1. **Dataset create and visualize: Dogs**

**Dataset can be found here:** [dataset\_task02](https://drive.google.com/drive/folders/1Mn-4wtsXXTqTjSsf87pAtwIIuJZ2TsZd?usp=drive_link)

**Code for visualization:**

plt.figure(figsize=(15, 6))

for i in range(min(10, len(dog\_images))):

img = dog\_images[i]

img\_rgb = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

plt.subplot(2, 5, i + 1)

plt.imshow(img\_rgb)

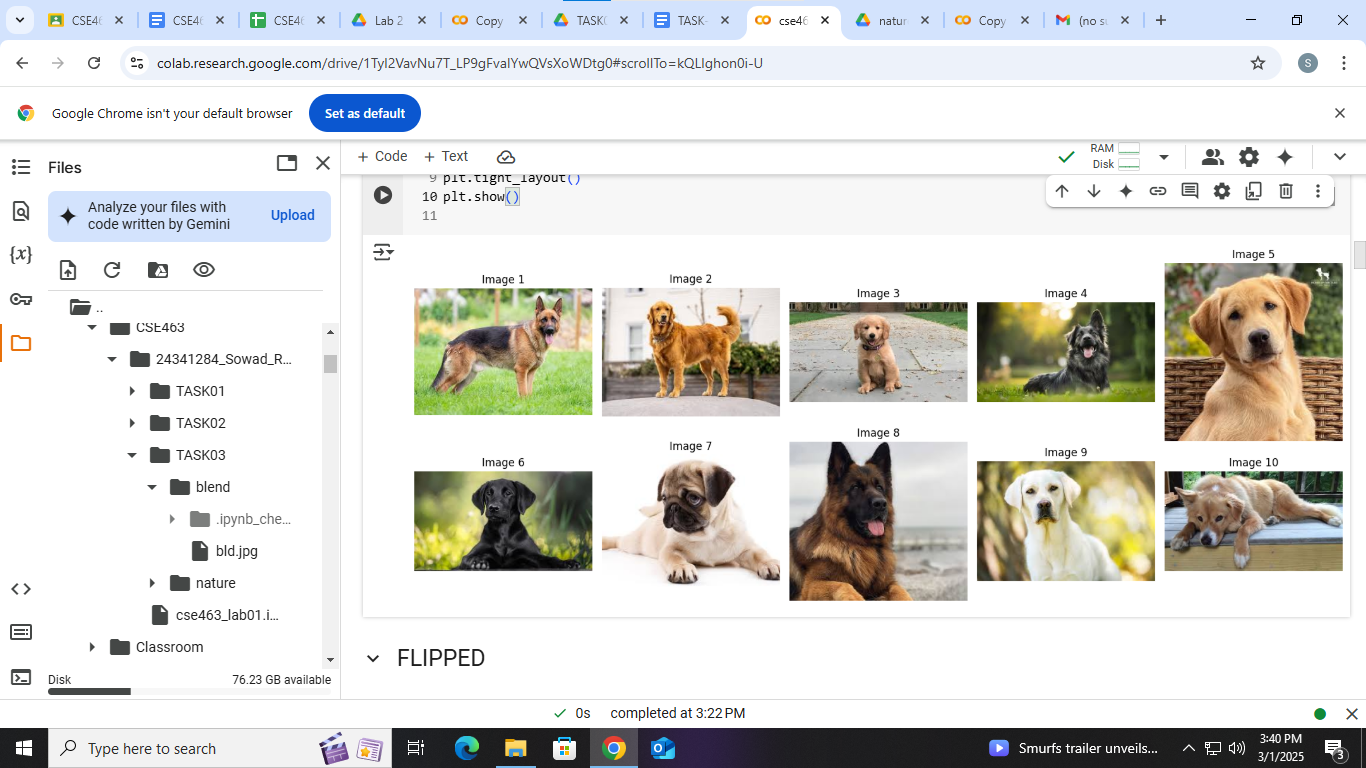
plt.axis('off')

plt.title(f'Image {i + 1}')

plt.tight\_layout()

plt.show()

**OUTPUT:**



1. Performed 5 different transformations:
2. **Flipped:**

**I applied a flip augmentation technique to increase the dataset size. When models will be trained on this dataset the model generalization will increase as in real world a dog might be in vertical or horizontal position.**

**Flipped Image can be found here:**[**flip**](https://drive.google.com/drive/folders/1-aXflerfasyxGg98TdvfvnPEggCUtjzk?usp=drive_link)

**Code:**

import cv2

import os

from google.colab.patches import cv2\_imshow

image\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/dataset\_task02'

flip\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/flip'

os.makedirs(flip\_folder, exist\_ok=True)

images = os.listdir(image\_folder)

flipped\_images = []

for file in images:

if file.endswith(('.jpg')):

img\_path = os.path.join(image\_folder, file)

img = cv2.imread(img\_path)

flipped\_image = cv2.flip(img, 0)

flipped\_images.append(flipped\_image)

flipped\_save\_path = os.path.join(flip\_folder, f"flipped\_{file}")

cv2.imwrite(flipped\_save\_path, flipped\_image)

plt.figure(figsize=(15, 6))

for i in range(min(10, len(flipped\_images))):

plt.subplot(2, 5, i + 1)

plt.imshow(cv2.cvtColor(flipped\_images[i], cv2.COLOR\_BGR2RGB))

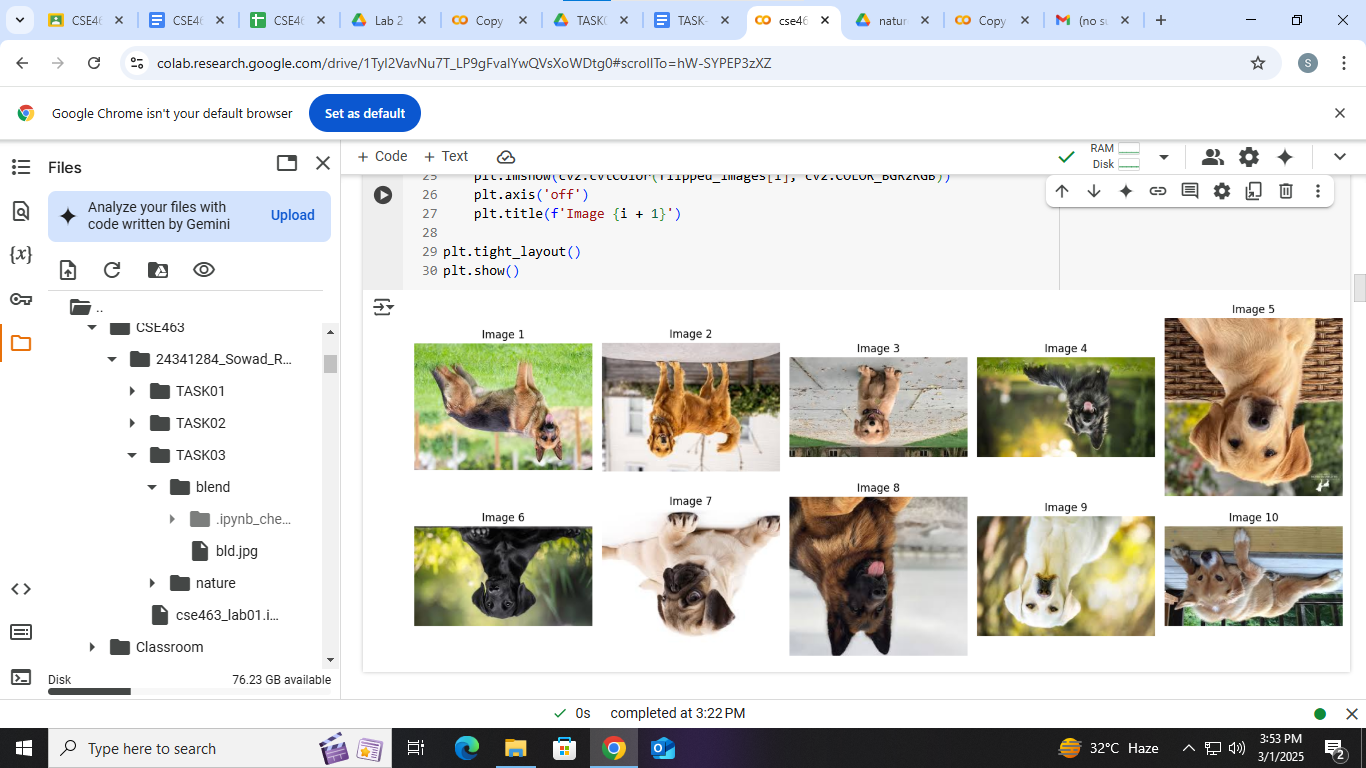
plt.axis('off')

plt.title(f'Image {i + 1}')

plt.tight\_layout()

plt.show()

**OUTPUT:**

****

1. **Cropped:**

**I applied a crop augmentation technique to increase the dataset variability. This augmentation technique helps a model to understand the object localization. Suppose only the face of the dog is given.**

**Cropped Image can be found here:**[**crop**](https://drive.google.com/drive/folders/10DAZ3eHX8mdKWo9vEELYSwjb3F4Opexg?usp=drive_link)

**Code:**

import cv2

import os

import matplotlib.pyplot as plt

image\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/dataset\_task02'

crop\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/crop'

os.makedirs(crop\_folder, exist\_ok=True)

images = os.listdir(image\_folder)

cropped\_images = []

for image\_file in images:

if image\_file.endswith(('.jpg')):

img\_path = os.path.join(image\_folder, image\_file)

img = cv2.imread(img\_path)

height, width = img.shape[:2]

print(height,width)

cropped\_image = img[50:height-10, 50:width-10]

cropped\_images.append(cropped\_image)

cropped\_save\_path = os.path.join(crop\_folder, f"cropped\_{image\_file}")

cv2.imwrite(cropped\_save\_path, cropped\_image)

plt.figure(figsize=(15, 6))

for i in range(min(10, len(cropped\_images))):

plt.subplot(2, 5, i + 1)

plt.imshow(cv2.cvtColor(cropped\_images[i], cv2.COLOR\_BGR2RGB))

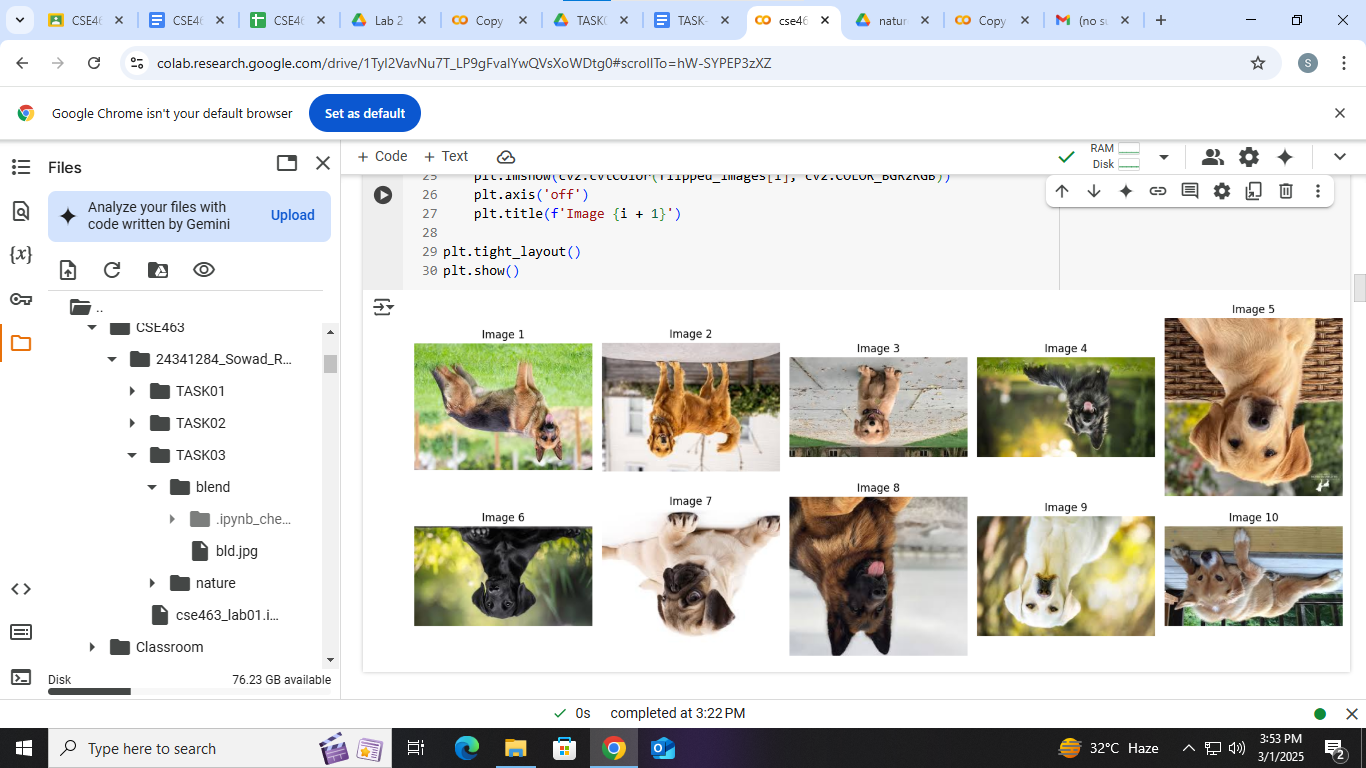
plt.axis('off')

plt.title(f'Image {i + 1}')

plt.tight\_layout()

plt.show()

**OUTPUT:**

****

1. **Rotated:**

**I applied a rotation augmentation technique so that the model can generalize. In real life a dog can appear in many angles so I applied rotation techniques so that the model can recognize a dog coming from different angles.**

**Rotated Image can be found here:**[**rotated**](https://drive.google.com/drive/folders/10hr2T5eUHK7O5XBPd7vO1lcEWFuiXCfq?usp=drive_link)

**CODE:**

import cv2

import os

import matplotlib.pyplot as plt

image\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/dataset\_task02'

crop\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/rotated'

os.makedirs(crop\_folder, exist\_ok=True)

images = os.listdir(image\_folder)

rotated\_images\_save=[]

for image\_file in images:

if image\_file.endswith(('.jpg')):

img\_path = os.path.join(image\_folder, image\_file)

img = cv2.imread(img\_path)

print(f'Original image shape: {img.shape}')

rows, cols = img.shape[:2]

M = cv2.getRotationMatrix2D((cols / 2, rows / 2), -90, 0.5)

rotated\_image = cv2.warpAffine(img, M, (cols, rows))

rotated\_images\_save.append(rotated\_image)

rotated\_save\_path = os.path.join(crop\_folder, f"rotated\_{image\_file}")

cv2.imwrite(rotated\_save\_path, rotated\_image)

plt.figure(figsize=(15, 6))

for i in range(min(10, len(rotated\_images\_save))):

plt.subplot(2, 5, i + 1)

plt.imshow(cv2.cvtColor(rotated\_images\_save[i], cv2.COLOR\_BGR2RGB))

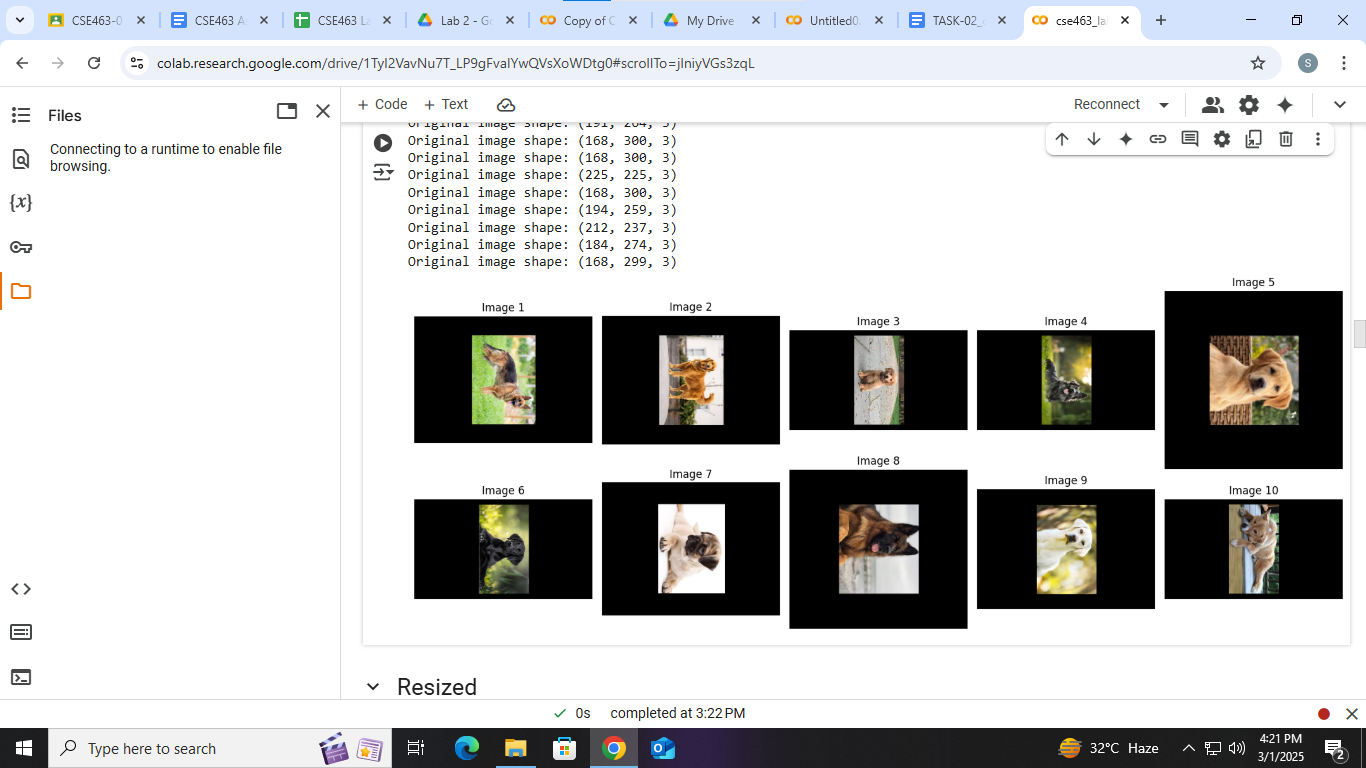
plt.axis('off')

plt.title(f'Image {i + 1}')

plt.tight\_layout()

plt.show()

**OUTPUT:**

****

1. **Resized:**

**Because datasize size will increase and variety. The model can generalize more.**

**Resized images can be found here:**[**resized**](https://drive.google.com/drive/folders/11J7qAg3bkrohkoQFjflJj9U-c_p7Z26y?usp=drive_link)

**CODE:** import cv2

import os

import matplotlib.pyplot as plt

image\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/dataset\_task02'

resize\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/resized'

os.makedirs(resize\_folder, exist\_ok=True)

images = os.listdir(image\_folder)

resized\_images = []

for image\_file in images:

if image\_file.endswith(('.jpg')):

img\_path = os.path.join(image\_folder, image\_file)

img = cv2.imread(img\_path)

width = int(img.shape[1] \* 0.5)

height = int(img.shape[0] \* 0.5)

dim = (width, height)

resized\_image = cv2.resize(img, dim, interpolation=cv2.INTER\_CUBIC)

resized\_images.append(resized\_image)

resized\_save\_path = os.path.join(resize\_folder, f"resized\_{image\_file}")

cv2.imwrite(resized\_save\_path, resized\_image)

plt.figure(figsize=(15, 6))

for i, resized\_image in enumerate(resized\_images):

plt.subplot(2, 5, i + 1)

plt.imshow(cv2.cvtColor(resized\_image, cv2.COLOR\_BGR2RGB))

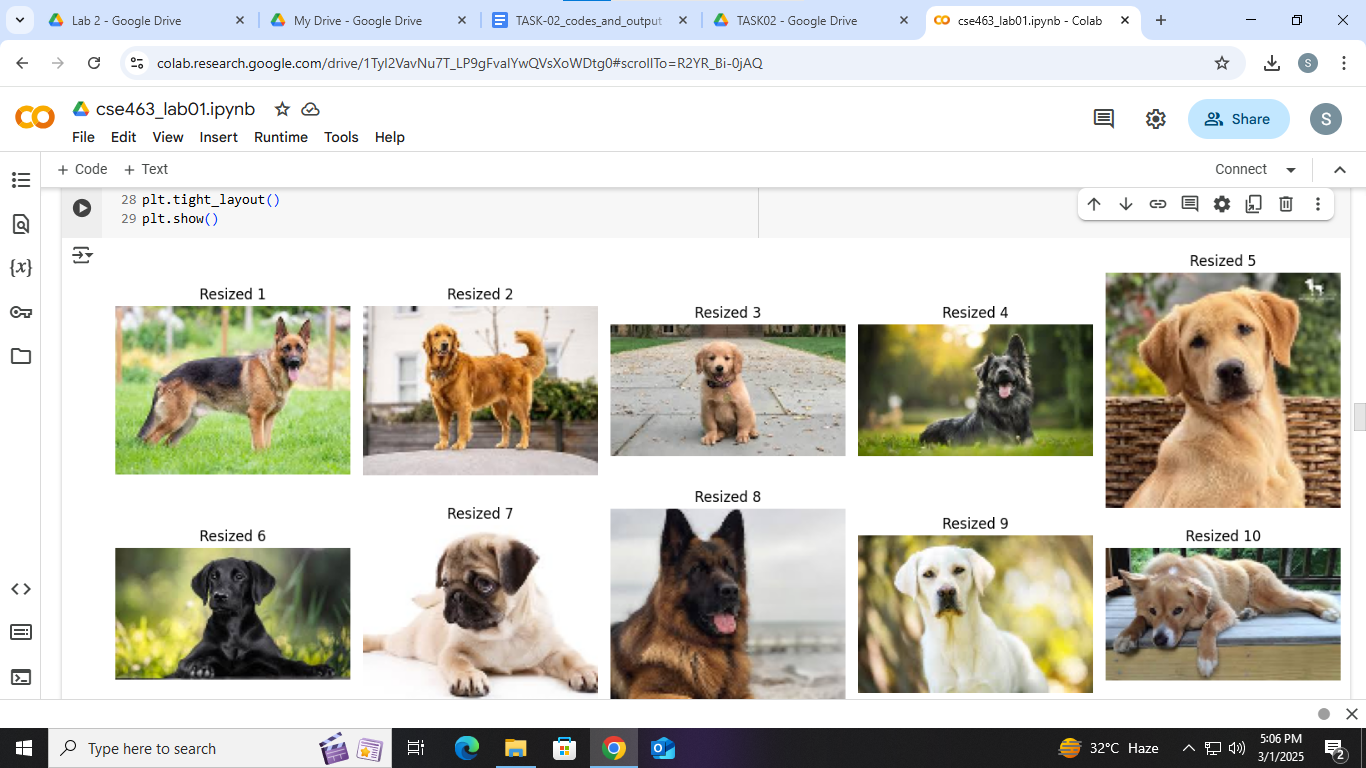
plt.title(f"Resized {i + 1}")

plt.axis('off')

plt.tight\_layout()

plt.show()

**OUTPUT:**

****

1. **Sheared:**

**I applied sheared augmentation technique so that dataset size and variation can be increased as in real life a dog might be seen from different angles.**

**Sheared images can be found here:**[**sheared**](https://drive.google.com/drive/folders/11oIkU_ekTbb3JIKXw0M1OSKzbTlC33v-?usp=drive_link)

**CODE:**

import cv2

import os

import numpy as np

import matplotlib.pyplot as plt

image\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/dataset\_task02'

sheared\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/sheared'

os.makedirs(sheared\_folder, exist\_ok=True)

images = os.listdir(image\_folder)

sheared\_images = []

for image\_file in images:

if image\_file.endswith(('.jpg')):

img\_path = os.path.join(image\_folder, image\_file)

img = cv2.imread(img\_path)

height, width = img.shape[:2]

shear\_x = 0.5

shear\_y = 0.0

shear\_matrix = np.float32([

[1, shear\_x, 0],

[shear\_y, 1, 0]

])

sheared\_image = cv2.warpAffine(img, shear\_matrix, (width + int(shear\_x \* height), height))

sheared\_images.append(sheared\_image)

sheared\_save\_path = os.path.join(sheared\_folder, f"sheared\_{image\_file}")

cv2.imwrite(sheared\_save\_path, sheared\_image)

plt.figure(figsize=(15, 6))

for i, sheared\_image in enumerate(sheared\_images):

plt.subplot(2, 5, i + 1)

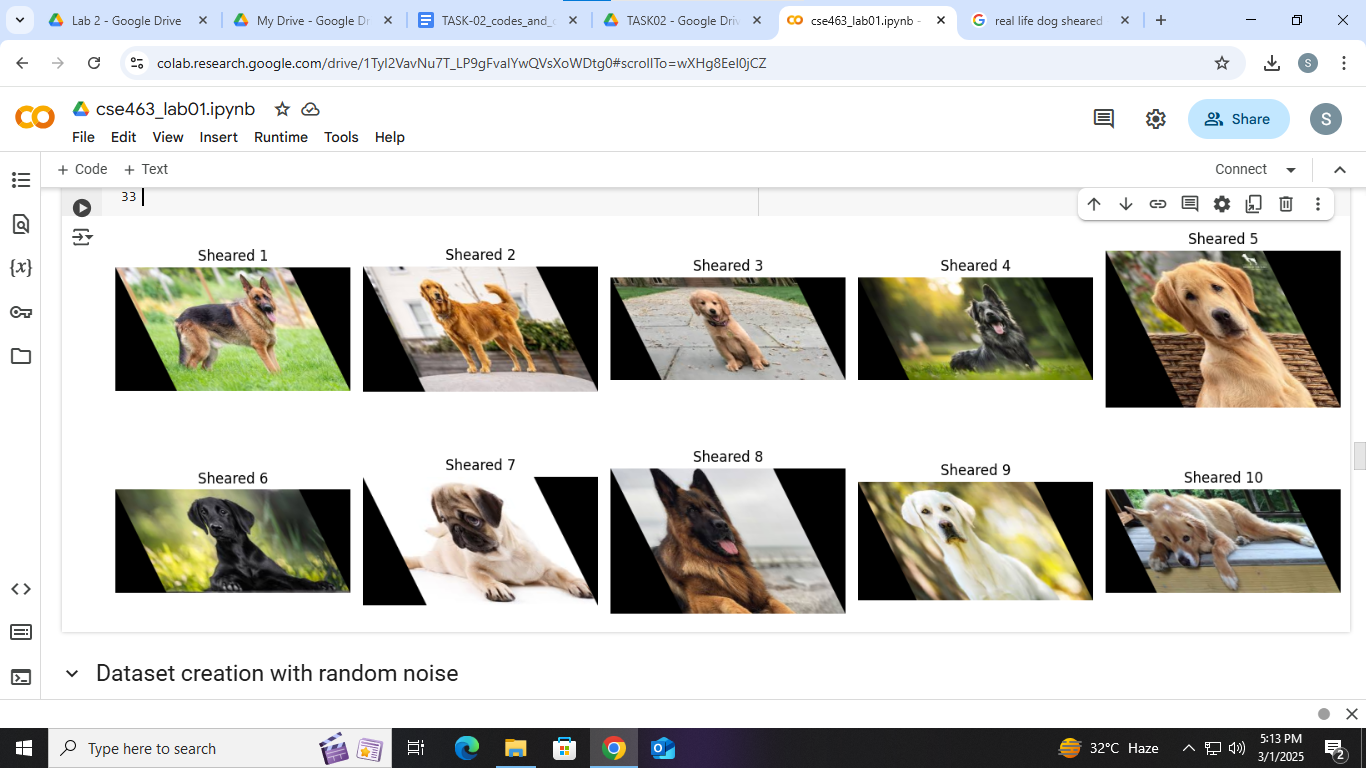
plt.imshow(cv2.cvtColor(sheared\_image, cv2.COLOR\_BGR2RGB))

plt.title(f"Sheared {i + 1}")

plt.axis('off')

plt.tight\_layout()

plt.show()

**OUTPUT:  
**

**3. Dataset creation with random noise:  
Original dataset(before) can be found here:**[**dataset\_task02**](https://drive.google.com/drive/folders/1Mn-4wtsXXTqTjSsf87pAtwIIuJZ2TsZd?usp=drive_link)

**After Applying Random Noise dataset can be found here:**[**noisy\_images\_dataset**](https://drive.google.com/drive/folders/12hBhAnmRwWodQmLKwdYVWJd226mX23rR?usp=drive_link)

**CODE:**

import cv2

import os

import numpy as np

import matplotlib.pyplot as plt

image\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/dataset\_task02'

noisy\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/noisy\_images\_dataset'

os.makedirs(noisy\_folder, exist\_ok=True)

images = os.listdir(image\_folder)

prob = 0.10

for image\_file in images:

if image\_file.endswith(('.jpg')):

img\_path = os.path.join(image\_folder, image\_file)

img = cv2.imread(img\_path)

row, col, ch = img.shape

noisy\_image = np.copy(img)

num\_salt = np.ceil(prob \* img.size \* 0.5).astype(int)

num\_pepper = np.ceil(prob \* img.size \* 0.5).astype(int)

salt\_coords = [np.random.randint(0, i - 1, num\_salt) for i in img.shape]

noisy\_image[salt\_coords[0], salt\_coords[1]] = 255

pepper\_coords = [np.random.randint(0, i - 1, num\_pepper) for i in img.shape]

noisy\_image[pepper\_coords[0], pepper\_coords[1]] = 0

noisy\_save\_path = os.path.join(noisy\_folder, f"noisy\_{image\_file}")

cv2.imwrite(noisy\_save\_path, noisy\_image)

plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)

plt.title("Original Image")

plt.imshow(cv2.cvtColor(img, cv2.COLOR\_BGR2RGB))

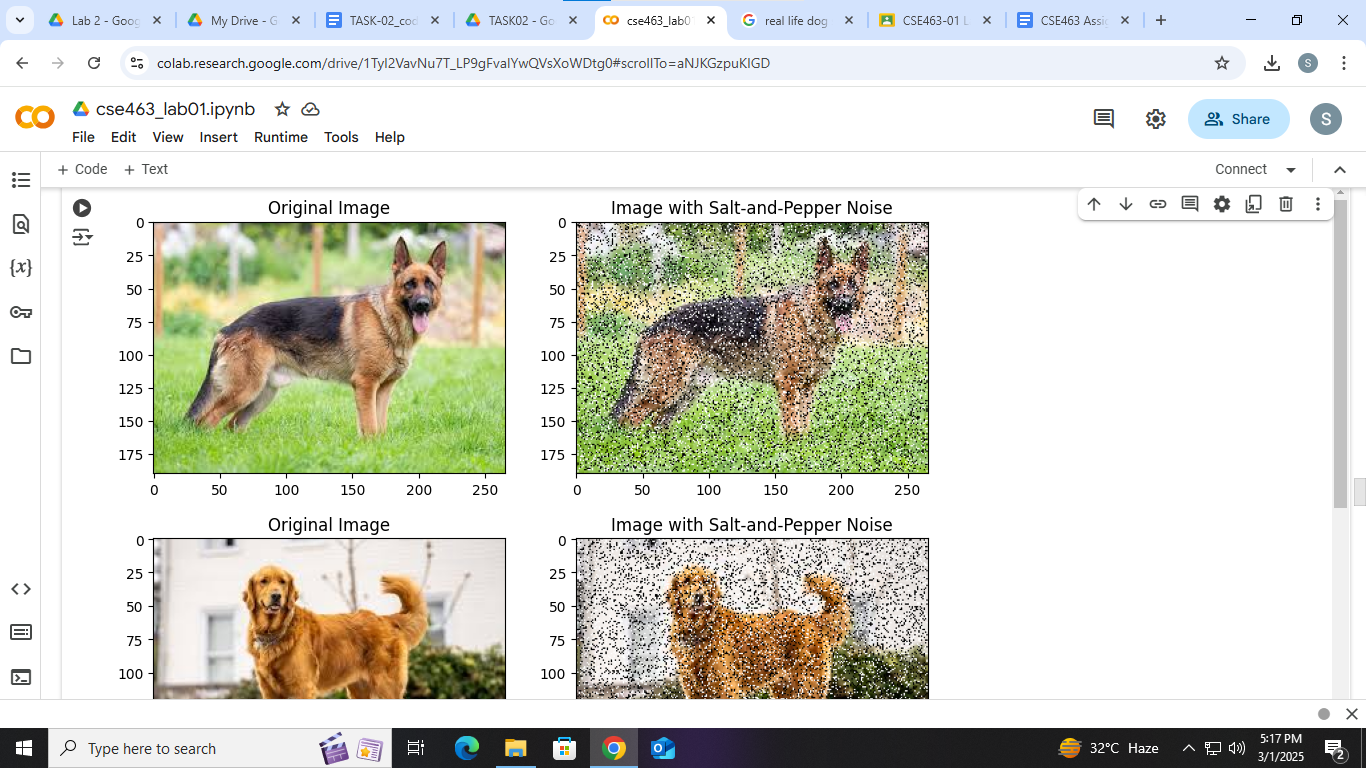
plt.subplot(1, 2, 2)

plt.title("Image with Salt-and-Pepper Noise")

plt.imshow(cv2.cvtColor(noisy\_image, cv2.COLOR\_BGR2RGB))

plt.show()

**OUTPUT:**

****

**4. Histogram**

**Plotted Histogram Images can be found here:**[**histograms**](https://drive.google.com/drive/folders/13rcspRLBxPTq0_Hgs0l8e-cSyRvVR0Bv?usp=drive_link)

**CODE:**import cv2

import os

import numpy as np

import matplotlib.pyplot as plt

noisy\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/noisy\_images\_dataset'

histogram\_folder = '/content/drive/MyDrive/CSE463/24341284\_Sowad\_Rahman\_LAB01/TASK02/histograms/'

os.makedirs(histogram\_folder, exist\_ok=True)

noisy\_images = os.listdir(noisy\_folder)

for noisy\_image\_file in noisy\_images:

if noisy\_image\_file.endswith(('.jpg')):

noisy\_image\_path = os.path.join(noisy\_folder, noisy\_image\_file)

noisy\_image = cv2.imread(noisy\_image\_path, cv2.IMREAD\_COLOR)

flattened\_noise = noisy\_image.flatten()

plt.figure(figsize=(10, 5))

plt.hist(flattened\_noise, bins=256, color='green', alpha=0.7)

plt.title(f"Histogram of Noisy Image: {noisy\_image\_file}")

plt.xlabel("Pixel Intensity")

plt.ylabel("Frequency")

histogram\_save\_path = os.path.join(histogram\_folder, f"histogram\_{noisy\_image\_file}.png")

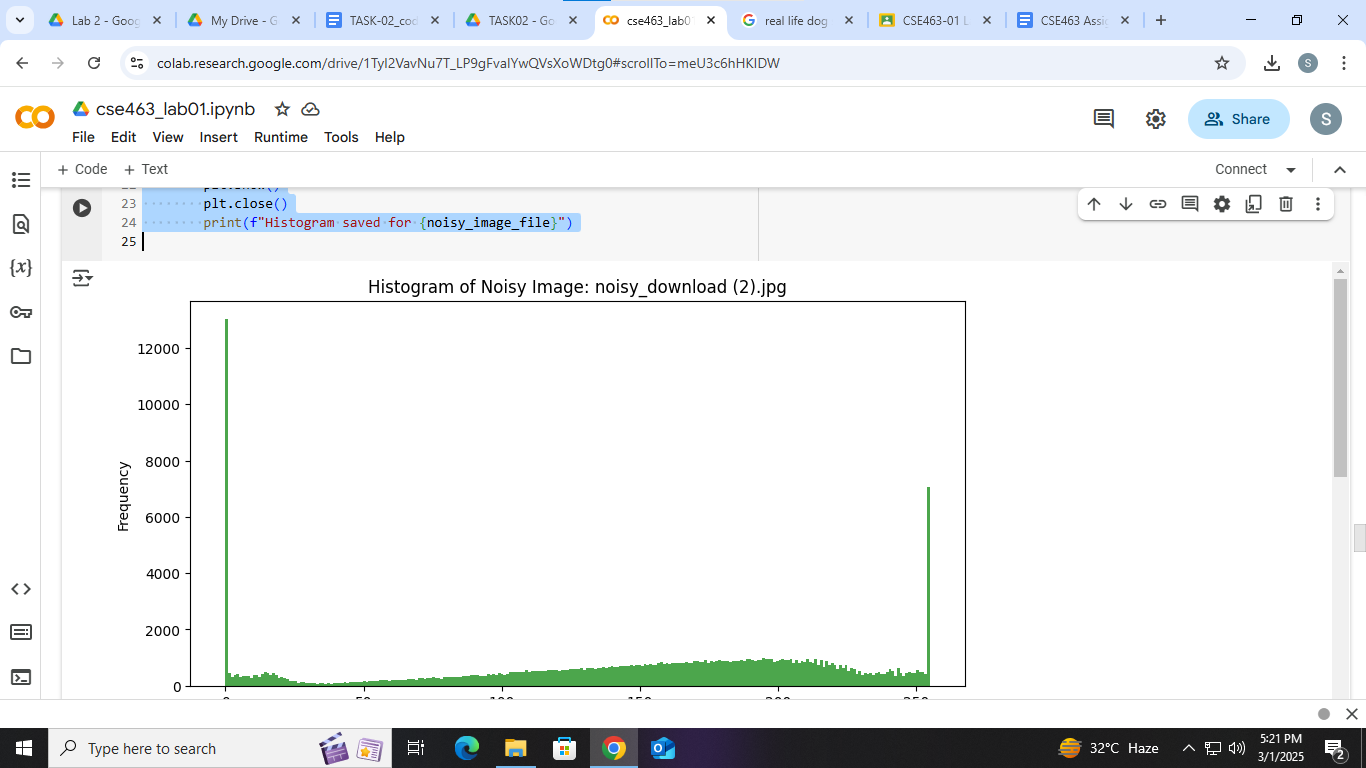
plt.savefig(histogram\_save\_path)

plt.show()

plt.close()

print(f"Histogram saved for {noisy\_image\_file}")

**OUTPUT:**

****