Source code

1. AlbumWindow.xaml.cs:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.IO;
using System.Linq;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Forms;
using System.Windows.Input;
using System.Windows.Media.Imaging;
using System.Windows.Threading;

namespace AlbumViewer
{
    public partial class AlbumWindow : Window
    {
        static private List<BitmapImage> images;
        private FolderBrowserDialog directory;

        public static List<BitmapImage> Images
        {
            get
            {
                return images;
            }

            set
            {
                images = value;
            }
        }


        private PhotoViewerWindow photoViewer;
        private SaveAlbumAsWindow saveAlbum;

        public AlbumWindow()
        {
            InitializeComponent();
            photoViewer = new PhotoViewerWindow();
            saveAlbum = new SaveAlbumAsWindow();
            Images = new List<BitmapImage>();
            add.IsEnabled = false;
            loading.Visibility = Visibility.Hidden;
            //bar.Visibility = Visibility.Hidden;
        }

        private void ListBox_MouseDoubleClick(object sender, MouseButtonEventArgs e)
        {
            if (!Images.Any())
                return;
            BitmapImage selectedImage = (BitmapImage)listBox.SelectedItem;
            if (photoViewer == null || photoViewer.IsClosed)
                photoViewer = new PhotoViewerWindow(selectedImage);
            else
                photoViewer.image.Source = selectedImage;
            photoViewer.Show();
        }

        BackgroundWorker worker = new BackgroundWorker();
        private async void open_Click(object sender, RoutedEventArgs e)
        {
            //worker.DoWork += DoIndependentWork;
            //worker.RunWorkerCompleted += OpenAlbum;
            //worker.RunWorkerAsync();
```

```
            //DoIndependentWork();
            await OpenAlbum();
            //DoIndependentWork();

            //bar.Visibility = Visibility.Hidden;
            //loading.Visibility = Visibility.Hidden;
        }

        async Task OpenAlbum()
        {

            Dispatcher.Invoke((Action)( () =>
            {
                if (Images.Any())
                    Images.Clear();

                directory = new FolderBrowserDialog();
                directory.Description = "Select an Album";
                directory.ShowDialog();


                if (!string.IsNullOrEmpty(directory.SelectedPath))
                {
                    GetImages("*.jpg");
                    GetImages("*.png");

                    //DataContext = null;
                    //DataContext = Images;
                    //await Task.Run(() =>
                    //{

                        string albumName =
directory.SelectedPath.Substring(directory.SelectedPath.LastIndexOf('\\') + 1,
directory.SelectedPath.Length - directory.SelectedPath.LastIndexOf('\\') - 1);
                        Title = "";
                        Title = "Album Viewer - " + albumName;
                        add.IsEnabled = true;
                        //loading.Visibility = Visibility.Hidden;
                        listBox.ItemsSource = null;
                        listBox.ItemsSource = Images;

                    //});
                }
            }));
            //return Images;



        }

        private void DoIndependentWork()
        {

            Dispatcher.Invoke((Action)(() =>
            {

                //bar.Visibility = Visibility.Visible;
                loading.Visibility = Visibility.Visible;
            }));

        }

        private void GetImages(string extension)
        {
            DirectoryInfo directoryInfo = new DirectoryInfo(directory.SelectedPath);
            foreach (FileInfo fileName in directoryInfo.GetFiles(extension))
            {
                Images.Add(new BitmapImage(new Uri(fileName.FullName)));
            }
        }
```

```
        Microsoft.Win32.OpenFileDialog openFileDialog = new Microsoft.Win32.OpenFileDialog();
        private void add_Click(object sender, RoutedEventArgs e)
        {
            openFileDialog.Multiselect = true;
            openFileDialog.Title = "Select images";
            openFileDialog.Filter = "All Image Files|*.jpg;*.jpeg;*;.jpe;*.jfif;*.png|" +
              "JPEG (*.jpg;*.jpeg;*.jpe;*.jfif;)|*.jpg;*.jpeg;.jpe;*.jfif|" +
              "PNG (*.png)|*.png";

            if (openFileDialog.ShowDialog() == true)
            {
                foreach (String fileName in openFileDialog.FileNames)
                {
                    Images.Add(new BitmapImage(new Uri(fileName)));
                }
            }
            listBox.ItemsSource = null;
            listBox.ItemsSource = Images;
        }

        private void new_Click(object sender, RoutedEventArgs e)
        {
            MessageBoxResult result = System.Windows.MessageBox.Show("Do you want to make new
album?", "New Album", MessageBoxButton.YesNo, MessageBoxImage.Question);
            if (result == MessageBoxResult.No)
                return;

            if (Images.Any())
            {
                Images.Clear();
                listBox.ItemsSource = null;
            }
            Title = "Album Viewer";
            add.IsEnabled = true;
        }

        private void saveAs_Click(object sender, RoutedEventArgs e)
        {
            saveAlbum = new SaveAlbumAsWindow();
            saveAlbum.ShowDialog();

            if (Images.Any())
                add.IsEnabled = true;
        }

        private void listBox_KeyDown(object sender, System.Windows.Input.KeyEventArgs e)
        {
            if (e.Key == Key.Delete)
            {
                MessageBoxResult result = System.Windows.MessageBox.Show("Do you want to delete
selected image?", "Delete selected image", MessageBoxButton.YesNo, MessageBoxImage.Question);
                if (result == MessageBoxResult.Yes)
                    Images.Remove((BitmapImage)listBox.SelectedItem);
            }
            listBox.ItemsSource = null;
            listBox.ItemsSource = Images;
        }

        private void exit_Click(object sender, RoutedEventArgs e)
        {
            System.Windows.Application.Current.Shutdown();
        }

        protected override void OnClosed(EventArgs e)
        {
            System.Windows.Application.Current.Shutdown();
        }
    }
}
```

2. PhotoViewerWindow.xaml.cs:

```
using Microsoft.Win32;
using System;
using System.Runtime.InteropServices;
using System.Windows;
using System.Windows.Media.Imaging;

namespace AlbumViewer
{
    public partial class PhotoViewerWindow : Window
    {
        private OpenFileDialog openFileDialog;

        private int rotation;
        public int Rotation
        {
            get
            {
                return rotation;
            }

            set
            {
                rotation = value;
            }
        }

        public PhotoViewerWindow()
        {
            InitializeComponent();
            openFileDialog = new OpenFileDialog();
        }

        public PhotoViewerWindow(BitmapImage image)
        {
            InitializeComponent();
            this.image.Source = image;
        }

        private void btnRight_Click(object sender, RoutedEventArgs e)
        {
            rotateTransform.Angle = (Rotation += 90);
        }

        private void btnNext_Click(object sender, RoutedEventArgs e)
        {
            if (scaleTransform.ScaleX == -1)
                scaleTransform.ScaleX = 1;

            rotateTransform.Angle = (Rotation = 0);
            BitmapImage tempImage = (BitmapImage)image.Source;
            int index = AlbumWindow.Images.IndexOf(tempImage);

            if (index != -1)
            {
                if (index != AlbumWindow.Images.Count - 1)
                    image.Source = AlbumWindow.Images[index + 1];
                else
                    image.Source = AlbumWindow.Images[0];
            }
        }

        private void btnPrevious_Click(object sender, RoutedEventArgs e)
        {
            if (scaleTransform.ScaleX == -1)
                scaleTransform.ScaleX = 1;

            rotateTransform.Angle = (Rotation = 0);
```

```
            BitmapImage tempImage = (BitmapImage)image.Source;
            int index = AlbumWindow.Images.IndexOf(tempImage);

            if (index != -1)
            {
                if (index != 0)
                    image.Source = AlbumWindow.Images[index - 1];
                else
                    image.Source = AlbumWindow.Images[AlbumWindow.Images.Count - 1];
            }
        }

        private void btnLeft_Click(object sender, RoutedEventArgs e)
        {
            rotateTransform.Angle = (Rotation -= 90);
        }

        public bool IsClosed { get; private set; }
        protected override void OnClosed(EventArgs e)
        {
            base.OnClosed(e);
            IsClosed = true;
        }

        private void btnFlip_Click(object sender, RoutedEventArgs e)
        {
            if (scaleTransform.ScaleX == 1)
                scaleTransform.ScaleX = -1;
            else
                scaleTransform.ScaleX = 1;
        }

        //Set as desktop background
        [DllImport("user32.dll", CharSet = CharSet.Auto)]
        private static extern Int32 SystemParametersInfo(UInt32 action, UInt32 uParam, String
vParam, UInt32 winIni);
        private static readonly UInt32 SPI_SETDESKWALLPAPER = 0x14;
        private static readonly UInt32 SPIF_UPDATEINIFILE = 0x01;
        private static readonly UInt32 SPIF_SENDWININICHANGE = 0x02;
        private void btnBackground_Click(object sender, RoutedEventArgs e)
        {
            BitmapImage background = (BitmapImage)image.Source;
            SystemParametersInfo(SPI_SETDESKWALLPAPER, 0,
Uri.UnescapeDataString(background.UriSource.AbsolutePath), SPIF_UPDATEINIFILE |
SPIF_SENDWININICHANGE);
        }
    }
}
```

3. SaveAlbumAsWindow.xaml.cs:

```
using System;
using System.IO;
using System.Windows;
using System.Windows.Forms;
using System.Windows.Media;

namespace AlbumViewer
{
    public partial class SaveAlbumAsWindow : Window
    {
        public SaveAlbumAsWindow()
        {
            InitializeComponent();
            canvas.Visibility = Visibility.Collapsed;
        }

        private void btnCancel_Click(object sender, RoutedEventArgs e)
```

```
        {
            Close();
        }

        public bool IsClosed { get; private set; }
        protected override void OnClosed(EventArgs e)
        {
            base.OnClosed(e);
            IsClosed = true;
        }

        private FolderBrowserDialog directory;
        private void btnBrowse_Click(object sender, RoutedEventArgs e)
        {

            directory = new FolderBrowserDialog();
            directory.ShowDialog();
            if (!string.IsNullOrEmpty(directory.SelectedPath))
            {
                DirectoryInfo directoryInfo = new DirectoryInfo(directory.SelectedPath);
                txtLocation.Text = directory.SelectedPath;
            }
        }

        private void btnOK_Click(object sender, RoutedEventArgs e)
        {
            if (string.IsNullOrEmpty(txtLocation.Text))
            {
                WrongLocation();
                textBlock.Text = "Please enter a location. For example: C:\\Users\\pc\\Documents";
                return;
            }

            txtName.BorderBrush = new SolidColorBrush(Colors.Gray);
            txtLocation.BorderBrush = new SolidColorBrush(Colors.Gray);
            string folderPath = txtLocation.Text;
            folderPath = folderPath.Replace("\\", "/");

            if (txtName.Text.IndexOfAny(new char[] { ':', '/', '\\', '*', '?', '"', '<', '>', '|' })
>= 0)
            {
                WrongName();
                textBlock.Text = "A name can't contain any of the following characters: / \\ :
* ? \" < > |.";
                return;
            }

            else if (Directory.Exists(folderPath + "/" + txtName.Text))
            {
                WrongName();
                textBlock.Text = "The folder " + txtName.Text + " already exists!\nPlease try
another name.";
                return;
            }

            else if (string.IsNullOrEmpty(txtName.Text))
            {
                WrongName();
                textBlock.Text = "Please enter a name.";
                return;
            }

            else if (!Directory.Exists(txtLocation.Text))
            {
                WrongLocation();
                textBlock.Text = "Could not find the location: " + txtLocation.Text + ".";
                return;
            }

            else if (!Directory.Exists(folderPath + "/" + txtName.Text))
```

```
            {
                Directory.CreateDirectory(System.IO.Path.Combine(folderPath, txtName.Text));
                txtName.BorderBrush = new SolidColorBrush(Colors.Green);
                canvas.Visibility = Visibility.Collapsed;
                Close();
            }

            foreach (var file in AlbumWindow.Images)
            {
                string temp = file.UriSource.AbsolutePath;
                int index = temp.LastIndexOf('/') + 1;
                string fileName = file.UriSource.AbsolutePath.Substring(index, temp.Length - index);
                string fullPath = file.UriSource.AbsolutePath;
                fullPath = Uri.UnescapeDataString(fullPath);
                fullPath = @System.IO.Path.GetFullPath(fullPath);
                //System.IO.File.Copy(file.UriSource.AbsolutePath.Replace("/", "\\"),
@System.IO.Path.Combine(directory.SelectedPath, txtName.Text + "\\" + fileName), true);
                System.IO.File.Copy(fullPath,
System.IO.Path.GetFullPath(@System.IO.Path.Combine(directory.SelectedPath, txtName.Text + "\\" +
fileName)), true);
            }
        }

        private void WrongName( )
        {
            txtName.BorderBrush = new SolidColorBrush(Colors.Red);
            canvas.Visibility = Visibility.Visible;
        }

        private void WrongLocation()
        {
            txtLocation.BorderBrush = new SolidColorBrush(Colors.Red);
            canvas.Visibility = Visibility.Visible;
        }
    }
}
```