1.Image read, write and save operation

image_loader.py

```
import cv2    #openCV use as cv2 in python

#Loads a color image. Any transparency of image will be neglected. It is the default flag.
#this function is used to read the image from location
img1 = cv2.imread('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\boat.jpg',1)
img1 = cv2.resize(img1,(1280,700))#width ,height
print(img1)
cv2.imshow("original",img1)  #It accept two parameters 1)- Name of screen ,2) -  Image
cv2.waitKey(0)  #here parameter inside waitkey handle the life duration of an image
cv2.destroyAllWindows()
```

Output

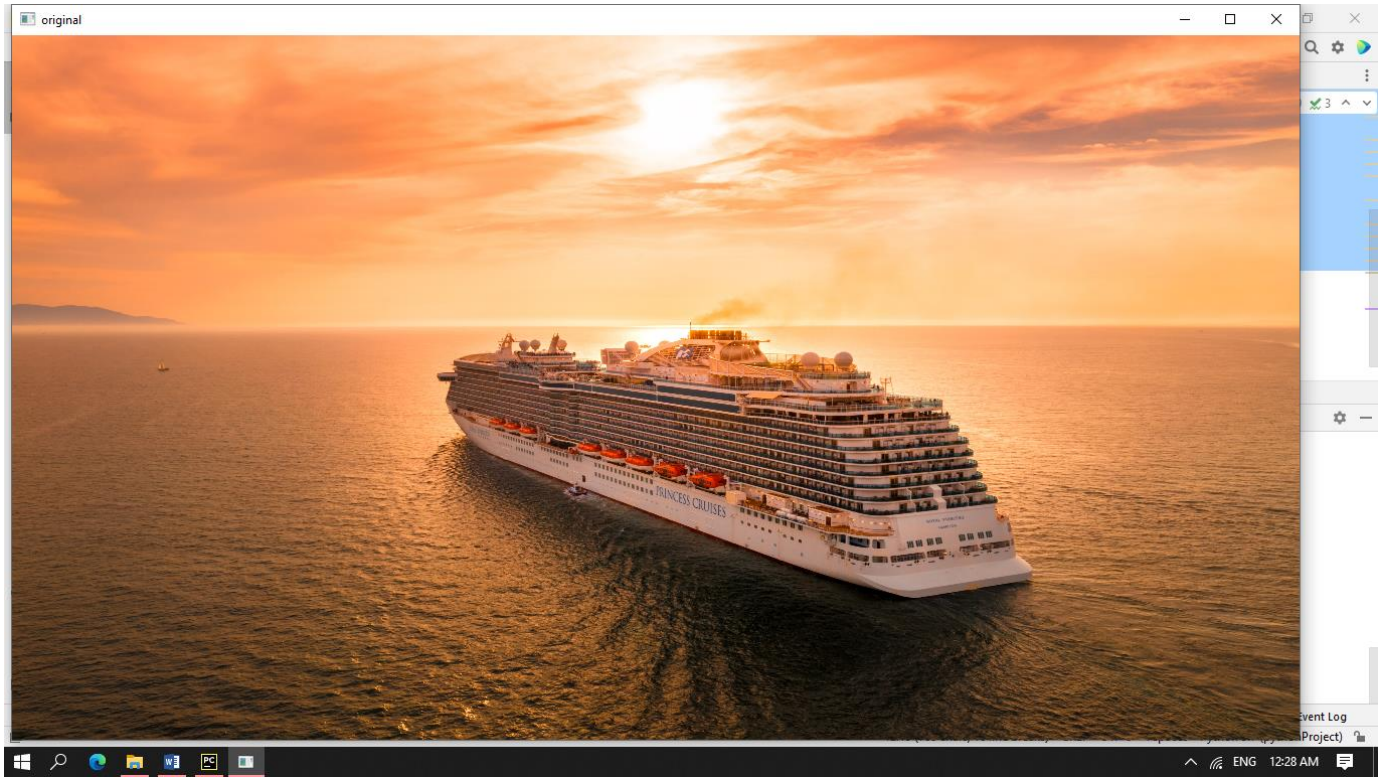
original

grayscale.py

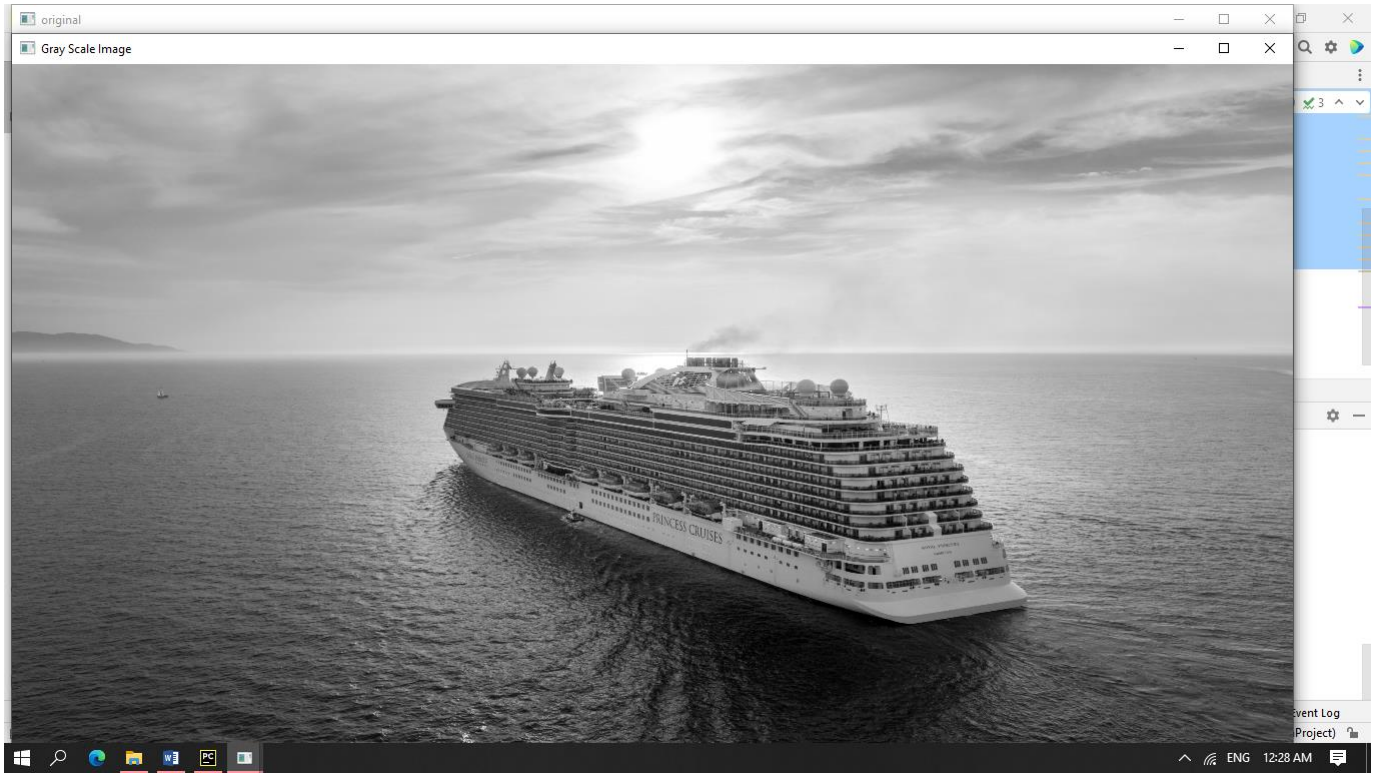
```
import cv2    #openCV use as cv2 in python

#Loads a color image. Any transparency of image will be neglected. It is the default flag.
#this function is used to read the image from location
img1 = cv2.imread('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\boat.jpg',1)
img1 = cv2.resize(img1,(1280,700))#width ,height
print(img1)
cv2.imshow("original",img1)  #It accept two parameters 1)- Name of screen ,2) -  Image

#cv2.IMREAD_GRAYSCALE : Loads image in grayscale mode
img2 = cv2.imread('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\boat.jpg',0)
img2 = cv2.resize(img2,(1280,700))#width ,height
cv2.imshow("Gray Scale Image",img2)
print("Image in gray scale==\n",img2)

cv2.waitKey()
cv2.destroyAllWindows()
```

Output

original



Gray Scale Image



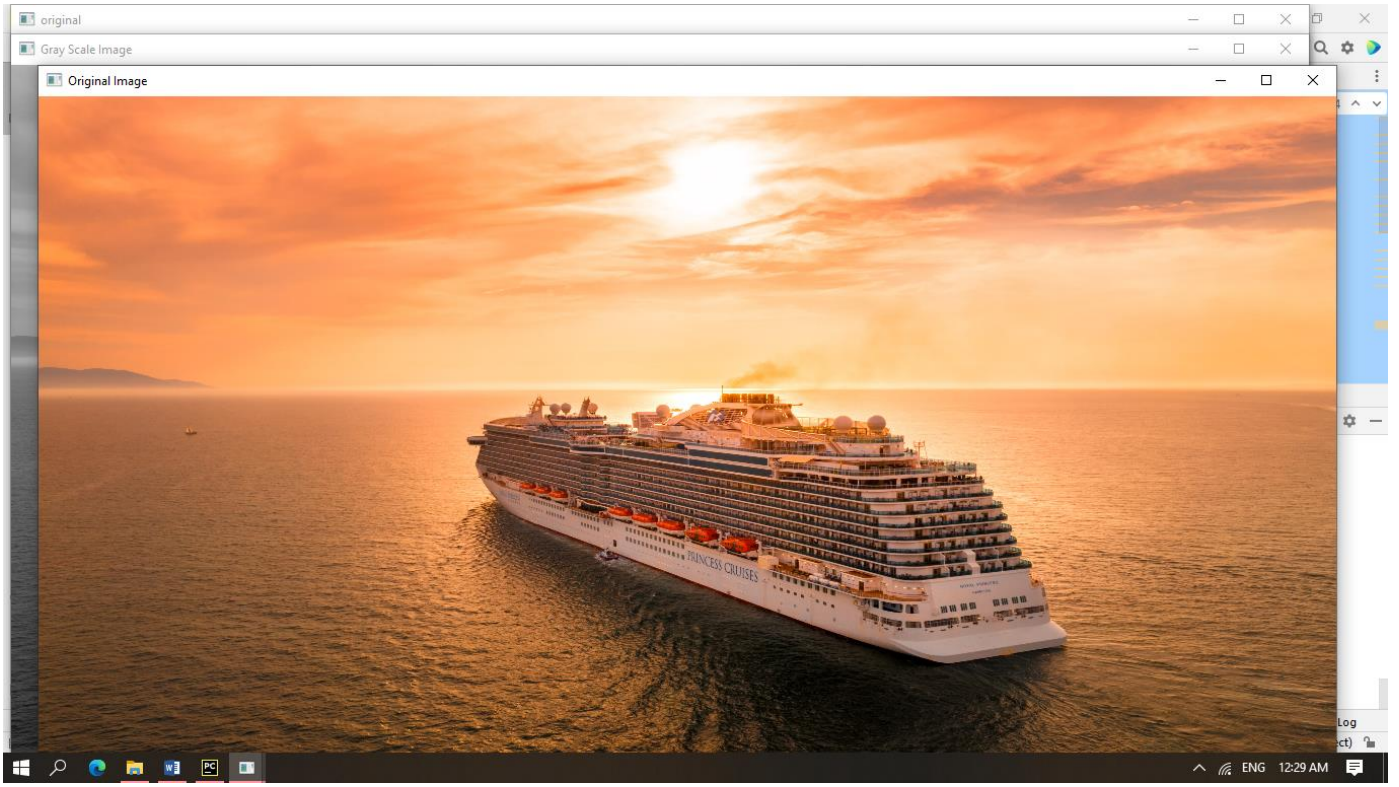
unchanged.py

```
import cv2    #openCV use as cv2 in python

#Loads a color image. Any transparency of image will be neglected. It is the default flag.
#this function is used to read the image from location
img1 = cv2.imread('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\boat.jpg',1)
img1 = cv2.resize(img1,(1280,700))#width ,height
print(img1)
cv2.imshow("original",img1)  #It accept two parameters 1)- Name of screen ,2) -  Image

#cv2.IMREAD_GRAYSCALE : Loads image in grayscale mode
img2 = cv2.imread('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\boat.jpg',0)
img2 = cv2.resize(img2,(1280,700))#width ,height
cv2.imshow("Gray Scale Image",img2)
print("Image in gray scale==\n",img2)
```

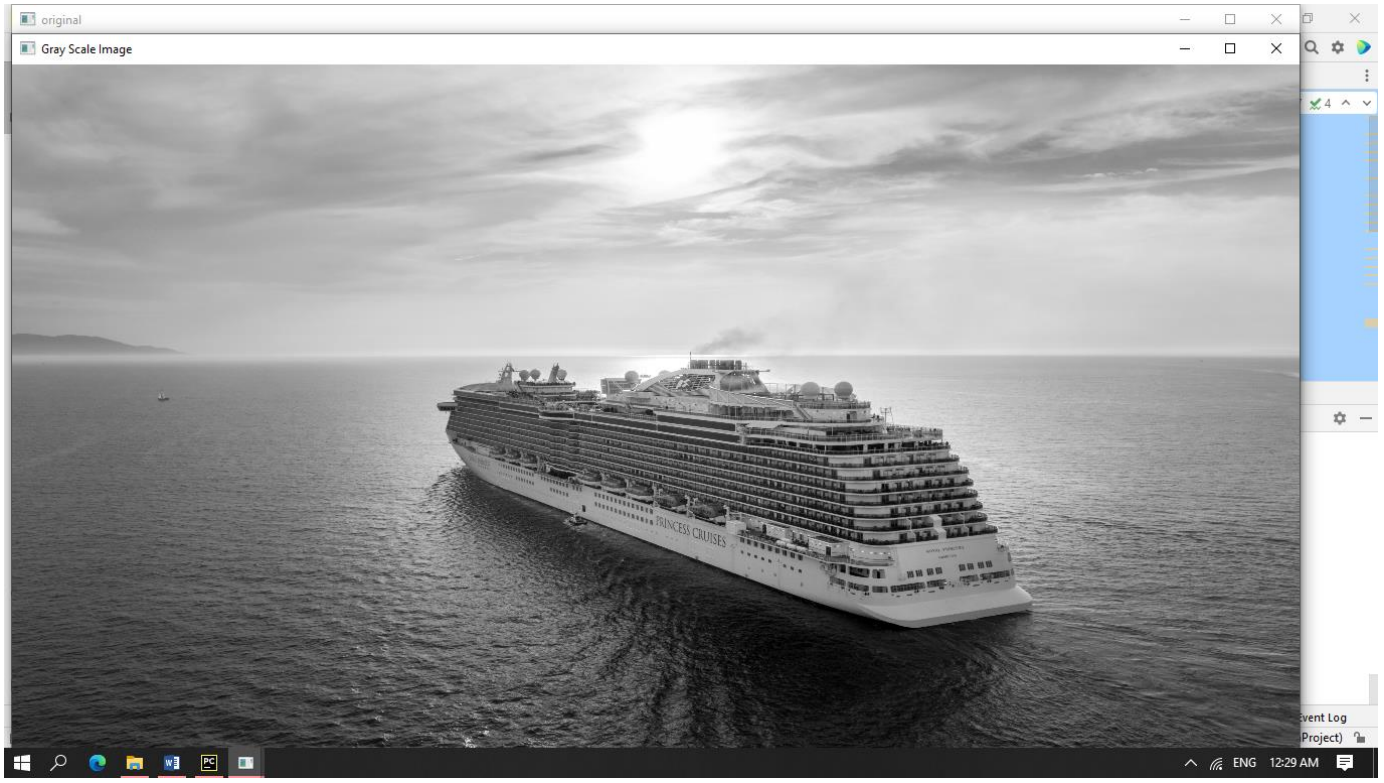
```
#cv2.IMREAD_UNCHANGED : Loads image as such including alpha channel
img3 = cv2.imread('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\boat.jpg',-1)
img3 = cv2.resize(img3,(1280,700))#width ,height
cv2.imshow("Original Image",img3)
print("Image in original value==\n",img3)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output

Original Image



Gray Scale Image



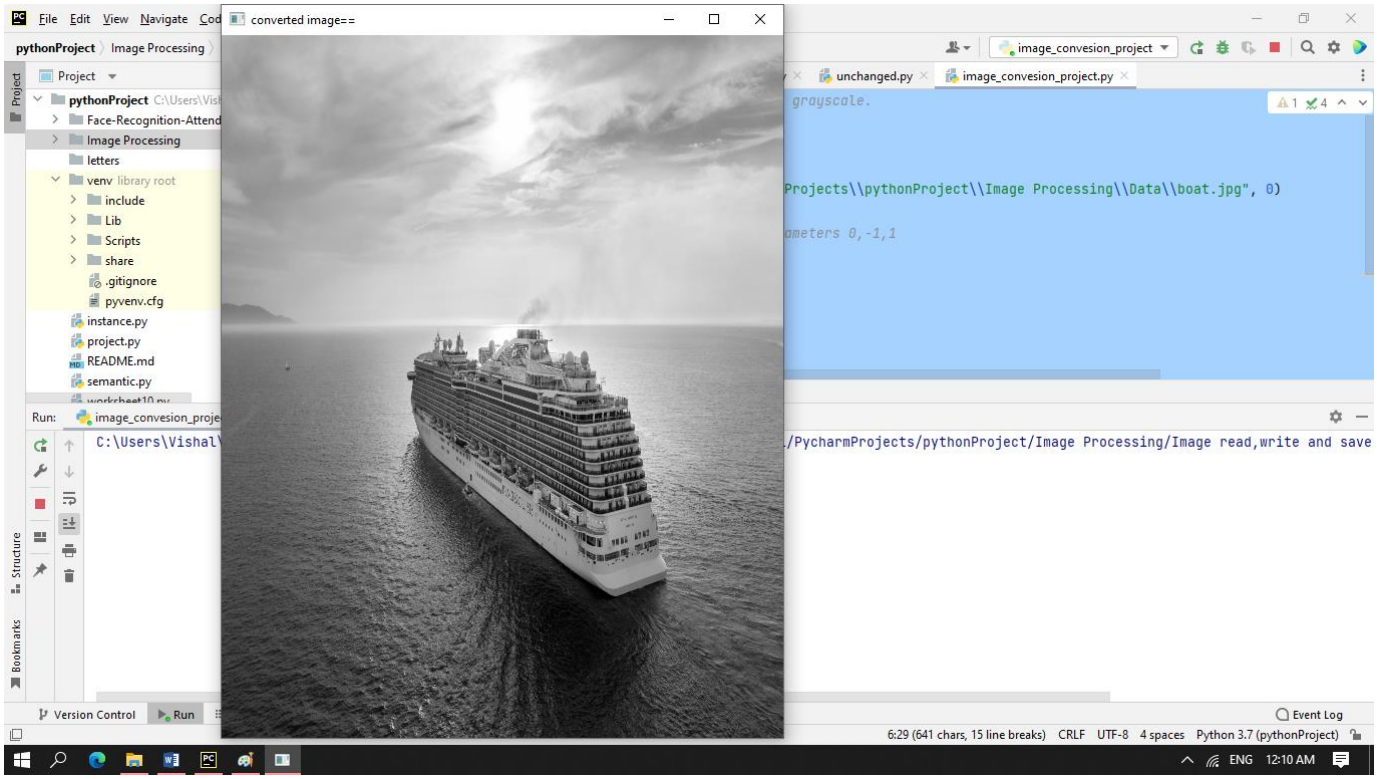
original

image_convesion_project.py

```
# Image conversion project colored image into grayscale.
import cv2
# First process is to read image
# convert image into grayscale
img1 = cv2.imread("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\boat.jpg", 0)
img1 = cv2.resize(img1, (560, 700))
# img1 = cv2.flip(img1, 0)  # it accept 3 parameters 0,-1,1
cv2.imshow("converted image==", img1)
k = cv2.waitKey(0) & 0xFF
if k == ord("q"):
    cv2.destroyAllWindows()

elif k == ord("s"):
    cv2.imwrite("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\ouput.png", img1)  # it accept name of image and data
    cv2.destroyAllWindows()
```

Output

converted image==



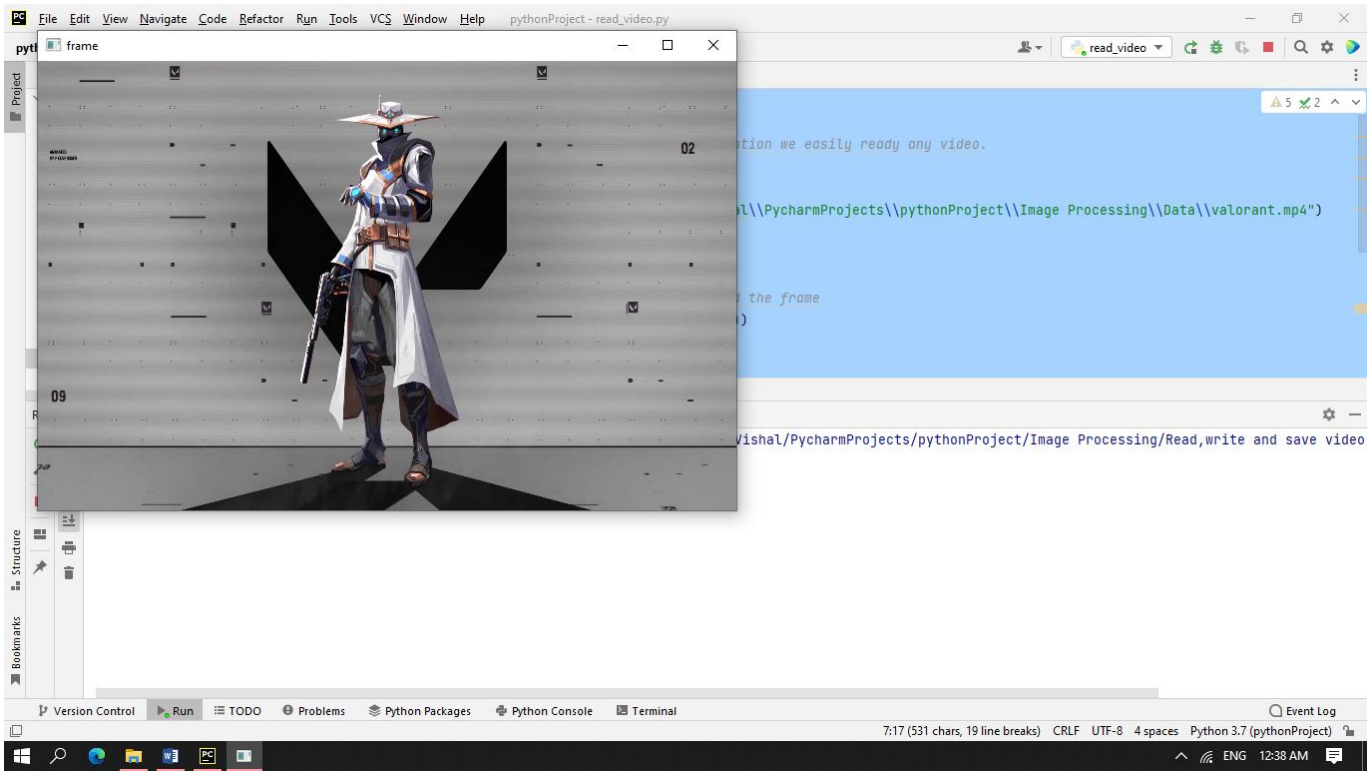
2.Read, write and save video using opencv

read_video.py

```
import cv2

#Here with the help of videoCapture function we easily ready any video.

#Here parameter is a path of any video
cap = cv2.VideoCapture("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\valorant.mp4")
print("cap",cap)

while True:
    ret, frame = cap.read()   #here read the frame
    frame = cv2.resize(frame, (700, 450))
    cv2.imshow('frame', frame)
    k = cv2.waitKey(25)
    if k == ord("q"):
        break
# Release everything if job is finished
cap.release()
cv2.destroyAllWindows()
```

Output

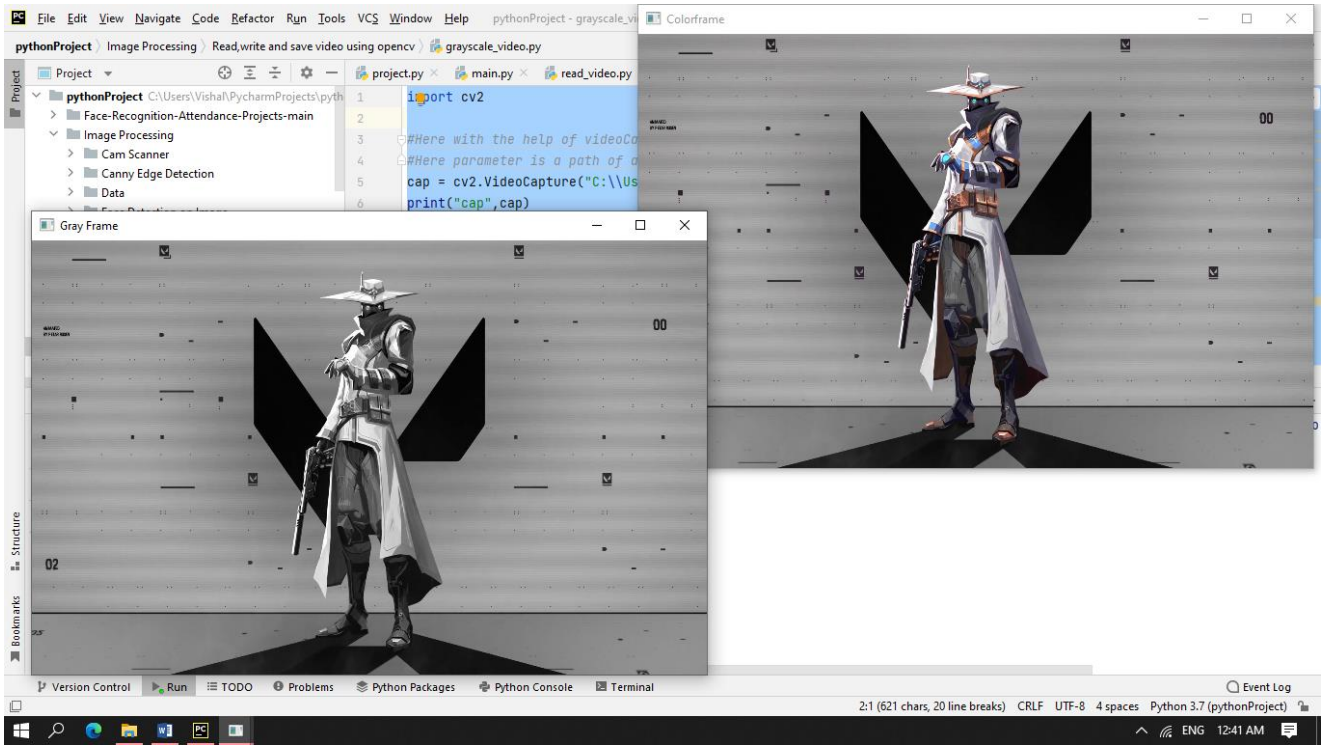
frame



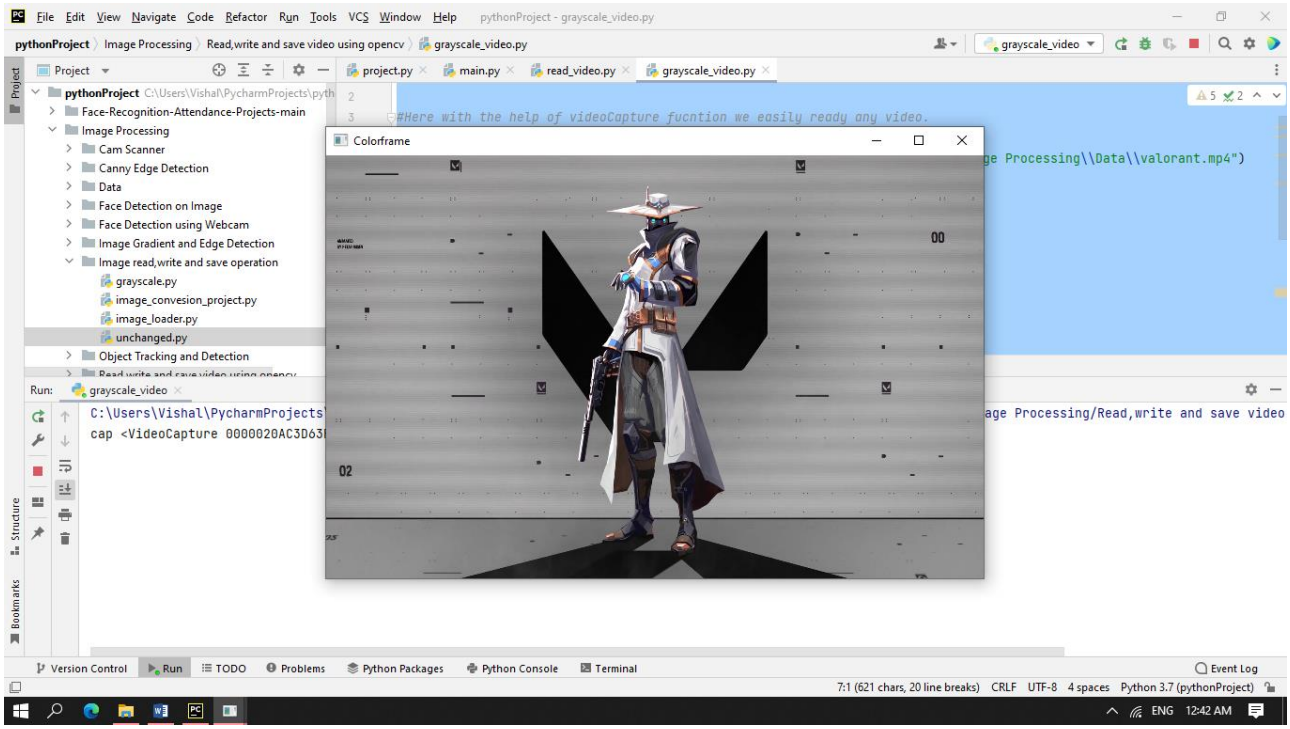
grayscale_video.py

```
import cv2

#Here with the help of videoCapture fucntion we easily ready any video.
#Here parameter is a path of any video
cap = cv2.VideoCapture("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\valorant.mp4")
print("cap",cap)

while True:
    ret, frame = cap.read()   #here read the frame
    frame = cv2.resize(frame, (700, 450))
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    cv2.imshow('Colorframe', frame)
    cv2.imshow("Gray Frame", gray)
    k = cv2.waitKey(25)
    if k == ord("q"):
        break
# Release everything if job is finished
cap.release()
cv2.destroyAllWindows()
```

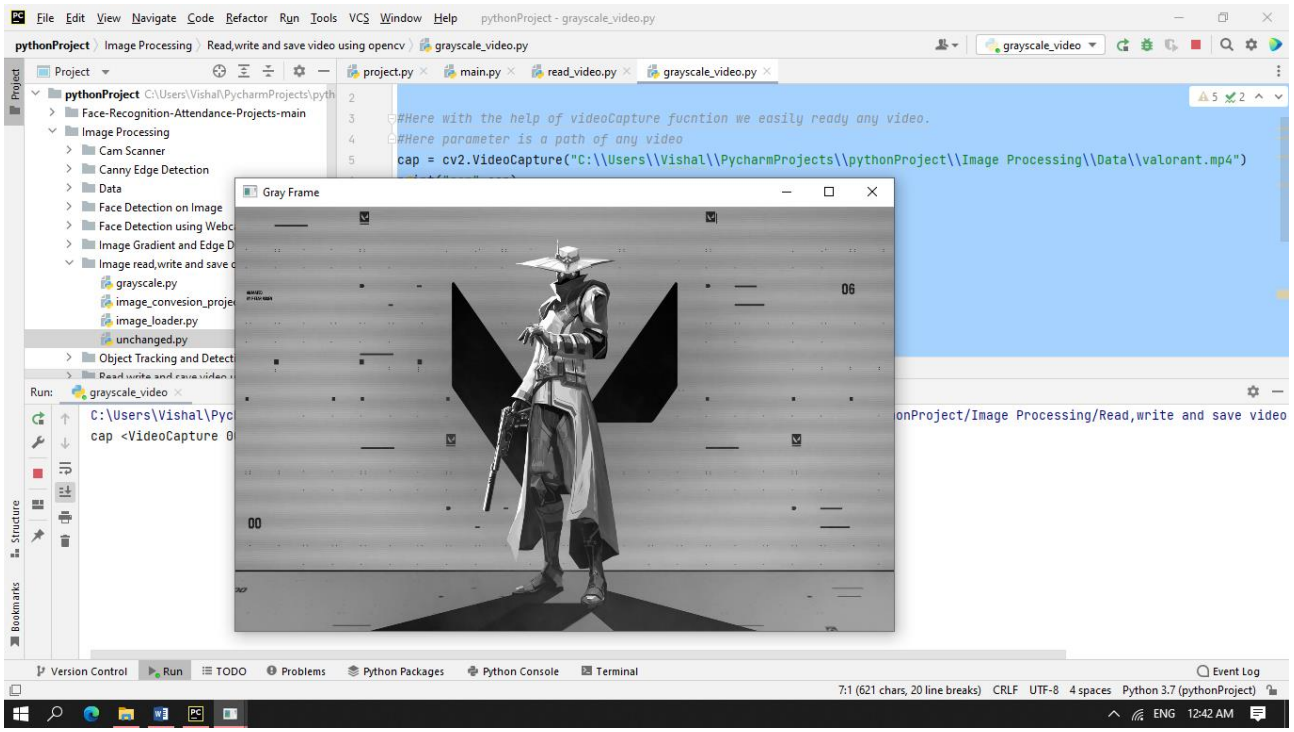
Output



Colorframe



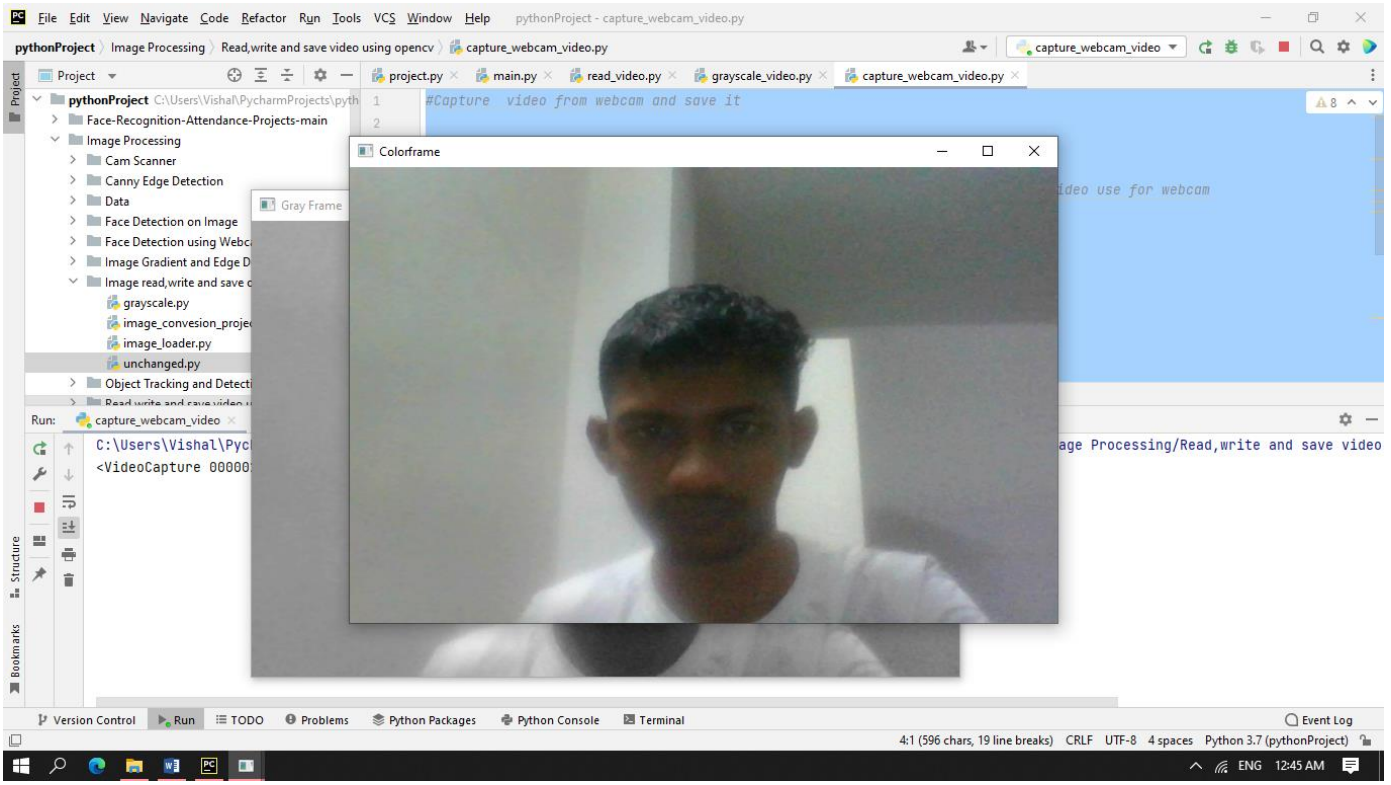
Gray Frame



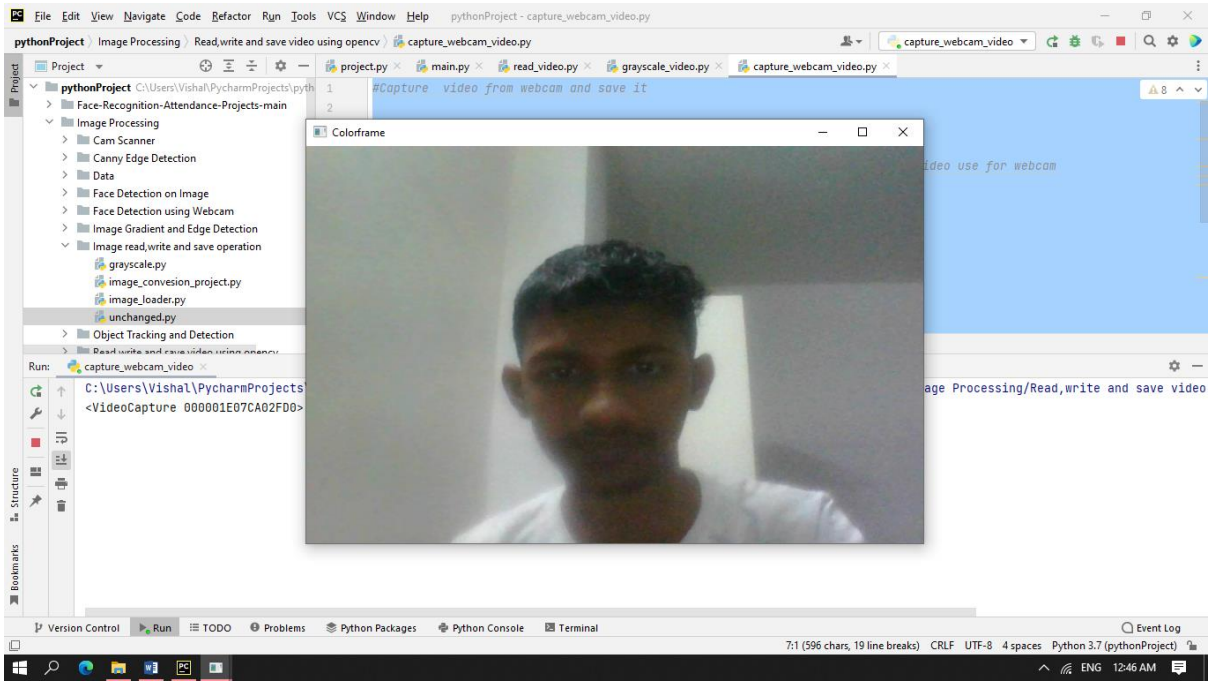
capture_webcam_video.py

```
#Capture  video from webcam and save it

import cv2

cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)   #Here parameter 0 is a path of any video use for webcam
print(cap)

while(cap.isOpened()):
    ret, frame = cap.read()   #here read the frame
    if ret == True:
        frame = cv2.resize(frame, (700, 450))
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        cv2.imshow("Gray Frame", gray)
        cv2.imshow('Colorframe', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):  # press to exit
            break

# Release everything if job is finished
cap.release()
cv2.destroyAllWindows()
```

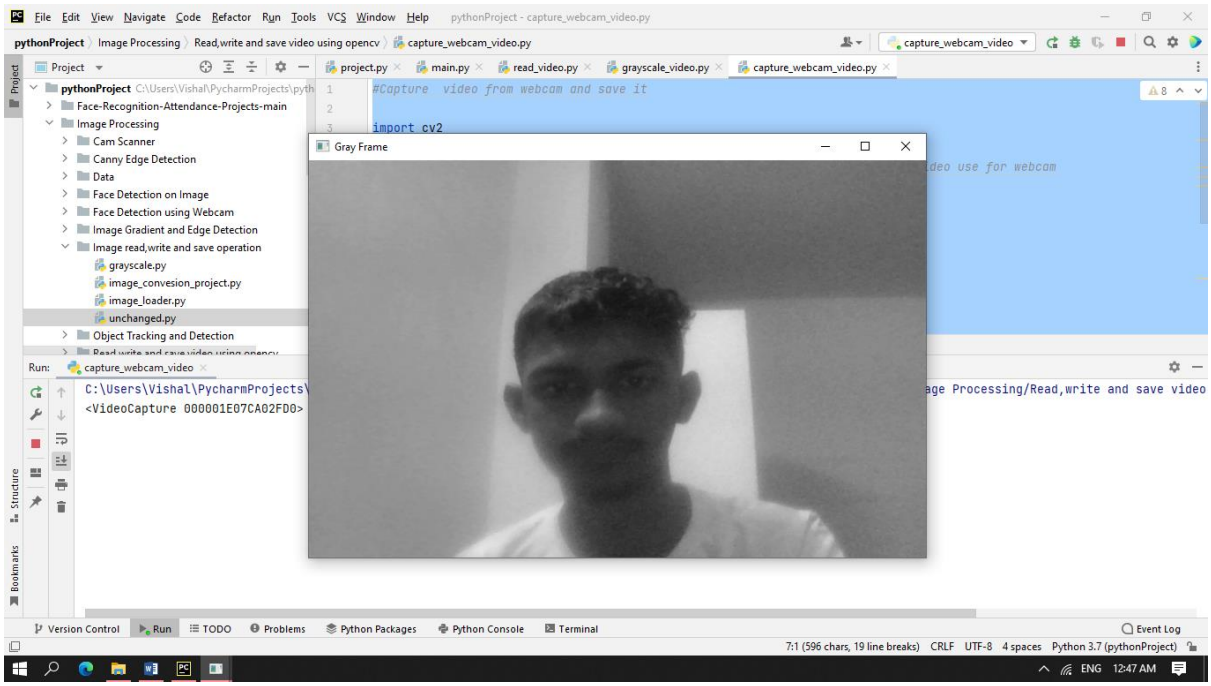
Output



Colorframe



Gray Frame

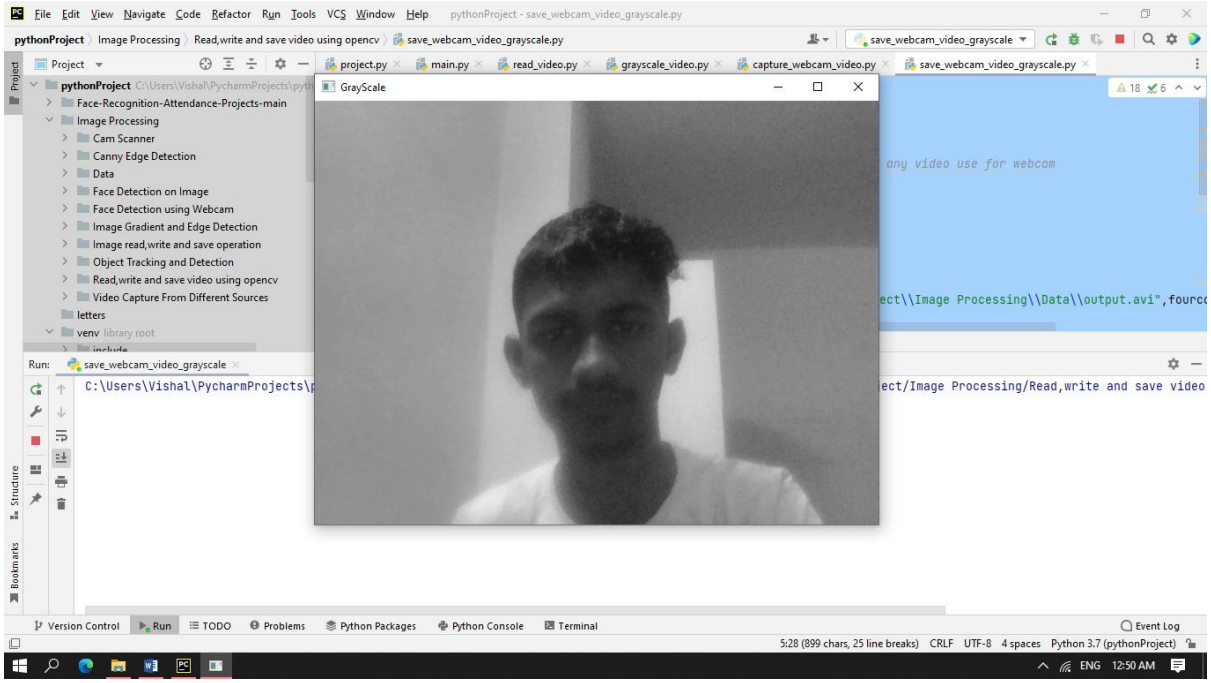


save_webcam_video_grayscale.py

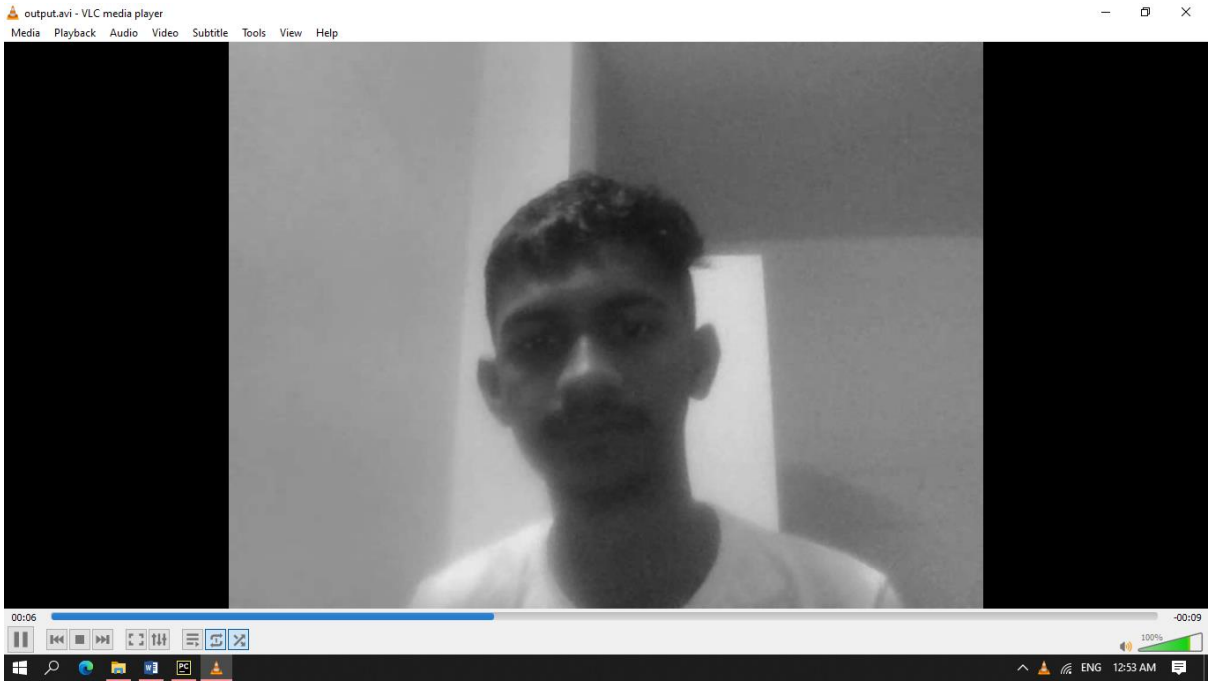
```
#Capture  video from webcam and save it
import cv2
cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)   #Here parameter 0 is a path of any video use for webcam

#it is 4 byte code which is use to specify the video codec
#Various codec --
#DIVX, XVID, MJPG, X264, WMV1, WMV2
fourcc = cv2.VideoWriter_fourcc(*"XVID")  # *"XVID"
#It contain 4 parameter , name, codec,fps,resolution
output = cv2.VideoWriter("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\output.avi",fourcc,20.0,(640,480),0)

while(cap.isOpened()):
    ret, frame = cap.read()   #here read the frame
    if ret == True:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        cv2.imshow('GrayScale', gray)
        output.write(gray)
        if cv2.waitKey(1) & 0xFF == ord('q'):  # press to exit
            break

# Release everything if job is finished
cap.release()
output.release()
cv2.destroyAllWindows()
```

Output

GrayScale



Video file saved with name *ouput.avi*

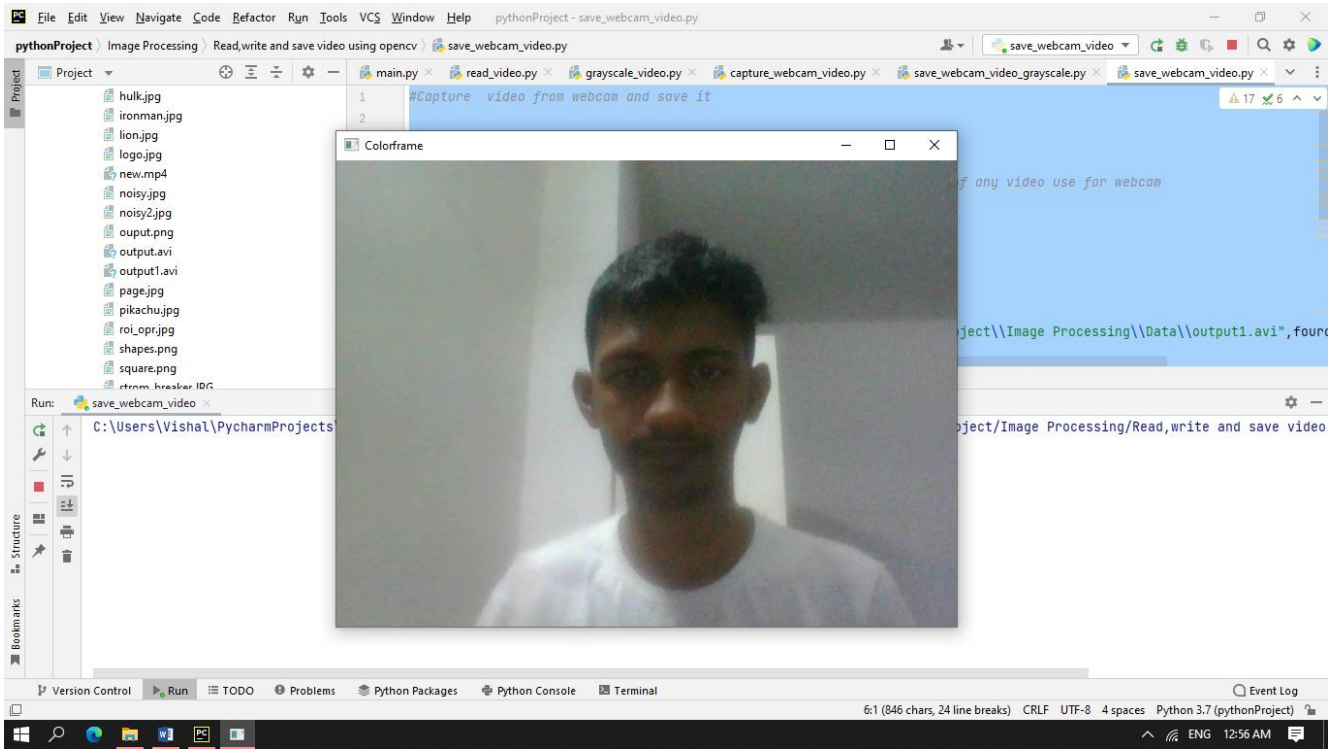


save_webcam_video.py

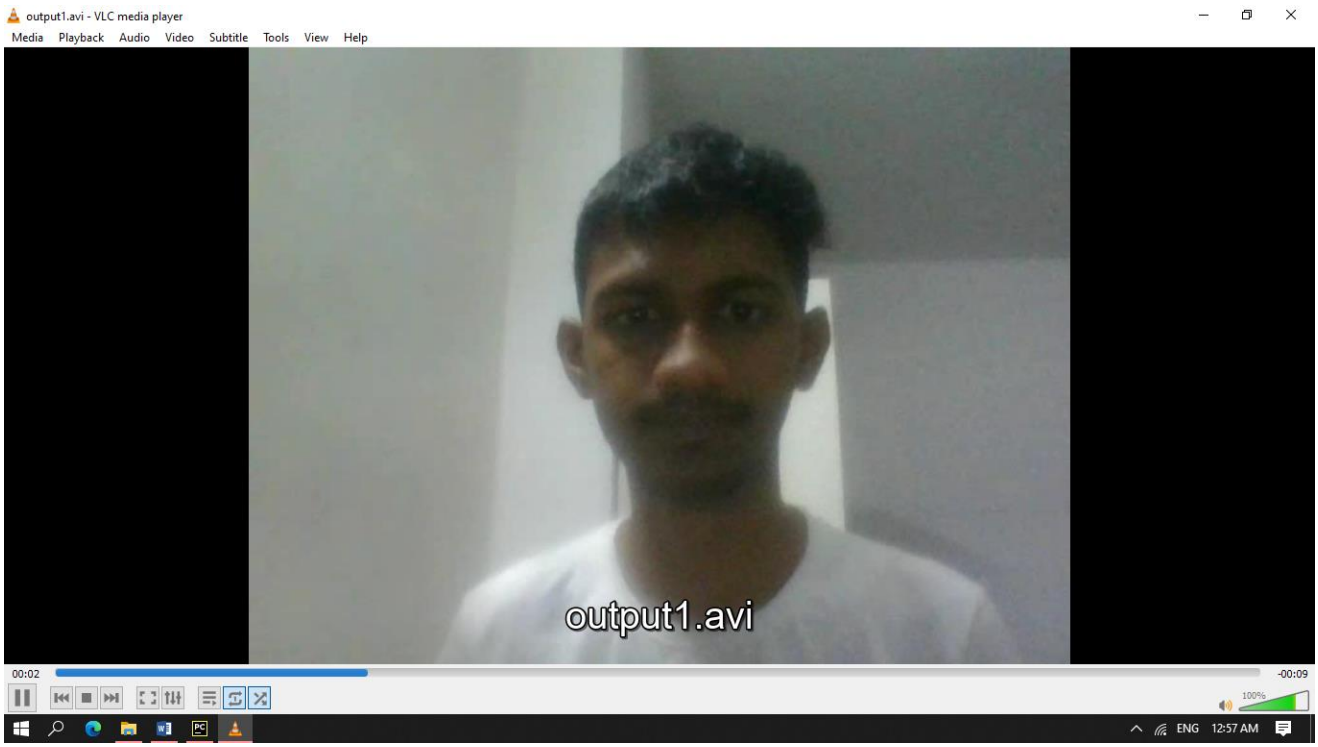
```
#Capture  video from webcam and save it
import cv2
cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)   #Here parameter 0 is a path of any video use for webcam

#it is 4 byte code which is use to specify the video codec
#Various codec --
#DIVX, XVID, MJPG, X264, WMV1, WMV2
fourcc = cv2.VideoWriter_fourcc(*"XVID")  # *"XVID"
#It contain 4 parameter , name, codec,fps,resolution
output = cv2.VideoWriter("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\output1.avi",fourcc,20.0,(640,480))

while(cap.isOpened()):
    ret, frame = cap.read()   #here read the frame
    if ret == True:
        cv2.imshow('Colorframe', frame)
        output.write(frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):  # press to exit
            break

# Release everything if job is finished
cap.release()
output.release()
cv2.destroyAllWindows()
```

Output

Colorframe



Video file saved with name *ouput1.avi*

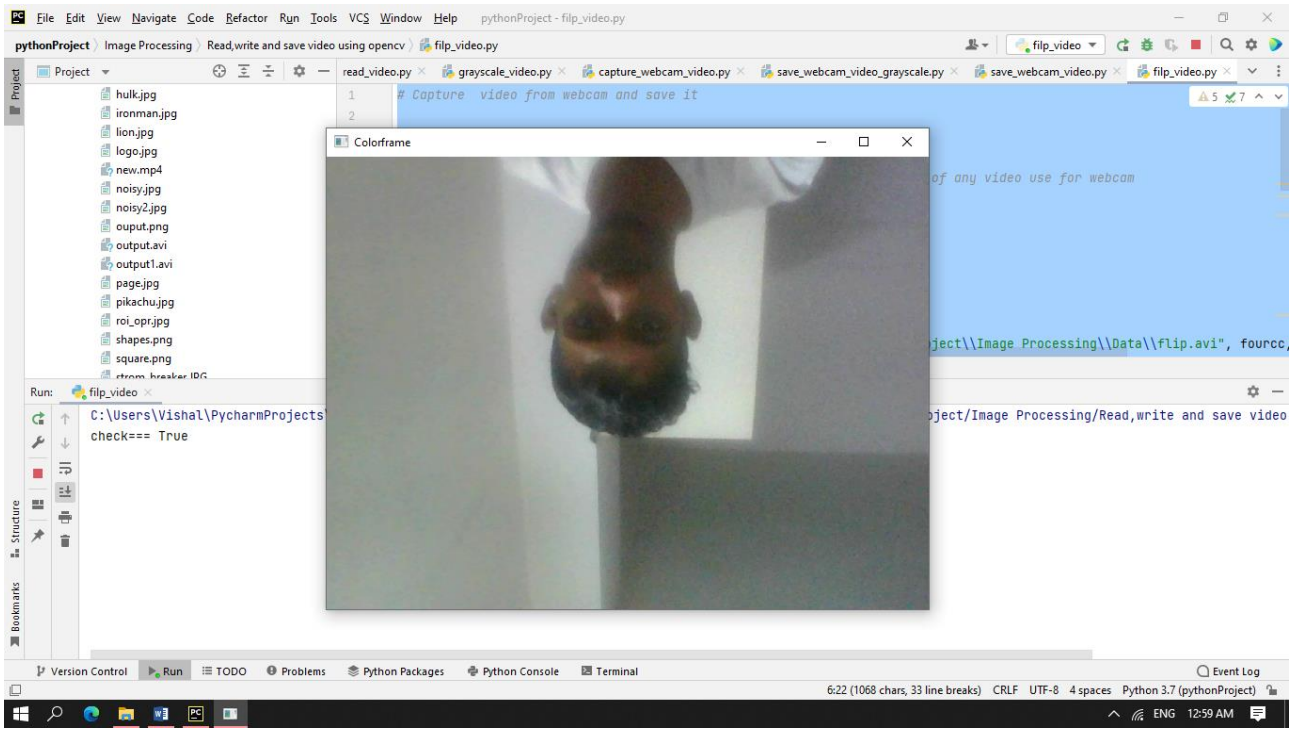


flip_video.py

```
# Capture  video from webcam and save it
import cv2
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)  # Here parameter 0 is a path of any video
use for webcam
print("check===", cap.isOpened())

# it is 4 byte code which is use to specify the video codec
# Various codec --
# DIVX, XVID, MJPG, X264, WMV1, WMV2
fourcc = cv2.VideoWriter_fourcc(*"XVID")  # *"XVID"
# It contain 4 parameter , name, codec,fps,resolution
output = cv2.VideoWriter("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image
Processing\\Data\\flip.avi", fourcc, 20.0, (640, 480))

while (cap.isOpened()):
    ret, frame = cap.read()  # here read the frame

    if ret == True:

        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        # here flip is used to lip the video at recording time
        frame = cv2.flip(frame, 0)
        output.write(frame)

        cv2.imshow('Colorframe', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):  # press to exit
            break
    else:
        break

# Release everything if job is finished
cap.release()
output.release()
cv2.destroyAllWindows()
```

Output

Colorframe



Video file saved with name *flip.avi*

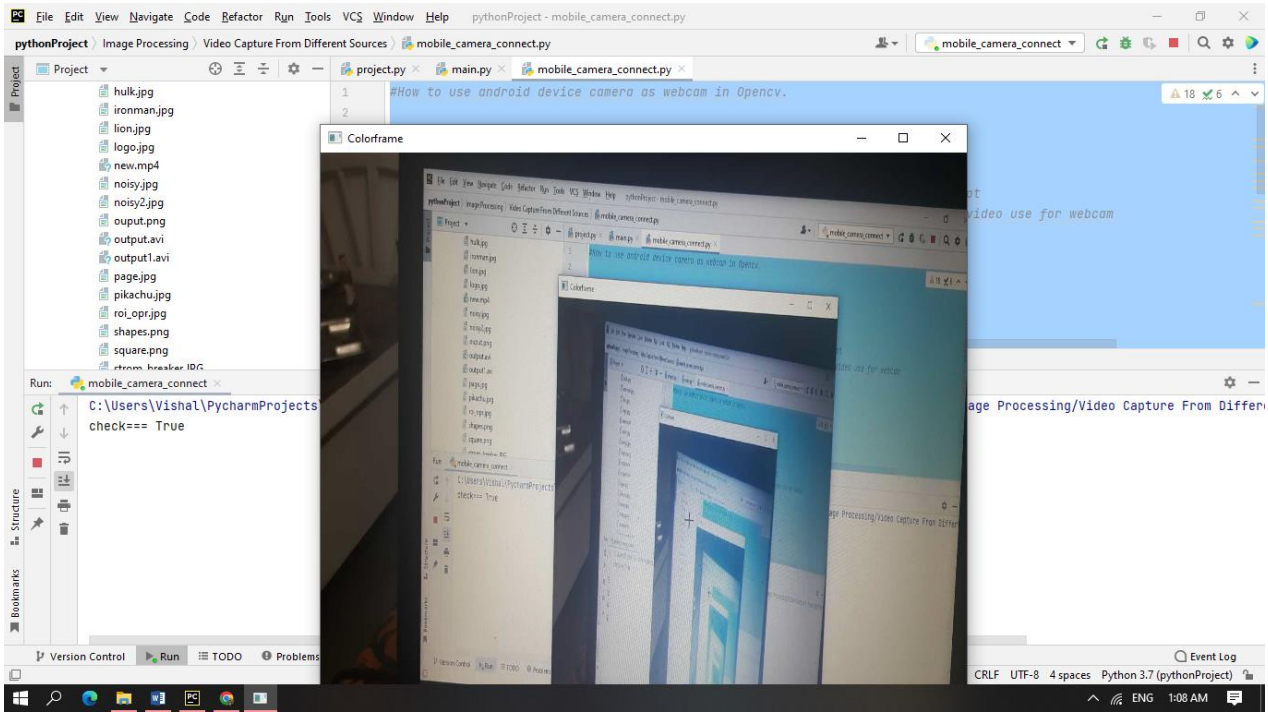
3.Video Capture from Different Sources

mobile_camera_connect.py

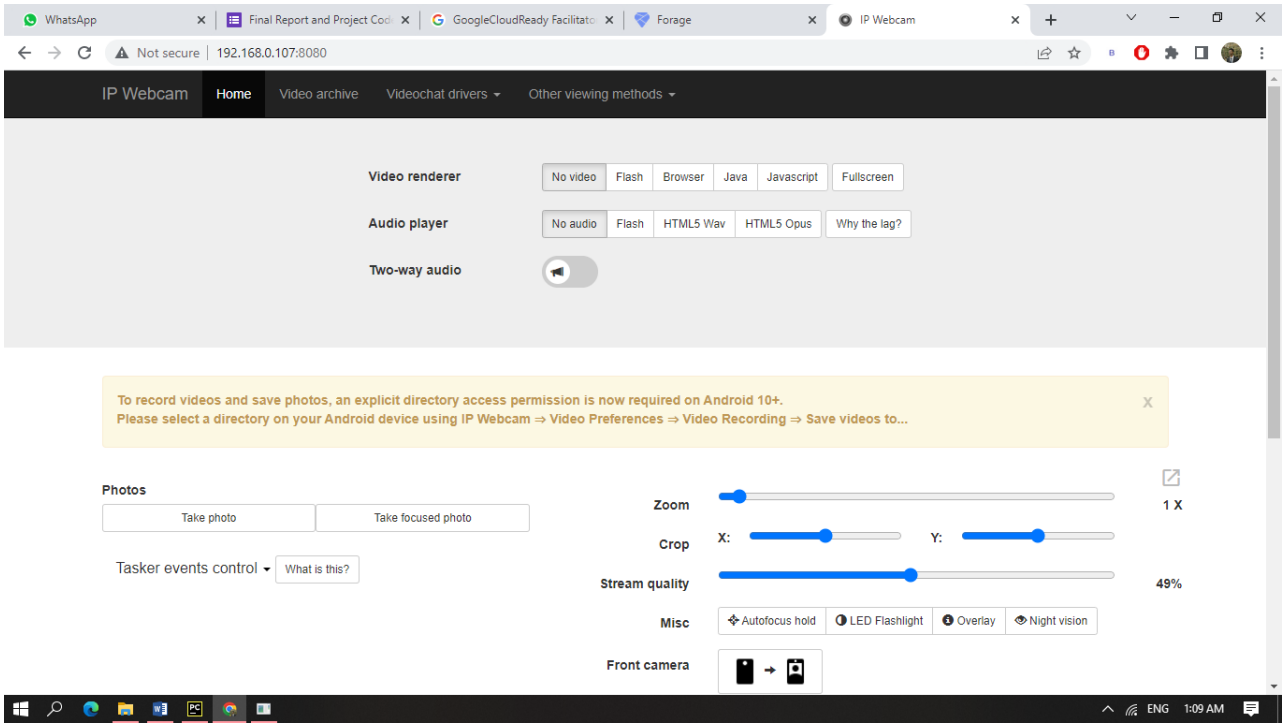
```
#How to use android device camera as webcam in Opencv.

import cv2

camera = "http://192.168.0.107:8080/video"
#connect your laptop and android device with same network either wifi or hotspot
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)   #Here parameter 0 is a path of any video use for webcam
cap.open(camera)
print("check===",cap.isOpened())

#it is 4 byte code which is use to specify the video codec
#Various codec --
#DIVX, XVID, MJPG, X264, WMV1, WMV2
fourcc = cv2.VideoWriter_fourcc(*'MP4V')  # *"XVID"
#It contain 4 parameter , name, codec,fps,resolution
output = cv2.VideoWriter("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\output.mp4",fourcc,20.0,(640,480))

while(cap.isOpened()):
    ret, frame = cap.read()   #here read the frame
    if ret == True:
        frame = cv2.resize(frame, (700, 700))
        cv2.imshow('Colorframe', frame)
        output.write(frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):  # press to exit
            break

# Release everything if job is finished
cap.release()
output.release()
cv2.destroyAllWindows()
```

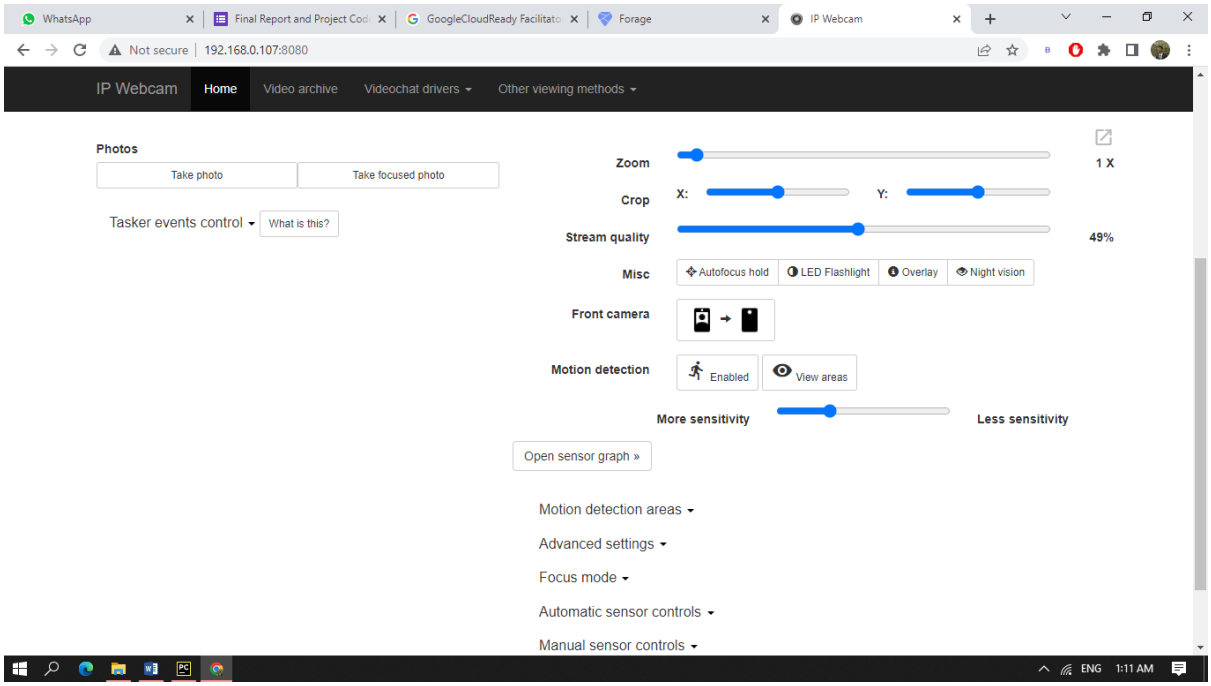
Output

Colorframe

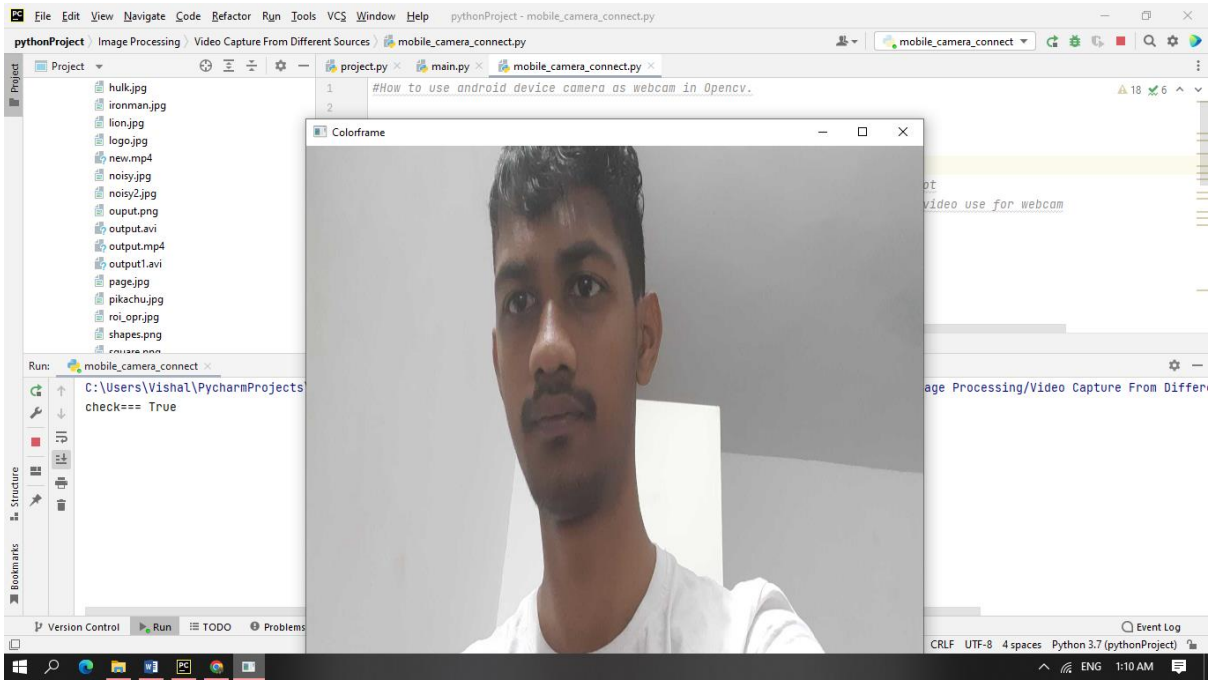


http://192.168.0.107:8080 opened on browser

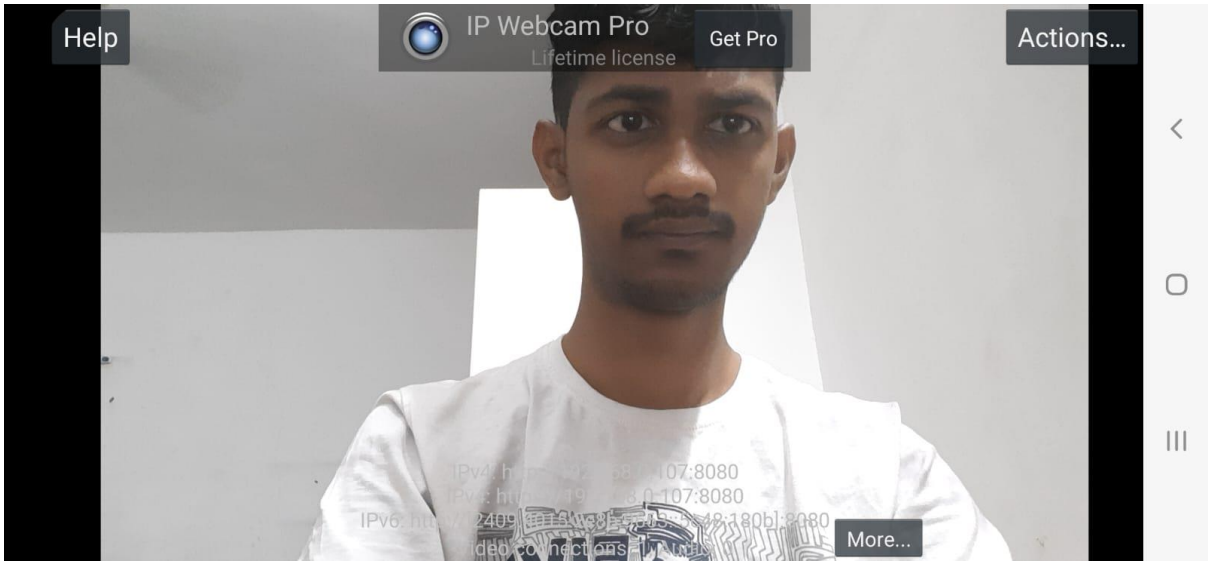




View from laptop



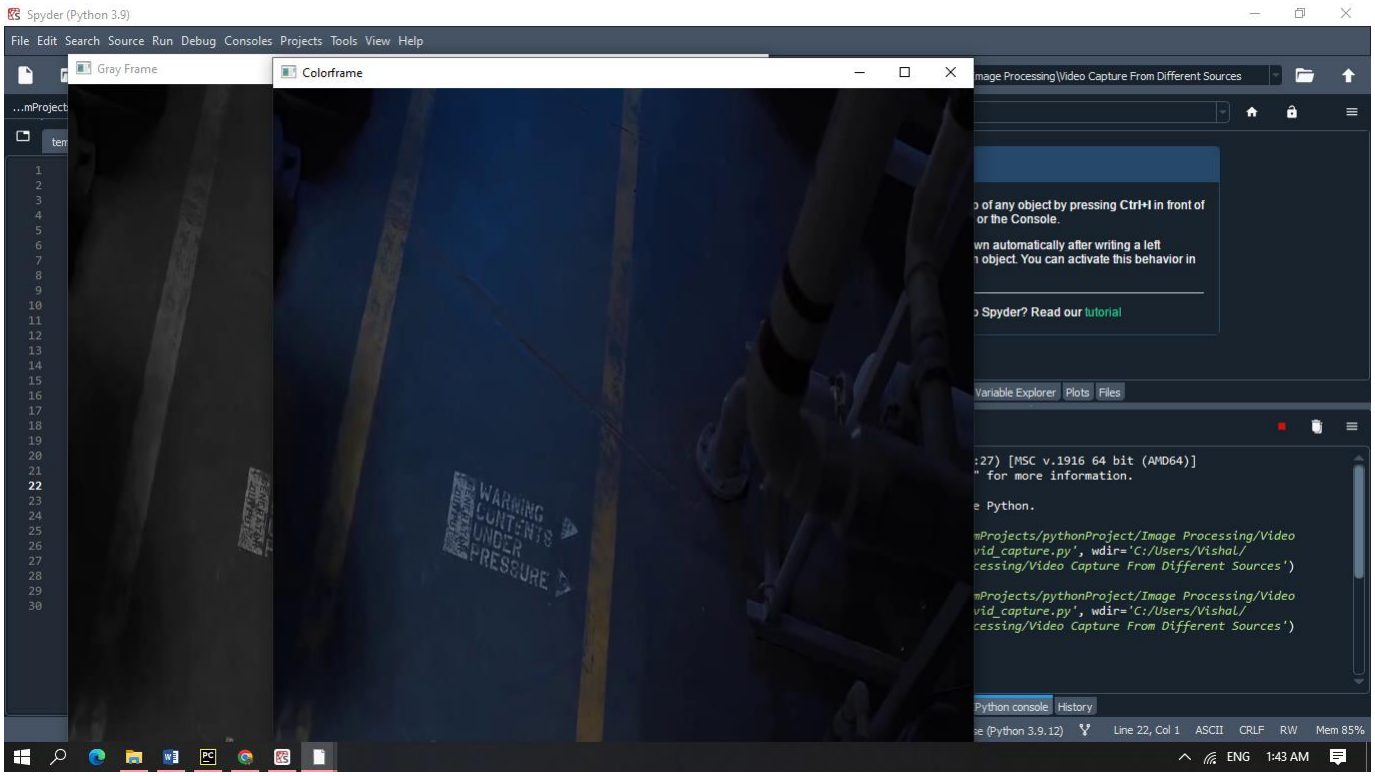
View from android on IP Webcam App

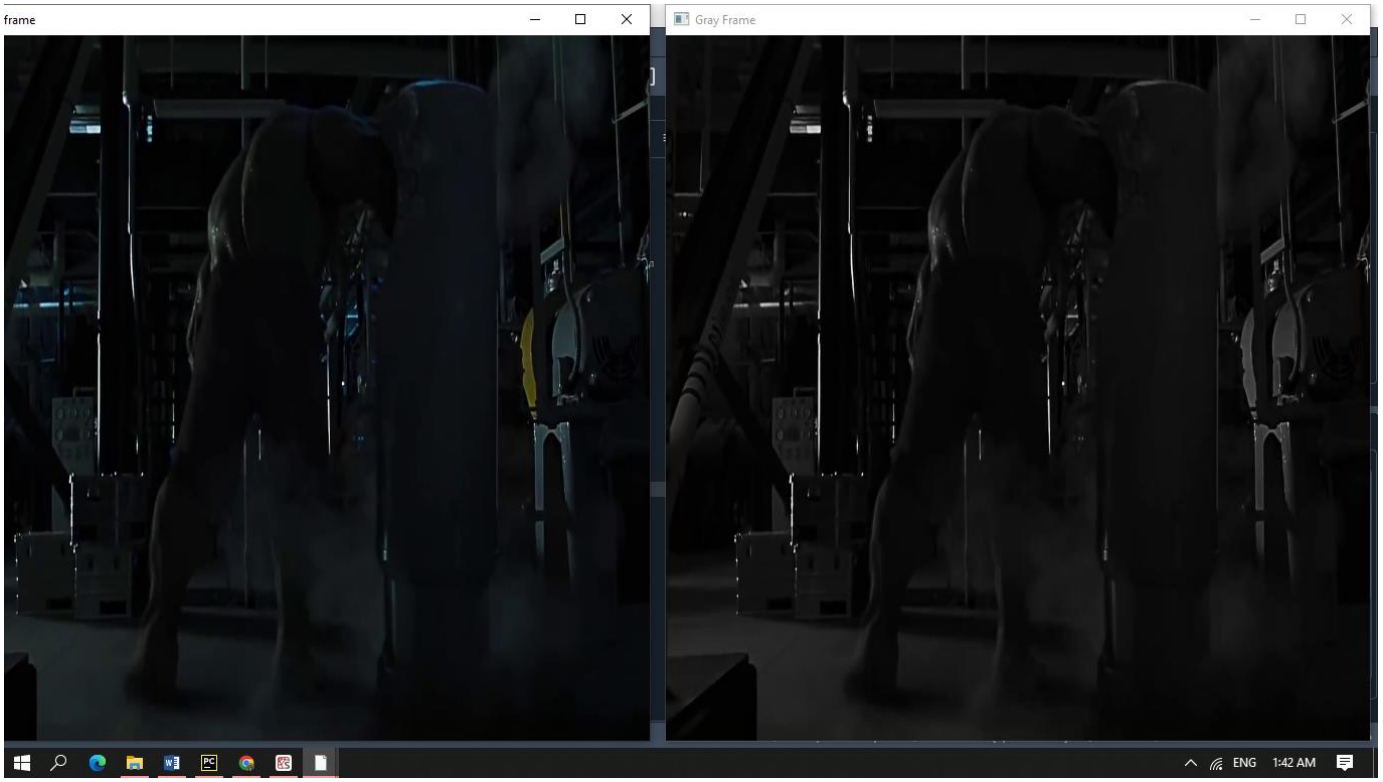


youtube_vid_capture.py

```
# Capture video from youtube
import pafy
import cv2
url = "https://www.youtube.com/watch?v=SLD9xzJ4oeU"
data = pafy.new(url )
data = data.getbest(preftype="mp4")
cap = cv2.VideoCapture()   #Here parameter 0 is a path of any video use for webcam
cap.open(data.url)

while(cap.isOpened()):
    ret, frame = cap.read()   #here read the frame

    if ret==True:
        frame = cv2.resize(frame, (700, 700))
        gray  = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
        #here flip is used to lip the video at recording time
        #frame = cv2.flip(frame,0)
        #output.write(gray)

        cv2.imshow("Gray Frame",gray)
        cv2.imshow('Colorframe',frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):   #press to exit
            break
    else:
        break

# Release everything if job is finished
cap.release()
cv2.destroyAllWindows()
```

Output


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Gray Frame
Colorframe
WARNING CONTENTS UNDER PRESSURE
Variable Explorer Plots Files
Python console History
ENG 1:43 AM

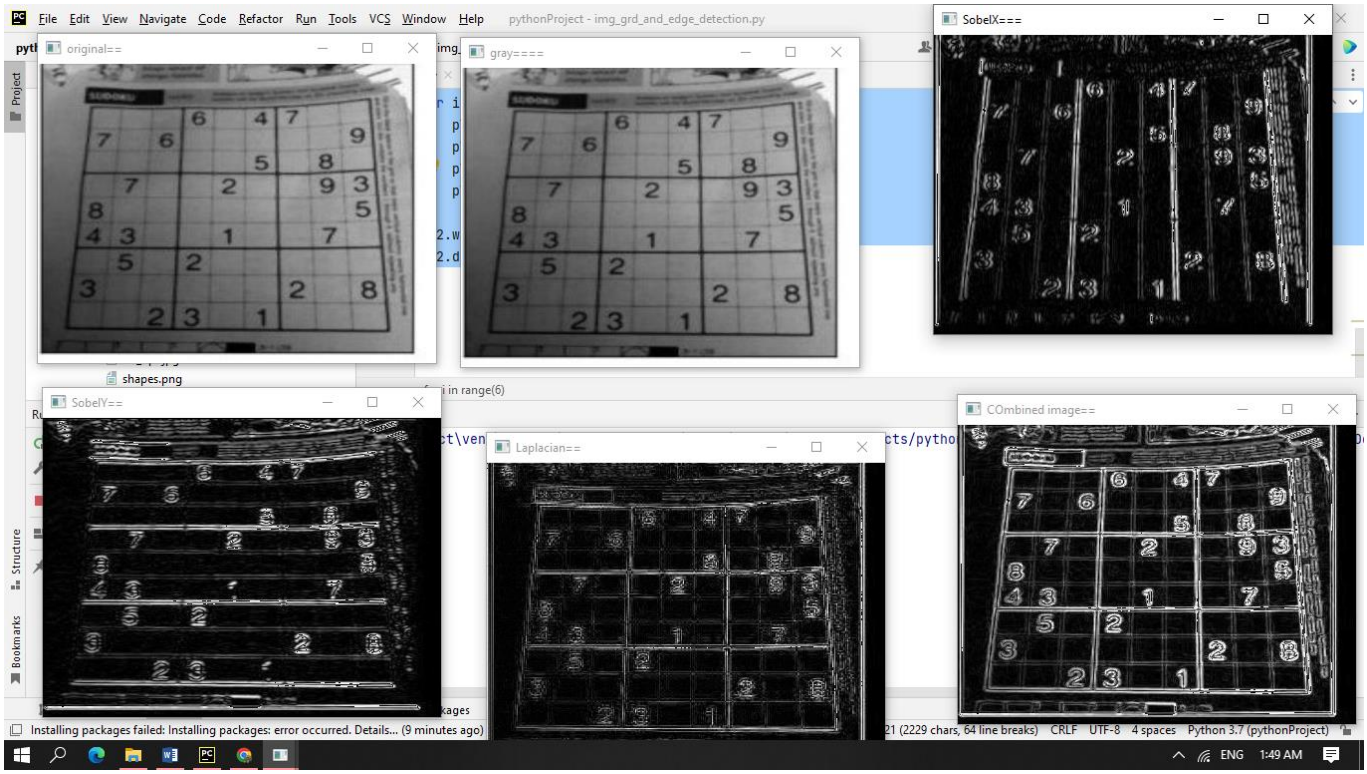

frame
Gray Frame
ENG 1:42 AM

4.Image Gradient and Edge Detection
img_grd_and_edge_detection.py

```
# Image Gradient--
# It is a directional change in the color or intensity in an image.
# It is most important part to find information from image
# Like finding edges within the images.
# There are various methods to find image gradient.
# These are - Laplacian Derivatives, SobelX and SobelY.
# All these functions have diff. mathematical approach to get result.
# All load image in the gray scale

# Don't Worry about Mathematics behind all these function
# we just use these to get our work in easy manner.

import cv2
import numpy as np

# load image into gray scale
img = cv2.imread("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image
Processing\\Data\\page.jpg")
img = cv2.resize(img, (400, 300))
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Laplacian Derivative---It calculate laplace derivate
# parameter(img,data_type for -ve val,ksize)
# kernal size must be odd
lap = cv2.Laplacian(img_gray, cv2.CV_64F, ksize=3)  # also pass kernal size
lap = np.uint8(np.absolute(lap))

# Sobel operation -
# is a joint Gausssian smoothing plus differentiation operation,
# so it is more  resistant to noise
# This is use for x and y bth
# parameter (img,type for -ve val,x = 1,y = 0,ksize)
# Sobel X focus on vertical lines
# Sobel y focus on horizontal lines

sobelX = cv2.Sobel(img_gray, cv2.CV_64F, 1, 0, ksize=3)  # here 1 means x direction
sobelY = cv2.Sobel(img_gray, cv2.CV_64F, 0, 1, ksize=3)  # here 1 means y direction

sobelX = np.uint8(np.absolute(sobelX))
sobelY = np.uint8(np.absolute(sobelY))

# finally combine sobelX and sobelY together
sobelcombine = cv2.bitwise_or(sobelX, sobelY)

cv2.imshow("original==", img)
cv2.imshow("gray====", img_gray)
```

```
cv2.imshow("Laplacian==", lap)
cv2.imshow("SobelX===", sobelX)
cv2.imshow("SobelY==", sobelY)
cv2.imshow("COmbined image==", sobelcombine)

# Now plot all the images on graph
titles = ["original", "gray", "laplacian", "sobelX", "sobelY", "combined"]
images = [img, img_gray, lap, sobelX, sobelY, sobelcombine]

# if you want then plot it
from matplotlib import pyplot as plt

for i in range(6):
    plt.subplot(3, 2, i + 1),
    plt.imshow(images[i], 'gray')
    plt.title(titles[i])
    plt.xticks([]), plt.yticks([])

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output



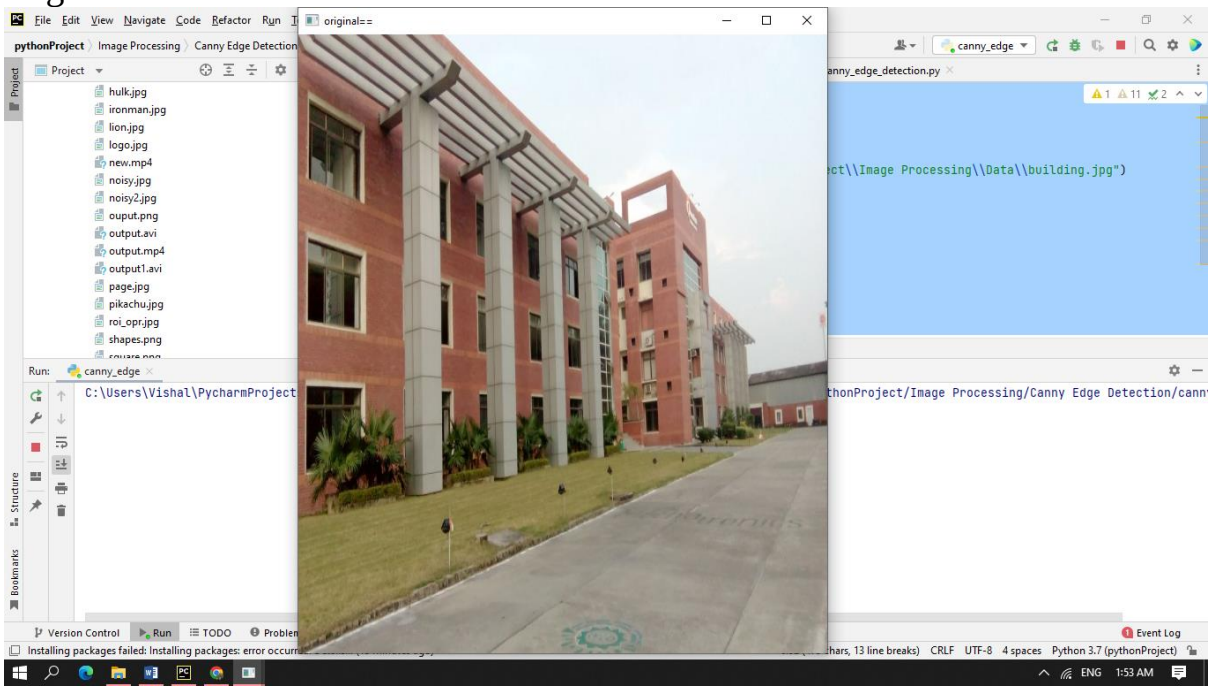
5.Canny Edge Detection

canny_edge.py

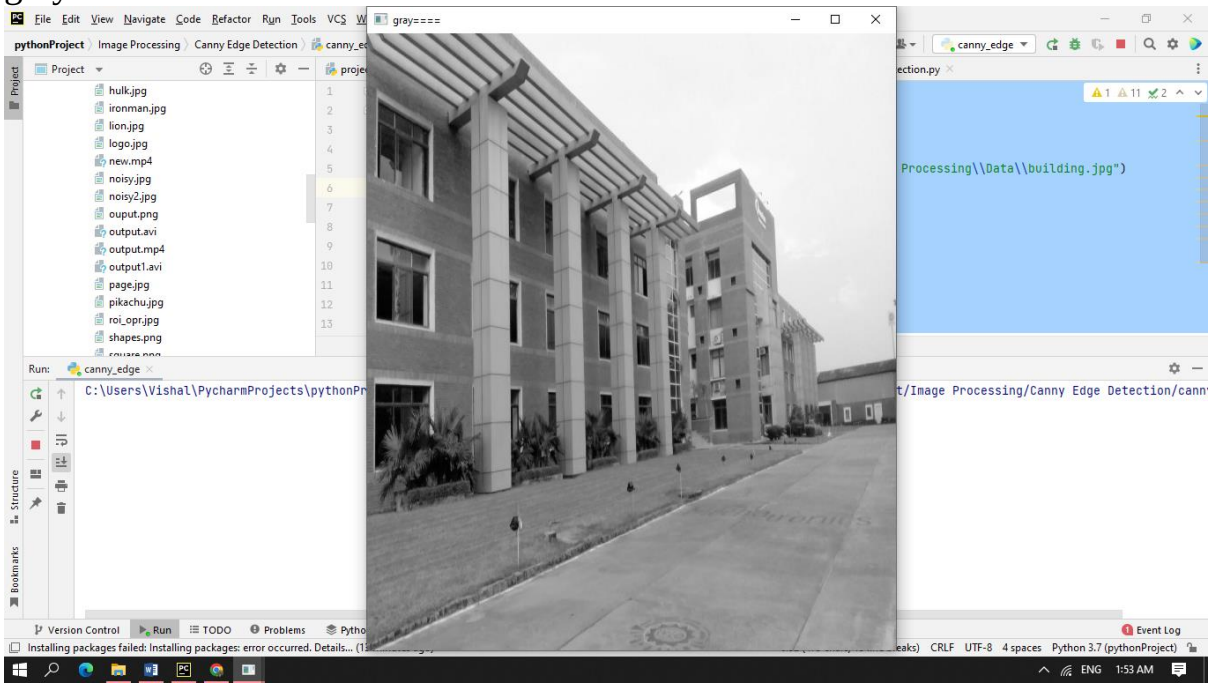
```
import cv2
import numpy as np

#load image into gray scale
img = cv2.imread("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\building.jpg")
img = cv2.resize(img,(600,700))
img_gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
#canny(img,thresh1,thres2)thresh 1 and thresh2 at different lvl
canny = cv2.Canny(img_gray,20,150)
cv2.imshow("original==",img)
cv2.imshow("gray====",img_gray)
cv2.imshow("canny==",canny)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output

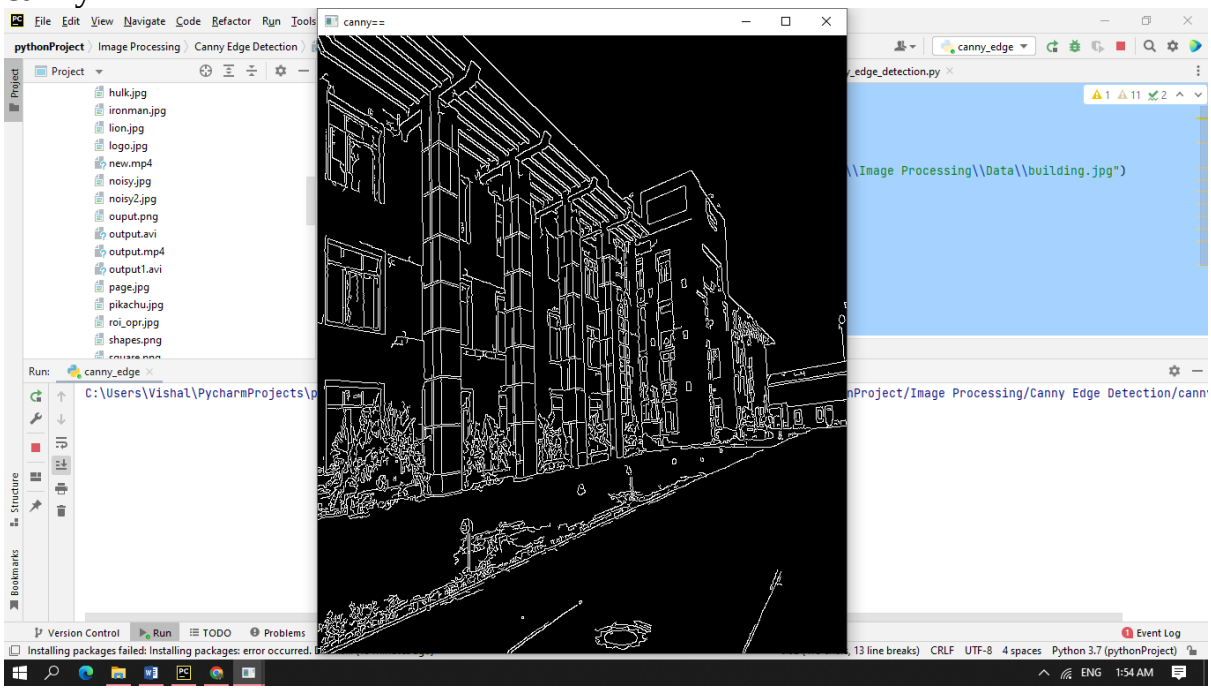
original==



gray====



canny==

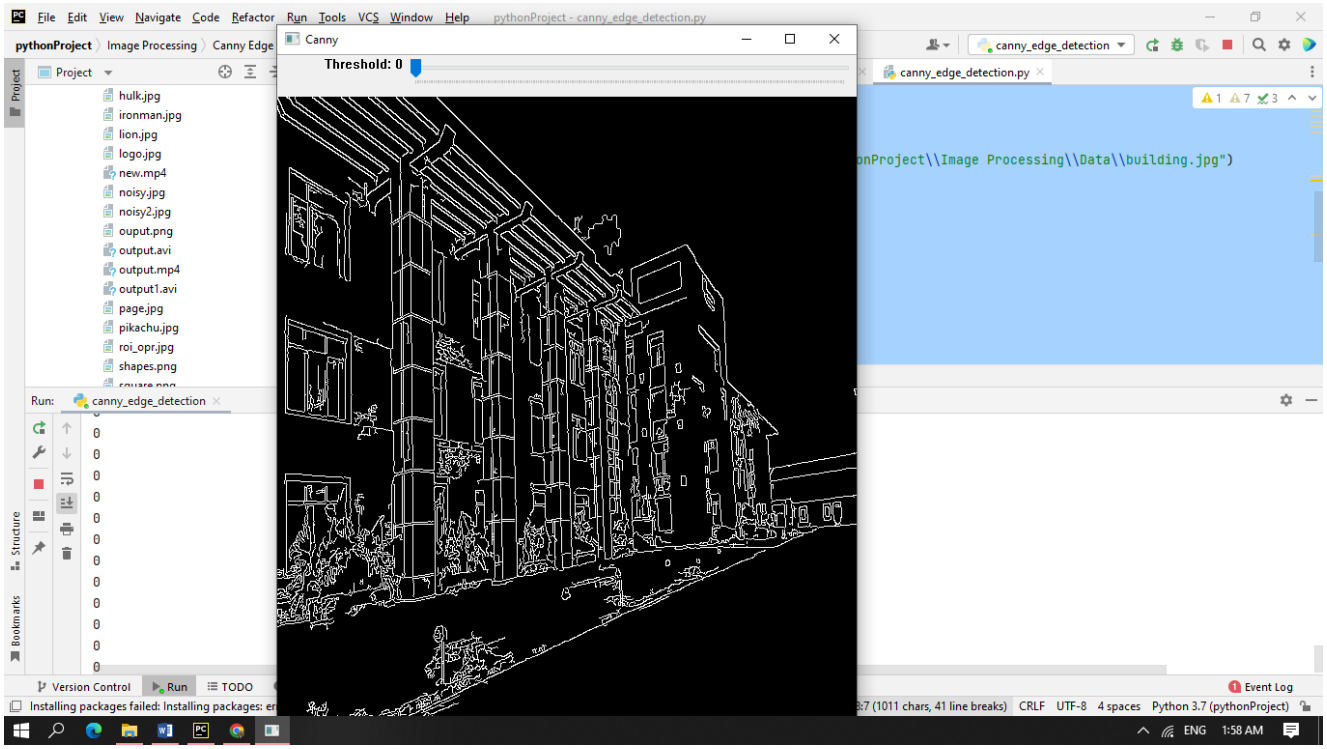

canny==
File Edit View Navigate Code Refactor Run Tools
pythonProject Image Processing Canny Edge Detection
canny_edge
Project
hulk.jpg
ironman.jpg
lion.jpg
logo.jpg
new.mp4
noisy.jpg
noisy2.jpg
ouput.png
output.avi
output.mp4
output1.avi
page.jpg
pikachu.jpg
roi_opr.jpg
shapes.png
\\Image Processing\\Data\\building.jpg")
Run: canny_edge
C:\Users\Vishal\PycharmProjects\p
Version Control
Run
TODO
Problems
Event Log
Installing packages failed: Installing packages: error occurred.
CRLF UTF-8 4 spaces Python 3.7 (pythonProject)
ENG 1:54 AM

canny_edge_detection.py

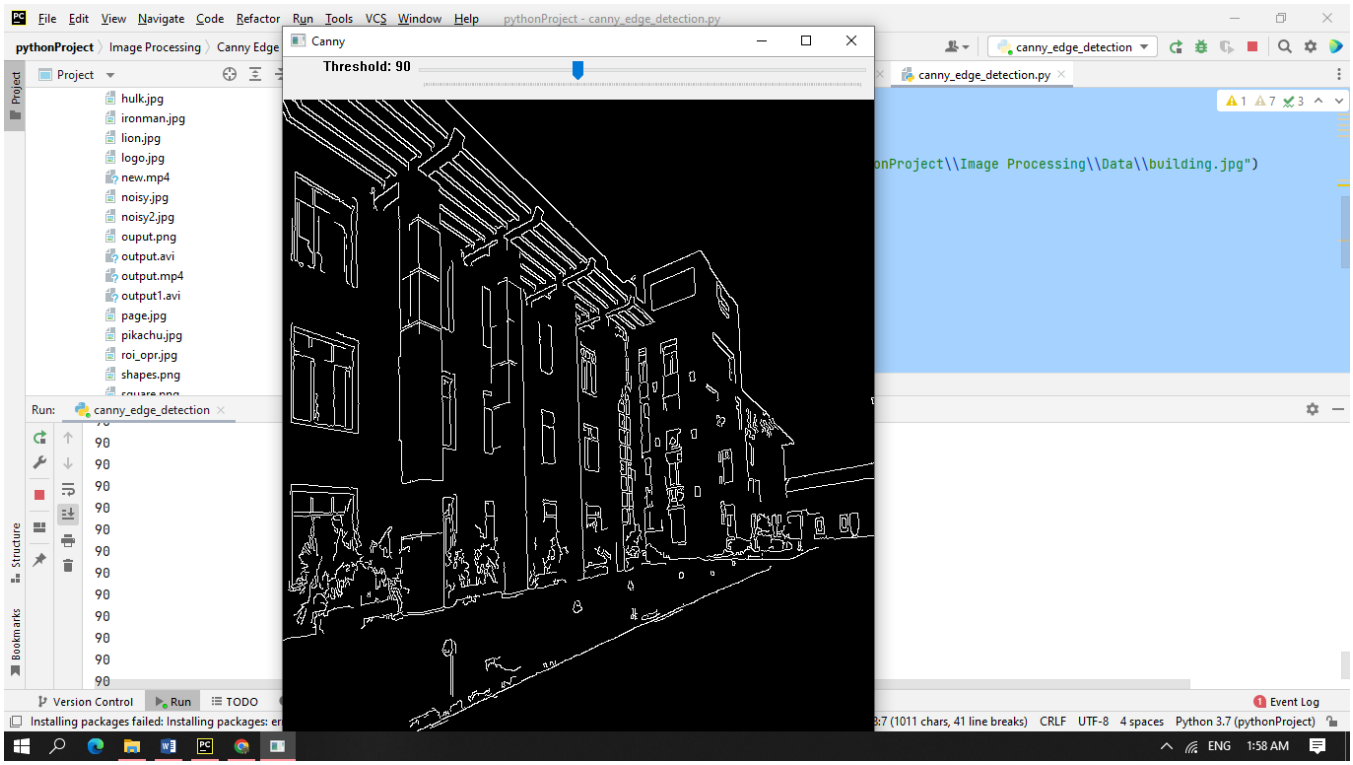
```
#Canny Edge Detection using OpenCV
#Canny Edge Detection is a popular edge detection approach.
#It is use  multi-stage algorithm to detect a edges.
#It was developed by John F. Canny in 1986.
#This approach combine with 5 steps.
# 1) -  Noise reduction(gauss)), 2) -Gradient calculation( ,
# 3) - Non-maximum suppression, 4) - Double Threshold,
# 5) - Edge Tracking by Hysteresis

import cv2
import numpy as np

# building trackbar with canny edge
# load image into gray scale
img = cv2.imread("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image
Processing\\Data\\building.jpg")
img = cv2.resize(img, (600, 700))
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)


def nothing(x):
    pass

cv2.namedWindow("Canny")
cv2.createTrackbar("Threshold", "Canny", 0, 255, nothing)

while True:
    a = cv2.getTrackbarPos('Threshold', 'Canny')

    print(a)
    res = cv2.Canny(img_gray, a, 255)
    cv2.imshow("Canny", res)
    k = cv2.waitKey(1) & 0xFF
    if k == 27:
        break

cv2.destroyAllWindows()
```

Output

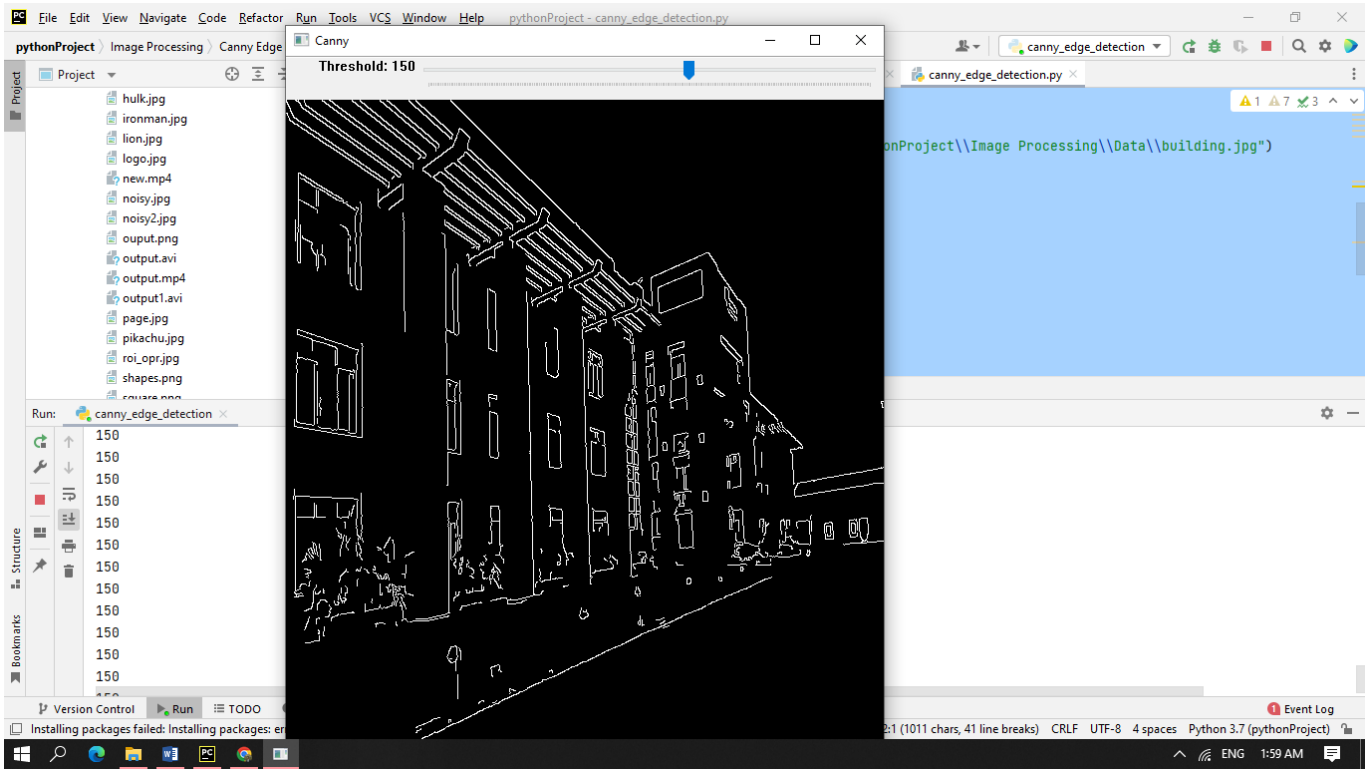
Threshold at 0



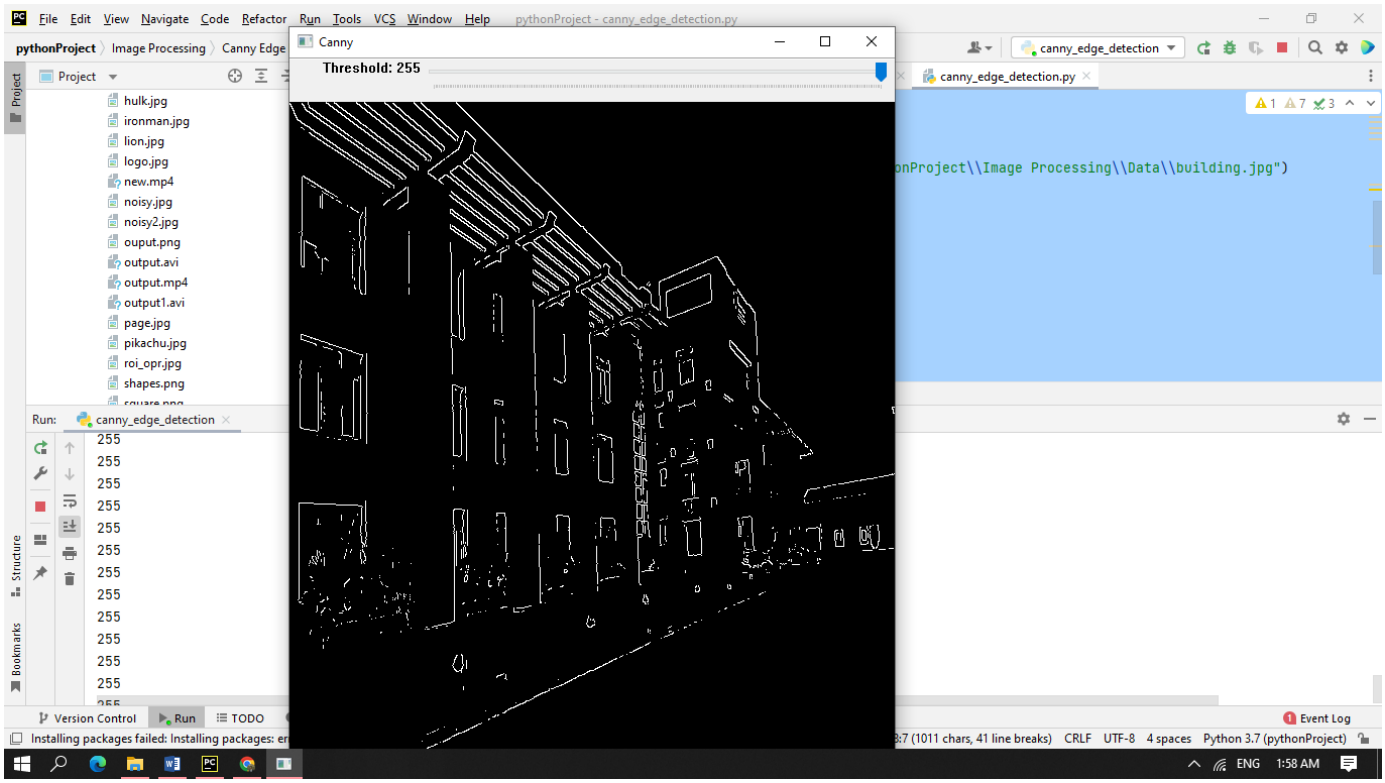
Threshold at 90



Threshold at 150



Threshold at 255



6.Face Detection on Image

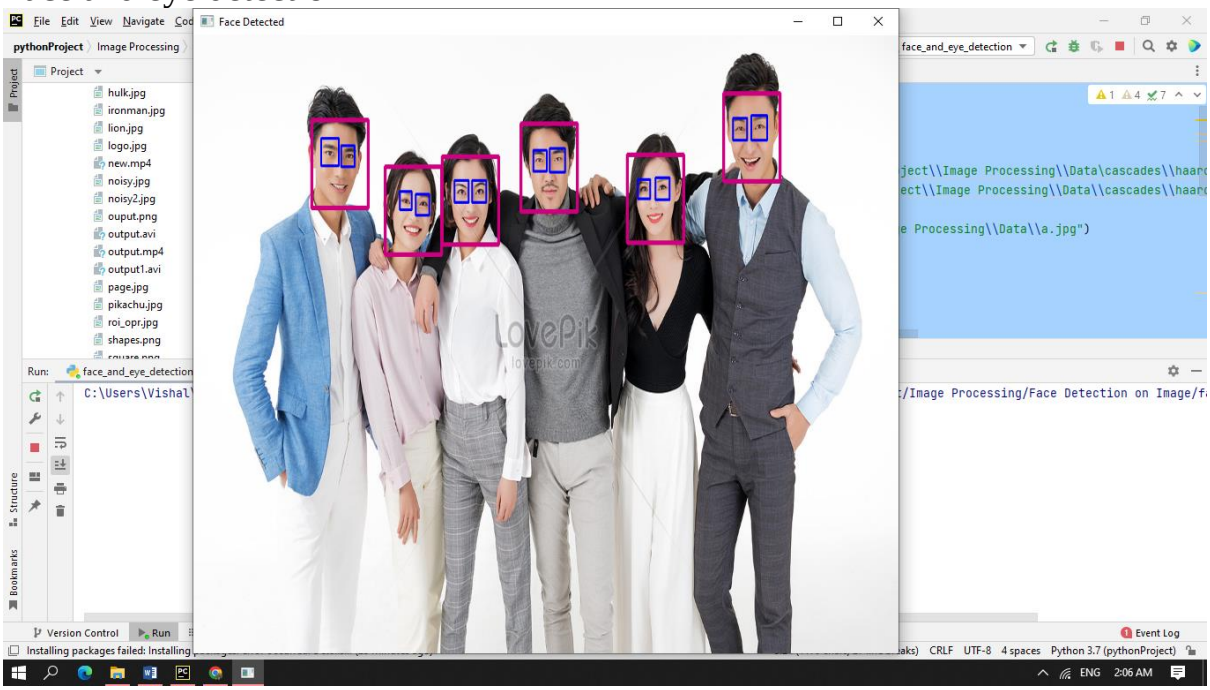
face_and_eye_detection.py

```
# Face Detection using haarcascade file
import cv2
import numpy

face = cv2.CascadeClassifier("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\cascades\\haarcascade_frontalface_default.xml")  # for detecting face
eye = cv2.CascadeClassifier('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\cascades\\haarcascade_eye.xml')  # for detecting eyes

image = cv2.imread("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\a.jpg")
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)  # convert into gray

# parameters(img,scale_factor[reduce image size],min_neighbour)
faces = face.detectMultiScale(gray, 4, 4)  # for  faces

for (x, y, w, h) in faces:

    image = cv2.rectangle(image, (x, y), (x + w, y + h), (127, 0, 205), 3)

    # Now detect eyes
    roi_gray = gray[y:y + h, x:x + w]
    roi_color = image[y:y + h, x:x + w]
    eyes = eye.detectMultiScale(roi_gray, 1.2, 1)
    for (ex, ey, ew, eh) in eyes:
        cv2.rectangle(roi_color, (ex, ey), (ex + ew, ey + eh), (255, 0, 0), 2)

image = cv2.resize(image, (800, 700))
cv2.imshow("Face Detected", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output

original image *a.jpg*

Face and eye detection

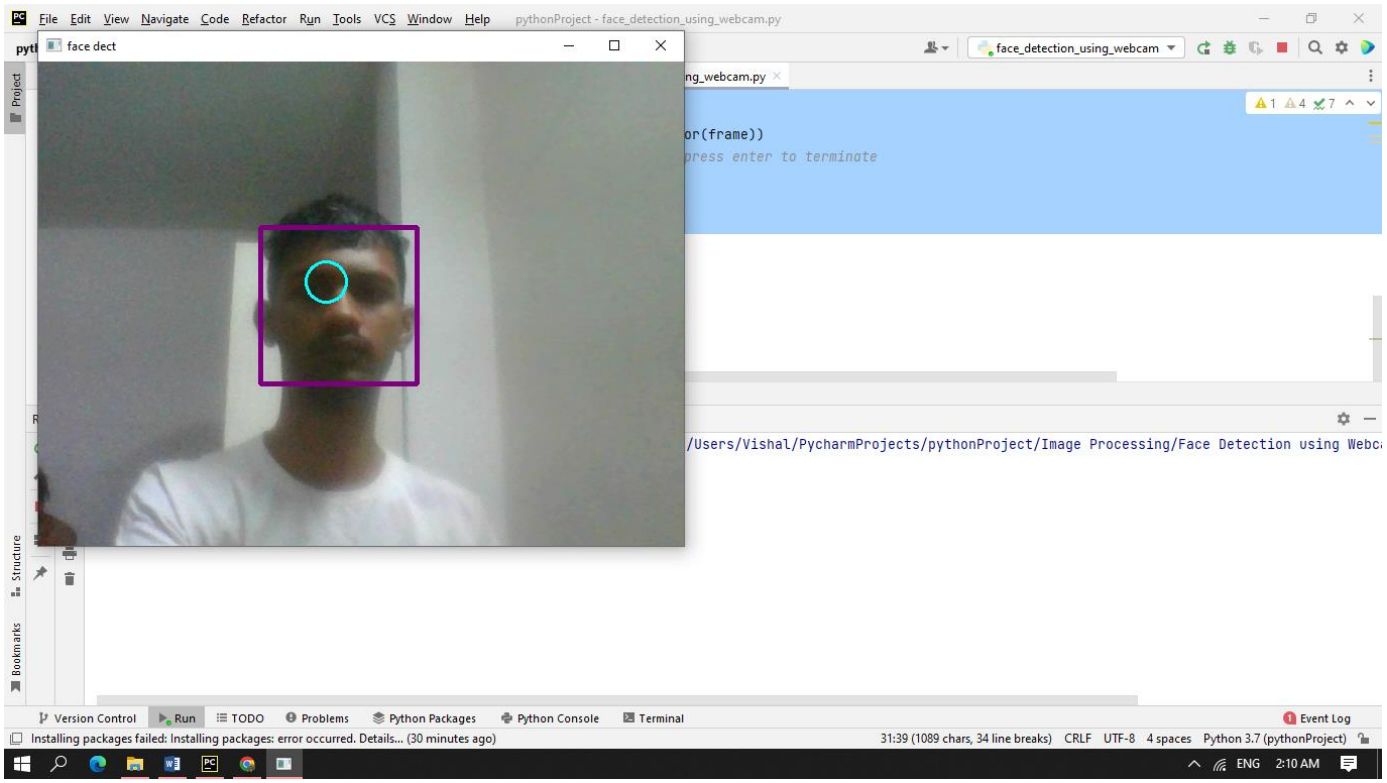


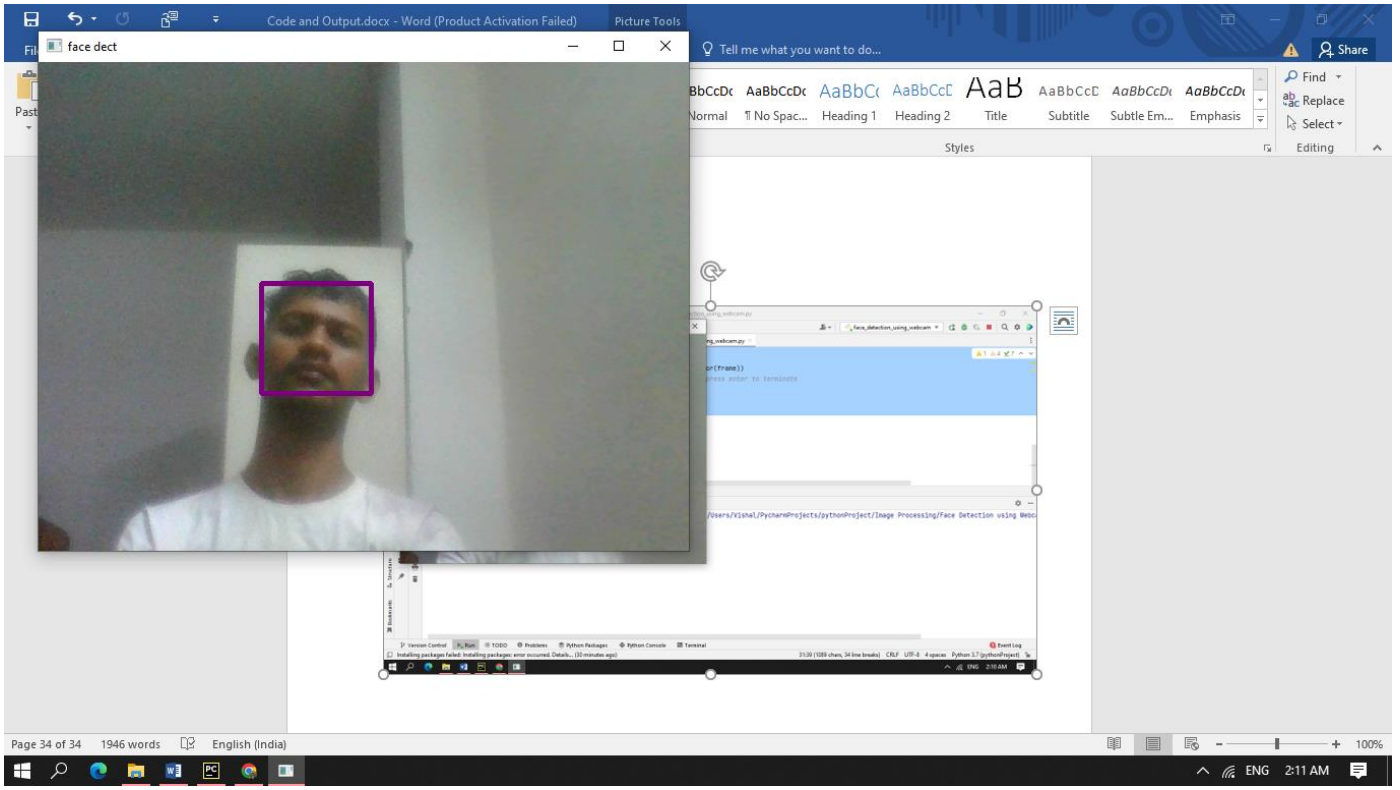
7.Face Detection using Webcam

face_detection_using_webcam.py

```
import cv2
import numpy

face = cv2.CascadeClassifier("C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\cascades\\haarcascade_frontalface_default.xml")
eye = cv2.CascadeClassifier('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\cascades\\haarcascade_eye.xml')  # for detecting eyes


def dector(img):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face.detectMultiScale(gray, 1.3, 5)

    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h), (127, 0, 125), 3)

        roi_gray = gray[y:y + h, x:x + w]

        roi_color = img[y:y + h, x:x + w]

        eyes = eye.detectMultiScale(roi_gray, 1.3, 3)
        for (ex, ey, ew, eh) in eyes:
            cv2.circle(roi_color, (ex + 27, ey + 27), 20, (255, 255, 0), 2)

    return img


cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
while True:
    ret, frame = cap.read()
    frame = cv2.flip(frame, 2)
    cv2.imshow("face dect", dector(frame))
    if cv2.waitKey(1) == 13:  # press enter to terminate
        break

cap.release()
cv2.destroyAllWindows()
```

Output


face dect
File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject - face_detection_using_webcam.py
face_detection_using_webcam
ng_webcam.py
or(frame))
press enter to terminate
/Users/Vishal/PycharmProjects/pythonProject/Image Processing/Face Detection using Webc
Version Control
Run
TODO
Problems
Python Packages
Python Console
Terminal
Event Log
Installing packages failed: Installing packages: error occurred. Details... (30 minutes ago)
31:39 (1089 chars, 34 line breaks)
CRLF
UTF-8
4 spaces
Python 3.7 (pythonProject)
ENG
2:10 AM


face dect
Code and Output.docx - Word (Product Activation Failed)
Picture Tools
Tell me what you want to do...
Share
Find
Replace
Select
Styles
Editing
Page 34 of 34
1946 words
English (India)
100%
ENG
2:11 AM

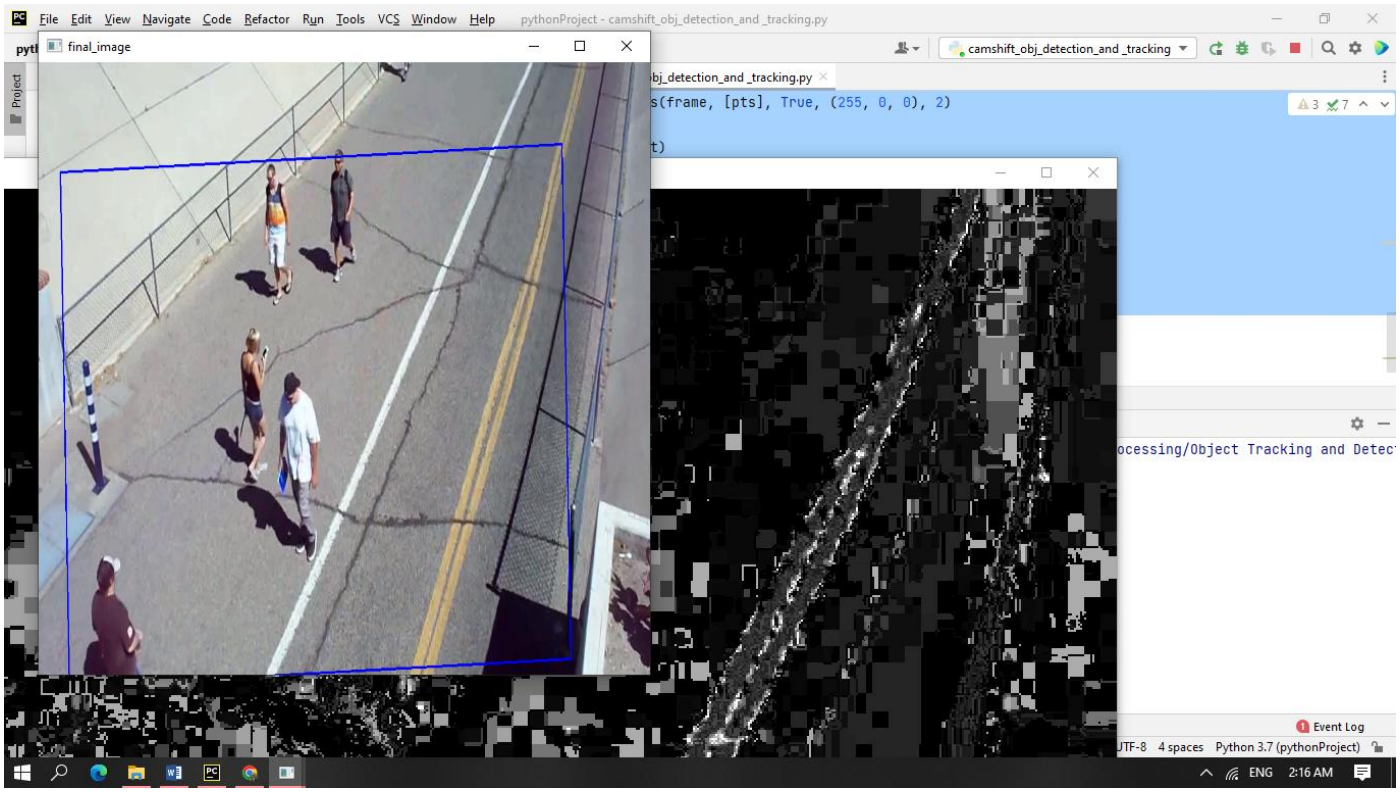
8.Object Tracking and Detection

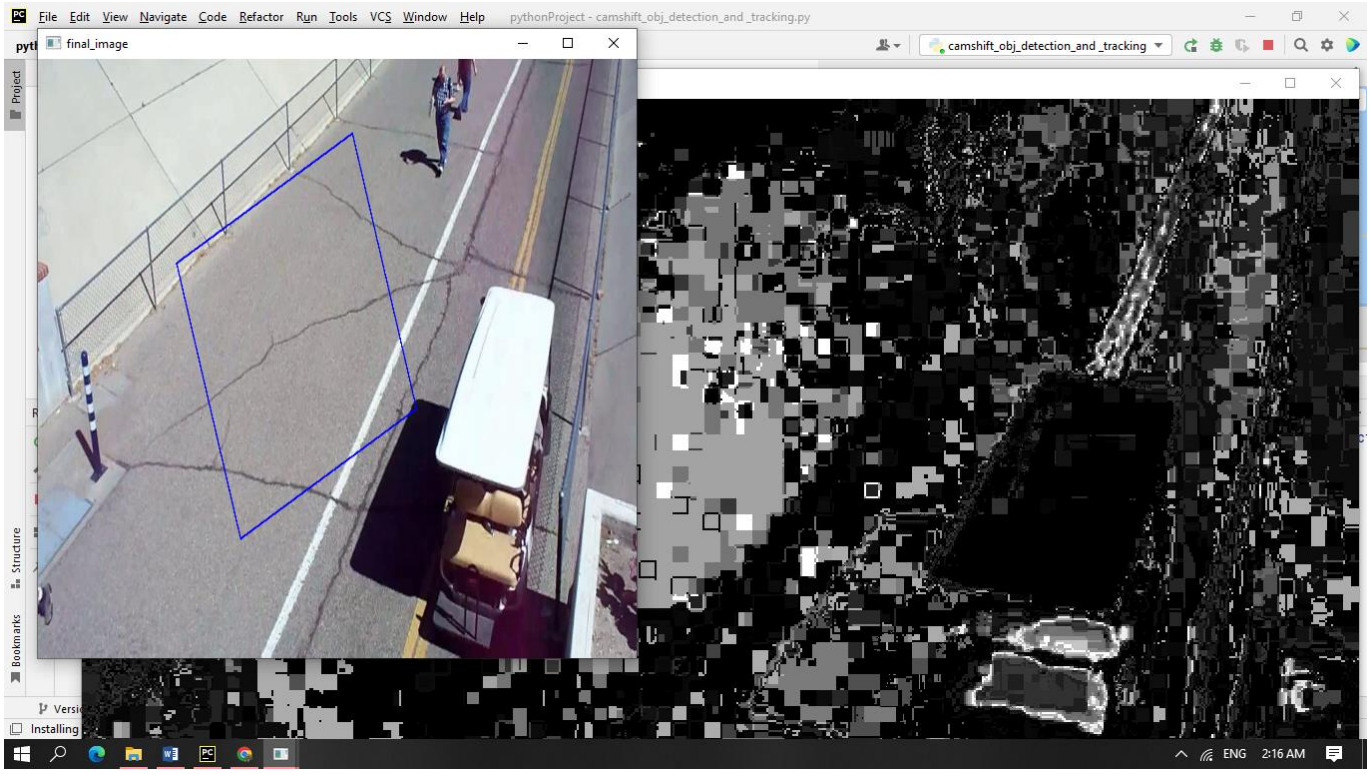
camshift_obj_detection_and _tracking.py

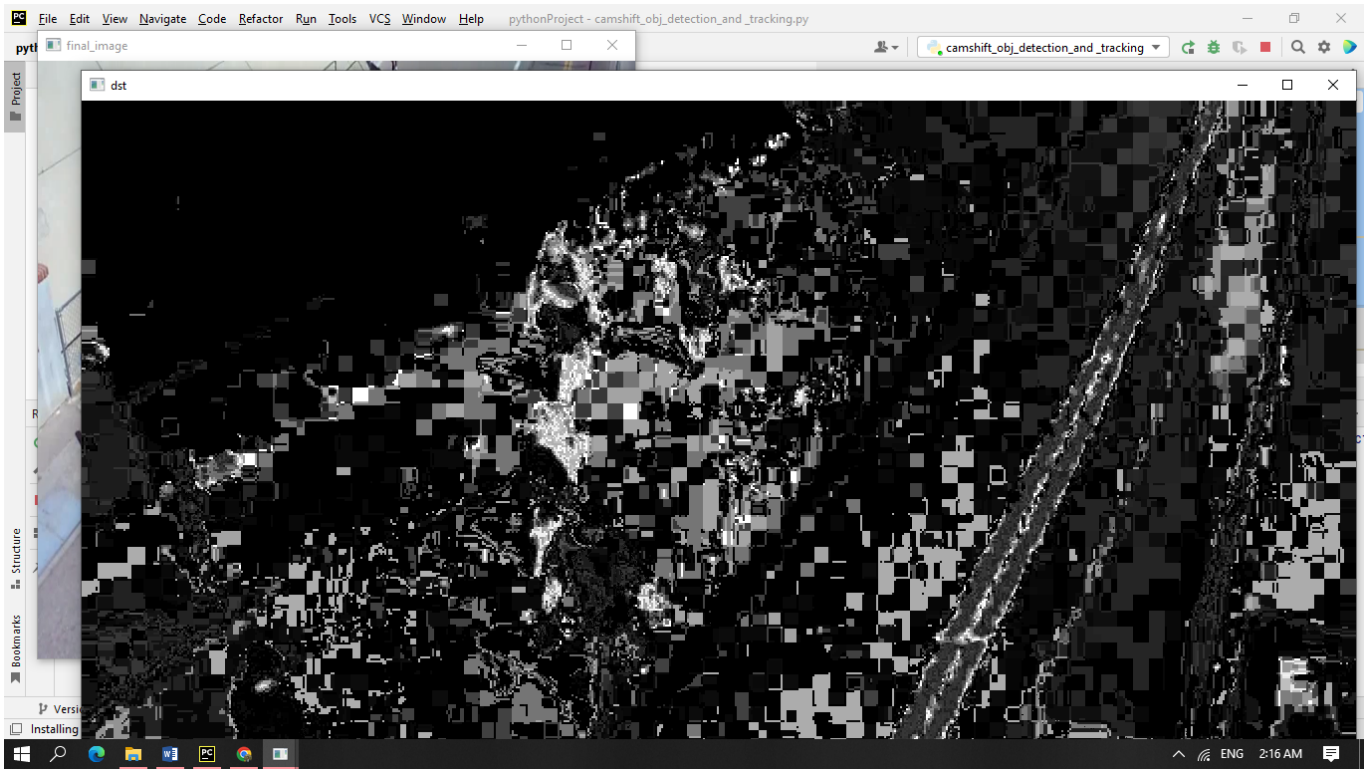
```
# CAMshift (Continuously Adaptive Meanshift)


import numpy as np
import cv2 as cv

cap = cv.VideoCapture('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\test2.mp4')

# take first frame of the video
ret, frame = cap.read()

# setup initial location of window
x, y, width, height = 580, 30, 80, 150
track = (x, y, width, height)

# set up the ROI for tracking
roi = frame[y:y + height, x: x + width]
hsv_roi = cv.cvtColor(roi, cv.COLOR_BGR2HSV)
mask = cv.inRange(hsv_roi, np.array((0., 60., 32.)), np.array((180., 255., 255)))
roi_hist = cv.calcHist([hsv_roi], [0], mask, [180], [0, 180])
cv.normalize(roi_hist, roi_hist, 0, 255, cv.NORM_MINMAX)

# Setup the termination criteria, either 10 iteration or move by atleast 1 pt
termntn = (cv.TERM_CRITERIA_EPS | cv.TERM_CRITERIA_COUNT, 10, 1)
cv.imshow('roi', roi)
while True:
    ret, frame = cap.read()

    if ret == True:

        hsv = cv.cvtColor(frame, cv.COLOR_BGR2HSV)
        dst = cv.calcBackProject([hsv], [0], roi_hist, [0, 180], 1)

        # apply meanshift to get the new location
        ret, track = cv.CamShift(dst, track, termntn)

        # Draw it on image
        # x,y,w,h = track
        # final = cv.rectangle(frame, (x,y), (x+w, y+h), (0,0,255), 3)
```

```
        # Try to draw rotating rectangle
        pts = cv.boxPoints(ret)
        pts = np.int64(pts)
        final = cv.polylines(frame, [pts], True, (255, 0, 0), 2)

        cv.imshow('dst', dst)
        frame = cv.resize(final, (600, 600))
        cv.imshow('final_image', frame)
        k = cv.waitKey(30) & 0xff
        if k == 27:
            break
    else:
        break
cv.destroyAllWindows()
```

Output




File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject - camshift_obj_detection_and _tracking.py
final_image
camshift_obj_detection_and _tracking
Project
Structure
Bookmarks
Installing
ENG
2:16 AM

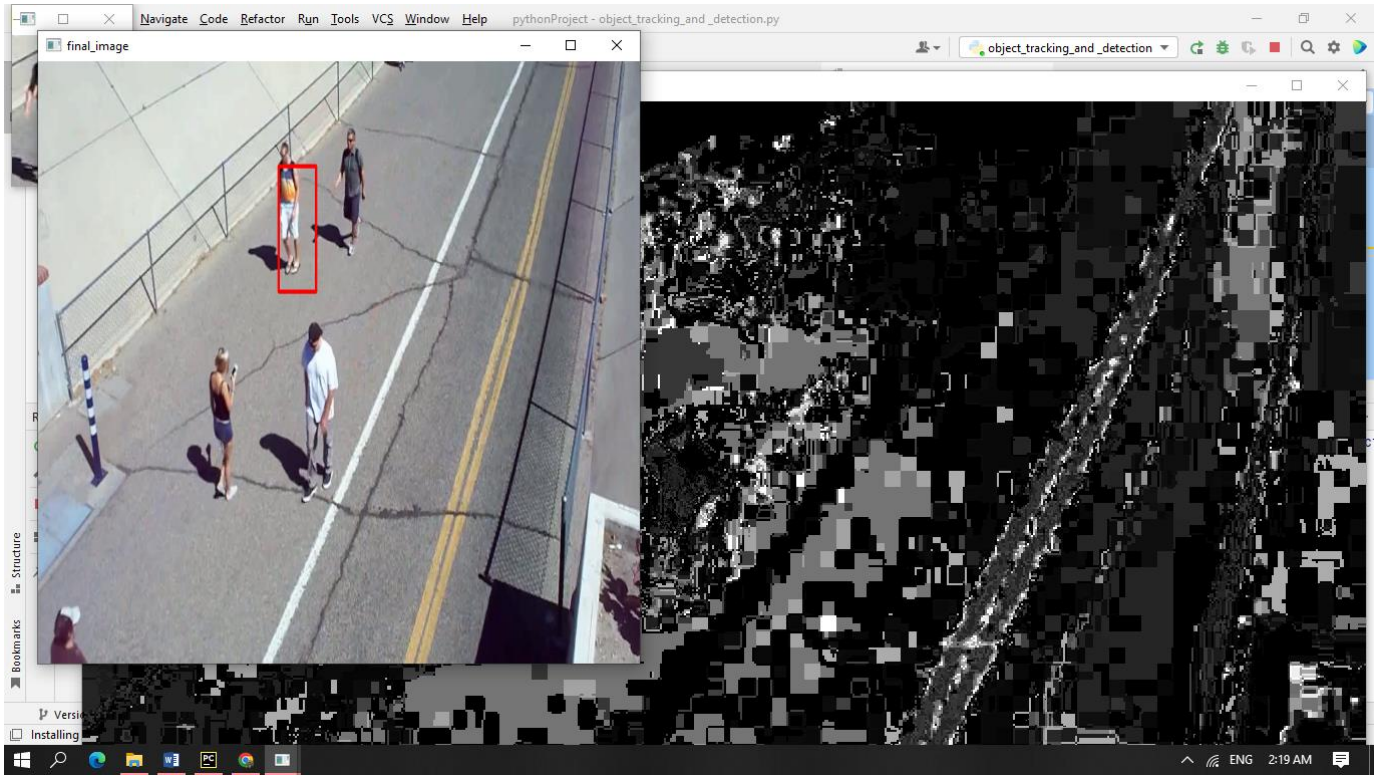

File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject - camshift_obj_detection_and _tracking.py
final_image
dst
camshift_obj_detection_and _tracking
Project
Structure
Bookmarks
Installing
ENG
2:16 AM

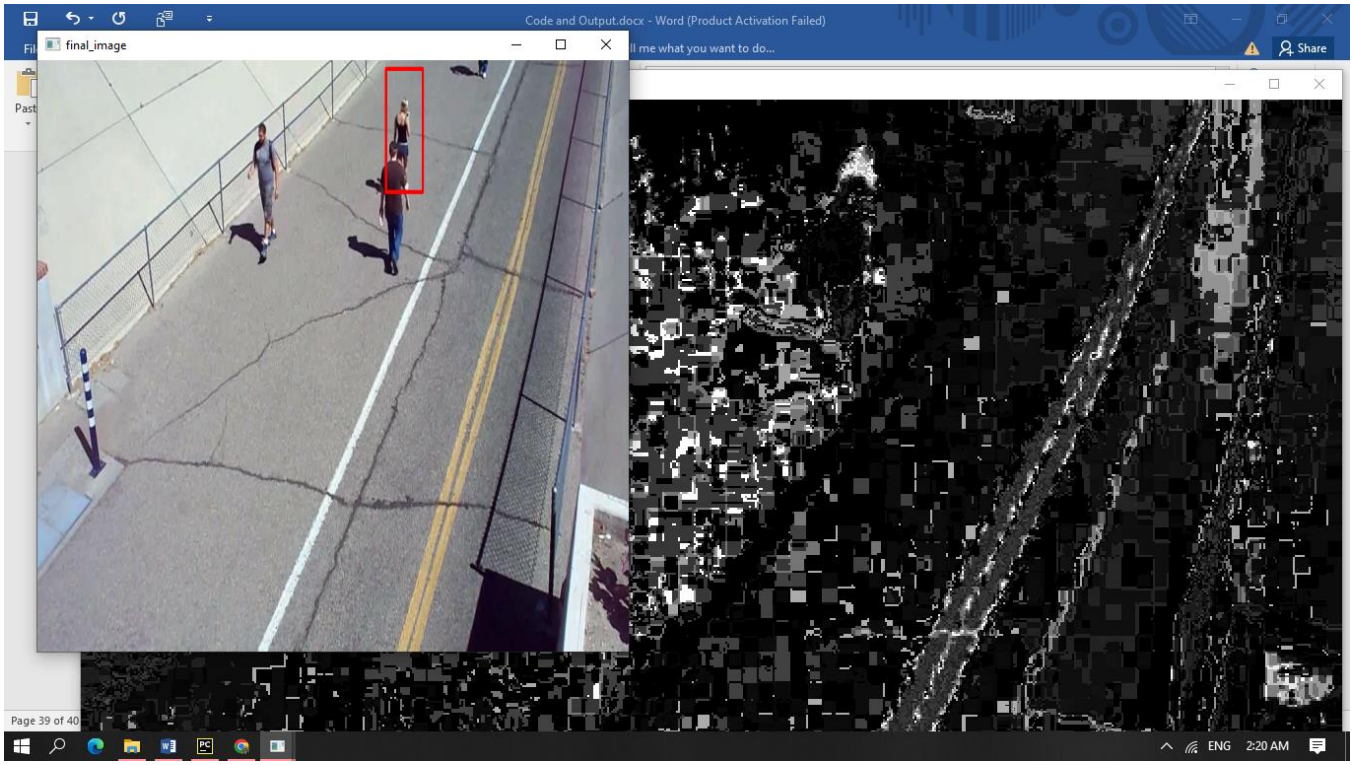
object_tracking_and _detection.py

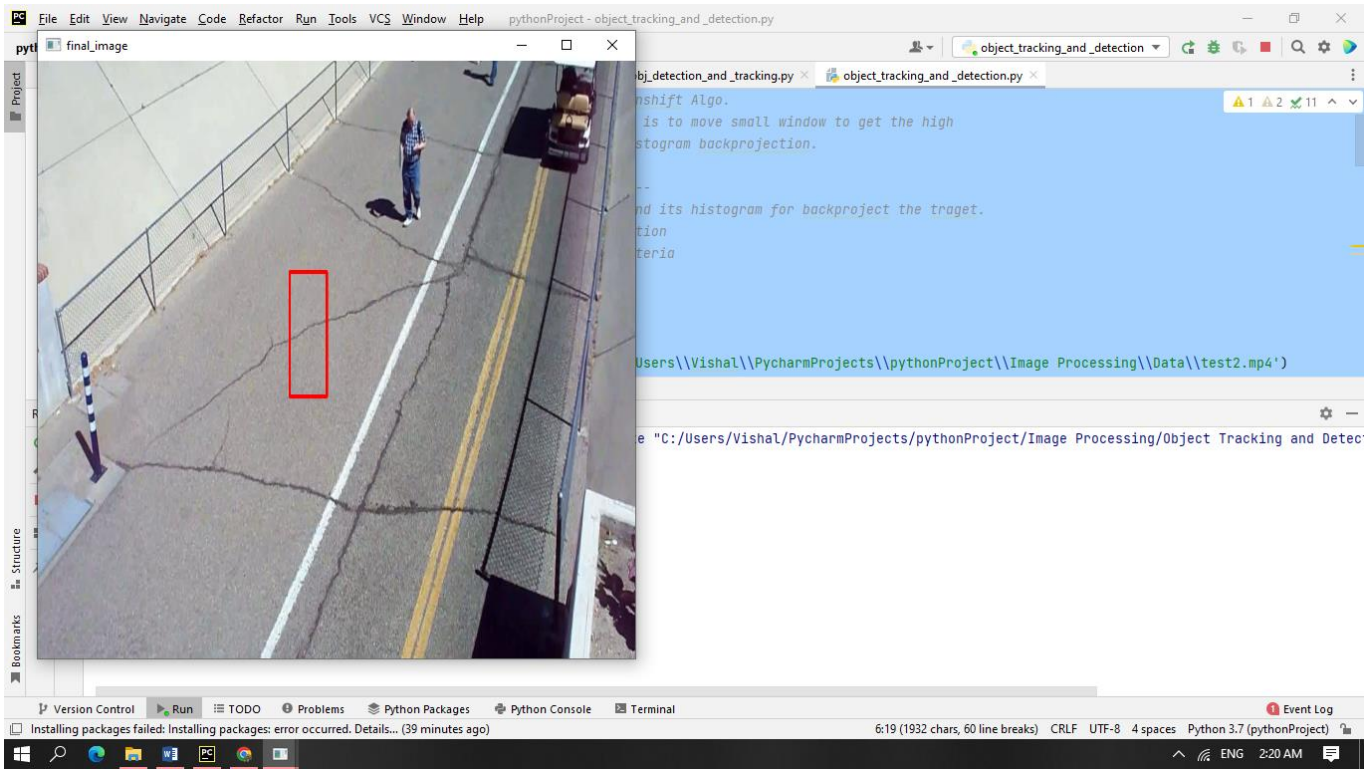
```
# Object Tracking using Meanshift Algo.
# The idea behind this algo is to move small window to get the high
# density pixels same as histogram backprojection.

# Steps to use this algo-----
# First setup target and find its histogram for backproject the target.
# Also set one initial location
# setup the termination criteria

import numpy as np
import cv2 as cv

cap = cv.VideoCapture('C:\\Users\\Vishal\\PycharmProjects\\pythonProject\\Image
Processing\\Data\\test2.mp4')

# take first frame of the video
ret, frame = cap.read()

# setup initial location of window
x, y, width, height = 580, 30, 80, 150
track = (x, y, width, height)

# set up the ROI for tracking
roi = frame[y:y + height, x: x + width]
hsv_roi = cv.cvtColor(roi, cv.COLOR_BGR2HSV)
mask = cv.inRange(hsv_roi, np.array((0., 60., 32.)),
                  np.array((180., 255., 255)))
roi_hist = cv.calcHist([hsv_roi], [0], mask, [180], [0, 180])
cv.normalize(roi_hist, roi_hist, 0, 255, cv.NORM_MINMAX)

# Setup the termination criteria, either 10 iteration or move by atleast 1 pt
termntn = (cv.TERM_CRITERIA_EPS | cv.TERM_CRITERIA_COUNT, 10, 1)
cv.imshow('roi', roi)
while True:
    ret, frame = cap.read()

    if ret == True:

        hsv = cv.cvtColor(frame, cv.COLOR_BGR2HSV)
        dst = cv.calcBackProject([hsv], [0], roi_hist, [0, 180], 1)

        # apply meanshift to get the new location
        ret, track = cv.meanShift(dst, track, termntn)
```

```
        # Draw it on image
        x, y, w, h = track
        final = cv.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 3)

        cv.imshow('dst', dst)
        frame = cv.resize(final, (600, 600))
        cv.imshow('final_image', frame)
        k = cv.waitKey(30) & 0xff
        if k == 27:
            break
    else:
        break
cv.destroyAllWindows()

# There are few disadvantages of this algo
# Fixe of target window should not be changed.
# to manually pass roi and then detect our target
```

Output




final_image
Code and Output.docx - Word (Product Activation Failed)
Share
Page 39 of 40
ENG
2:20 AM


final_image
File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject - object_tracking_and _detection.py
object_tracking_and _detection
object_tracking_and _detection.py
Users\\Vishal\\PycharmProjects\\pythonProject\\Image Processing\\Data\\test2.mp4')
Version Control
Run
TODO
Problems
Python Packages
Python Console
Terminal
Event Log
Installing packages failed: Installing packages: error occurred. Details... (39 minutes ago)
6:19 (1932 chars, 60 line breaks)
CRLF
UTF-8
4 spaces
Python 3.7 (pythonProject)
ENG
2:20 AM

9.Cam Scanner

Scanner.py

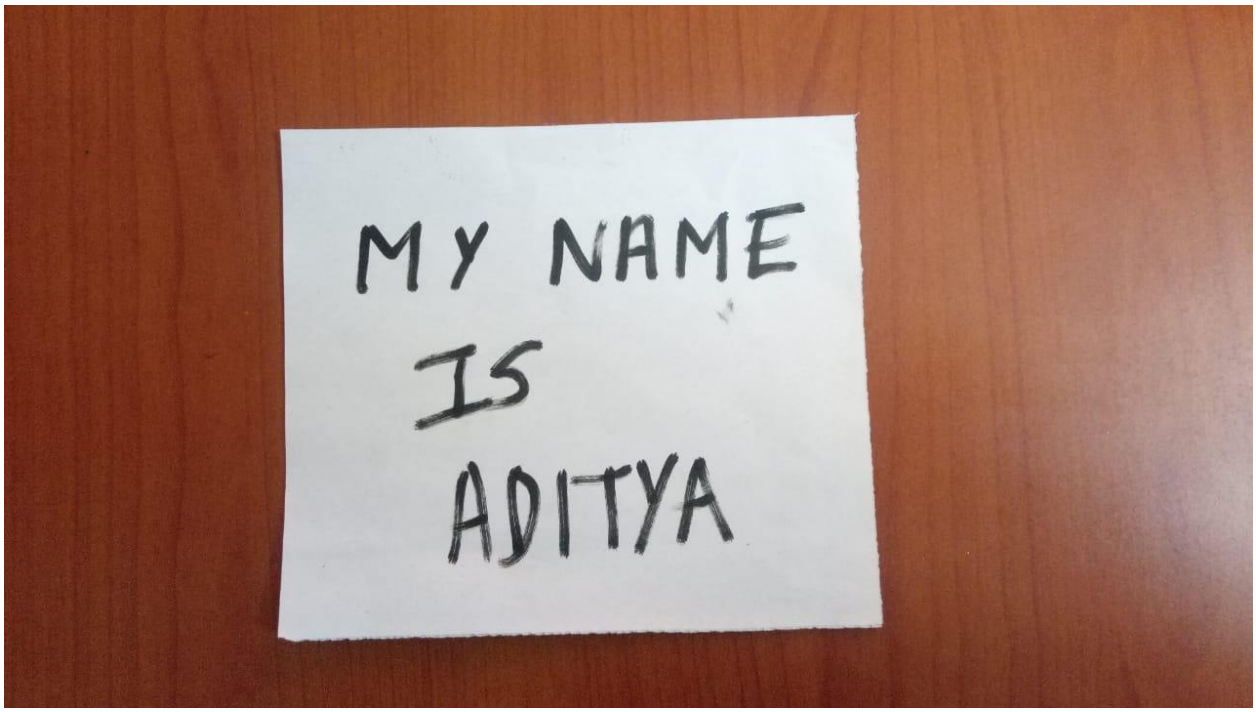
```
import cv2
import numpy as np
import mapper
image=cv2.imread("test_img.jpg")   #read in the image
image=cv2.resize(image,(1300,800)) #resizing because opencv does not work well with
bigger images
orig=image.copy()

gray=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)  #RGB To Gray Scale
cv2.imshow("Gray Scale",gray)

blurred=cv2.GaussianBlur(gray,(5,5),0)  #(5,5) is the kernel size and 0 is sigma that
determines the amount of blur
cv2.imshow("Blur",blurred)

edged=cv2.Canny(blurred,30,50)  #30 MinThreshold and 50 is the MaxThreshold
cv2.imshow("Canny",edged)


contours,hierarchy=cv2.findContours(edged,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMP
LE)  #retrieve the contours as a list, with simple approximation model
contours=sorted(contours,key=cv2.contourArea,reverse=True)

#the loop extracts the boundary contours of the page
for c in contours:
    p=cv2.arcLength(c,True)
    approx=cv2.approxPolyDP(c,0.02*p,True)

    if len(approx)==4:
        target=approx
        break
approx=mapper.mapp(target) #find endpoints of the sheet

pts=np.float32([[0,0],[800,0],[800,800],[0,800]])  #map to 800*800 target window

op=cv2.getPerspectiveTransform(approx,pts)  #get the top or bird eye view effect
dst=cv2.warpPerspective(orig,op,(800,800))
```

```
cv2.imshow("Scanned",dst)
# press q or Esc to close
cv2.waitKey(0)
cv2.destroyAllWindows()
```

mapper.py

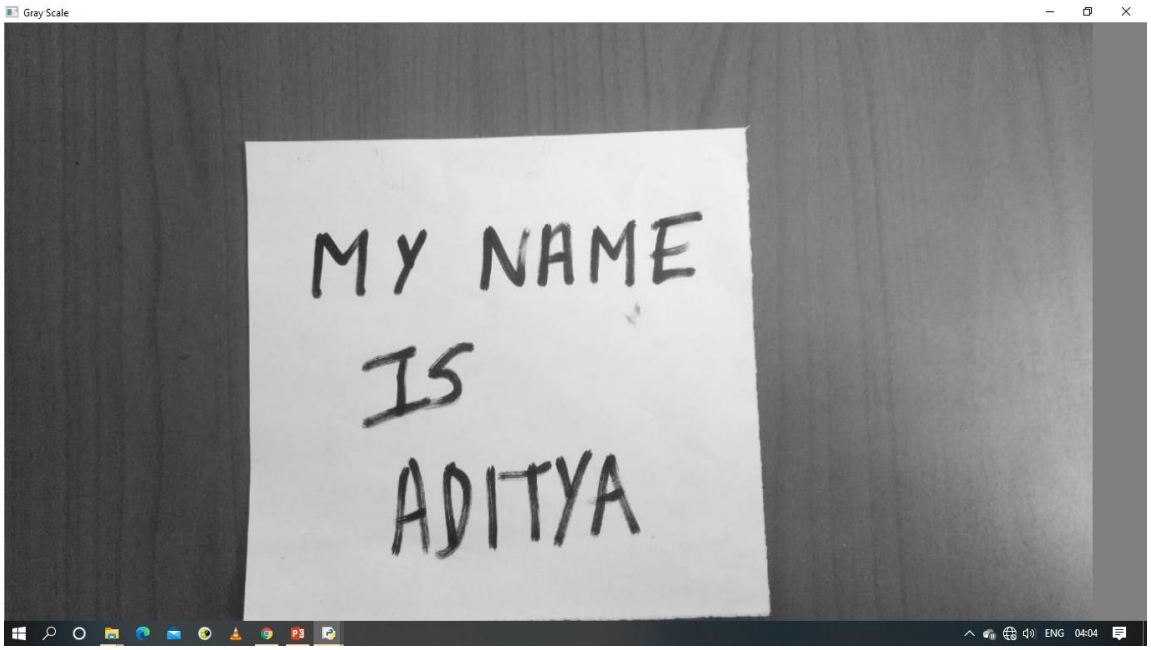
```
import numpy as np

def mapp(h):
    h = h.reshape((4,2))
    hnew = np.zeros((4,2),dtype = np.float32)

    add = h.sum(1)
    hnew[0] = h[np.argmin(add)]
    hnew[2] = h[np.argmax(add)]

    diff = np.diff(h,axis = 1)
    hnew[1] = h[np.argmin(diff)]
    hnew[3] = h[np.argmax(diff)]

    return hnew
```

Original Image

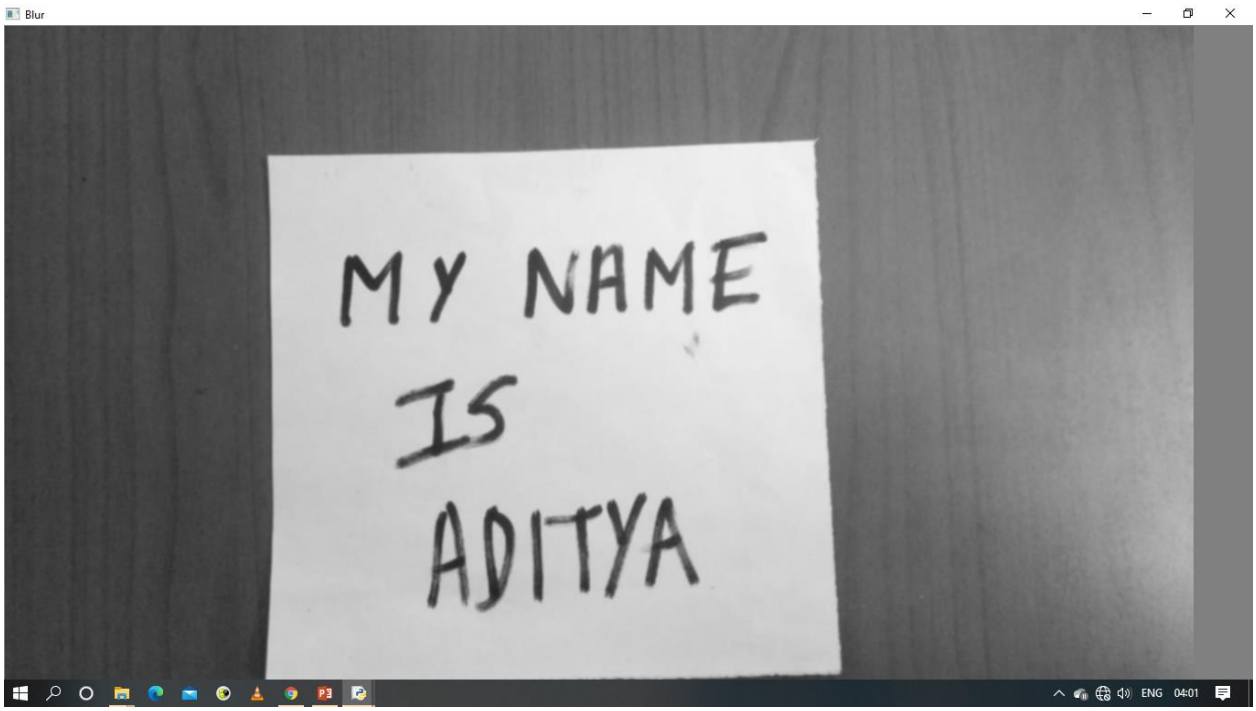


Output after running the Python Code for Scanner.py

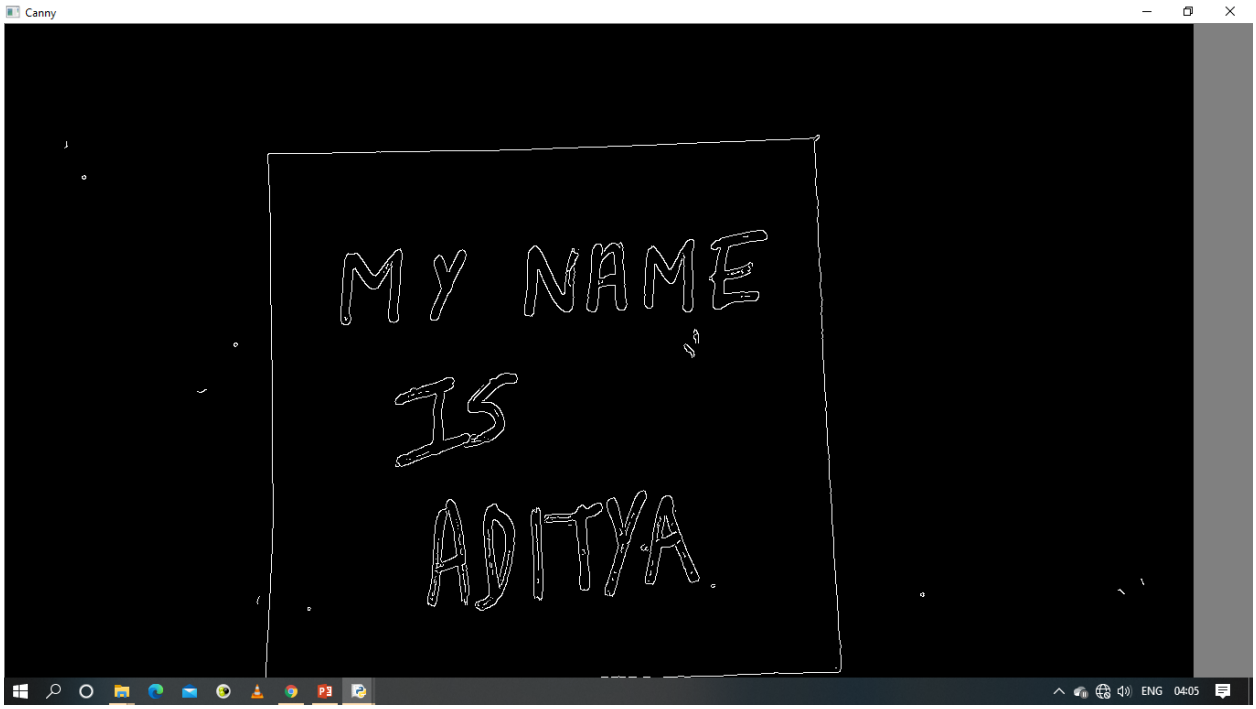
First we get the gray scale image entitled “Gray Scale” as output.



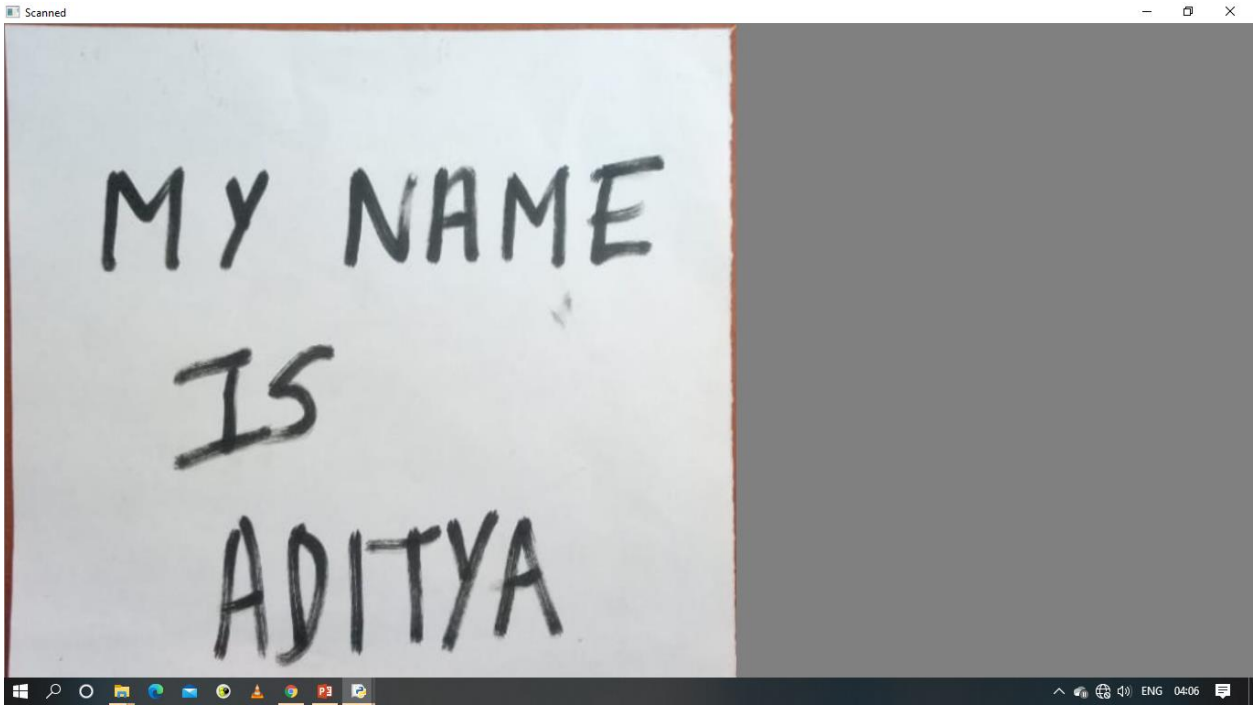
Secondly we get the Blurred Image entitled “Blur”.



Thirdly we get the Canny Detected Image entitled “Canny”.



Fourthly we get the Scanned Image entitled “Scanned” which is the final output.



10.Screen Recorder

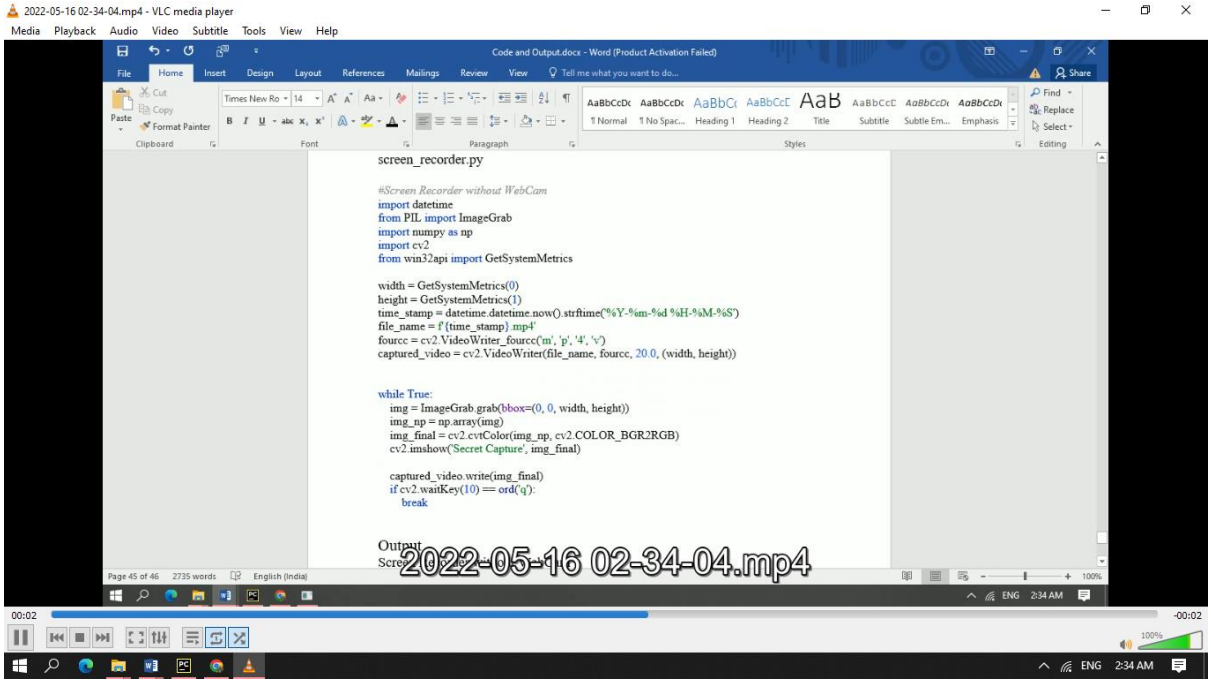
screen_recorder.py

```
#Screen Recorder without WebCam
import datetime
from PIL import ImageGrab
import numpy as np
import cv2
from win32api import GetSystemMetrics

width = GetSystemMetrics(0)
height = GetSystemMetrics(1)
time_stamp = datetime.datetime.now().strftime('%Y-%m-%d %H-%M-%S')
file_name = f'{time_stamp}.mp4'
fourcc = cv2.VideoWriter_fourcc('m', 'p', '4', 'v')
captured_video = cv2.VideoWriter(file_name, fourcc, 20.0, (width, height))


while True:
    img = ImageGrab.grab(bbox=(0, 0, width, height))
    img_np = np.array(img)
    img_final = cv2.cvtColor(img_np, cv2.COLOR_BGR2RGB)
    cv2.imshow('Secret Capture', img_final)

    captured_video.write(img_final)
    if cv2.waitKey(10) == ord('q'):
        break
```

Output

Screen Recorder without WebCam

Screen Recorder saved the video file with time stamp in .mp4 format



screenrecorderwithwebcam.py

```
#Screen Recorder With WebCam
import datetime
from PIL import ImageGrab
import numpy as np
import cv2
from win32api import GetSystemMetrics

width = GetSystemMetrics(0)
height = GetSystemMetrics(1)
time_stamp = datetime.datetime.now().strftime('%Y-%m-%d %H-%M-%S')
file_name = f'{time_stamp}.mp4'
fourcc = cv2.VideoWriter_fourcc('m', 'p', '4', 'v')
captured_video = cv2.VideoWriter(file_name, fourcc, 20.0, (width, height))

webcam = cv2.VideoCapture(0)

while True:
    img = ImageGrab.grab(bbox=(0, 0, width, height))
    img_np = np.array(img)
    img_final = cv2.cvtColor(img_np, cv2.COLOR_BGR2RGB)
    _, frame = webcam.read()
    fr_height, fr_width, _ = frame.shape
    img_final[0:fr_height, 0: fr_width, :] = frame[0: fr_height, 0: fr_width, :]
    cv2.imshow('Secret Capture', img_final)
```

```
    # cv2.imshow('webcam', frame)

    captured_video.write(img_final)
    if cv2.waitKey(10) == ord('q'):
        break
```

Output

Screen Recorder saved the video file with time stamp in .mp4 format