# FaceAppWithGPT2

## FaceAppWithGPT2

### Dependencies

- DlibDotNet v19.21.0.20220724
- Emgu.CV v4.9.0.5494
- Xabe.FFmpeg v5.2.6

### /FaceAppWithGPT2/Program.cs

```
namespace FaceAppWithGPT2
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

## ImageProcessingLibrary

### Dependencies

- DlibDotNet v19.21.0.20220724
- Emgu.CV v4.9.0.5494
- Xabe.FFmpeg v5.2.6

### /ImageProcessingLibrary/Helpers/DirectoryHelper.cs

```
using System;
using System.Collections.Generic;
using System.IO;

namespace ImageProcessingLibrary.Helpers
{
    internal static class DirectoryHelper
    {
        /// <summary>
        /// Validates if the given directory path exists. If it doesn't exist, throws a Dire
        /// </summary>
        /// <param name="directoryPath">The path of the directory to validate.</param>
        public static void ValidateDirectory(string directoryPath)
        {
            if (string.IsNullOrWhiteSpace(directoryPath))
```

```
            {
                throw new ArgumentException("Directory path cannot be null or empty.", name
            }

            if (!Directory.Exists(directoryPath))
            {
                throw new DirectoryNotFoundException($"The directory '{directoryPath}' does
            }
        }

        /// <summary>
        /// Gets all image files (JPG, PNG) from the specified directory.
        /// </summary>
        /// <param name="directoryPath">The path of the directory to search for image files
        /// <returns>A list of file paths for the images found in the directory.</returns>
        public static List<string> GetImageFiles(string directoryPath)
        {
            ValidateDirectory(directoryPath);

            // Define allowed image extensions
            string[] allowedExtensions = { ".jpg", ".jpeg", ".png" };

            // Get all files with allowed extensions
            var imageFiles = new List<string>();
            foreach (var file in Directory.GetFiles(directoryPath))
            {
                if (Array.Exists(allowedExtensions, ext => ext.Equals(Path.GetExtension(file
                {
                    imageFiles.Add(file);
                }
            }

            return imageFiles;
        }
    }
}
```

### /ImageProcessingLibrary/Interfaces/IImageResizer.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ImageProcessingLibrary.Interfaces
```

```
{
    internal interface IImageResizer
    {
        void ResizeImage(string inputPath, string outputPath, int width, int height);
    }
}
```

### /ImageProcessingLibrary/PictureSizeAdaptation/ImageResizer.cs

```
using ImageProcessingLibrary.Interfaces;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ImageProcessingLibrary.PictureSizeAdaptation
{
    internal class ImageResizer : IImageResizer
    {
    }
}
```

### /ImageProcessingLibrary/Class1.cs

```
namespace ImageProcessingLibrary
{
    public class Class1
    {

    }
}
```

## FaceMorphingLibrary

### Dependencies

- DlibDotNet v19.21.0.20220724
- Emgu.CV v4.9.0.5494
- Xabe.FFmpeg v5.2.6

### /FaceMorphingLibrary/Class1.cs

```
namespace FaceMorphingLibrary
{
    public class Class1
    {
```

```
    }
}
```

## VideoGenerationLibrary

### Dependencies

- DlibDotNet v19.21.0.20220724
- Emgu.CV v4.9.0.5494
- Xabe.FFmpeg v5.2.6

### /VideoGenerationLibrary/Class1.cs

```
namespace VideoGenerationLibrary
{
    public class Class1
    {

    }
}
```

## Sonstige Dateien

### Dependencies

- No dependencies found