let server = photo.Attribute("server").Value.ToString()
127                        let farm = photo.Attribute("farm").Value.ToString()
128                        select new FlickerImage
129                        {
130                            Title = title,
131                            Url = string.Format("https://farm
                       {0}.staticflickr.com/{1}/{2}_{3}.jpg", farm, server, id,
                       secret)
132                        };
133                 FlickerList.Clear();
134
135                 if (flickrPhotos.Any())
136                 {
137                     foreach (var list in flickrPhotos)
```

```
138                     {
139                         FlickerList.Add(new FlickerImage() { Title =
                        list.Title, Url = list.Url });
140                     }
141
142                 }
143                 else
144                 {
145                     fi.Title = "No Match Found!";
146                     FlickerList.Add(fi);
147                 }
148             }
149             catch(WebException)
150             {
151                 if (fTask.Status == TaskStatus.Faulted)
152                     MessageBox.Show("Unable to get results from Flickr",
153                        "Flickr Error", MessageBoxButton.OK,
154                        MessageBoxImage.Error);
155                 FlickerList.Clear();
156                 fi.Title = "Error Occured";
157                 FlickerList.Add(fi);
158             }
159
160             var selectedURL =
161                   ((FlickerImage)FlickerList[0]).Url;
162             DisplayPicture(selectedURL);
163
164         }
165
166         /// <summary>
167         /// Getters and Setters
168         /// </summary>
169         public ICommand SearchEngine { get; private set; }
170         public ICommand SelectedItemCommand { get; private set; }
171         public string Title
172         {
173             get
174             {
175                 return title;
176             }
177             set
178             {
179                 title = value;
180                 RaisePropertyChanged("Title");
181             }
182         }
183         public string Url
184         {
185             get
```

```
186             {
187                 return url;
188             }
189             set
190             {
191                 url = value;
192                 RaisePropertyChanged("Url");
193             }
194         }
195 
196         public List<string> TitleName
197         {
198             get
199             {
200                 return titleName;
201             }
202             set
203             {
204                 titleName = value;
205                 RaisePropertyChanged("TitleName");
206             }
207         }
208         public ObservableCollection<FlickerImage> FlickerList
209         {
210             get
211             {
212                 return fList;
213             }
214             set
215             {
216                 fList = value;
217                 RaisePropertyChanged("FlickerList");
218             }
219         }
220 
221         public FlickerImage SelectedItem
222         {
223             get
224             {
225                 return selectedItem;
226             }
227             set
228             {
229                 selectedItem = value;
230                 RaisePropertyChanged("SelectedItem");
231             }
232         }
233 
234         public string Display
```

```
235         {
236             get
237             {
238                 return Title;
239             }
240         }
241
242     }
243 }
```