Dip Project:

Group Members

1. Name: Saad Dastgir
2. Name: Muhammad Umair Khan
3. Name: Ahsan Abbasi
4. Name: Hassan Baig

Image Editor Documentation

Overview

The **Image Editor** is a web-based application that allows users to load an image and apply various image filters, such as Grayscale, Black & White, Negative, Threshold, Brightness adjustment, Contrast adjustment, Opacity adjustment, and a newly added Blur filter. The user interface is designed to be intuitive and user-friendly.

User Interface

Canvas

The canvas is the central area where the loaded image is displayed and where filters are applied. The canvas is responsive and adjusts its size based on the dimensions of the loaded image.

File Loader

Located above the canvas, the File Loader section allows users to select an image file from their device using the provided file input. The selected image is then loaded into the canvas for editing.

Filter Buttons

The Filter Buttons are located on the left side of the canvas. These buttons trigger specific image filters when clicked. The available filters include:

- Grayscale
- Black & White
- Negative
- Threshold
- Blur (newly added)

Control Sliders

Located on the right side of the canvas, these sliders allow users to adjust image properties in real-time:

- Brightness: Adjust the overall brightness of the image.
- Contrast: Control the difference between light and dark areas.

- Opacity: Set the transparency of the image.

Blur Filter

The Blur filter is applied by clicking the "Blur" button. It smoothens the image by averaging the color values of neighboring pixels.

How to Use

Loading an Image

1. Click on the "Choose File" button in the File Loader section.
2. Select an image file from your device.
3. The selected image will be loaded into the canvas automatically.

Applying Filters

1. Choose one or more filters from the Filter Buttons section (Grayscale, Black & White, Negative, Threshold, Blur).
2. Adjust the control sliders (Brightness, Contrast, Opacity) to achieve the desired visual effect.
3. The image on the canvas will update in real-time based on the applied filters and adjustments.

Adjusting Controls

1. Use the Brightness slider to control the overall brightness of the image.
2. Use the Contrast slider to adjust the contrast between light and dark areas.
3. Use the Opacity slider to set the transparency of the image.

Some Code Snippets:

```
window.onload = function () {
  var canvas = document.getElementById("imageCanvas");
  var ctx = canvas.getContext("2d");
  var imageLoader = document.getElementById("imageLoader");
  var originalImage = null;

  imageLoader.addEventListener("change", handleImage, false);
```

```
function handleImage(e) {
  var reader = new FileReader();
  reader.onload = function (event) {
    var img = new Image();
    img.onload = function () {
      canvas.width = img.width;
      canvas.height = img.height;
      ctx.drawImage(img, 0, 0);
      originalImage = img;
      applyFilters(); // Apply filters on image load
    };
    img.src = event.target.result;
  };
  reader.readAsDataURL(e.target.files[0]);
}

var brightnessControl = document.getElementById("brightness");
var contrastControl = document.getElementById("contrast");
var opacityControl = document.getElementById("opacity");
var negativeButton = document.getElementById("negativeFilter");
var thresholdButton = document.getElementById("thresholdFilter");
var grayscaleButton = document.getElementById("grayscaleFilter");
var blackWhiteButton = document.getElementById("blackWhiteFilter");
var blurButton = document.getElementById("blurFilter");

brightnessControl.addEventListener("input", applyFilters);
contrastControl.addEventListener("input", applyFilters);
opacityControl.addEventListener("input", applyFilters);
```

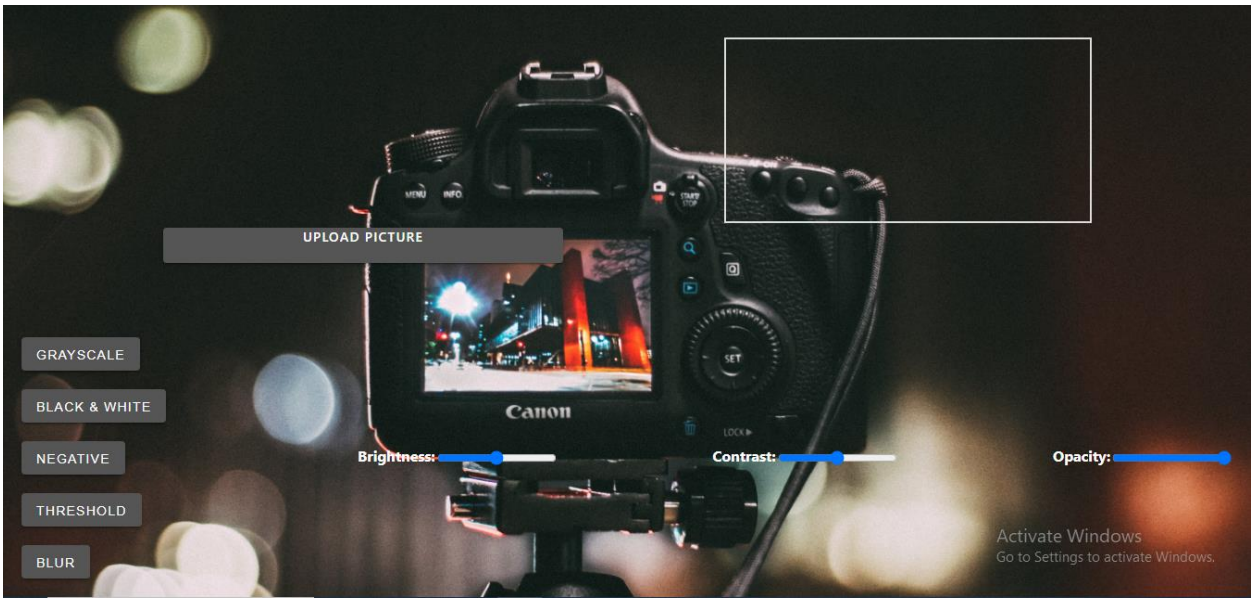
```
    thresholdButton.addEventListener("click", applyThreshold);
    negativeButton.addEventListener("click", applyNegative);
    grayscaleButton.addEventListener("click", applyGrayscale);
    blackWhiteButton.addEventListener("click", applyBlackAndWhite);
    blurButton.addEventListener("click", applyBlur);

    function applyFilters() {
      if (originalImage) {
        var brightnessValue = brightnessControl.value;
        var contrastValue = contrastControl.value;
        var opacityValue = opacityControl.value;

        ctx.clearRect(0, 0, canvas.width, canvas.height);

        ctx.globalAlpha = opacityValue / 100;

        ctx.filter = `brightness(${100 + parseInt(brightnessValue)}%) contrast(${100 +
parseInt(contrastValue)}%)`;
        ctx.drawImage(originalImage, 0, 0);
      }
    }

    function applyThreshold() {
      // Implement threshold filter if needed
    }

    function applyNegative() {
      // Implement negative filter if needed
    }
```

```
function applyGrayscale() {
  if (originalImage) {
    ctx.drawImage(originalImage, 0, 0);
    var imageData = ctx.getImageData(0, 0, canvas.width, canvas.height);
    var data = imageData.data;

    for (var i = 0; i < data.length; i += 4) {
      var avg = (data[i] + data[i + 1] + data[i + 2]) / 3;
      data[i] = avg;
      data[i + 1] = avg;
      data[i + 2] = avg;
    }

    ctx.putImageData(imageData, 0, 0);
  }
}

function applyBlackAndWhite() {
  if (originalImage) {
    ctx.drawImage(originalImage, 0, 0);
    var imageData = ctx.getImageData(0, 0, canvas.width, canvas.height);
    var data = imageData.data;

    for (var i = 0; i < data.length; i += 4) {
      var avg = (data[i] + data[i + 1] + data[i + 2]) / 3;
      var color = avg > 127 ? 255 : 0;
      data[i] = color;
      data[i + 1] = color;
```

```
      data[i + 2] = color;
    }

    ctx.putImageData(imageData, 0, 0);
  }
}

function applyBlur() {
  if (originalImage) {
    ctx.drawImage(originalImage, 0, 0);
    var imageData = ctx.getImageData(0, 0, canvas.width, canvas.height);
    var data = imageData.data;

    for (var i = 0; i < data.length; i += 4) {
      var avg = (
        data[i] +
        data[i + 1] +
        data[i + 2] +
        data[i + 4] +
        data[i + 5] +
        data[i + 6] +
        data[i + 8] +
        data[i + 9] +
        data[i + 10]
      ) / 9;

      data[i] = avg;
      data[i + 1] = avg;
      data[i + 2] = avg;
```

```
        }

        ctx.putImageData(imageData, 0, 0);
      }
    }
  };
```

User Interface



Hosted Link:

https://diplabproject.netlify.app/