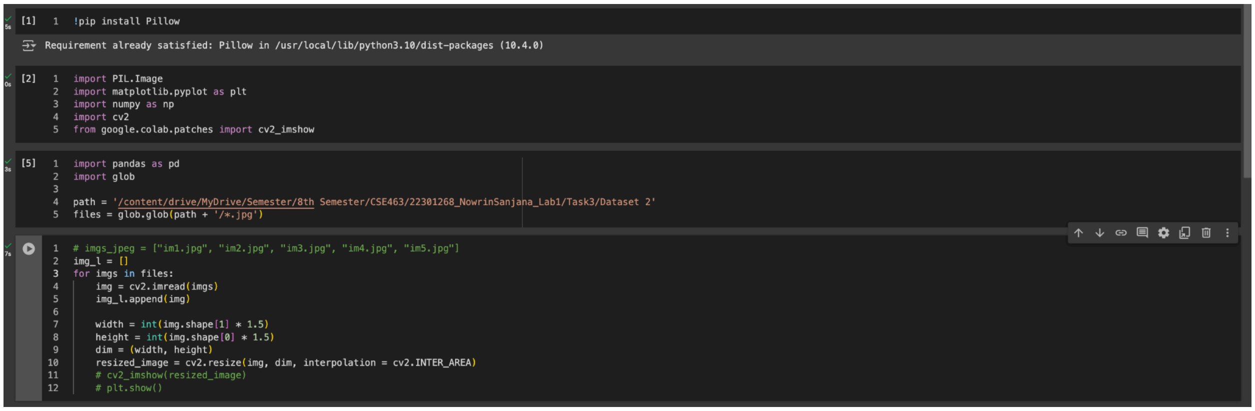
Task3(Code) blend of 5 images and gaussian noise in the blended image:



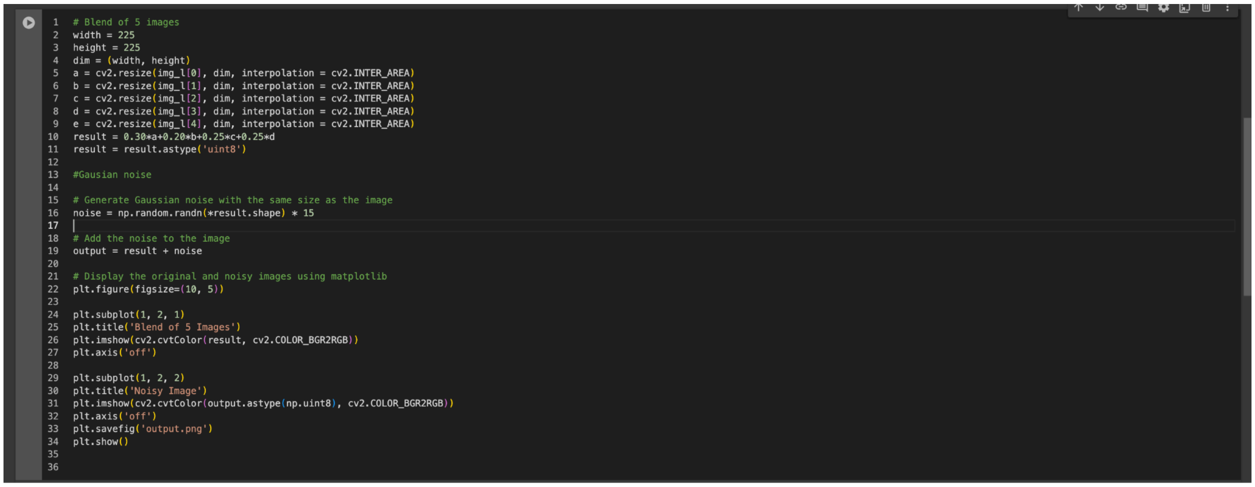
```
[1]
!pip install Pillow
```

```
Requirement already satisfied: Pillow in /usr/local/lib/python3.10/dist-packages (10.4.0)
```

```
[2]
import PIL.Image
import matplotlib.pyplot as plt
import numpy as np
import cv2
from google.colab.patches import cv2_imshow
```

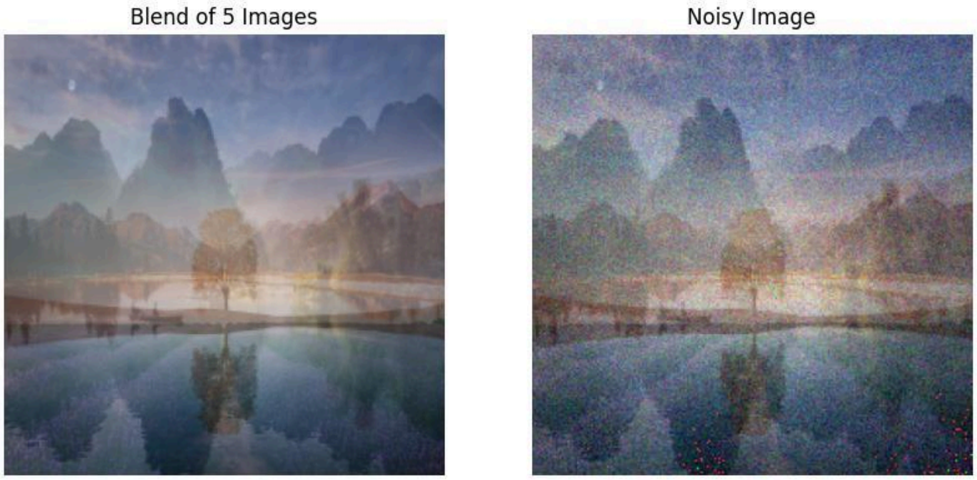
```
[5]
import pandas as pd
import glob

path = '/content/drive/MyDrive/Semester/8th Semester/CSE463/22301268_NowrinSanjana_Lab1/Task3/Dataset 2'
files = glob.glob(path + '/*.jpg')
```

```
# imgs_jpeg = ["im1.jpg", "im2.jpg", "im3.jpg", "im4.jpg", "im5.jpg"]
img_l = []
for imgs in files:
    img = cv2.imread(imgs)
    img_l.append(img)

    width = int(img.shape[1] * 1.5)
    height = int(img.shape[0] * 1.5)
    dim = (width, height)
    resized_image = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
    # cv2_imshow(resized_image)
    # plt.show()
```

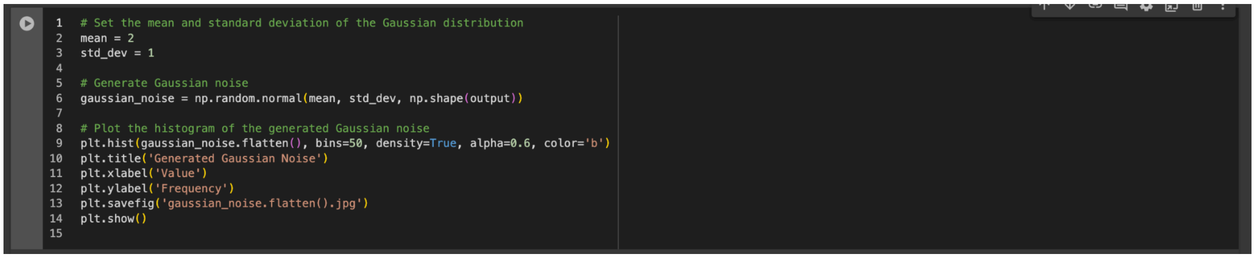


```
# Blend of 5 images
width = 225
height = 225
dim = (width, height)
a = cv2.resize(img_l[0], dim, interpolation = cv2.INTER_AREA)
b = cv2.resize(img_l[1], dim, interpolation = cv2.INTER_AREA)
c = cv2.resize(img_l[2], dim, interpolation = cv2.INTER_AREA)
d = cv2.resize(img_l[3], dim, interpolation = cv2.INTER_AREA)
e = cv2.resize(img_l[4], dim, interpolation = cv2.INTER_AREA)
result = 0.30*a+0.20*b+0.25*c+0.25*d
result = result.astype('uint8')

#Gausian noise

# Generate Gaussian noise with the same size as the image
noise = np.random.randn(*result.shape) * 15

# Add the noise to the image
output = result + noise

# Display the original and noisy images using matplotlib
plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)
plt.title('Blend of 5 Images')
plt.imshow(cv2.cvtColor(result, cv2.COLOR_BGR2RGB))
plt.axis('off')

plt.subplot(1, 2, 2)
plt.title('Noisy Image')
plt.imshow(cv2.cvtColor(output.astype(np.uint8), cv2.COLOR_BGR2RGB))
plt.axis('off')
plt.savefig('output.png')
plt.show()
```

Task3(Output) blend of 5 images and gaussian noise in the blended image:

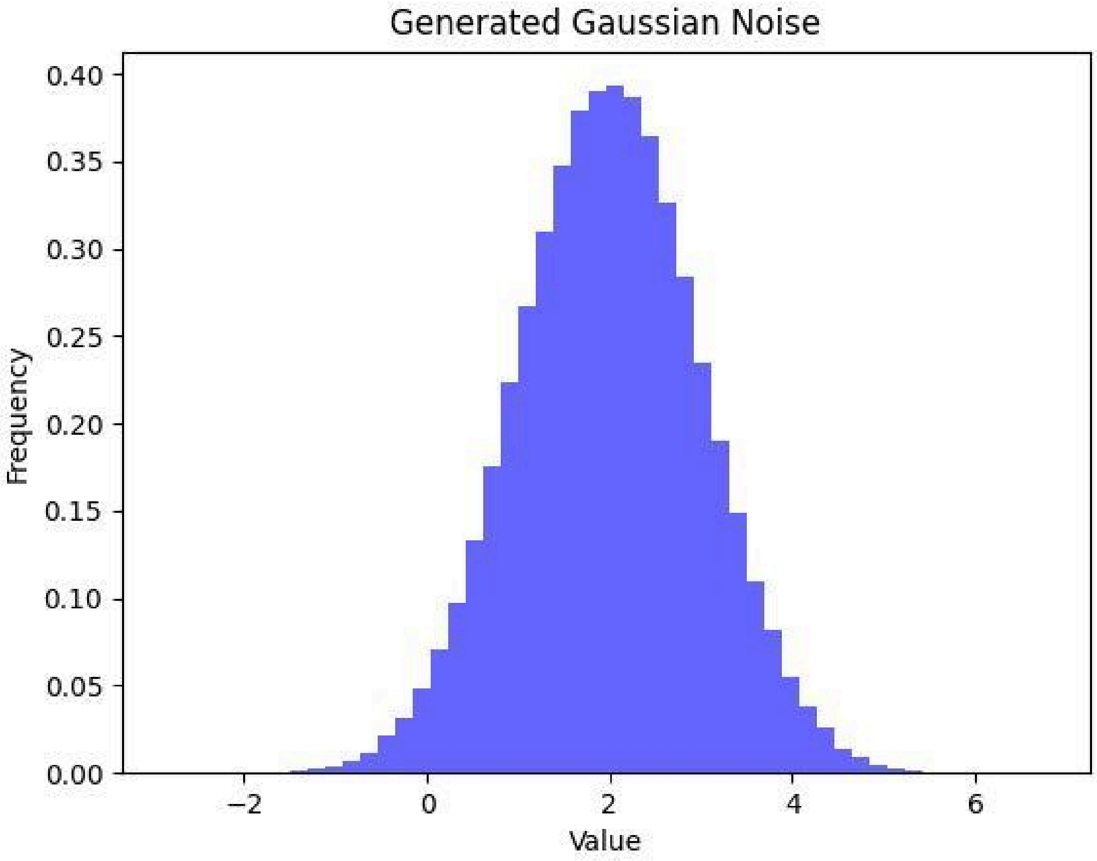


Task3(Code) Histogram plot:



```python
# Set the mean and standard deviation of the Gaussian distribution
mean = 2
std_dev = 1

# Generate Gaussian noise
gaussian_noise = np.random.normal(mean, std_dev, np.shape(output))

# Plot the histogram of the generated Gaussian noise
plt.hist(gaussian_noise.flatten(), bins=50, density=True, alpha=0.6, color='b')
plt.title('Generated Gaussian Noise')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.savefig('gaussian_noise.flatten().jpg')
plt.show()
```

Task3(output) Histogram plot:


Generated Gaussian Noise
Frequency
Value