Image Processing Application - USEME.md

This file provides instructions on how to operate the Image Processing Application. You can utilize this application through Command Line Interface (CLI) and Graphical based User Interface (GUI).

This application allows users to process images by running commands directly from a command prompt or through a script file. It supports a wide range of operations, including color manipulation, image transformations, and enhancement effects. Users can also preview split views of specific manipulations, where only part of the image is modified.

The application offers a graphical user interface (GUI) that complements its command-line capabilities, providing users with an intuitive and user-friendly way to interact with the system. Through the GUI, users can effortlessly load images, save them in various formats, and perform all the operations available in the command-line interface, such as applying filters or modifying image properties.

Moreover, the GUI introduces an exclusive feature: Image Downscaling, allowing users to resize images to smaller dimensions while maintaining their aspect ratio. This operation is not available through the command-line interface, making the GUI a more versatile and powerful tool for image processing tasks.

This application features a command-based interface for executing various image processing operations. Below is an overview of the available commands, their functionality, examples of how to use them, and any applicable conditions or dependencies.

Table of Contents

Command line Based Interaction

All the sections below provide detailed information on how to use the program in command-line mode.

Command Syntax and Structure

Each command has a specific format and requires certain arguments. Commands are summarized below.

Note : Users must strictly adhere to the exact format for all commands described below. Any deviation or formatting error may lead to execution failures. Additionally, altering the order of arguments will result in unexpected outcomes.

Note : An image must be loaded before applying any operations, and the save command should be used to save any processed image to your system.

Here's an enhanced version of the **Supported Commands** section with more detailed descriptions and additional instructions.

Supported Commands

Each command must adhere strictly to the specified syntax. If any command is incorrect or the image is not in the specified file path, the application will throw an exception. Additionally, for any operation to work on an image, it must already be loaded into the application's memory with the specified name.

1. Loading and Saving

- **Load**: Loads an image from the provided absolute file path and assigns a custom name for in-app reference.

  This image name is crucial for subsequent operations, as it allows the application to identify which image to manipulate.

  It is mandatory to load an image first in order to use any following commands.

  Ensure the image exists in the specified path; otherwise, an exception will be thrown.

  The application only supports images in **PNG, JPG, JPEG, and PPM** formats. If an image with a different extension is provided, an error will be thrown.

```
load absolute/path/to/image.jpg image-name
```

Example:

```
load /images/sample.jpg sample
```

- **Save**: Saves the processed image to a specified path with the provided name. Make sure the image you want to save has been processed and exists in the application's memory.

  Ensure the filepath is valid; otherwise, an exception will be thrown.

  The application only supports images in **PNG, JPG, JPEG, and PPM** formats. If an image with a different extension is provided, an error will be thrown.

  ```
  save absolute/path/to/output.png image-name
  ```

  Example:

  ```
  save /results/sample-output.png sample
  ```

2. ***Color and GreyScale Components***

These commands isolate individual color channels (Red, Green, Blue) or specific grayscale components (Intensity, Value, Luma) of an image.

- **Red/Green/Blue Component**: Extracts a specific color component (red, green, or blue) from the image and saves the result under a new name. The output image will contain only the selected color channel information and in the other channels the value of the selected channels will be set.
  So for red-component of the image values will be (R,R,R) ; for the green-component image value will be (G,G,G) and for blue-component image value will be (B,B,B).

  It is mandatory that given input image name should be present in the application's memory.

  When used with the `split` argument, the application will apply the operation only to the specified percentage of the image, starting from the left side.

  Value of split percentage must be within **0 and 100** and (decimal values are allowed).

  This operation also supports `partial image manipulation`, which requires a mask image to be loaded into the system. Initially, the operation is applied to the entire image. Then, the mask image guides the final manipulation: regions of the mask containing black pixels indicate where the resulting values from the operation should be applied in the final image.

For the operation to be carried out successfully, the mask image must be loaded into the system, and its dimensions must match those of the image on which the operation is to be performed.

Application will store output image into its memory. User can use `save` command to get the output image.

```
red-component image-name dest-image-name
green-component image-name dest-image-name
blue-component image-name dest-image-name
```

Example:

```
red-component sample sample-red
green-component sample sample-green
blue-component sample sample-blue
```

The commands with the split operations are.

```
red-component image-name dest-image-name split percentage
green-component image-name dest-image-name split percentage
blue-component image-name dest-image-name split percentage
```

Example:

```
red-component sample sample-red split 58.9
green-component sample sample-green split 34
blue-component sample sample-blue split 89
```

The command with partial image manipulation are.

```
red-component image-name mask-image-name dest-image-name
green-component image-name mask-image-name dest-image-name
blue-component image-name mask-image-name dest-image-name
```

Example:

```
red-component sample sample-red-mask sample-red
green-component sample sample-green-mask sample-green
blue-component sample sample-blue-mask sample-blue
```

- **Intensity/Value/Luma Component**: Creates a grayscale image based on different brightness calculations:

  Intensity averages the RGB values.

  Value uses the highest RGB value.

  Luma uses a weighted sum of the RBG values.

It is mandatory that given input image name should be present in the application's memory.

When used with the `split` argument, the application will apply the operation only to the specified percentage of the image, starting from the left side.

Value of split percentage must be within **0 and 100** and (decimal values are allowed).

This operation also supports `partial image manipulation`, which requires a mask image to be loaded into the system. Initially, the operation is applied to the entire image. Then, the mask image guides the final manipulation: regions of the mask containing black pixels indicate where the resulting values from the operation should be applied in the final image.

For the operation to be carried out successfully, the mask image must be loaded into the system, and its dimensions must match those of the image on which the operation is to be performed.

Application will store output image into its memory. User can use `save` command to get the output image.

```
value-component image-name dest-image-name
intensity-component image-name dest-image-name
luma-component image-name dest-image-name
```

Example:

```
value-component sample sample-red
intensity-component sample sample-intensity
luma-component sample sample-luma
```

The commands with the split operations are.

```
value-component image-name dest-image-name split percentage
intensity-component image-name dest-image-name split percentage
luma-component image-name dest-image-name split percentage
```

Example:

```
value-component sample sample-value split 58.9
intensity-component sample sample-intensity split 34
luma-component sample sample-luma split 89
```

The commands with the partial image manipulation.

```
value-component image-name masked-image-name dest-image-name
intensity-component image-name masked-image-name dest-image-name
luma-component image-name masked-image-name dest-image-name
```

Example:

```
    value-component sample sample-value-masked sample-value
    intensity-component sample sample-intensity-masked sample-intensity-
masked
    luma-component sample sample-luma-masked sample-luma
```

3. Transformations

- **Horizontal Flip**: Flips the image horizontally, creating a mirror image along the vertical axis.

  It is mandatory that given input image name should be present in the application's memory.

  Application will store output image into its memory. User can use `save` command to get the output image.

  ```
  horizontal-flip image-name dest-image-name
  ```

  Example:

  ```
  horizontal-flip sample sample-flipped
  ```

- **Vertical Flip**: Flips the image vertically, creating a mirror image along the horizontal axis.

  It is mandatory that given input image name should be present in the application's memory.

  Application will store output image into its memory. User can use `save` command to get the output image.

- `vertical-flip image-name dest-image-name`

  Example:

  ```
  vertical-flip sample sample-flipped
  ```

4. Brightness Adjustment

- **Brighten** or **Darken**: Adjusts the brightness of an image by a specified increment (positive for brightening, negative for darkening).

  It is mandatory that given input image name should be present in the application's memory.

  The brightness value affects all color channels equally.

  The increment value the user can enter should be an integer.

  User can enter negative increment value to darken the image.

  Application will store output image into its memory. User can use `save` command to get the output image.

```
brighten increment image-name dest-image-name
```

Example for Brighten command :

```
brighten 20 sample sample-brightened
```

Example for Darken command :

```
brighten -20 sample sample-brightened
```

5. RGB Split and Combine

- **RGB Split**: Separates the image into three separate images for each color channel (red, green, blue).

  It is mandatory that given input image name should be present in the application's memory.

  Application will store output images into its memory. User can use save command to get the output images.

  ```
  rgb-split image-name dest-image-name-red dest-image-name-green dest-image-name-blue
  ```

  Example: `plaintext   rgb-split sample sample-red sample-green sample-blue`

- **RGB Combine**: Combines three images, each containing one of the RGB color channels, into a single image.

  All the input images must have **the same dimensions**.

  Order of the input images **must not change**.

  And to generate the correct image the ordering of the images to be combined must be maintained, that is first the red image, then green and at last the blue image must be provided.

  ```
  rgb-combine image-name red-image green-image blue-image
  ```

  Example:

  ```
    rgb-combine sample-new sample-red sample-green sample-blue
  ```

6. Blur and Sharpen

- **Blur**: Applies a blurring effect to soften the image, reducing details and smoothing transitions.

  It is mandatory that given input image name should be present in the application's memory.**

When used with the `split` argument, the application will apply the operation only to the specified percentage of the image, starting from the left side.

Value of split percentage must be within **0 and 100** and (decimal values are allowed).

This operation also supports `partial image manipulation`, which requires a mask image to be loaded into the system. Initially, the operation is applied to the entire image. Then, the mask image guides the final manipulation: regions of the mask containing black pixels indicate where the resulting values from the operation should be applied in the final image.

For the operation to be carried out successfully, the mask image must be loaded into the system, and its dimensions must match those of the image on which the operation is to be performed.

Application will store output image into its memory. User can use `save` command to get the output image.

```
blur image-name dest-image-name
```

Example:

```
  blur sample sample-blur
```

The blur command with split parameter is given as.

```
  blur image-name dest-image-name split percentage
```

Example: `plaintext   blur sample sample-blur split 45.7`

The blur command with partial image manipulation. `plaintext   blur image-name masked-image-name dest-image-name` **Example**:

```
    blur sample sample-masked sample-blur
```

- **Sharpen**: Enhances the edges in an image, making details crisper.

  It is mandatory that given input image name should be present in the application's memory.

  When used with the `split` argument, the application will apply the operation only to the specified percentage of the image, starting from the left side.

  Value of split percentage must be within **0 and 100** and (decimal values are allowed).

  This operation also supports `partial image manipulation`, which requires a mask image to be loaded into the system. Initially, the operation is applied to the entire image. Then, the mask image guides the final manipulation: regions of the mask containing black pixels indicate where the resulting values from the operation should be applied in the final image.

For the operation to be carried out successfully, the mask image must be loaded into the system, and its dimensions must match those of the image on which the operation is to be performed.

Application will store output image into its memory. User can use `save` command to get the output image.

```
sharpen image-name dest-image-name
```

Example:

```
  sharpen sample sample-sharpen
```

The sharpen command with split parameter is given as.

```
sharpen image-name dest-image-name split percentage
```

Example: `plaintext   sharpen sample sample-sharpen split 45.7`

The sharpen command with partial image manipulation. `plaintext   sharpen image-name masked-image-name dest-image-name` **Example**:

```
  sharpen sample sample-masked sample-sharpen
```

7. Sepia Tone

- **Sepia**: Applies a sepia filter to give the image a warm, aged appearance.

  It is mandatory that given input image name should be present in the application's memory.

  When used with the `split` argument, the application will apply the operation only to the specified percentage of the image, starting from the left side.

  Value of split percentage must be within **0 and 100** and (decimal values are allowed).

  This operation also supports `partial image manipulation`, which requires a mask image to be loaded into the system. Initially, the operation is applied to the entire image. Then, the mask image guides the final manipulation: regions of the mask containing black pixels indicate where the resulting values from the operation should be applied in the final image.

  For the operation to be carried out successfully, the mask image must be loaded into the system, and its dimensions must match those of the image on which the operation is to be performed.

  Application will store output image into its memory. User can use `save` command to get the output image.

```
sepia image-name dest-image-name
```

Example:

```
sepia sample sample-sharpen
```

The Sepia command with split parameter is given as.

```
sepia image-name dest-image-name split percentage
```

Example:

```
sepia sample sample-sharpen split 56.7
```

The sepia command with partial image manipulation.

```
sepia image-name masked-image-name dest-image-name
```

Example:

```
sepia sample sample-masked sample-sepia
```

8. Compression

- **Compress**: Reduces the file size of an image by a specified percentage, maintaining resolution but potentially affecting quality. The percentage parameter should lie between 0 and 100. It can be a decimal value as well.

  Value of percentage must be within **0 and 100** and (decimal values are allowed).

  Application will store output image into its memory. User can use save command to get the output image.

```
compress percentage image-name dest-image-name
```

Example:

```
compress 56.8 sample sample-sharpen
```

9. Histogram

- **Histogram**: Generates a histogram of the image's color distribution.

  It is mandatory that given input image name should be present in the application's memory.

  Application will store output image (histogram) into its memory. User can use save command to get the output image.

  ```
  histogram image-name dest-image-name
  ```

  Example:

  ```
  histogram sample sample-hist
  ```

10. Color Correct:

- **Color Correct**: *Automatically adjusts colors in the image to improve balance and tone.

  It is mandatory that given input image name should be present in the application's memory.

  When used with the `split` argument, the application will apply the operation only to the specified percentage of the image, starting from the left side.

  Value of split percentage must be within **0 and 100** and (decimal values are allowed).

  Application will store output image into its memory. User can use `save` command to get the output image.

  ```
  color-correct image-name dest-image-name
  ```

  Example:

```
color-correct sample sample-color-correct
```

The color-correct command with split parameter is given as.

```
color-correct image-name dest-image-name split percentage
```

Example:

```
color-correct sample sample-color-correct split 57.9
```

11. Levels Adjustment

- **Levels Adjust**: Adjusts black, mid, and white points to improve contrast and brightness. The black mid and white values provided should lie in 0-255 range. Also, the black, mid and white values should be in ascending order.

  It is mandatory that given input image name should be present in the application's memory.

  Values of `b`, `m` and `w` should be **an integer** and between **0 and 255**.

  Order of `b`, `m` and `w` **must not change**.

  Values of `b`, `m` and `w` should be in ascending order (b < m < w).

  When used with the `split` argument, the application will apply the operation only to the specified percentage of the image, starting from the left side.

  Value of split percentage must be within **0 and 100** and (decimal values are allowed).

Application will store output image into its memory. User can use `save` command to get the output image.

```
levels-adjust b m w image-name dest-image-name
```

Example: `plaintext    levels-adjust 20 100 245 sample sample-level` The levels-adjust command with split parameter is given as.

```
levels-adjust b m w image-name dest-image-name split percentage
```

Example: `plaintext    levels-adjust 20 100 245 sample sample-level split 57.9`

12. Quit Application

- **Quit**: Exits the application. When the application is closed, it will lose its memory, resulting in all loaded and processed images being discarded.

  ```
  quit
  ```

13. Run Script File

- **Run Script**: Executes a list of commands from a script file. Each command in the script must follow the specified syntax.

  Script file must be **a text file**.

  Path of the script file must be valid.

  If an invalid command is encountered in the script file, an error message for that specific command will be displayed, and the script will proceed to execute the remaining commands.

  Application will store output images into its memory. User can use `save` command to get the output images.

  ```
  run absolute/path/to/script.txt
  ```

  Example: `plaintext    run res/script.txt`

Split Preview Mode

Certain commands support **Split Preview** mode, where only part of the image is modified while the rest remains unchanged. To use split preview, append `split p` to the command. The 'p' parameter provided by the user should lie in 0-100 range, it can be a decimal value. Supported commands for split preview include:

- `blur`, `sharpen`, `sepia`, `red-component`, `green-component`, `blue-component`, `value-component`, `luma-component`, `intensity-component`, `color-correct`, `levels-adjust`.

Example: `plaintext     blur sample sample-blur split p`

Note: Commands in Split Preview mode should strictly follow the syntax above to avoid errors.

Partial Image Manipulation

Certain commands support **Partial Image Manipulation operation** mode, where only part of the image in where the mask image has black pixel is modified while the rest remains unchanged. To support this operation command has a mask image name in added to the command. Supported commands for partial image manipulation include:

- `blur`, `sharpen`, `sepia`, `red-component`, `green-component`, `blue-component`, `value-component`, `luma-component`, `intensity-component`.

Example:

```
blur sample sample-mask-image sample-blur
```

Note: Commands in image manipulation mode should strictly follow the syntax above to avoid errors.

Example Workflow

1. **Load** an image:

   ```
   load /images/photo.png photo
   ```

2. **Apply operations**:

   ```
   sepia photo photo-sepia
   ```

3. **Save** the modified image:

   ```
   save /results/photo-sepia.png photo-sepia
   ```

4. **Quit** the application:

   ```
   quit
   ```

Here’s a refined and more detailed version of the steps for your **USEME.md** file:

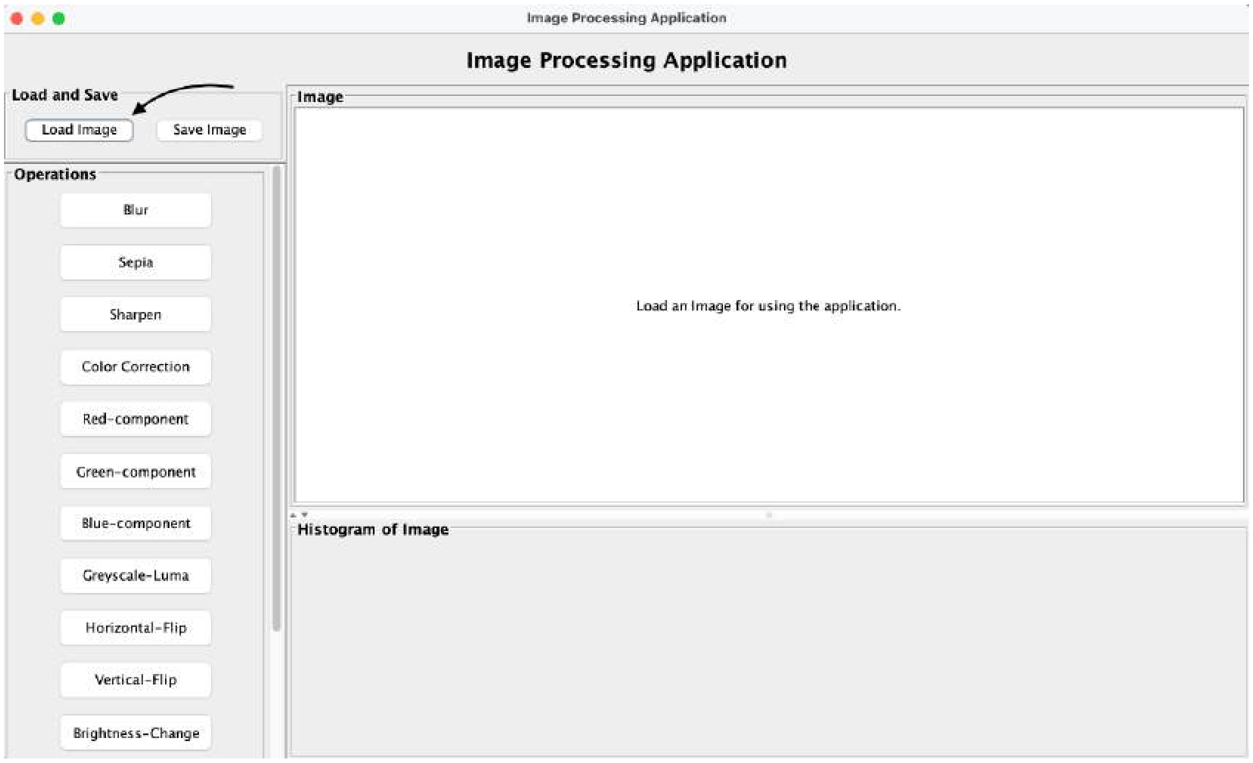
Graphical User Interface (GUI) Based Interaction

This section provides a comprehensive guide on how to use the application’s GUI to load images, perform various operations, and save the processed results. Follow the step-by-step instructions for a seamless experience.

Loading an Image

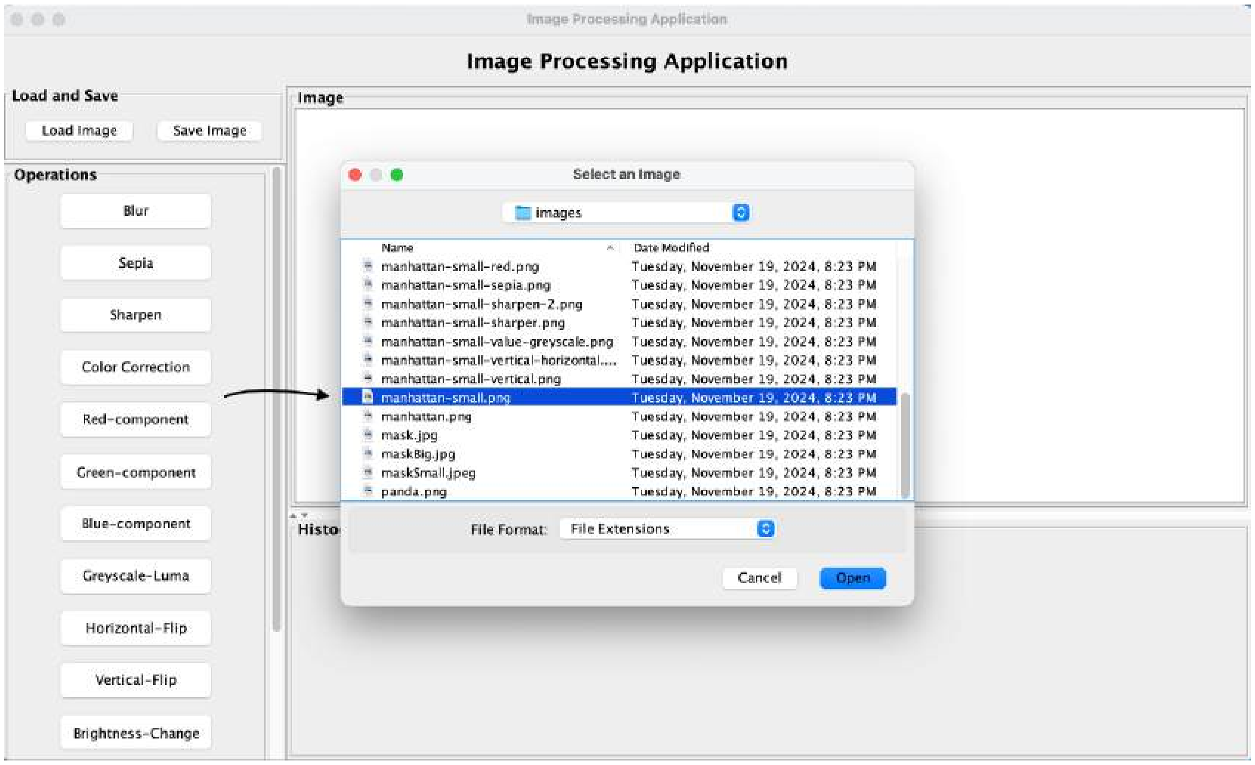
1. **Initiating the Load Process**

To begin, click on the **“Load”** button located at the top-left corner of the interface.



2. **Selecting an Image**

A file selection dialog will appear, allowing you to navigate through directories and select an image file.



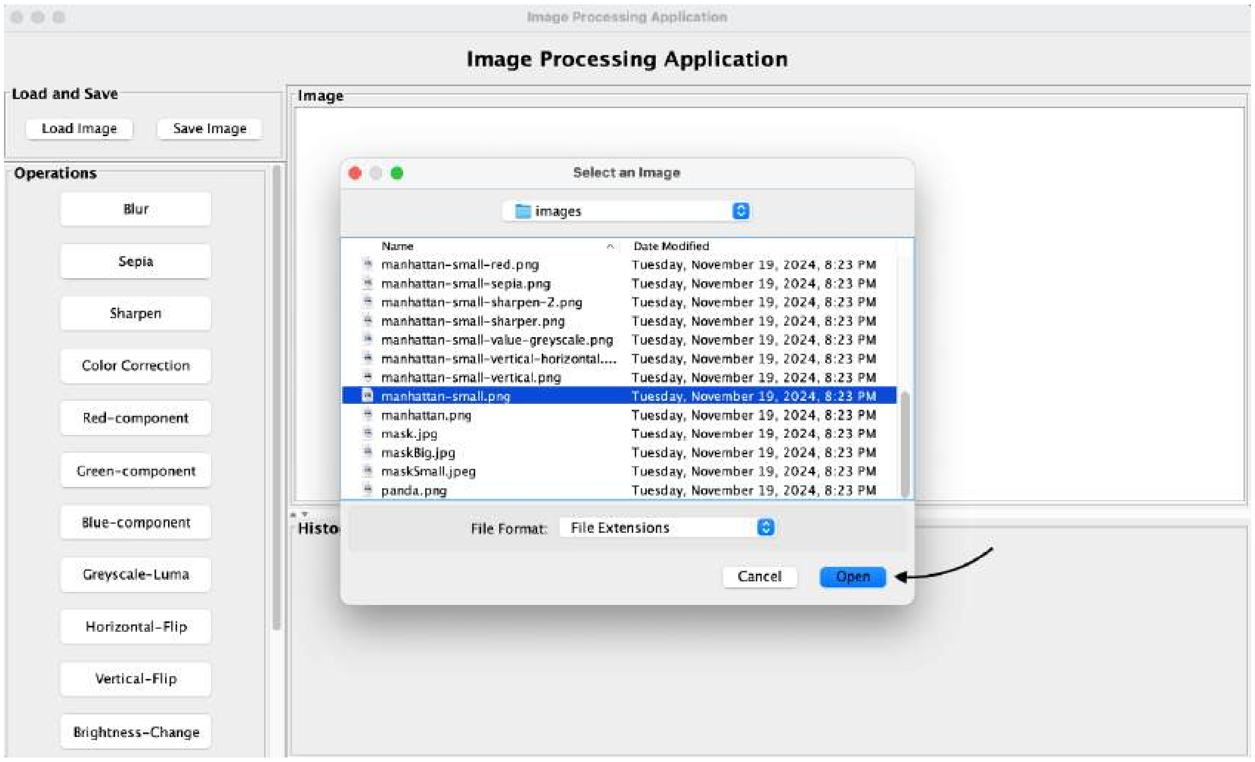
images

Name	Date Modified
manhattan-small-red.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-sepia.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-sharpen-2.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-sharper.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-value-greyscale.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-vertical-horizontal....	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-vertical.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small.png	Tuesday, November 19, 2024, 8:23 PM
manhattan.png	Tuesday, November 19, 2024, 8:23 PM
mask.jpg	Tuesday, November 19, 2024, 8:23 PM
maskBig.jpg	Tuesday, November 19, 2024, 8:23 PM
maskSmall.jpeg	Tuesday, November 19, 2024, 8:23 PM
panda.png	Tuesday, November 19, 2024, 8:23 PM

File Format: File Extensions

3. **Opening the Image**

Once you've chosen the desired image, click the **"Open"** button. This action will load the image into the application.



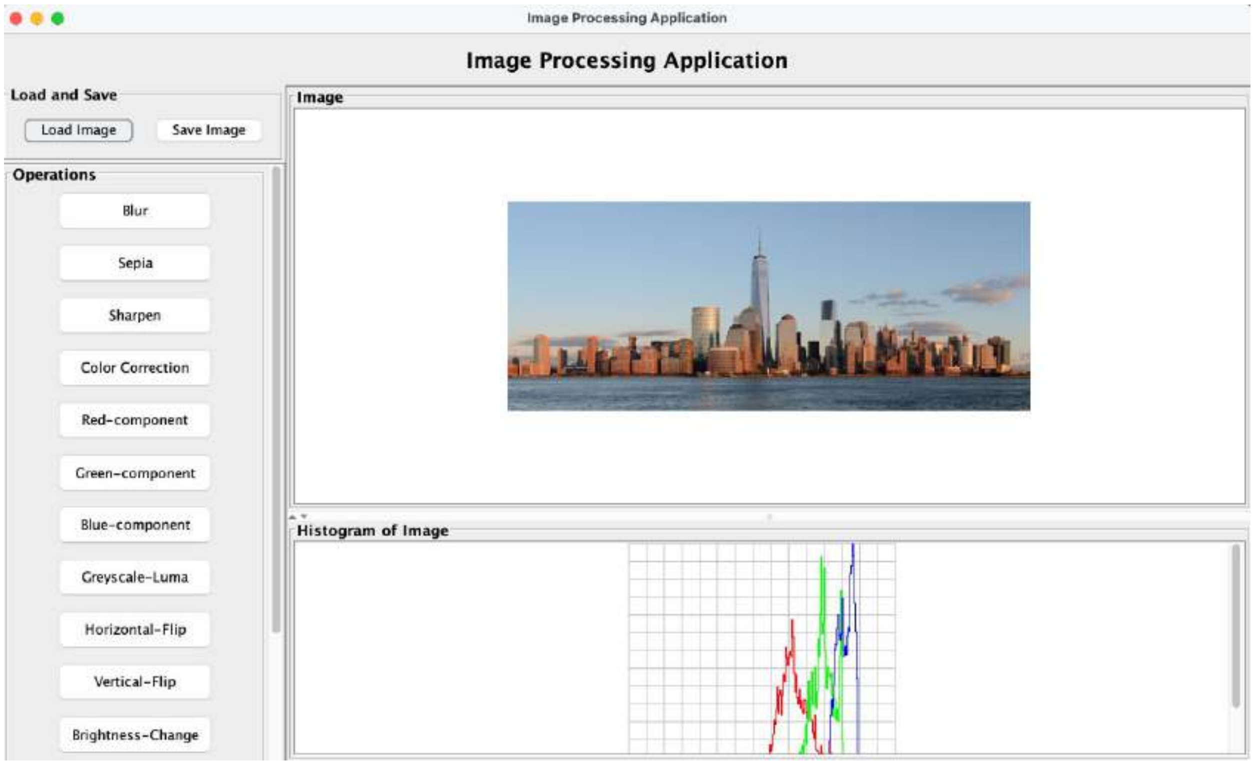
images

Name	Date Modified
manhattan-small-red.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-sepia.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-sharpen-2.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-sharper.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-value-greyscale.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-vertical-horizontal....	Tuesday, November 19, 2024, 8:23 PM
manhattan-small-vertical.png	Tuesday, November 19, 2024, 8:23 PM
manhattan-small.png	Tuesday, November 19, 2024, 8:23 PM
manhattan.png	Tuesday, November 19, 2024, 8:23 PM
mask.jpg	Tuesday, November 19, 2024, 8:23 PM
maskBig.jpg	Tuesday, November 19, 2024, 8:23 PM
maskSmall.jpeg	Tuesday, November 19, 2024, 8:23 PM
panda.png	Tuesday, November 19, 2024, 8:23 PM

File Format: File Extensions

4. **Viewing the Loaded Image**

The selected image will now appear in the main frame of the application. A histogram graph corresponding to the image is also displayed below it, providing a visual representation of its pixel intensity distribution.



5. **Cancel** (Optional)

You can directly click **"Cancel"** after selecting the operation or can click the cross mark on the top right of the panel to close the operation.

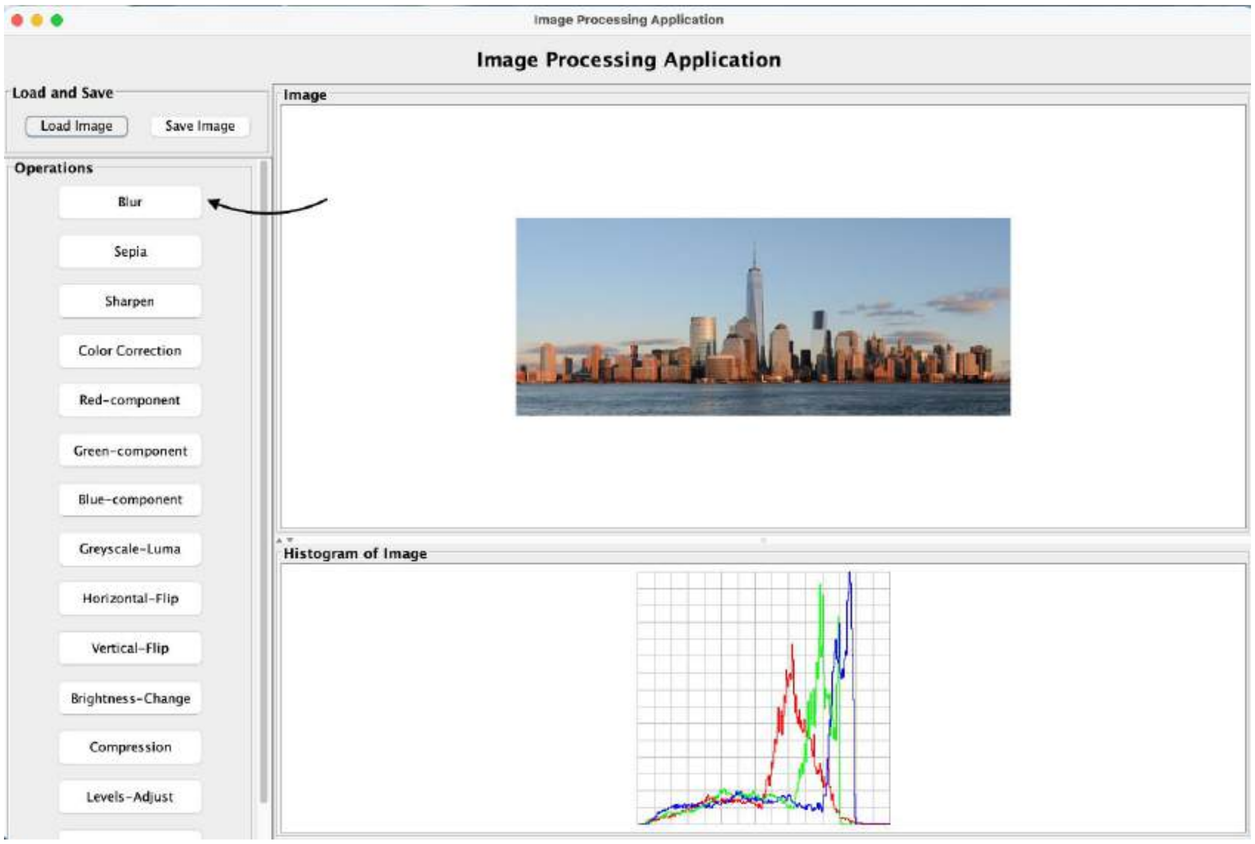
Split Supported Operations

Common Steps for Operations

The following instructions apply to the following operations: **Blur**, **Sharpen**, **Sepia**, **Red Component**, **Green Component**, **Blue Component**, **Luma component (Greyscale)**, and **Color-Correct.**

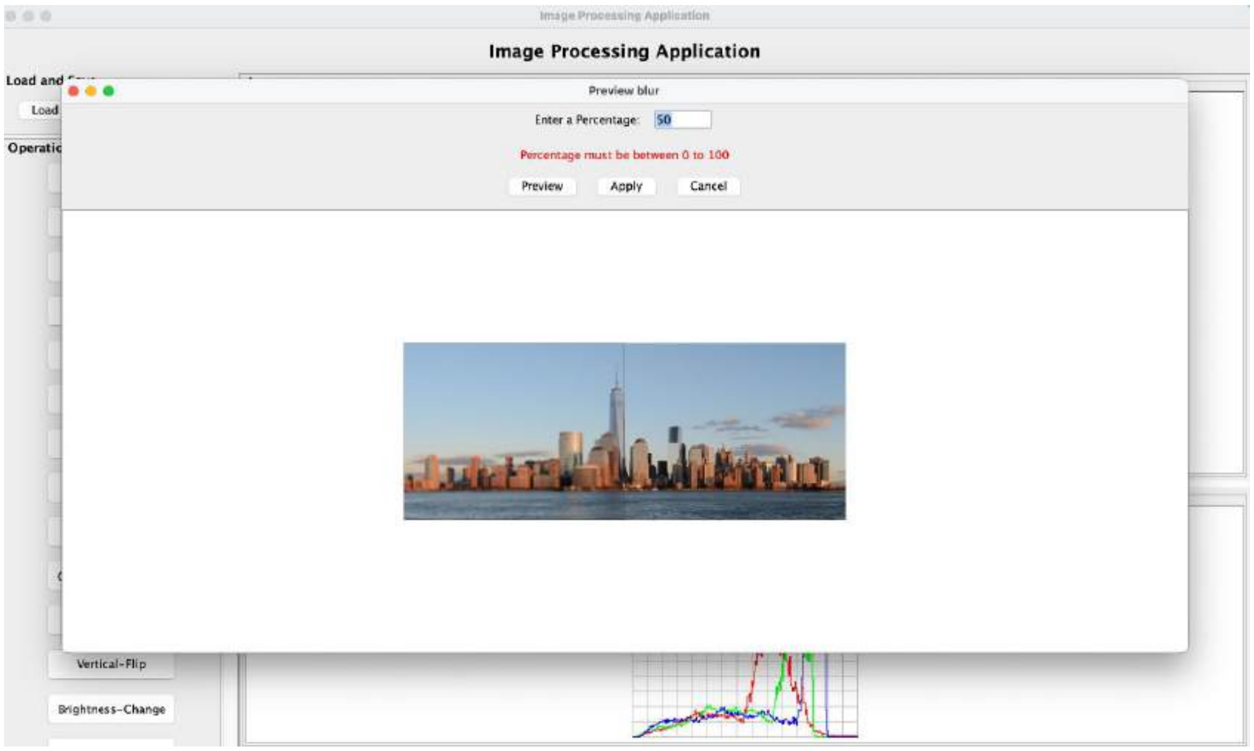
1. **Selecting an Operation**

Click on the button corresponding to the desired operation, such as **"Blur"**.



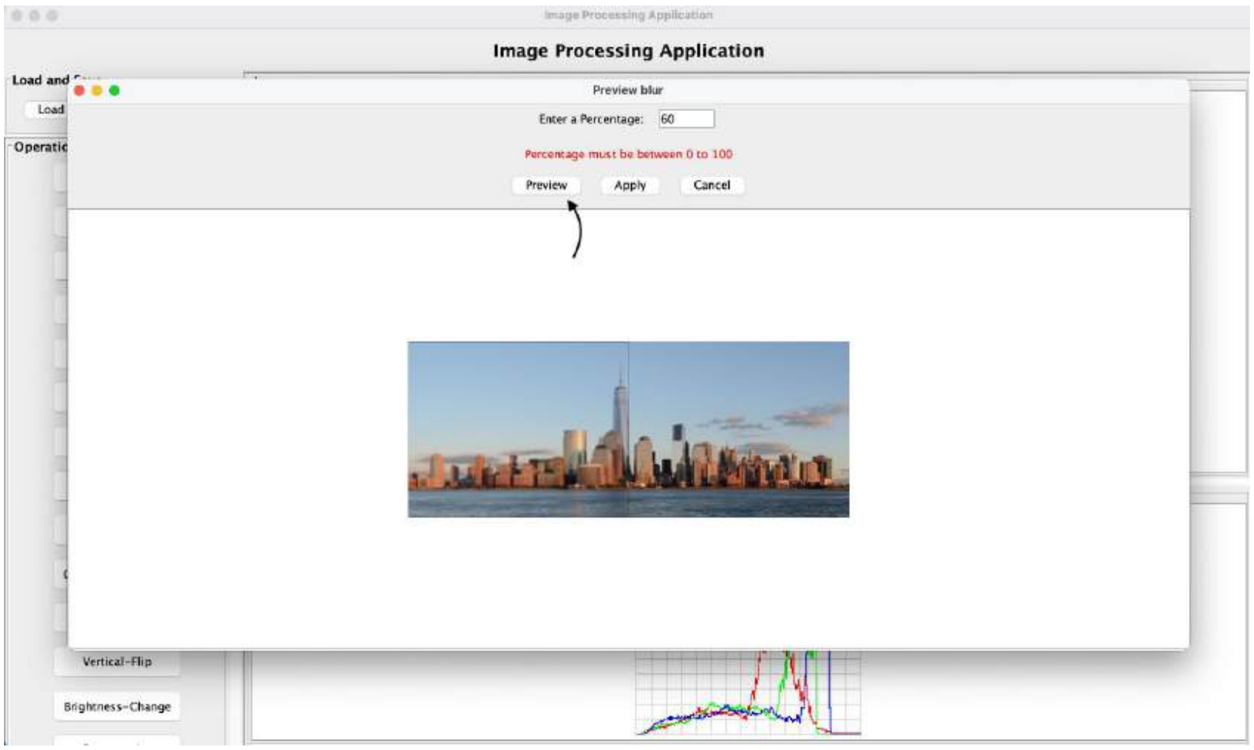
2. **Previewing the Operation**

A **"Preview Panel"** will appear, allowing you to preview the effect of the operation in split mode. By default, the split percentage is set to 50%. You can adjust this value to customize how much of the image is affected in the preview.



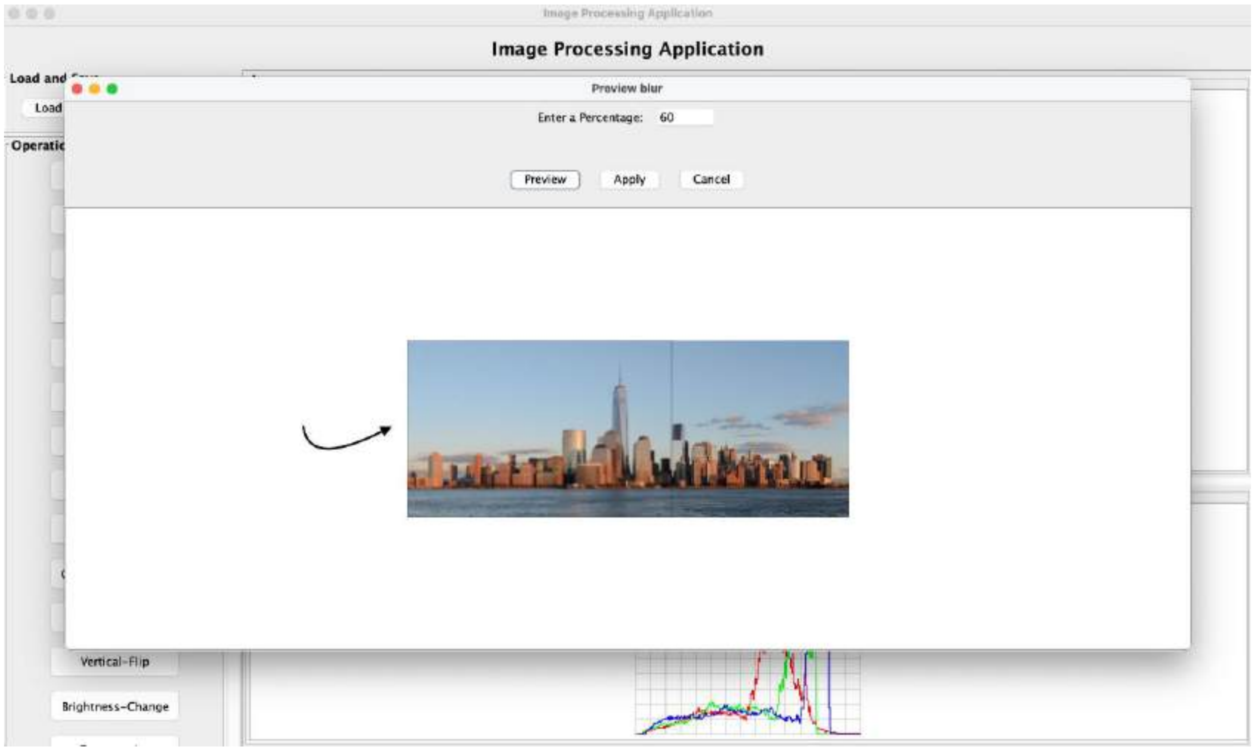
3. **Adjusting the Split Percentage**

Modify the split percentage to your preference and click **"Preview"** again to update the preview.



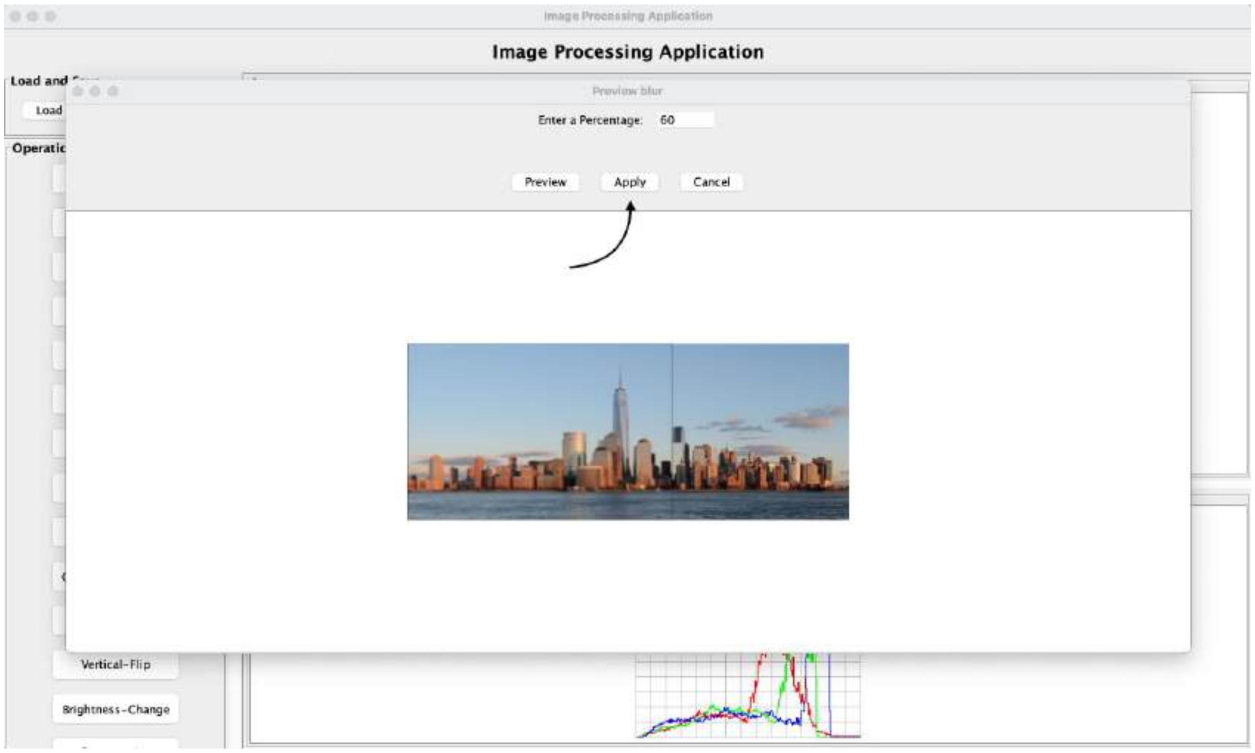
4. **Preview Output**

The updated preview will display the operation applied to the specified portion of the image. For example, if the split percentage is 60%, 60% of the image will show the operation applied, while the remaining 40% remains unchanged.

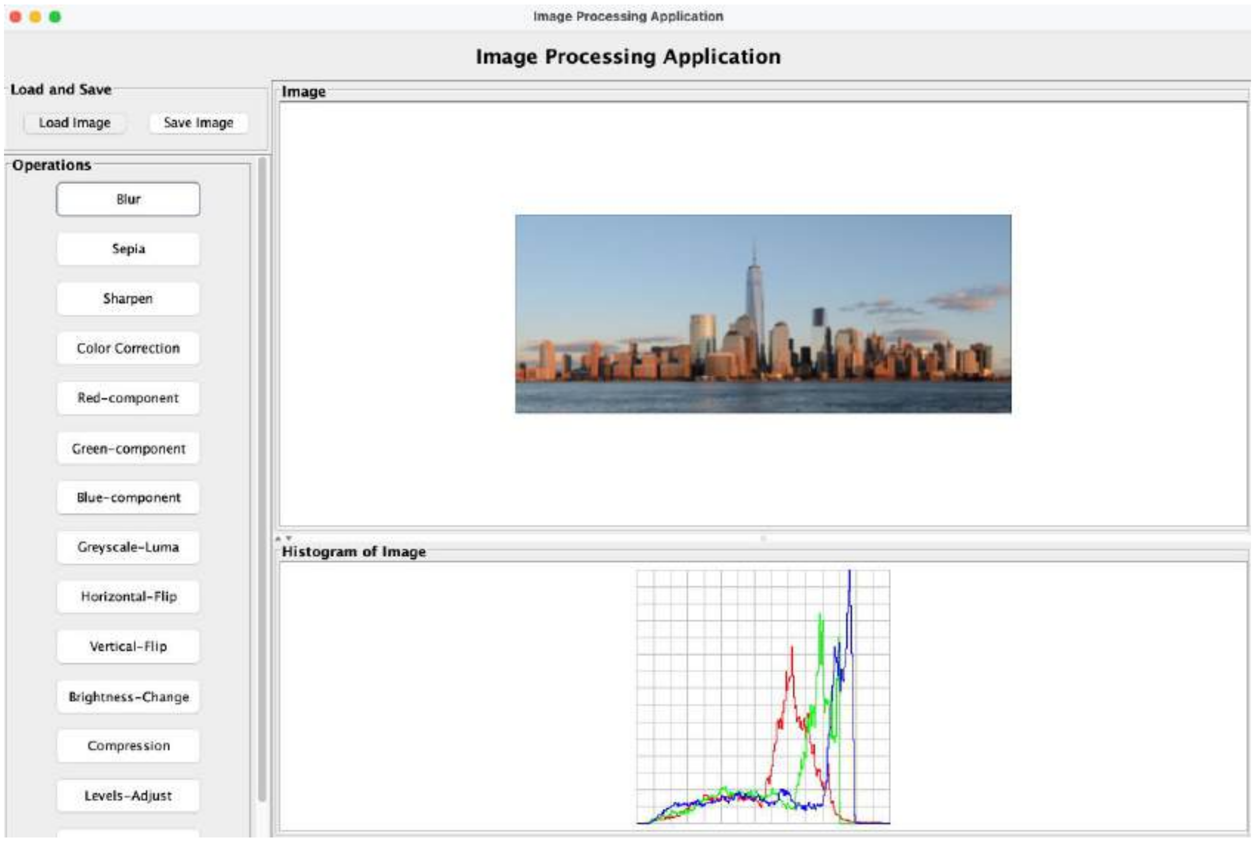


5. **Applying the Operation**

To finalize the operation, click **"Apply".** The application will process the image and display a confirmation popup stating **"Filter applied successfully."**



The operation is now permanently applied to the main image.



6. **Skipping the Preview** (Optional)

You can directly click **"Apply"** after selecting the operation without previewing it if you are confident about the effect.

7. **Cancel the Preview** (Optional)

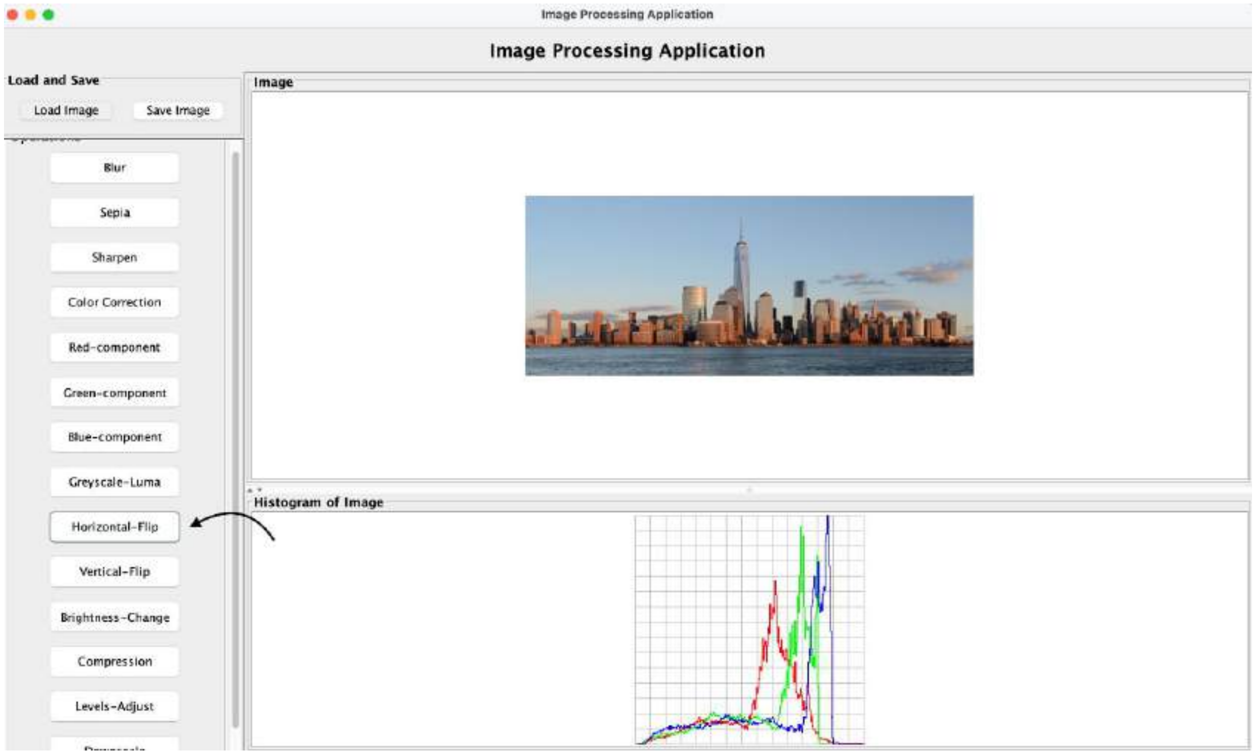
You can directly click **"Cancel"** after selecting the operation without previewing or can click the cross mark on the top right of the panel to close the preview operation.

Flip Operations

Horizontal Flip and **Vertical Flip** are direct transformations with no preview feature.

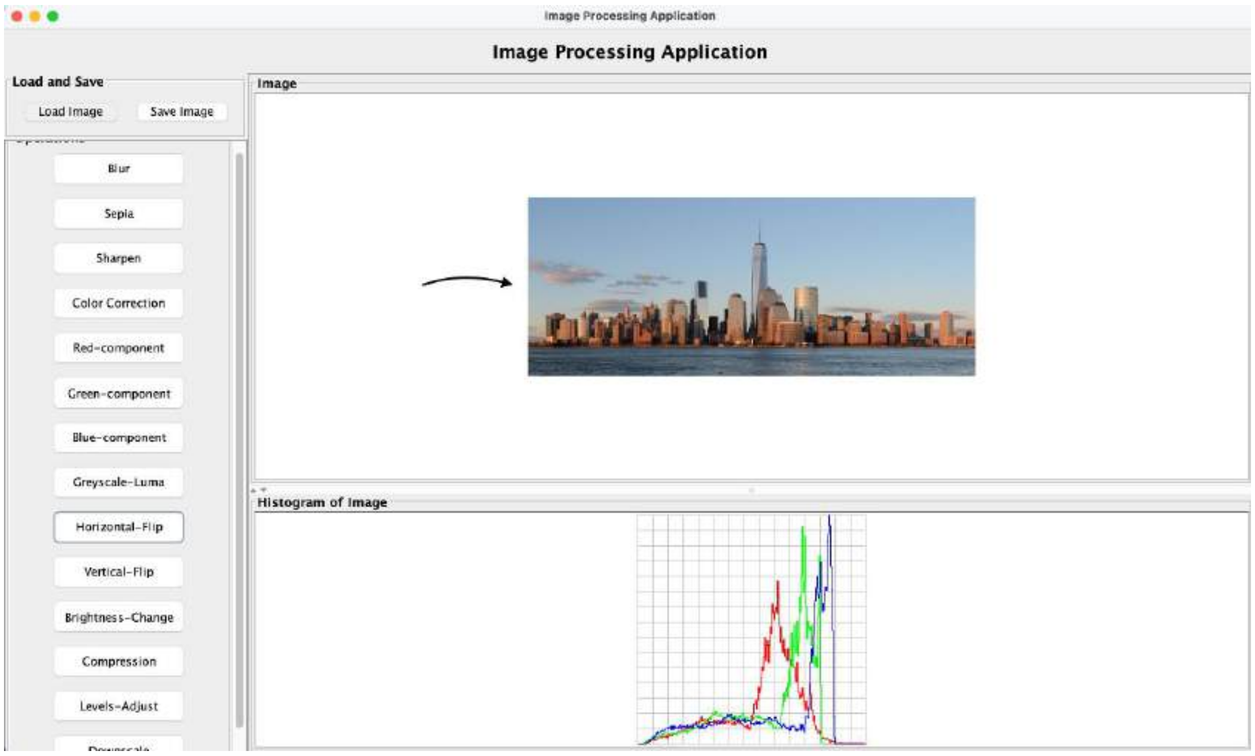
1. **Selecting Flip Options**

Click the **"Horizontal-Flip"** or **"Vertical-Flip"** button to flip the image accordingly.



2. **Viewing the Result**

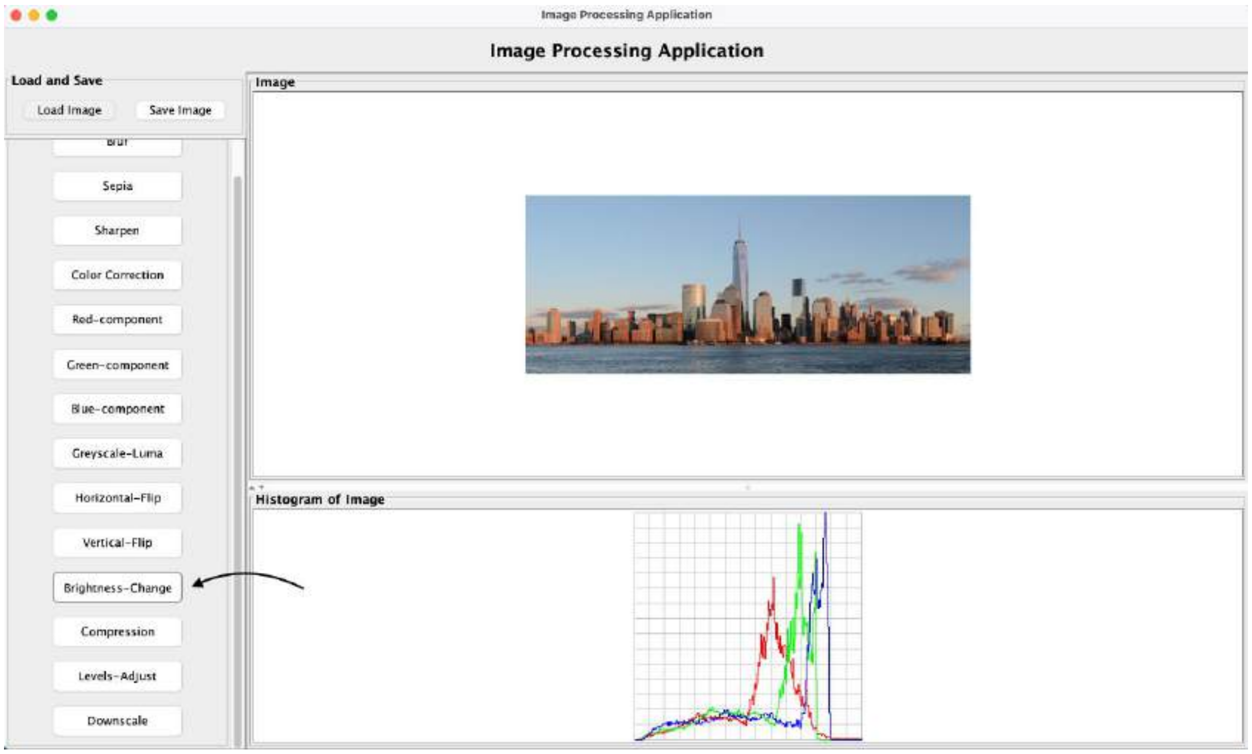
The application immediately updates the image with the chosen flip transformation.



Brightening and Darkening

1. **Adjusting Brightness**

Click on the **“Brightness-Change”** button to open the adjustment panel.



2. **Setting Intensity**
   Enter the desired intensity value:
   - A positive value brightens the image.
   - A negative value darkens the image.

Click **“Preview”** to see the changes.

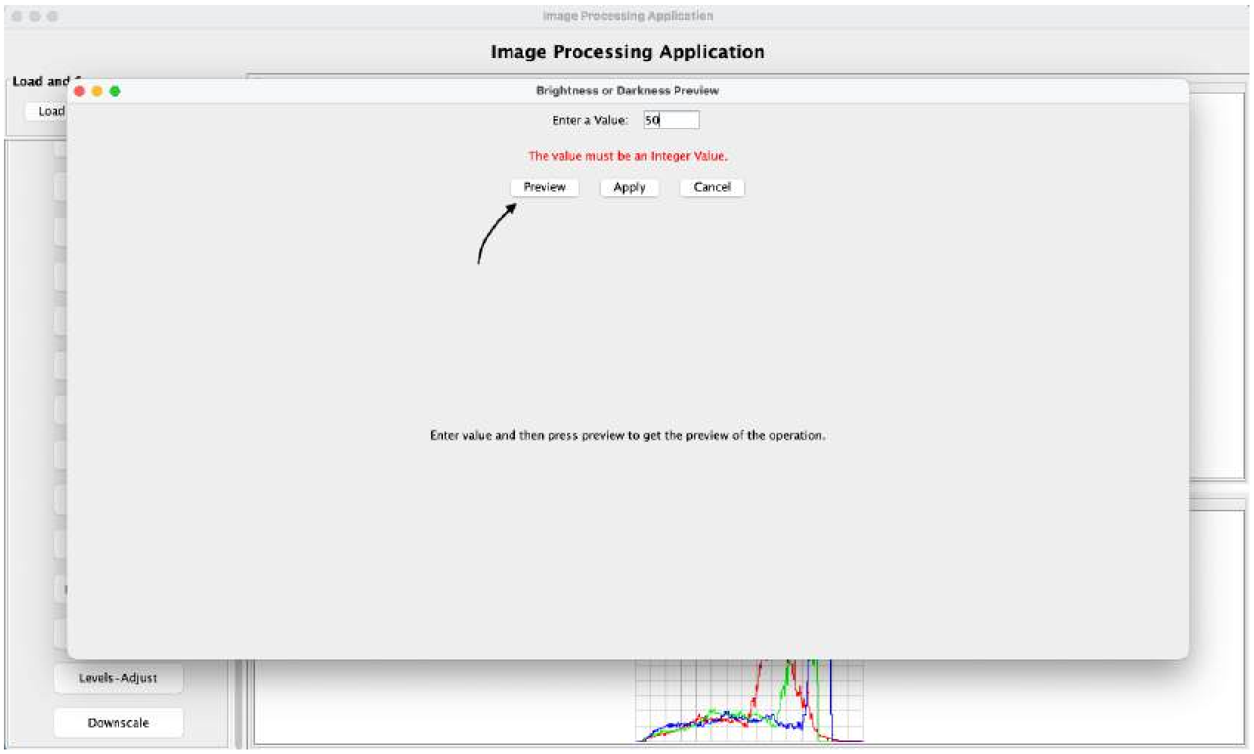


Image Processing Application
Image Processing Application
Load and
Load
Brightness or Darkness Preview
Enter a Value: 50
Preview
Apply
Cancel
Levels-Adjust
Downscale

The value must be an Integer Value.

Enter value and then press preview to get the preview of the operation.

3. **Preview Output**

The preview window displays the adjusted brightness or darkness applied to the image.

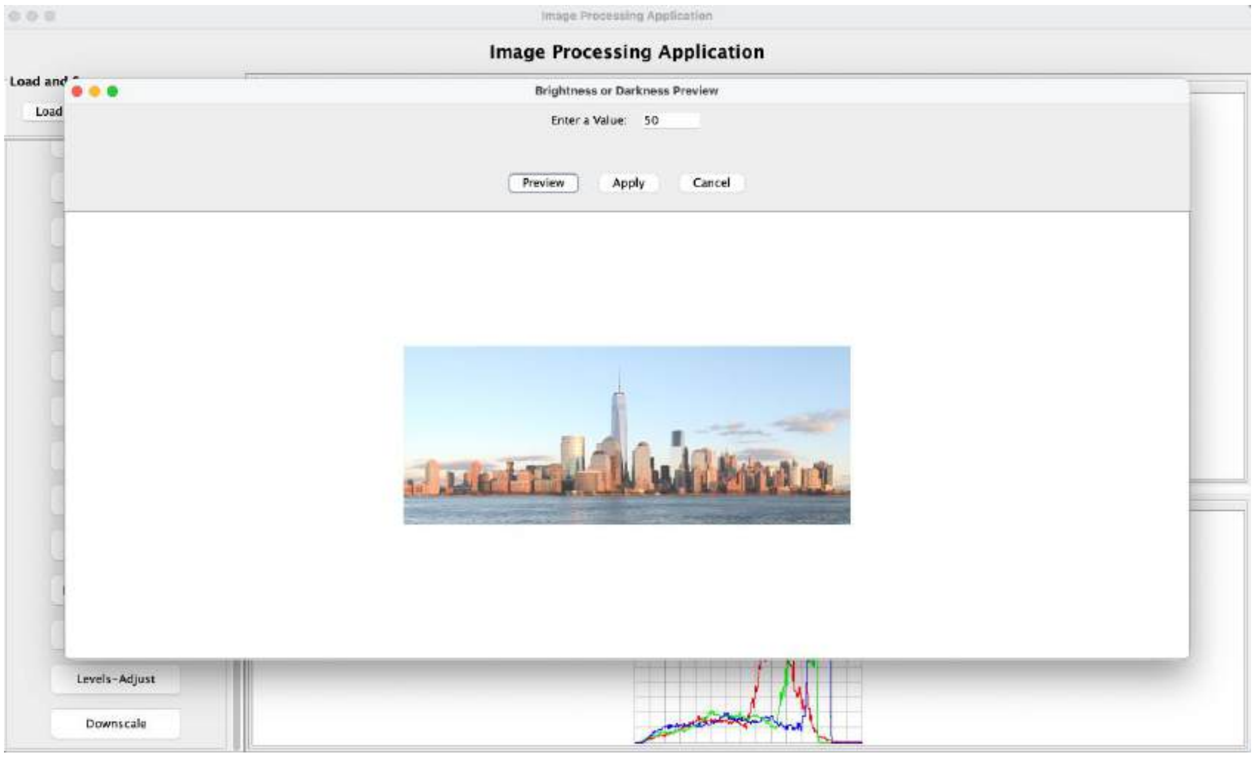


Image Processing Application
Image Processing Application
Load and
Load
Brightness or Darkness Preview
Enter a Value: 50
Preview
Apply
Cancel
Levels-Adjust
Downscale

4. **Applying Changes**

Once satisfied with the preview, click **"Apply"** to update the original image in the main frame.

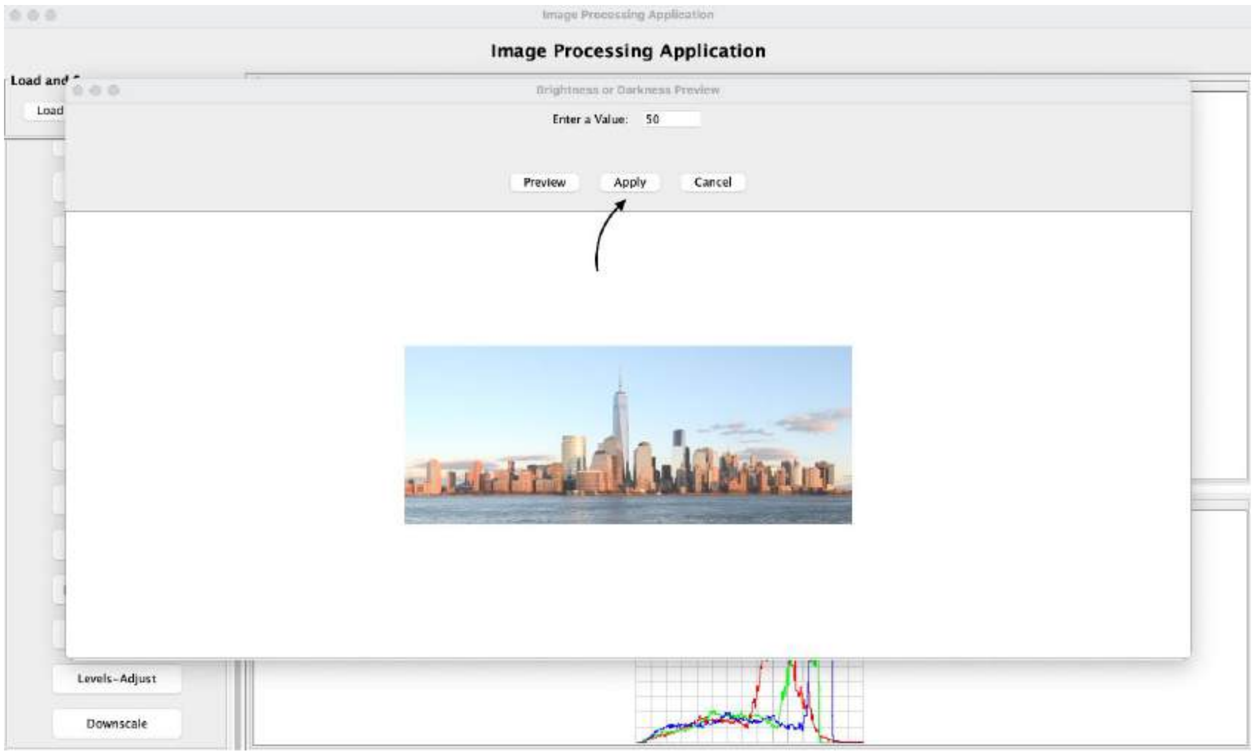


Image Processing Application
Image Processing Application
Load and
Load
Brightness or Darkness Preview
Enter a Value: 50
Preview
Apply
Cancel
Levels-Adjust
Downscale

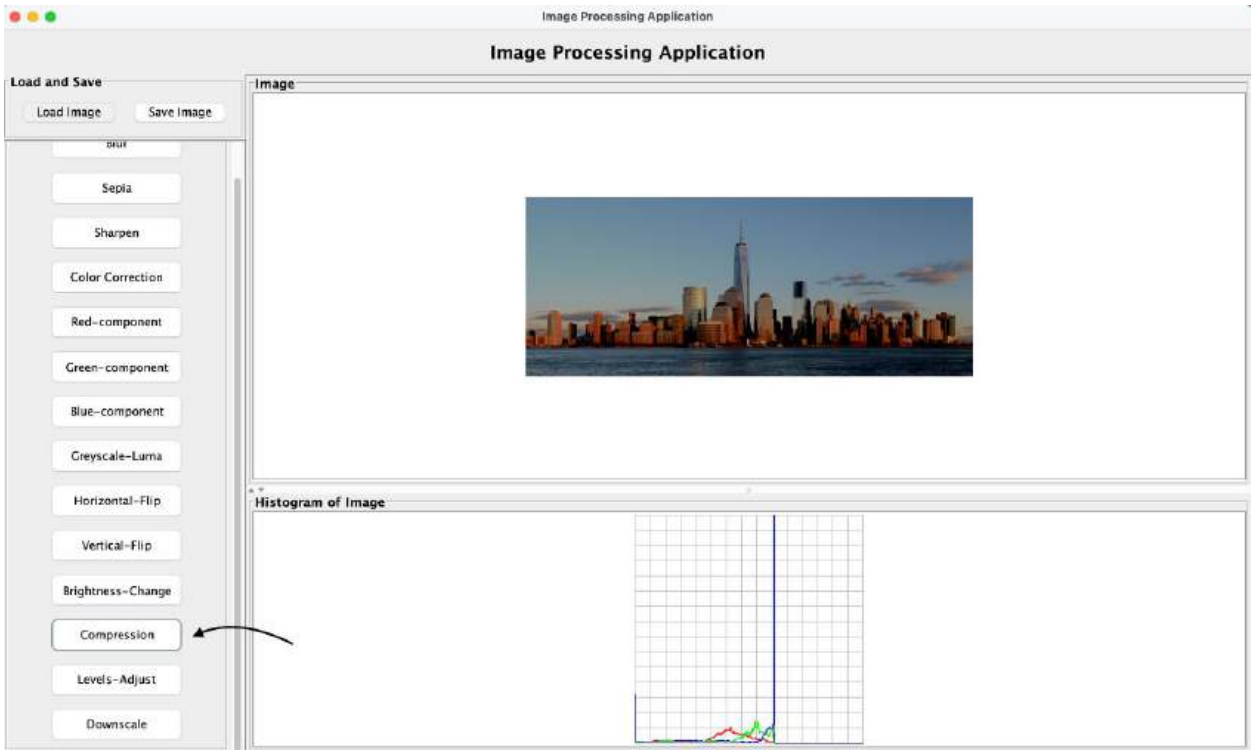
5. **Cancel the Preview** (Optional)

You can directly click **"Cancel"** after selecting the operation without previewing or can click the cross mark on the top right of the panel to close the preview operation.

Compressing an Image

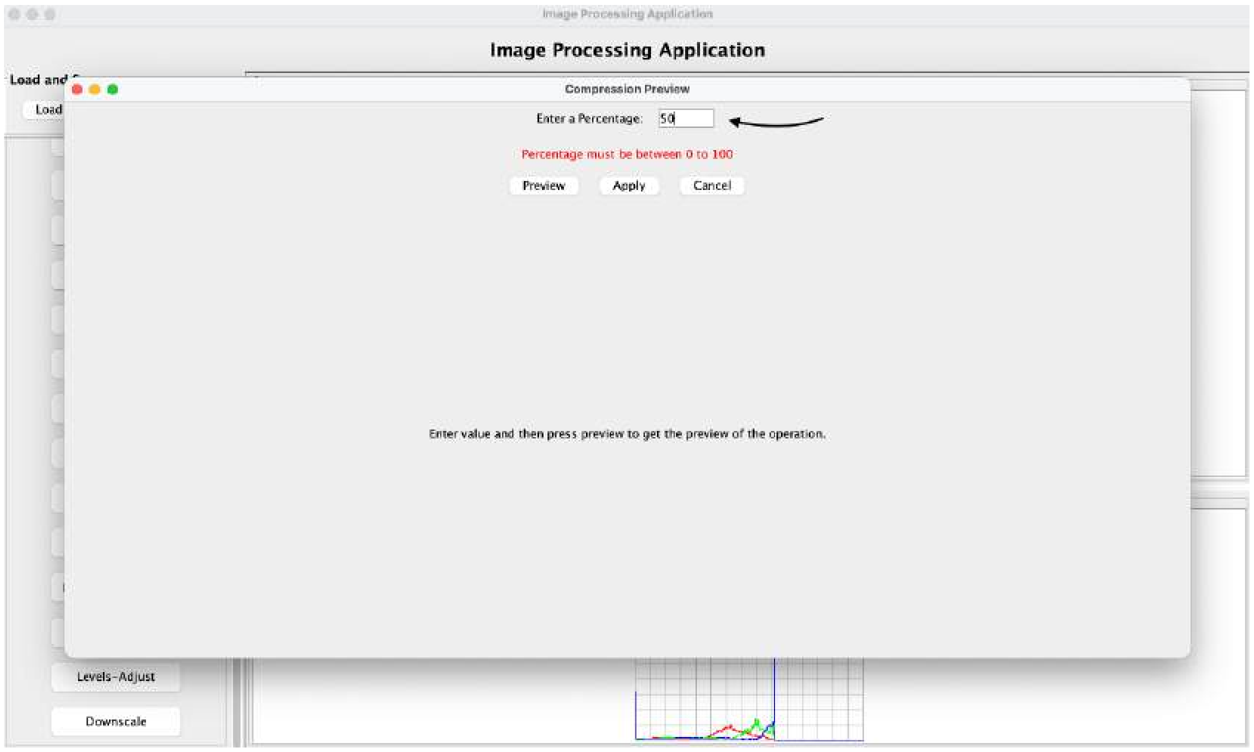
1. **Initiating Compression**

Click on the **"Compression"** button to open the compression dialog.



2. **Setting Compression Level**

Enter the percentage by which you want to compress the image. Click **"Preview"** to view the compressed version.

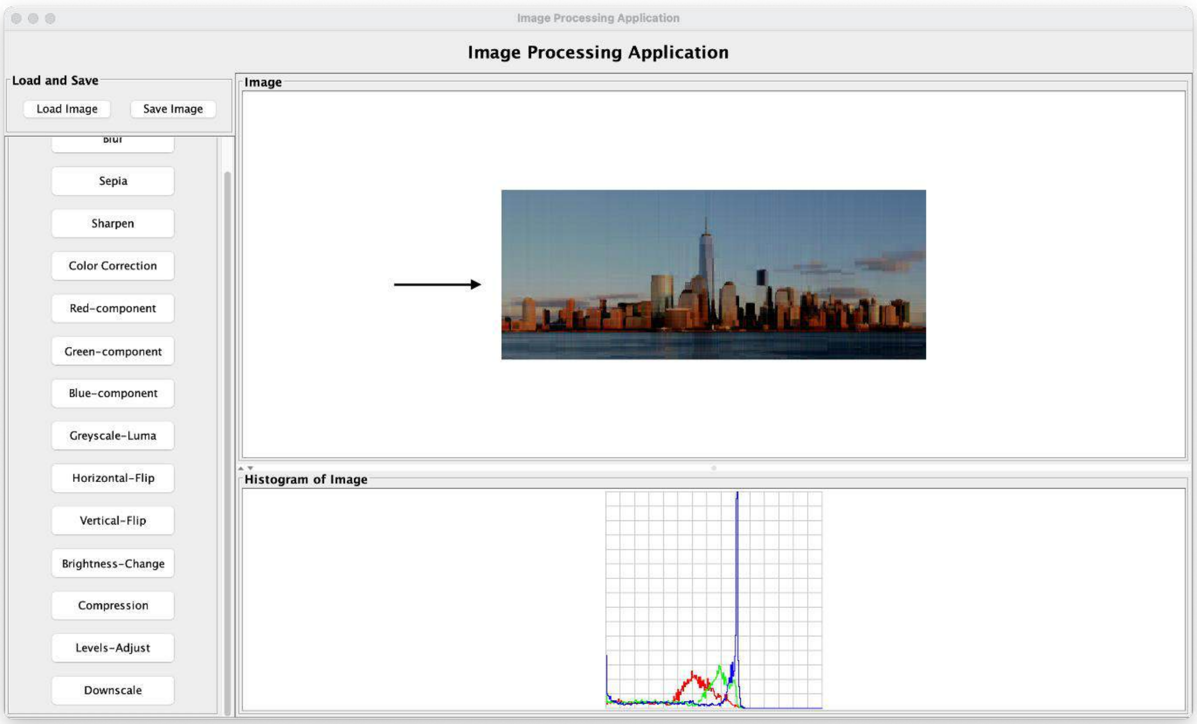


Percentage must be between 0 to 100

Enter value and then press preview to get the preview of the operation.

3. **Finalizing Compression**

Once satisfied, click **"Apply"** to compress the main image.



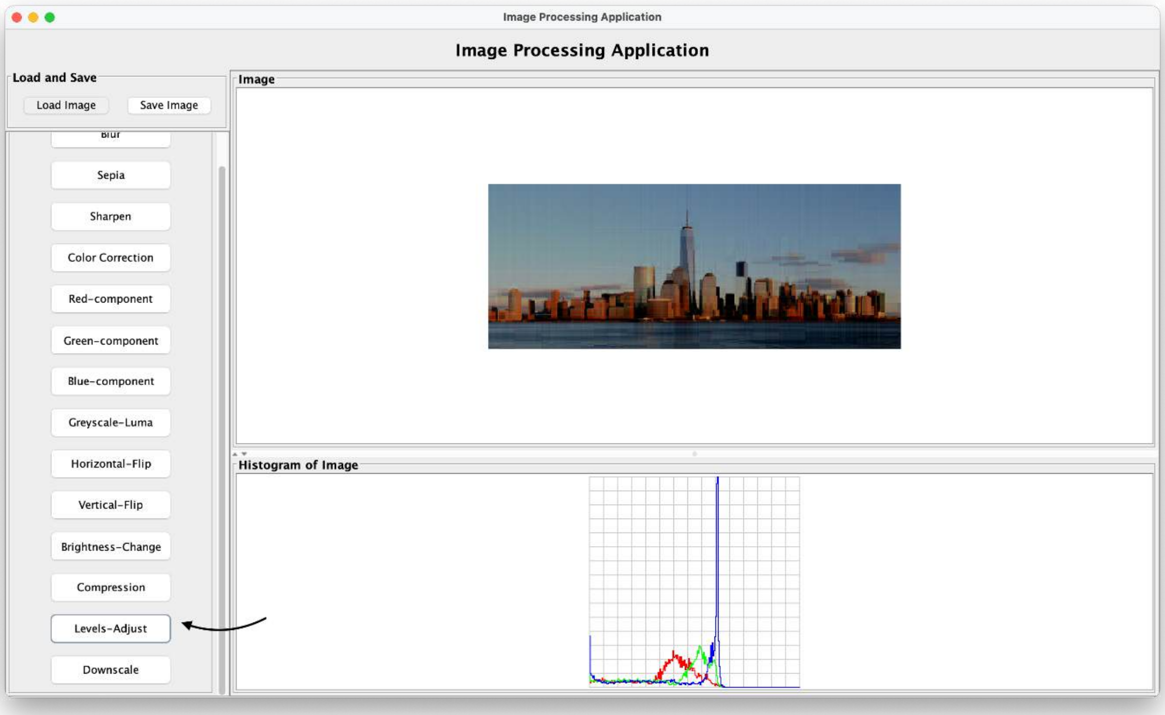
4. **Cancel the Preview** (Optional)

You can directly click **"Cancel"** after selecting the operation without previewing or can click the cross mark on the top right of the panel to close the preview operation.

Adjusting Levels

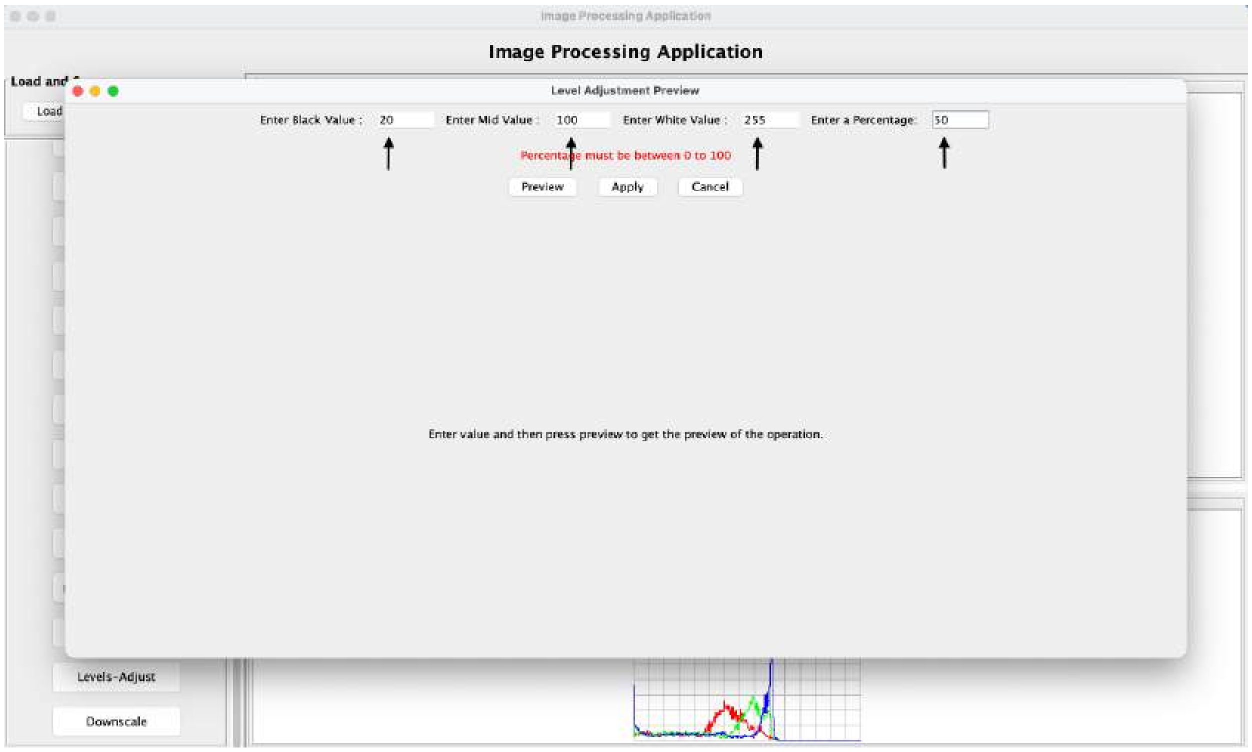
1. **Opening Levels Adjustment**

Click on the **"Levels-Adjust"** button.



2. **Providing Parameters**

Enter values for **Black**, **Mid**, and **White** points. These values must be between 0 and 255, and in ascending order. Specify the split percentage for preview as well.

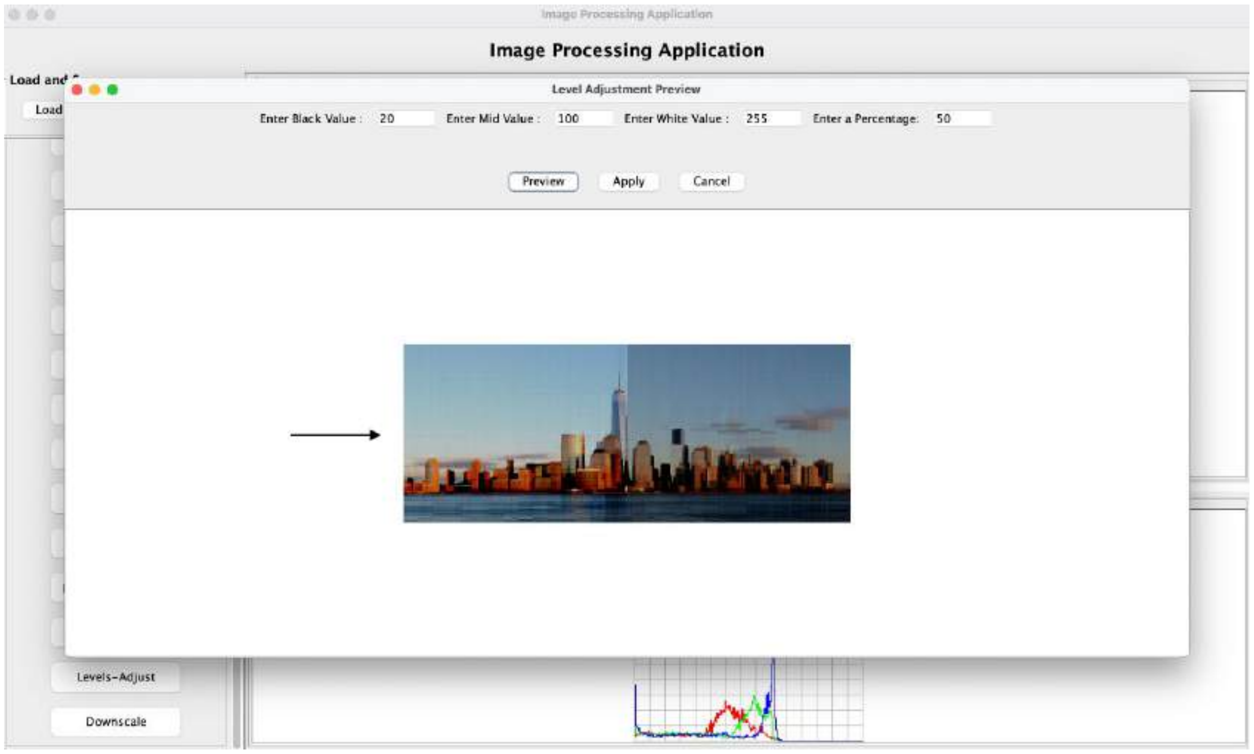


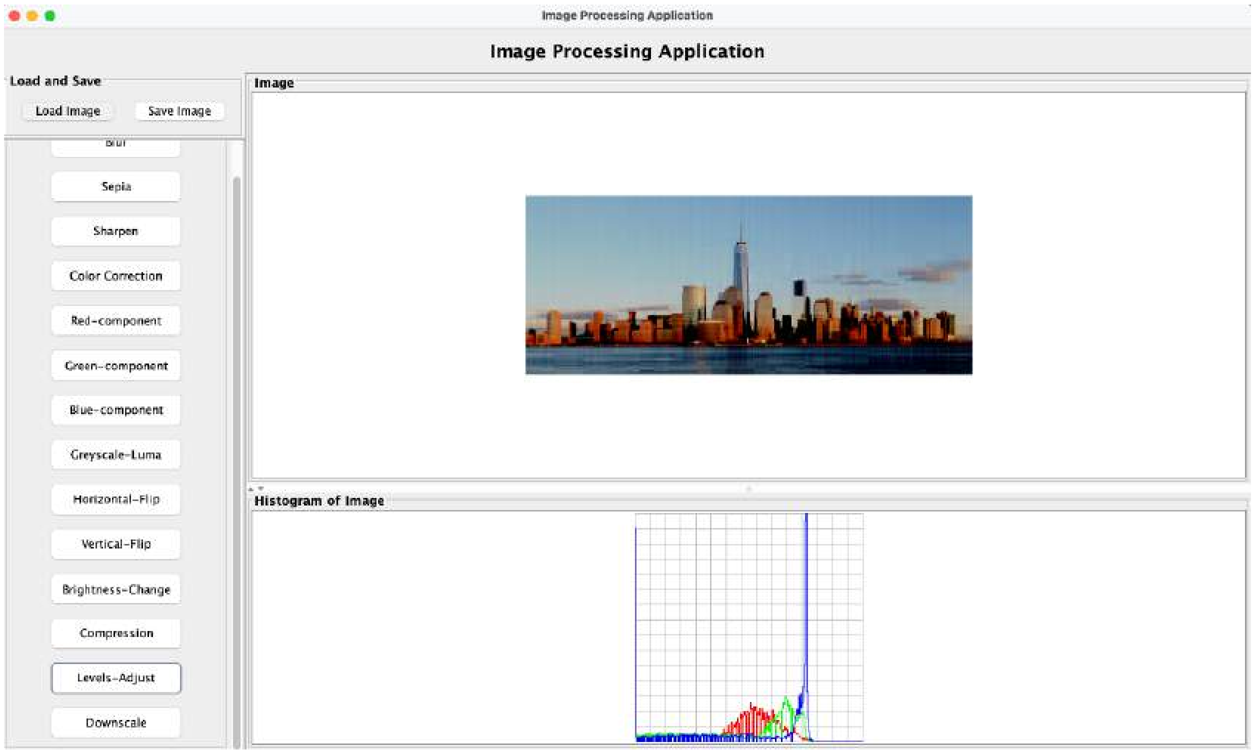
Percentage must be between 0 to 100

Enter value and then press preview to get the preview of the operation.

3. **Previewing and Applying**

Click **"Preview"** to view the effect and adjust if needed. Once satisfied, click **"Apply"** to finalize the adjustment.





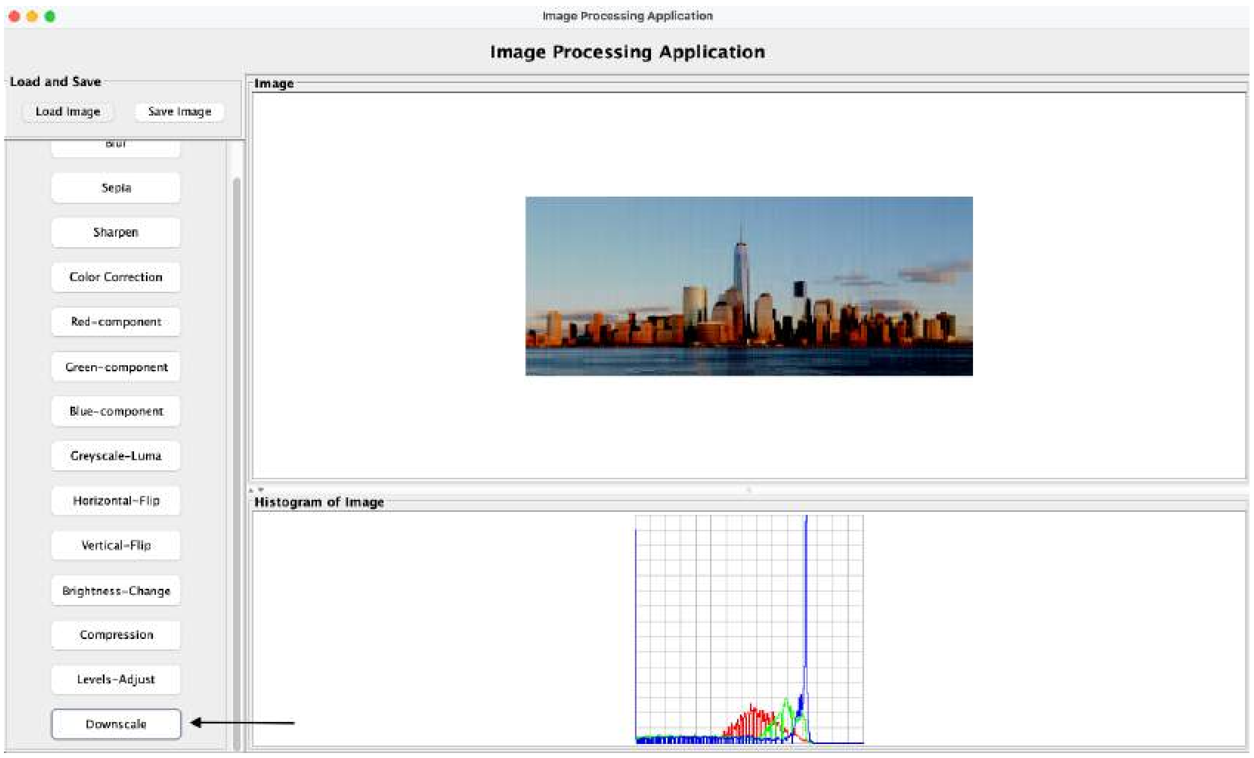
4. **Cancel the Preview** (Optional)

You can directly click **"Cancel"** after selecting the operation without previewing or can click the cross mark on the top right of the panel to close the preview operation.

Downscaling an Image

1. **Opening Downscale Tool**

Click on the **"Downscale"** button.



2. **Specifying Dimensions**

Enter the new target height and width. Ensure these values are smaller than the current image dimensions.

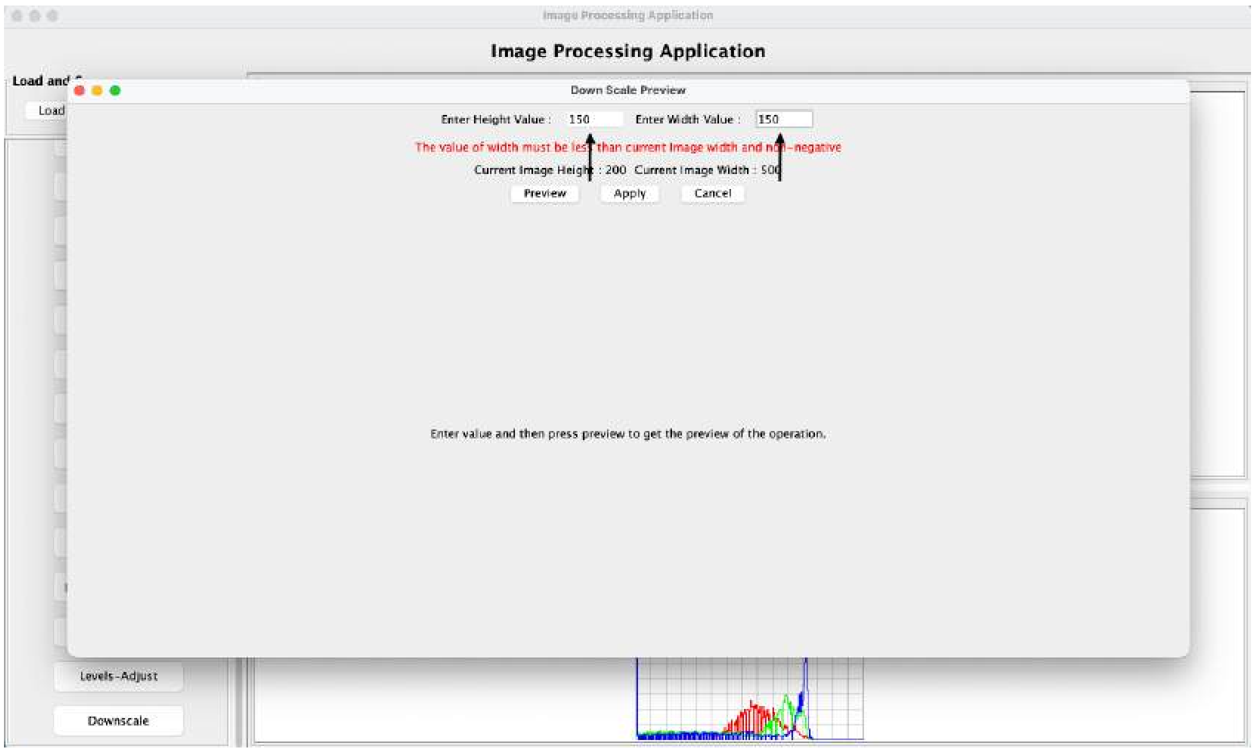


Image Processing Application

Down Scale Preview

Enter Height Value : 150 Enter Width Value : 150

The value of width must be less than current Image width and non-negative

Current Image Height : 200 Current Image Width : 500

Enter value and then press preview to get the preview of the operation.

3. **Previewing the Result**

Click **"Preview"** to see the downscaled version in the preview window.

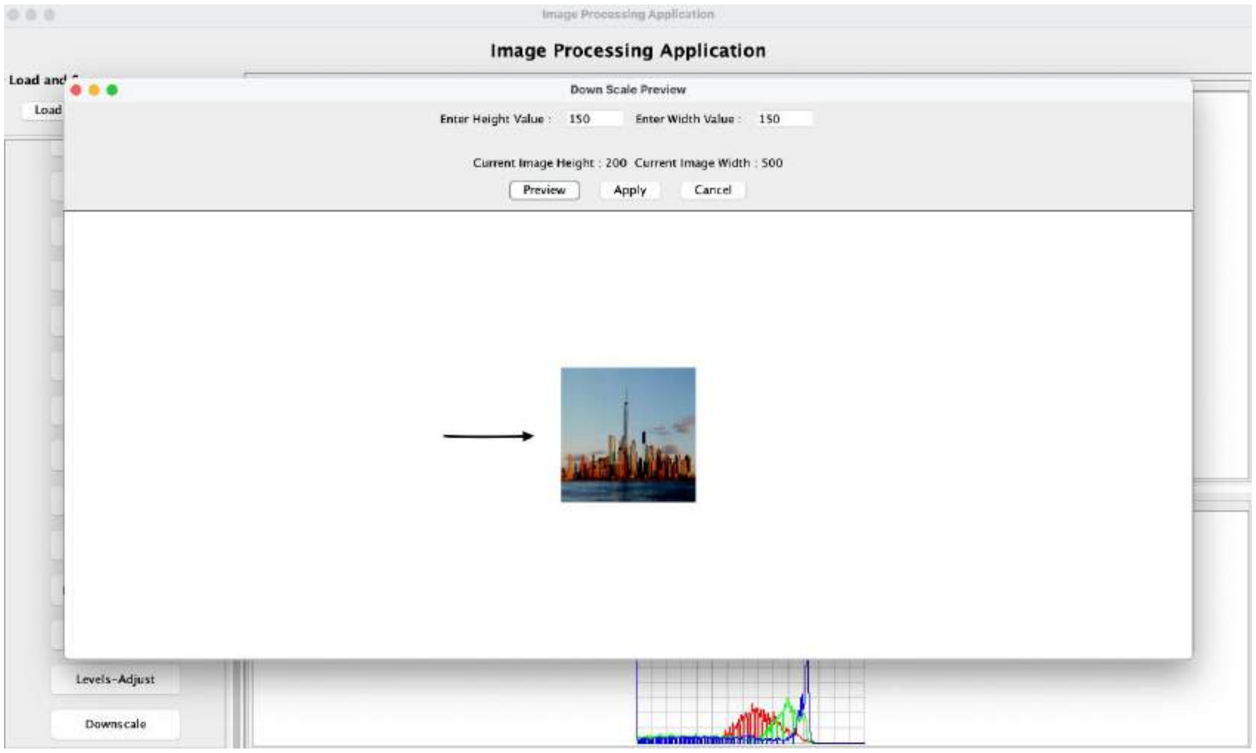


Image Processing Application

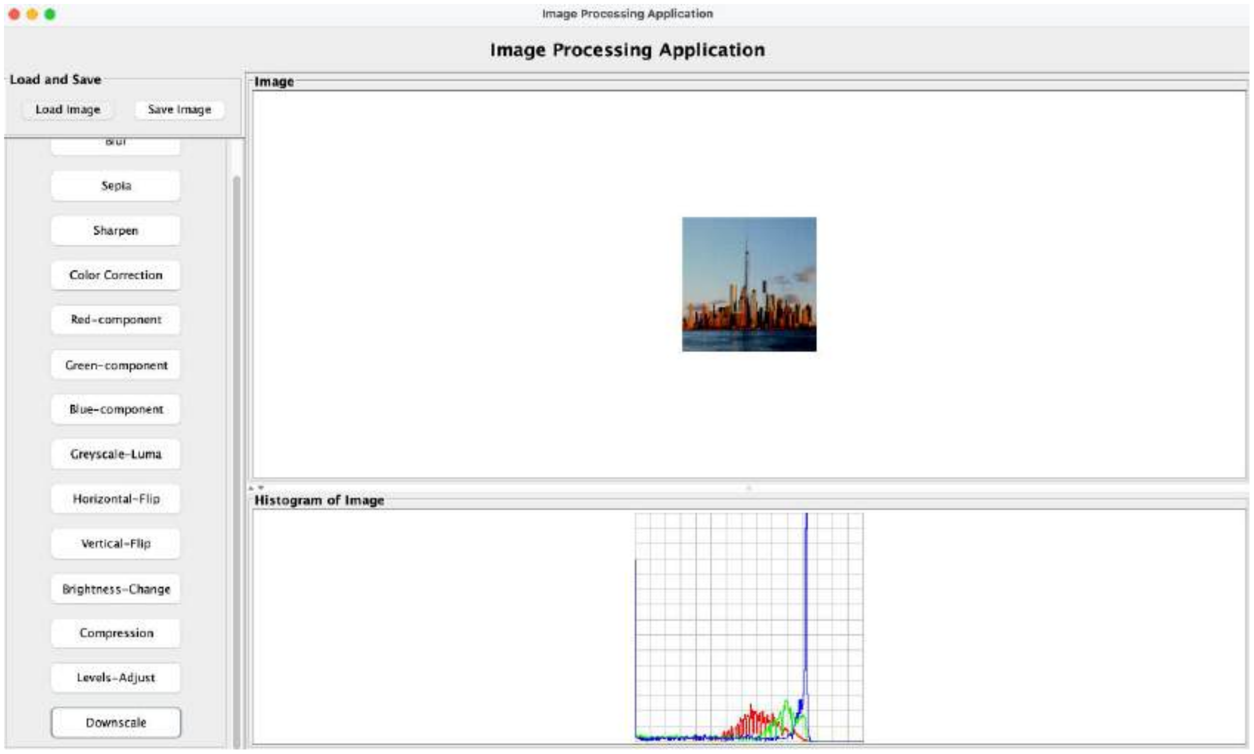
Down Scale Preview

Enter Height Value : 150 Enter Width Value : 150

Current Image Height : 200 Current Image Width : 500

4. **Applying the Changes**

Once satisfied, click **“Apply”** to update the main image with the downscaled dimensions.



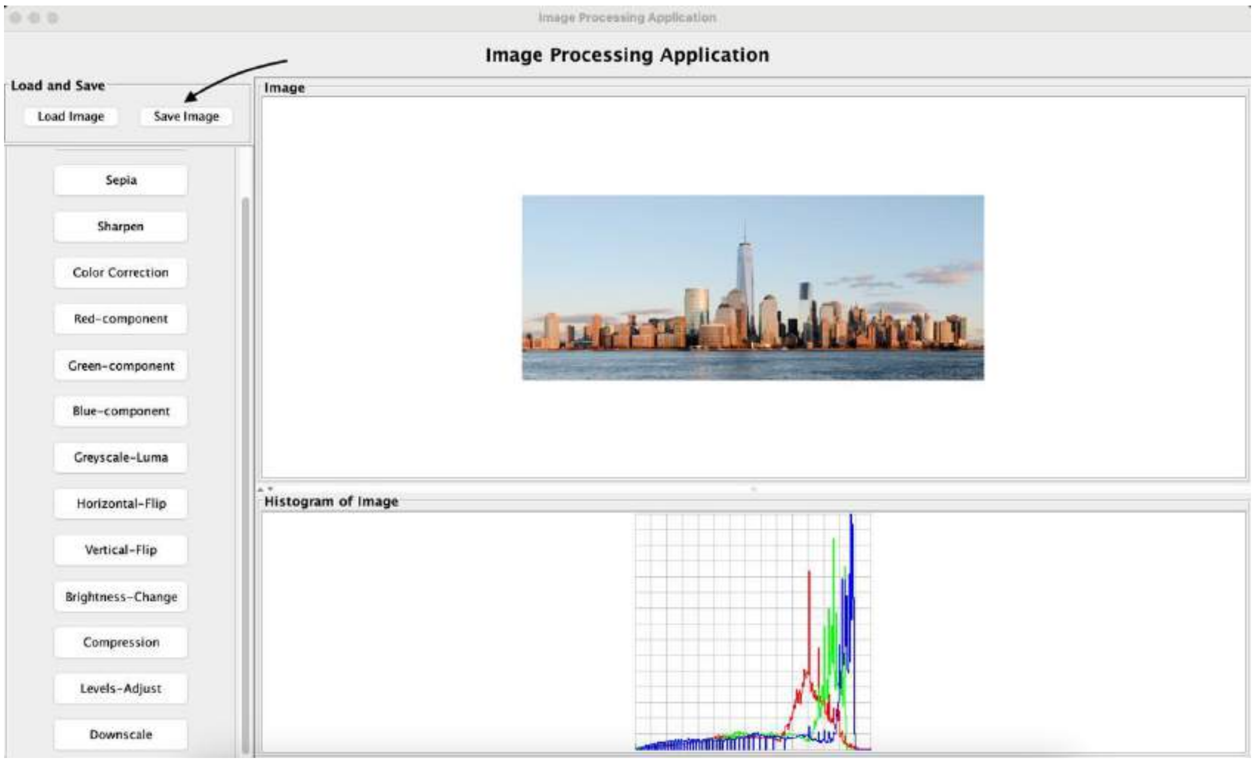
5. **Cancel the Preview** (Optional)

You can directly click **“Cancel”** after selecting the operation without previewing or can click the cross mark on the top right of the panel to close the preview operation.

Saving an Image

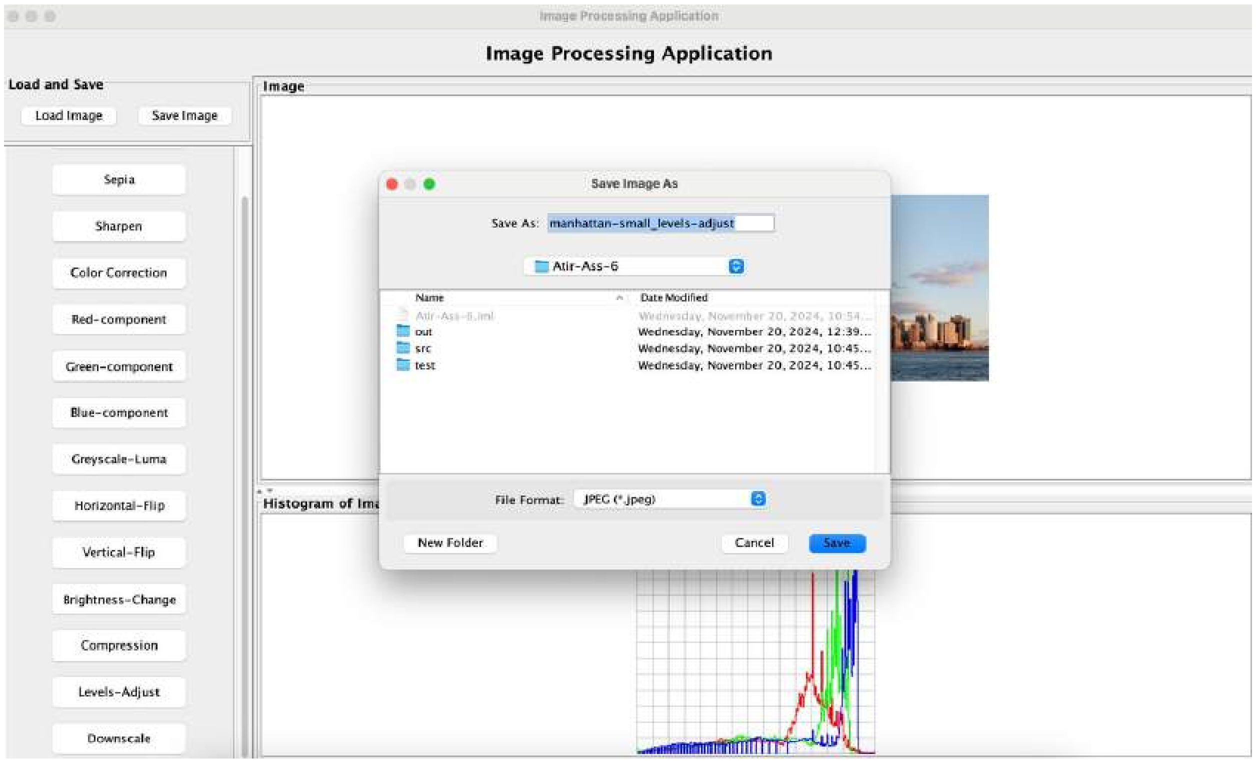
1. **Initiating the Save Process**

To begin, click on the **“Save”** button located at the top-left corner of the interface.



2. **Selecting an Image**

A file selection dialog will appear, allowing you to navigate through directories and select an image file where the image is to be stored. Also, a drop-down option is available, which allows the user to choose extension with which image is to be stored.

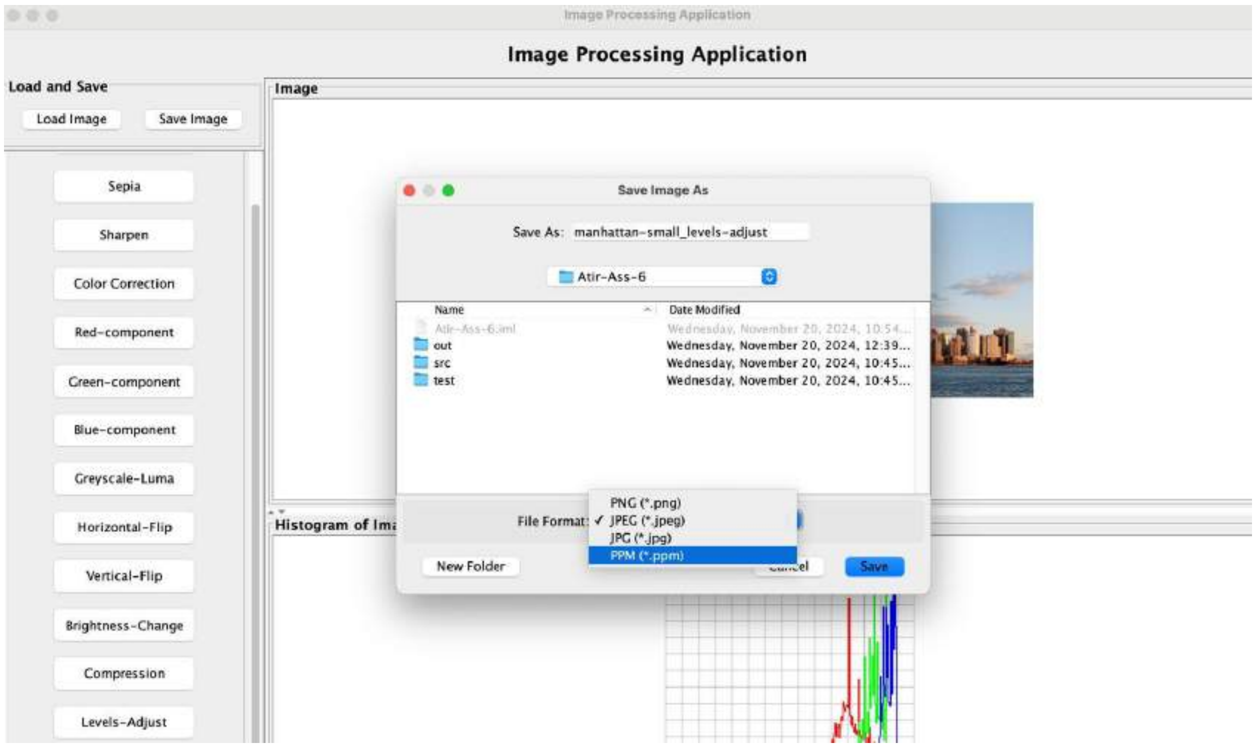


Save As: manhattan-small_levels-adjust

Atir-Ass-6

Name	Date Modified
Atir-Ass-6.iml	Wednesday, November 20, 2024, 10:54...
out	Wednesday, November 20, 2024, 12:39...
src	Wednesday, November 20, 2024, 10:45...
test	Wednesday, November 20, 2024, 10:45...

File Format: JPEG (*.jpeg)



Save As: manhattan-small_levels-adjust

Atir-Ass-6

Name	Date Modified
Atir-Ass-6.iml	Wednesday, November 20, 2024, 10:54...
out	Wednesday, November 20, 2024, 12:39...
src	Wednesday, November 20, 2024, 10:45...
test	Wednesday, November 20, 2024, 10:45...

File Format: PNG (*.png) / ✓ JPEG (*.jpeg) / JPG (*.jpg) / PPM (*.ppm)

3. **Saving the Image**

Once you've chosen the desired image, click the **"Save"** button. This action will save the image at the choosen location.

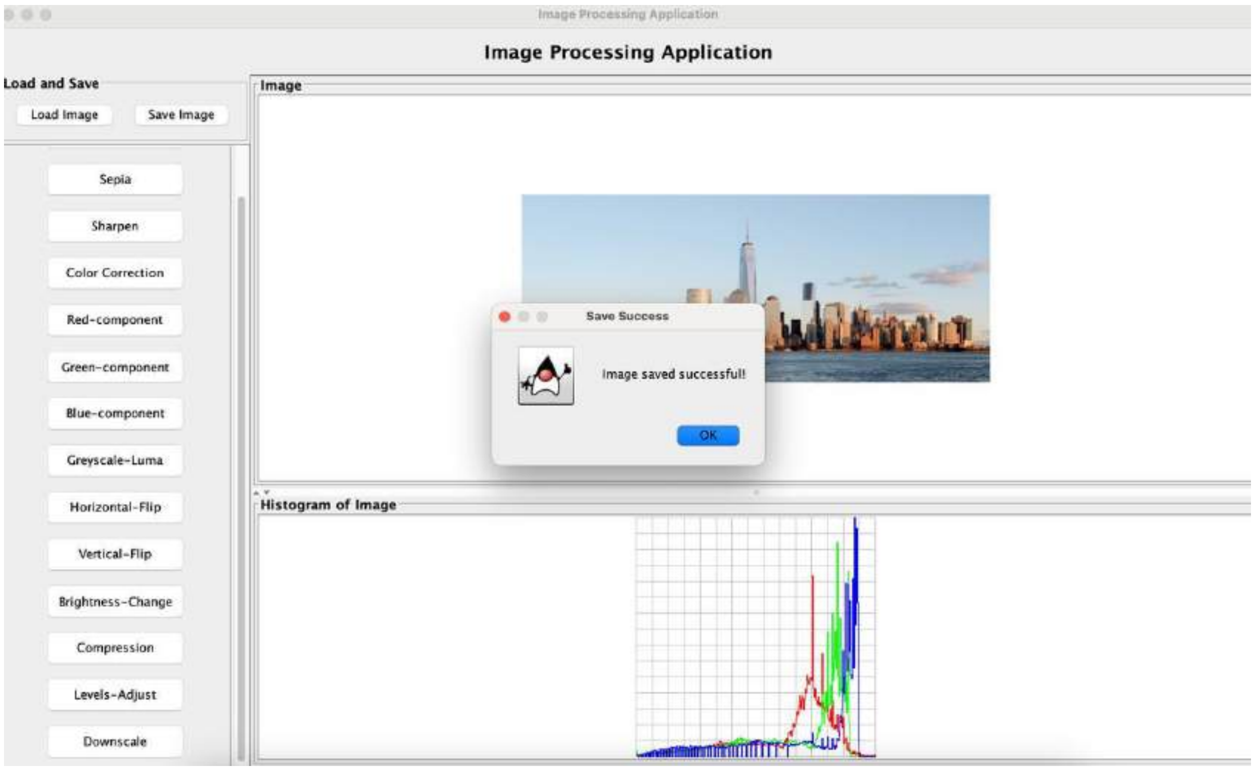


Image saved successfull!

4. **Cancel the Preview** (Optional)

You can directly click **“Cancel”** after selecting the operation or can click the cross mark on the top right of the panel to close the operation.

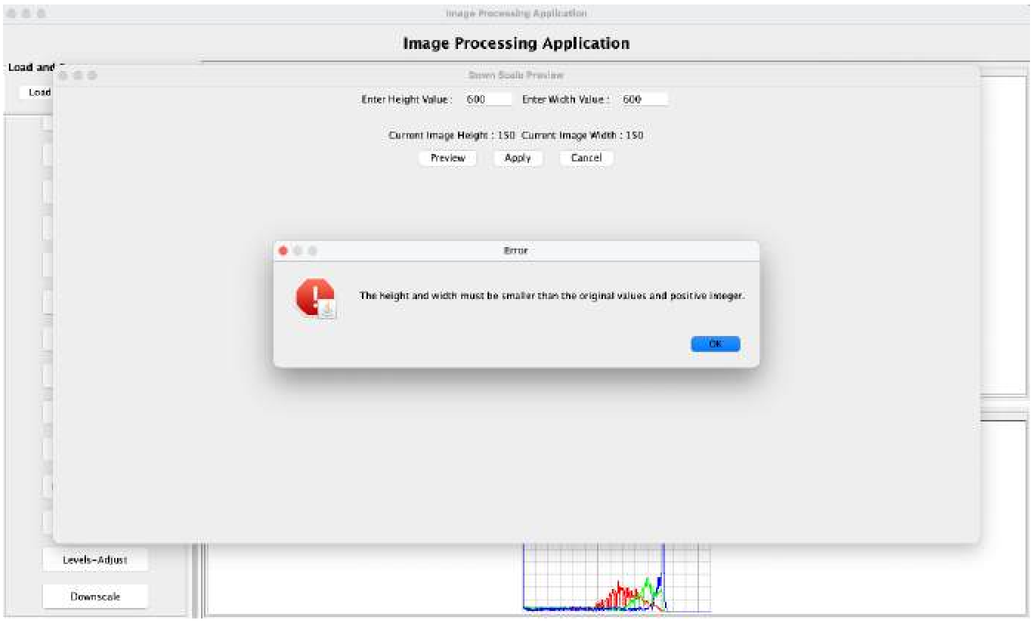
Below is the content for the “Error Handling” section for your GUI documentation. It is modeled in a similar style to the error pop-ups shown in your screenshots.

Error Handling

This application includes comprehensive error handling to ensure users provide valid inputs for operations. Error messages are displayed as pop-up alerts, providing specific guidance to users. Below are examples of the types of errors handled:

1. **Invalid Downscale Dimensions**

   - **Error Message:** “The height and width must be smaller than the original values and positive integers.”
   - **Description:** This error occurs if the user enters downscale dimensions (height or width) that are greater than or equal to the current image dimensions or if the values are non-positive integers.
   - **Example Scenario:**
     - Current Image Height: 150, Current Image Width: 150
     - Input: Height = 600, Width = 600 (Invalid)



Current Image Height : 150 Current Image Width : 150

The height and width must be smaller than the original values and positive integer.

- **Resolution:** Enter values smaller than the current dimensions and ensure the values are positive integers.

2. **Invalid Level Adjustment Values**

   - **Error Message:** “All values (B, M, and W) must be between 0 to 255.”
   - **Description:** This error is triggered when the user enters values for Black (B), Mid (M), or White (W) levels that are outside the range of 0-255. This error will be shown when the user clicks the preview or the apply button.
   - **Example Scenario:**
     - Input: Black = -90, Mid = -20, White = 30 (Invalid)

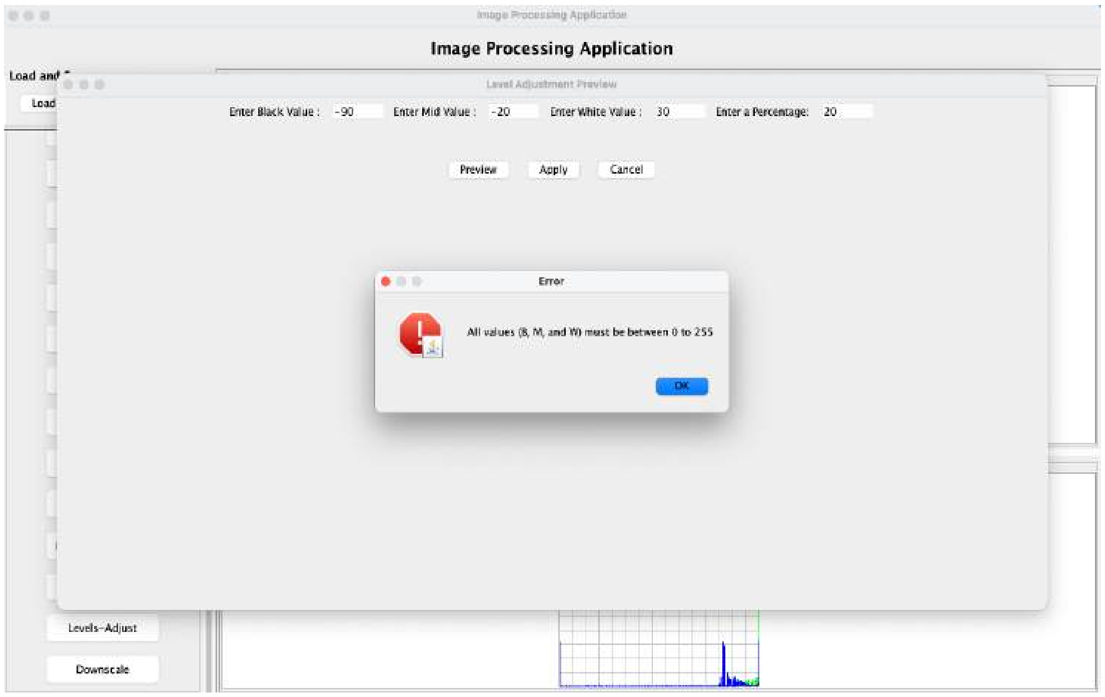


Image Processing Application

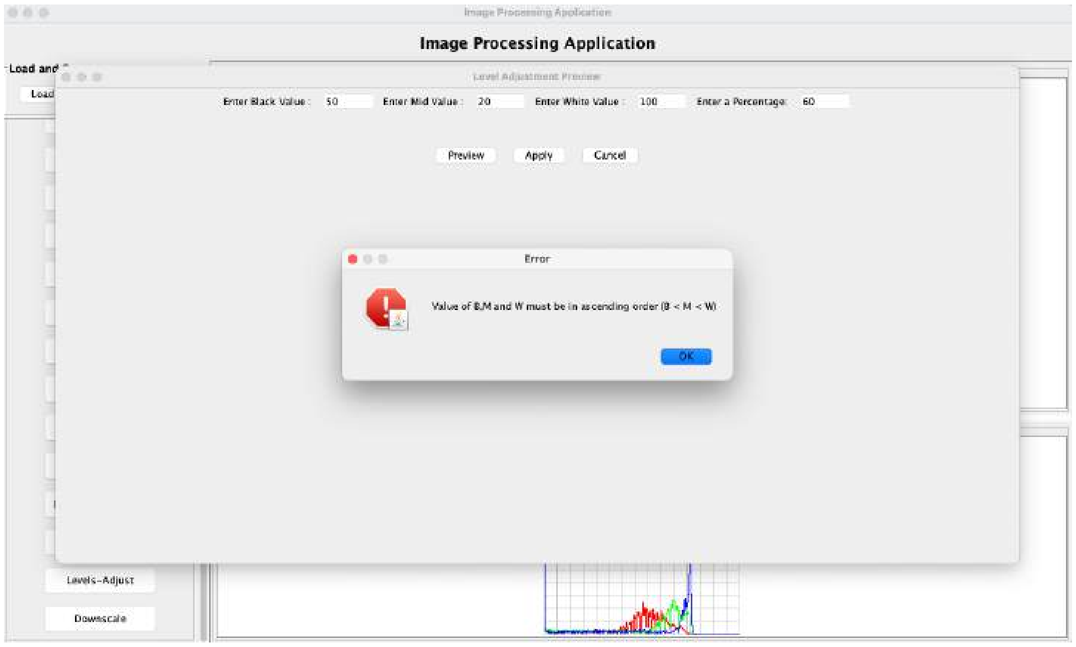
Level Adjustment Preview

Enter Black Value : -90 Enter Mid Value : -20 Enter White Value : 30 Enter a Percentage: 20

Error

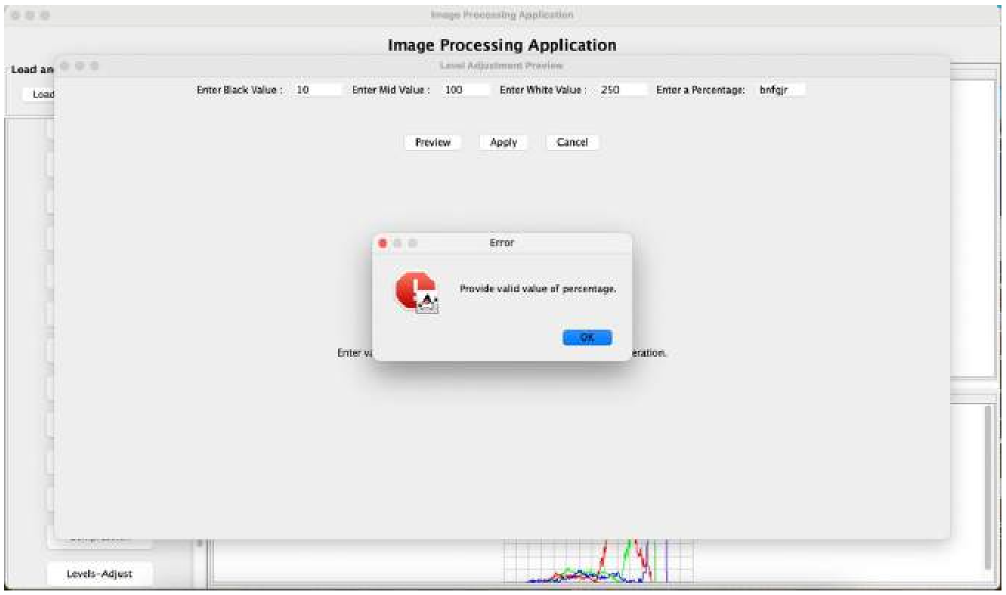
All values (B, M, and W) must be between 0 to 255

   - **Resolution:** Ensure that all level values are within the valid range of 0 to 255.
   - **Error Message:** “Value of B, M and W must be in ascending order (B < M < W).
   - **Description:** This error is triggered when the user enter values for Black (B), Mid (M), and White (W) levels in order which is not ascending. This error will be shown when the user clicks the preview or the apply button.
   - **Example Scenario:**
     - Input: Black = 50, Mid = 20, White = 100 (Invalid)



Value of B,M and W must be in ascending order (B < M < W)

- **Resolution:** Ensure that all values are in ascending order.
- **Error Message:** "Provide a valid value of percentage."
- **Description:** This error occurs if the percentage value for a certain operation is not valid, such as negative values or non-numeric entries. This error will be shown when user clicks the preview button.

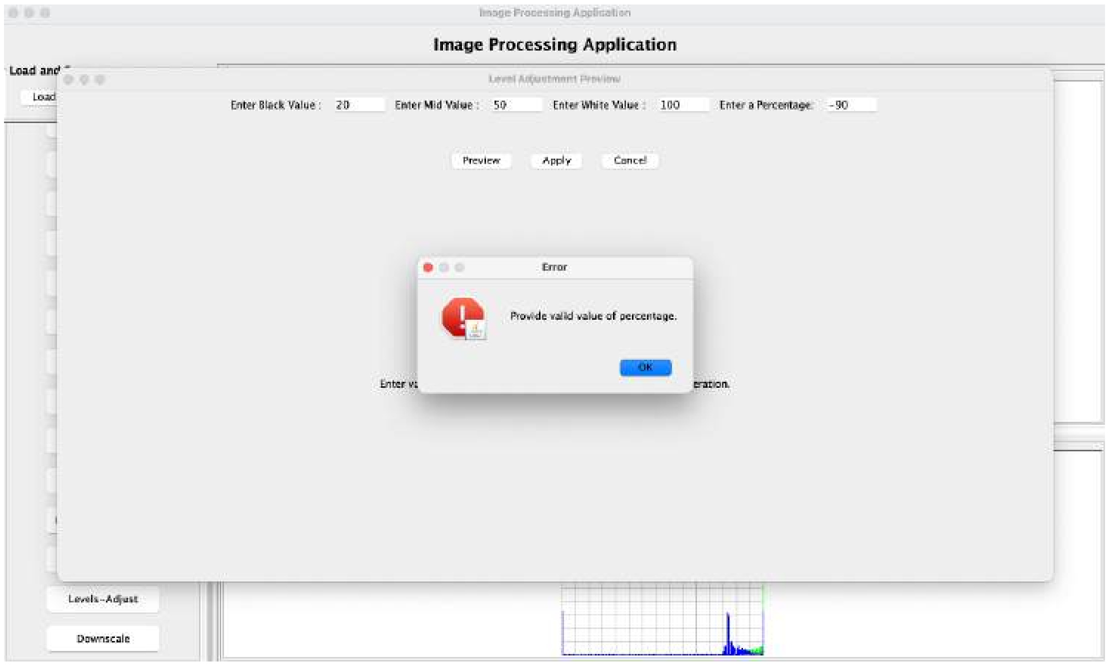


Provide valid value of percentage.

- **Resolution:** Provide a valid percentage value greater than or equal to 0, less than or equal to 100 and a numeric value.

3. **Invalid Percentage Value for all Split Preview Operation**

- **Error Message:** “Provide a valid value of percentage.”
- **Description:** This error occurs if the percentage value for a certain operation is not valid, such as negative values or non-numeric entries. This error will be shown when the user clicks the preview button.
- **Example Scenario:**
  - Input: Percentage = -90 (Invalid)



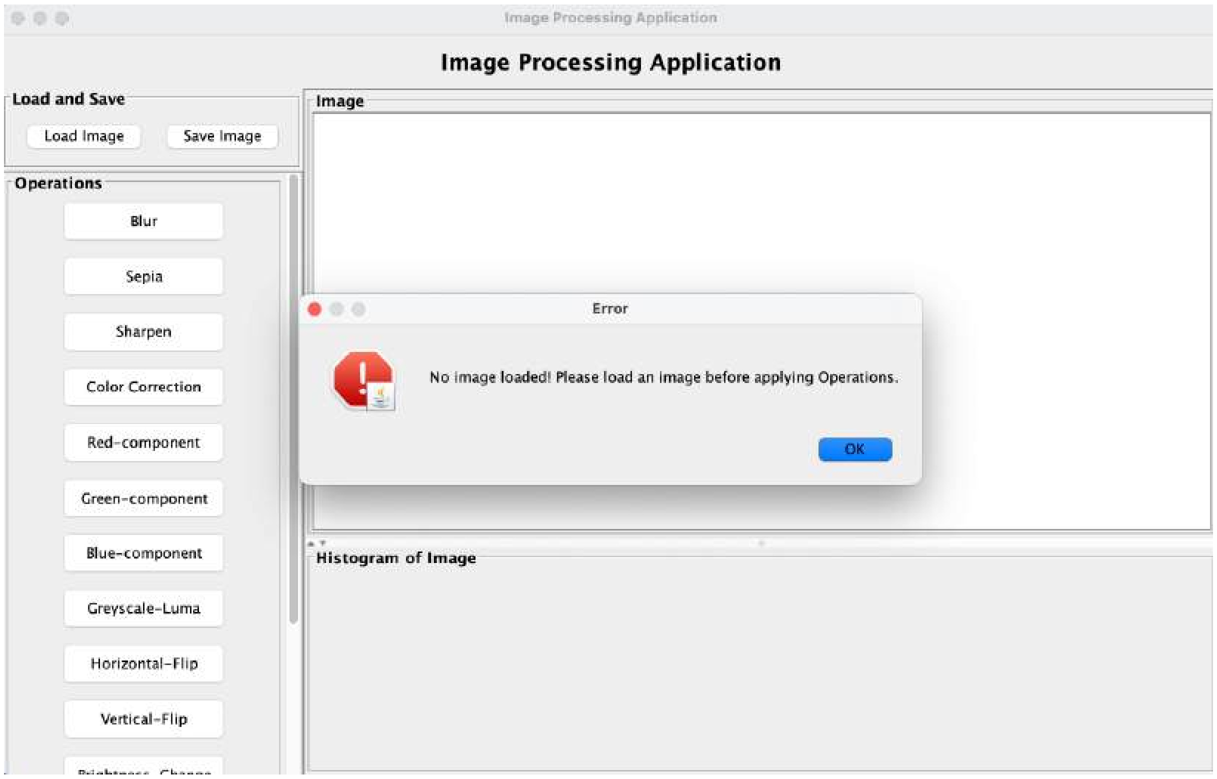
Provide valid value of percentage.

- **Resolution:** Provide a valid percentage value greater than or equal to 0, less than or equal to 100 and a numeric value.

4. **No Image Loaded**

- **Error Message:** “No image loaded! Please load an image before applying in Operations.”

- **Description:** This error occurs when you try to apply any operation or click on “Save” before loading an image.



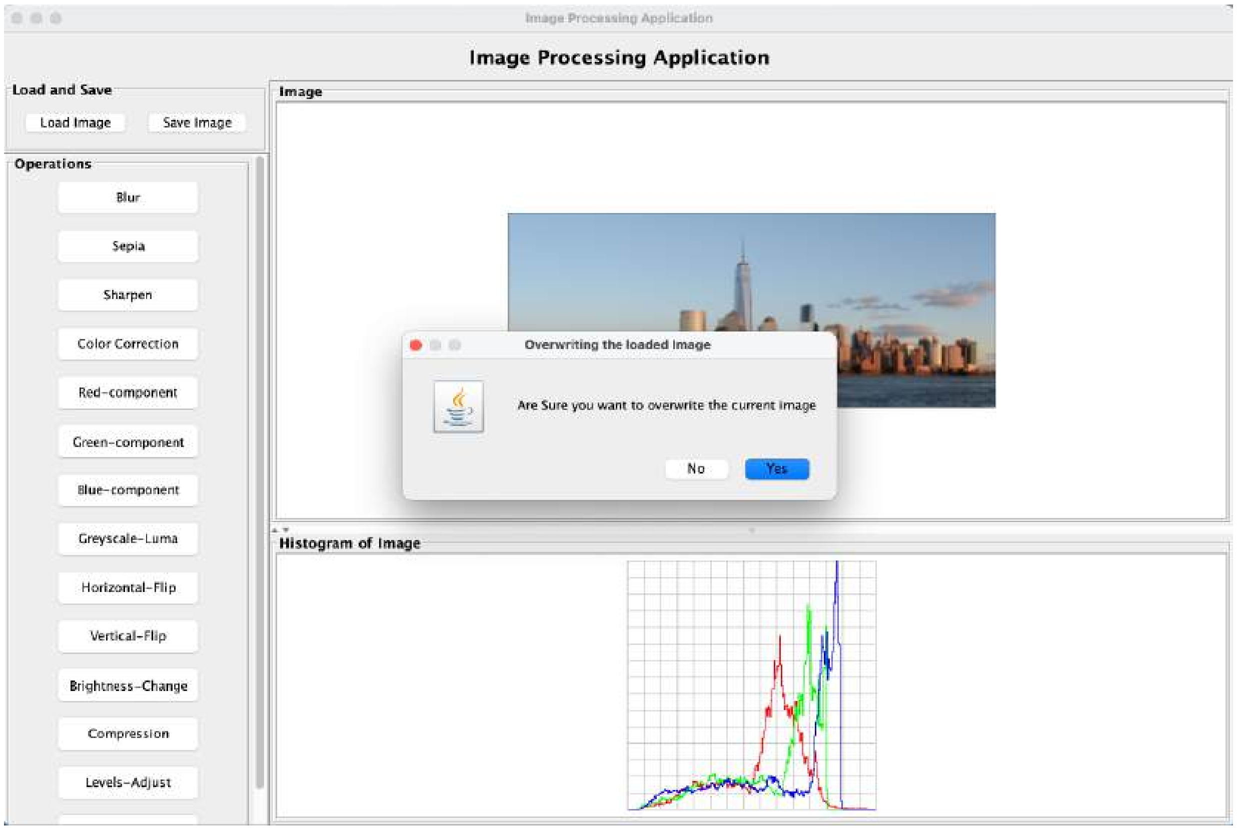
No image loaded! Please load an image before applying Operations.

– **Resolution:** Load an image before doing anything else.

5. **Overwriting Image**

   – **Warning Message:** "Are you sure you want to overwrite the current image.

   – **Description:** This popup appears when you try loading an image when there is already a loaded image in the application.



Overwriting the loaded Image

Are Sure you want to overwrite the current image

- **Resolution:** Clicking on "Yes" overwrites the current image in application.

6. **Invalid Compress Percentage Value**

   - **Error Message:** Provide valid value of percentage.

   - **Description:** When performing compress operation in the GUI is the value of percentage is not between 0 to 100 and not numeric then this error is thrown. This error is showen to user when user doesn't provide a valid percentage value when performing preview button is clicked and also when apply button is clicked by the user.

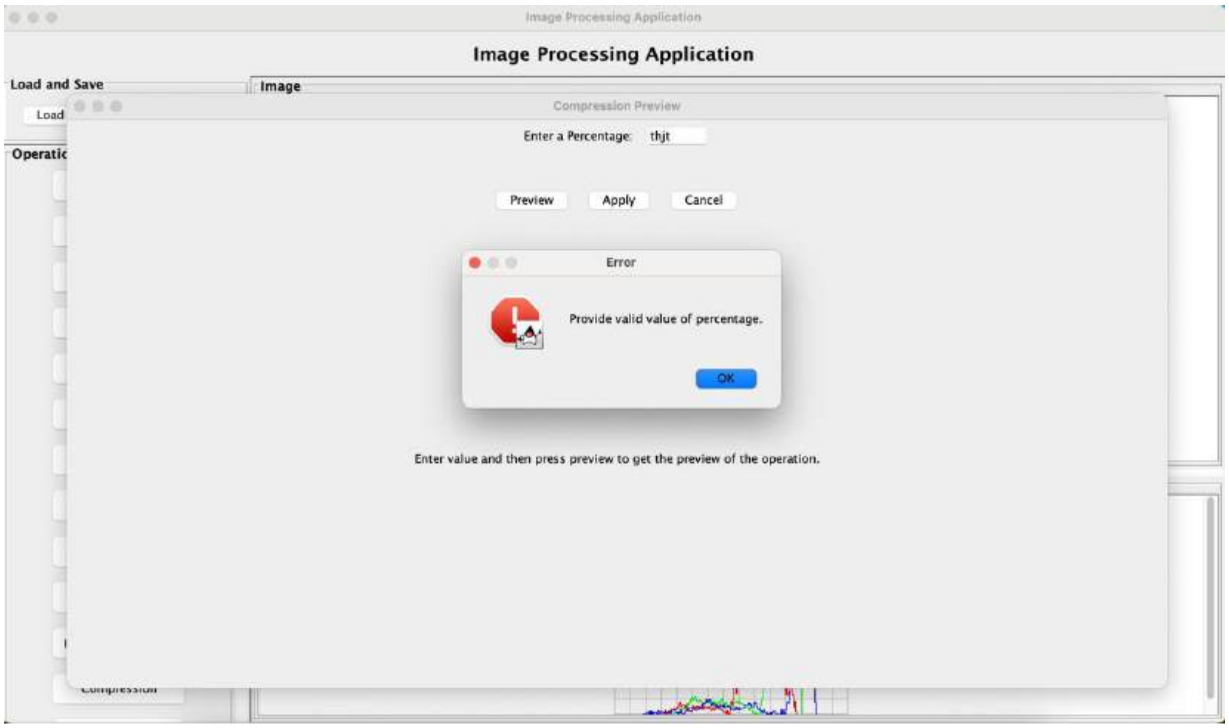


Image Processing Application

Compression Preview

Enter a Percentage: thjt

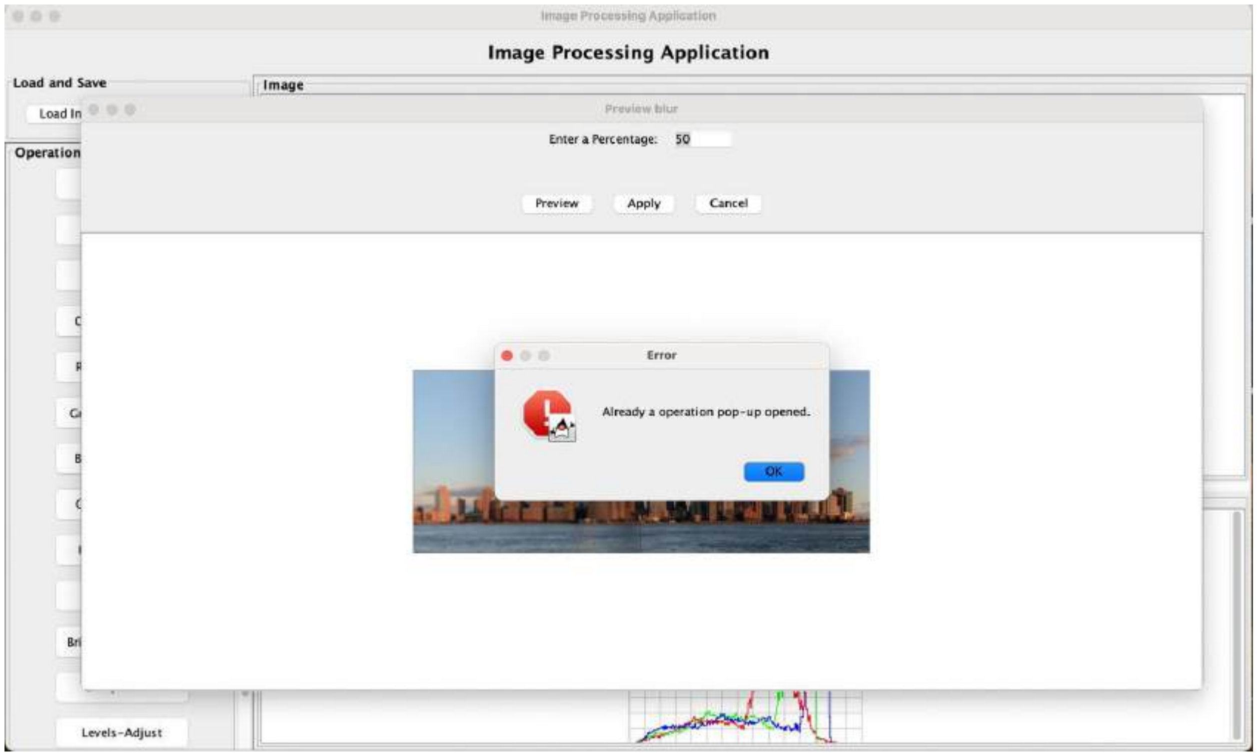
Error

Provide valid value of percentage.

Enter value and then press preview to get the preview of the operation.

- **Resolution:** Clicking on "Yes" overwrites the current image in application.

7. **Trying to open multiple preview frame at same time**
    - **Error Message:** Already a operation pop-up opened.
    - **Description:** When the user tries to open multiple preview frames by clicking multiple buttons when already a preview frame is opened.



Already a operation pop-up opened.

Each error message is designed to be clear and actionable, helping users quickly identify and resolve input issues.

This step-by-step guide ensures users can easily navigate through the GUI to perform a variety of operations with clarity and precision. This `USEME.md` file provides a comprehensive guide to all commands, syntax, and examples. Follow each step and condition carefully to ensure a smooth experience with the application.