LAMPIRAN A

Proses Augmentasi

```
      # Created a function to perform anti clockwise
rotation def anticlockwise_rotation(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img,
    (224,224)) angle=
    random.randint(0,180) return
    rotate(img, angle)

      # Create a function to perform clockwise
rotation def clockwise_rotation(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img,
    (224,224)) angle=
    random.randint(0,180) return
    rotate(img, -angle)

      # Create a function to flip images up and
down def flip_up_down(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img,
    (224,224)) return
    np.flipud(img)
      # Create a function to give a bright effect to
the image def add_brightness(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img, (224,224))
          img = adjust_gamma(img,
    gamma=0.5,gain=1) return img

      # Create a function to give the image a blur/blur
effect def blur_image(img):
          img = cv2.cvtColor(img, 0)
```

```
      # Create a function to give the image a sheared
effect def sheared(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img, (224,224))
    transform = AffineTransform(shear=0.2)
          shear_image = warp(img, transform,
    mode="wrap") return shear_image

      # Create a function to perform warp
shifts def warp_shift(img):
          img = cv2.cvtColor(img, 0)
          img = cv2.resize(img, (224,224))
          transform =
    AffineTransform(translation=(0,40)) warp_image =
    warp(img, transform, mode="wrap") return
    warp_image
```

```
      # Create a transformation variable that
will hold all the preprocessing processes that
have been made above
Transformations={'rotateanticlockwise':anticlockw
ise_rotat ion,
                      'rotate clockwise':
                       clockwise_rotation, 'warp
                       shift': warp_shift,
                             'blurring image':
                       blur_image,
                             'add
                       brightness' :
                       add_brightness,
                       'flip up down':
                       flip_up_down, 'shear
                       image': sheared
    }

      images_path="Dataset-
fix/Dataset/train/fresh" # Path for the
original image
      augmented_path=" Dataset-
fix/Dataset/train/fresh " # Path to put the
augmented image
      images=[] # To save images that have
been preprocessed from the folder

      # Read image name from folder and add path
into "images" array
for im in os.listdir(images_path):
    images.append(os.path.join(images_path,im))

      # The number of images that will be
added with the results of the augmentation
transformation, the number is adjusted
according to needs
      # Variable to iterate up to a predefined
number of images_to_generate
      ima
ges_to_ge
nerate=60
0 i=1

while
    i<=images_to_ge
    nerate:
    image=random.cho
    ice(images) try:
              original image = io.imread(image)
```

```
      select and call methods

  transformed_image =
            transformations[key](original_image) n = n +
            1

       new_image_path="%s/augmented_image_%s.jpg"
      %(augmented_path, i)

    transformed_image = img_as_ubyte(transformed_image) #
Convert images to unsigned byte format, with values in
[0, 255]
         cv2.imwrite(new_image_path, transformed_image) #
Save the result of the augmentation transformation on the
image into the specified path
              i =i+1
          except ValueError as e:
           print('could not read the',image ,':',e,'hence
skipping it.')
```