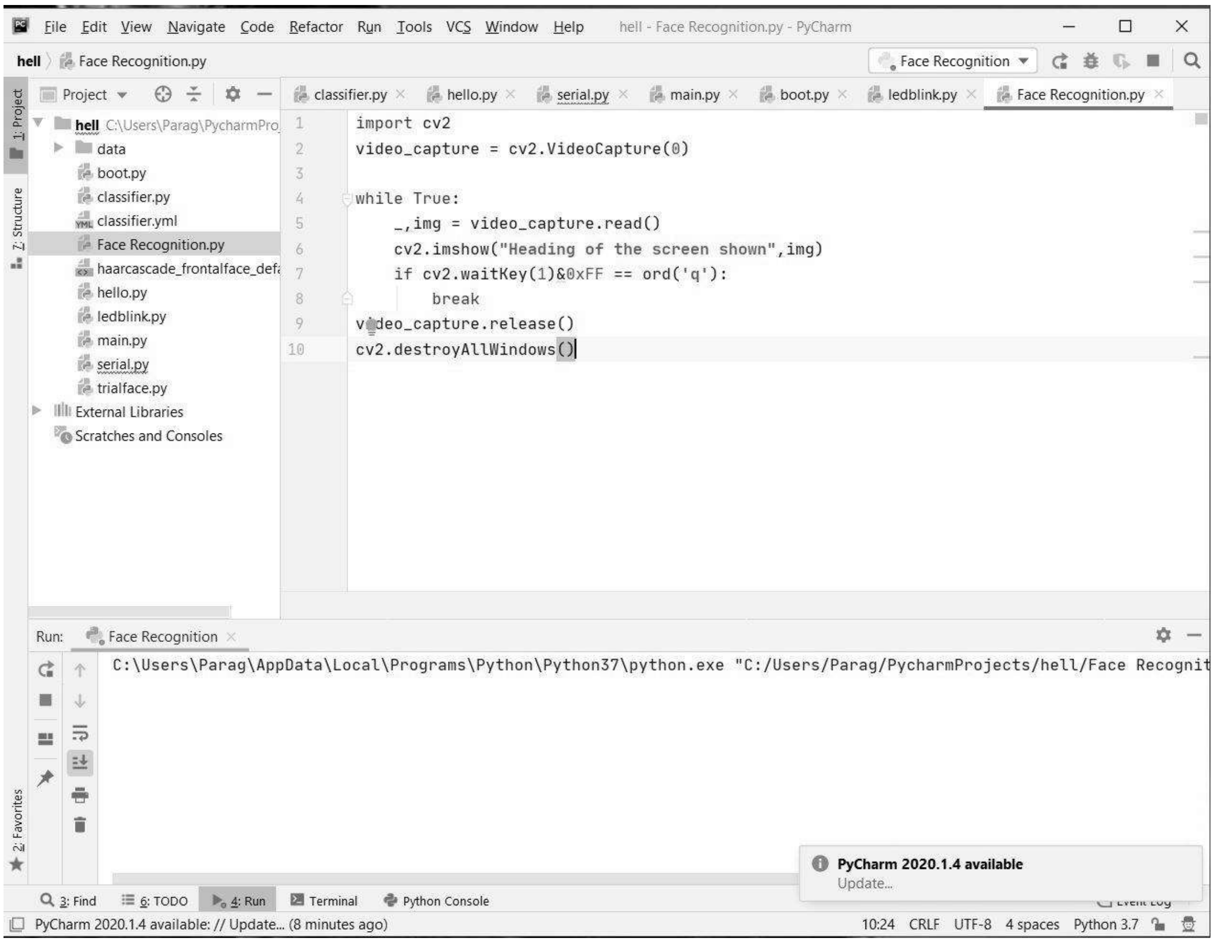
PROJECT EXECUTION

1) Download Python 3
2) Download PyCharm
3) Download Arduino IdE

Start writing the code in PyCharm

*_Steps to be performed for TOUCHLESS AUTHENTICATION SYSTEM_*

<u>Step 1: Reading live Webcam (built-in camera) video stream.</u>



```
import cv2
video_capture = cv2.VideoCapture(0)

while True:
    _,img = video_capture.read()
    cv2.imshow("Heading of the screen shown",img)
    if cv2.waitKey(1)&0xFF == ord('q'):
        break
video_capture.release()
cv2.destroyAllWindows()
```

```
C:\Users\Parag\AppData\Local\Programs\Python\Python37\python.exe "C:/Users/Parag/PycharmProjects/hell/Face Recognit
```

Fig 2: Step1 program code for reading live video stream.

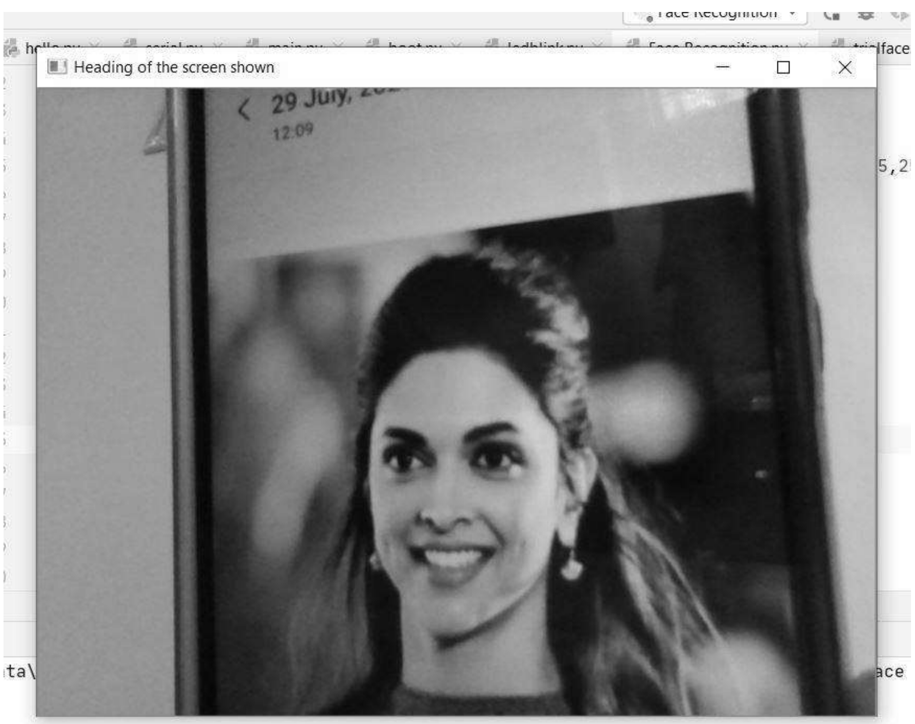
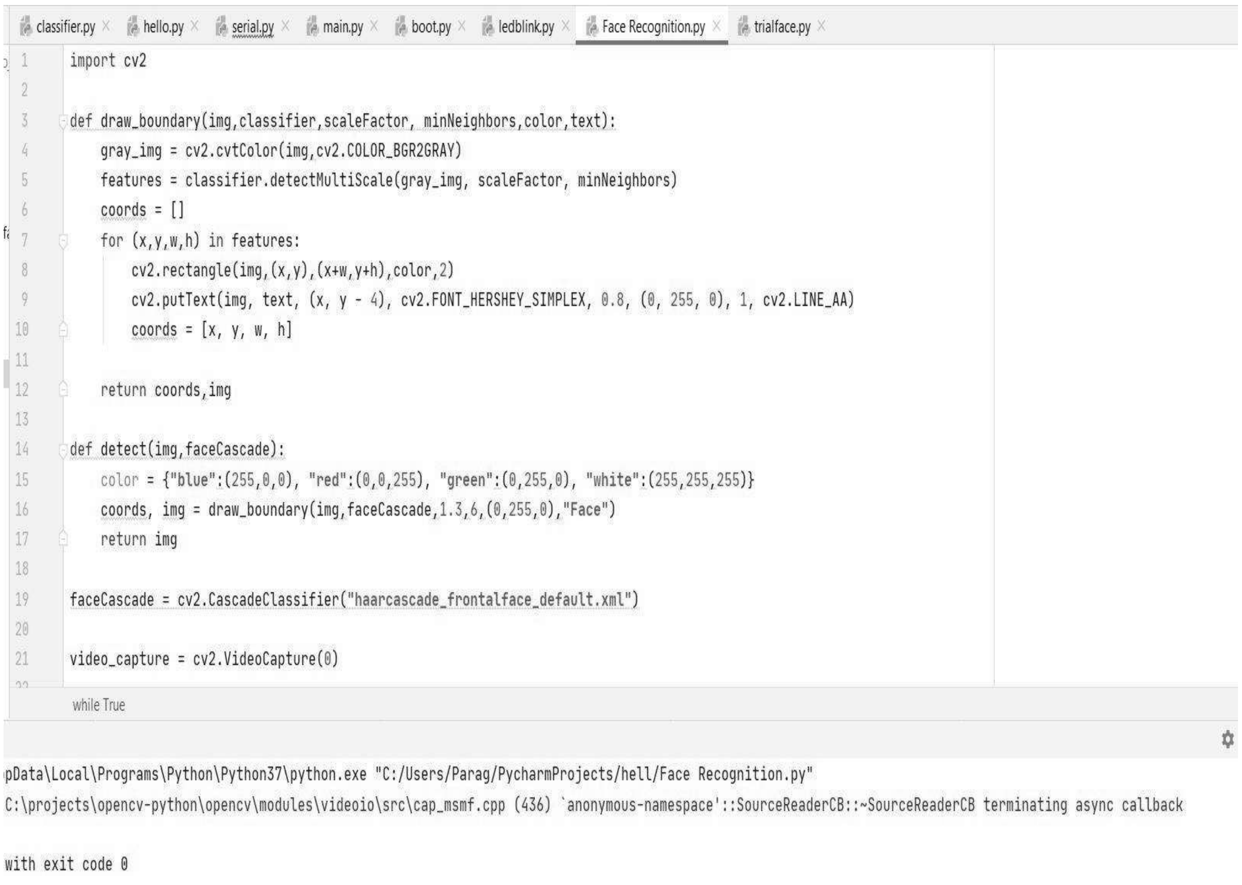


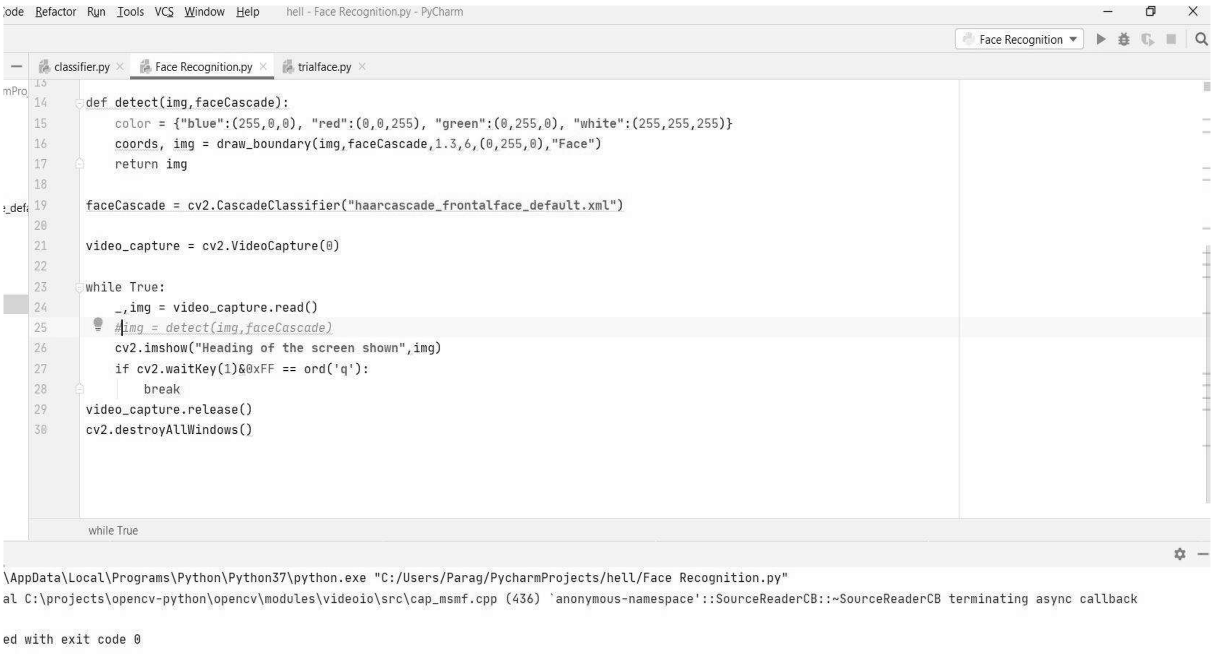
Fig 3: Step1 Output after running program code for reading live video stream.

<u>Step 2: Face Detection using Haar Cascade</u>



```
import cv2

def draw_boundary(img,classifier,scaleFactor, minNeighbors,color,text):
    gray_img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    features = classifier.detectMultiScale(gray_img, scaleFactor, minNeighbors)
    coords = []
    for (x,y,w,h) in features:
        cv2.rectangle(img,(x,y),(x+w,y+h),color,2)
        cv2.putText(img, text, (x, y - 4), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 0), 1, cv2.LINE_AA)
        coords = [x, y, w, h]

    return coords,img

def detect(img,faceCascade):
    color = {"blue":(255,0,0), "red":(0,0,255), "green":(0,255,0), "white":(255,255,255)}
    coords, img = draw_boundary(img,faceCascade,1.3,6,(0,255,0),"Face")
    return img

faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

video_capture = cv2.VideoCapture(0)
```

```
pData\Local\Programs\Python\Python37\python.exe "C:/Users/Parag/PycharmProjects/hell/Face Recognition.py"
C:\projects\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (436) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async callback

with exit code 0
```

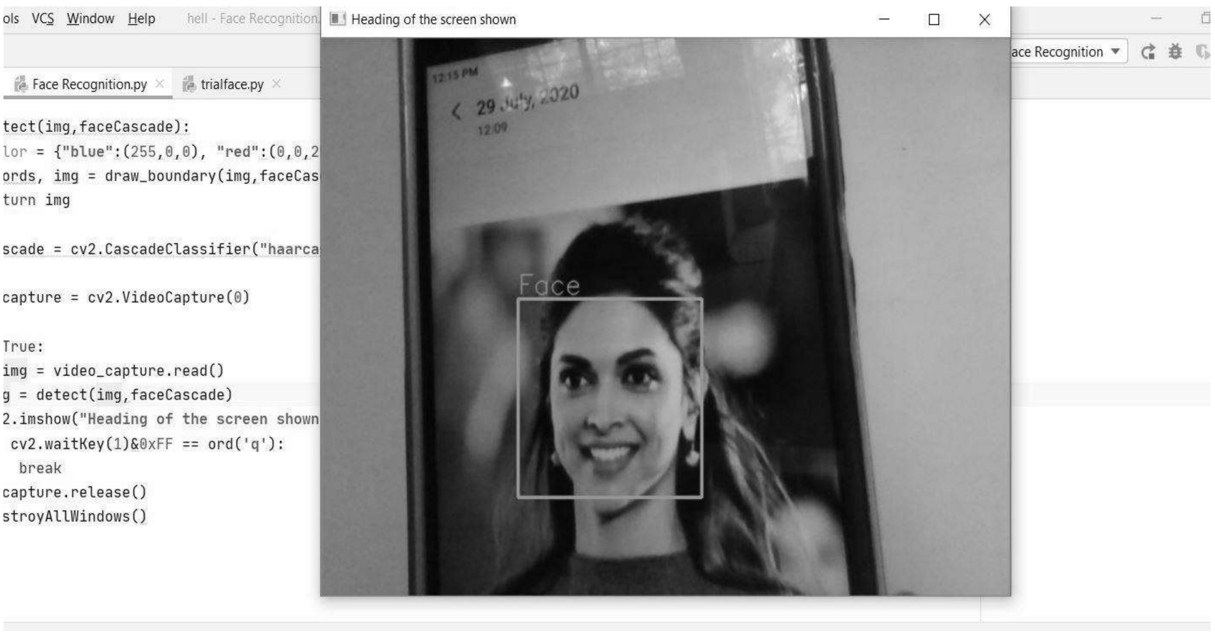


ode Refactor Run Tools VCS Window Help hell - Face Recognition.py - PyCharm
Face Recognition
classifier.py
Face Recognition.py
trialface.py
while True

```
def detect(img,faceCascade):
    color = {"blue":(255,0,0), "red":(0,0,255), "green":(0,255,0), "white":(255,255,255)}
    coords, img = draw_boundary(img,faceCascade,1.3,6,(0,255,0),"Face")
    return img

faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

video_capture = cv2.VideoCapture(0)

while True:
    _,img = video_capture.read()
    #img = detect(img,faceCascade)
    cv2.imshow("Heading of the screen shown",img)
    if cv2.waitKey(1)&0xFF == ord('q'):
        break
video_capture.release()
cv2.destroyAllWindows()
```

```
\AppData\Local\Programs\Python\Python37\python.exe "C:/Users/Parag/PycharmProjects/hell/Face Recognition.py"
al C:\projects\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (436) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async callback

ed with exit code 0
```

Fig 4 a&b: Step2 Program code for Face Detection using Haar Cascade.



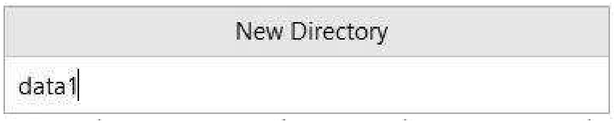
```
tect(img,faceCascade):
lor = {"blue":(255,0,0), "red":(0,0,2
ords, img = draw_boundary(img,faceCas
turn img

scade = cv2.CascadeClassifier("haarca

capture = cv2.VideoCapture(0)

True:
img = video_capture.read()
g = detect(img,faceCascade)
2.imshow("Heading of the screen shown
 cv2.waitKey(1)&0xFF == ord('q'):
  break
capture.release()
stroyAllWindows()
```

Fig 5: Step2 Output after running program code for Face Detection using Haar Cascade.

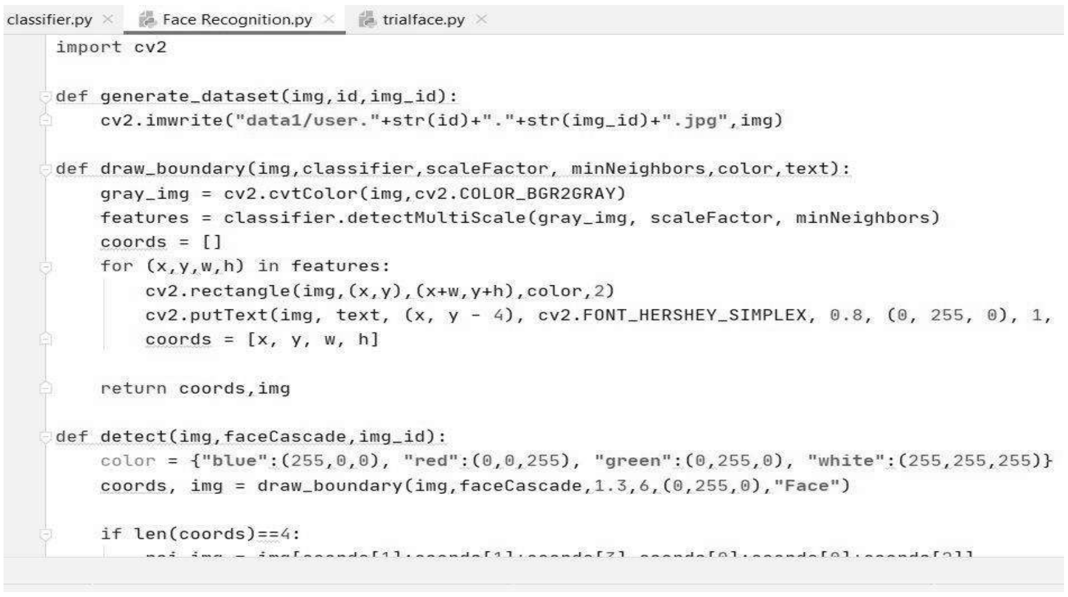
Step 3: Generating Dataset to Train Classifier

First you need to create a directory called ad data1 to store the captured photos of a person or employee.

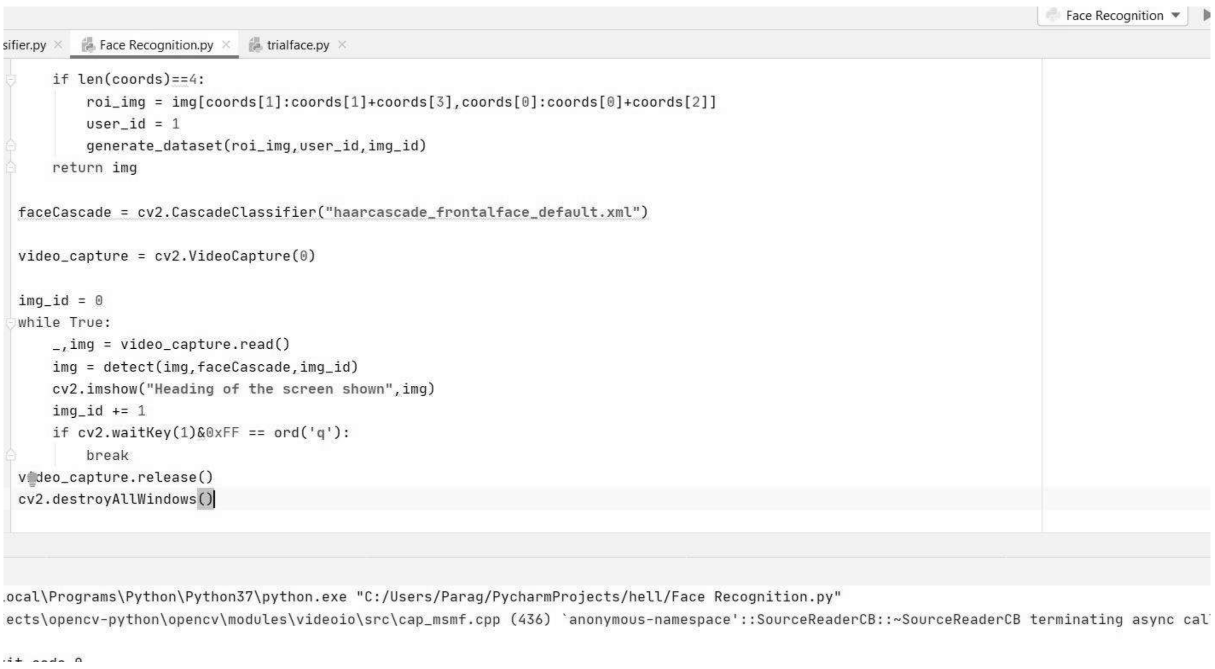
Create a new file in that you will see director option create a directory.



After creating directory start writing code in PyCharm



```python
import cv2

def generate_dataset(img,id,img_id):
    cv2.imwrite("data1/user."+str(id)+"."+str(img_id)+".jpg",img)

def draw_boundary(img,classifier,scaleFactor, minNeighbors,color,text):
    gray_img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    features = classifier.detectMultiScale(gray_img, scaleFactor, minNeighbors)
    coords = []
    for (x,y,w,h) in features:
        cv2.rectangle(img,(x,y),(x+w,y+h),color,2)
        cv2.putText(img, text, (x, y - 4), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 0), 1,
        coords = [x, y, w, h]

    return coords,img

def detect(img,faceCascade,img_id):
    color = {"blue":(255,0,0), "red":(0,0,255), "green":(0,255,0), "white":(255,255,255)}
    coords, img = draw_boundary(img,faceCascade,1.3,6,(0,255,0),"Face")

    if len(coords)==4:
```



```python
    if len(coords)==4:
        roi_img = img[coords[1]:coords[1]+coords[3],coords[0]:coords[0]+coords[2]]
        user_id = 1
        generate_dataset(roi_img,user_id,img_id)
    return img

faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

video_capture = cv2.VideoCapture(0)

img_id = 0
while True:
    _,img = video_capture.read()
    img = detect(img,faceCascade,img_id)
    cv2.imshow("Heading of the screen shown",img)
    img_id += 1
    if cv2.waitKey(1)&0xFF == ord('q'):
        break
video_capture.release()
cv2.destroyAllWindows()
```

```
.ocal\Programs\Python\Python37\python.exe "C:/Users/Parag/PycharmProjects/hell/Face Recognition.py"
ects\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (436) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async cal
```

Fig 6 a&b : Step3 Program code for generating dataset to train classifier.

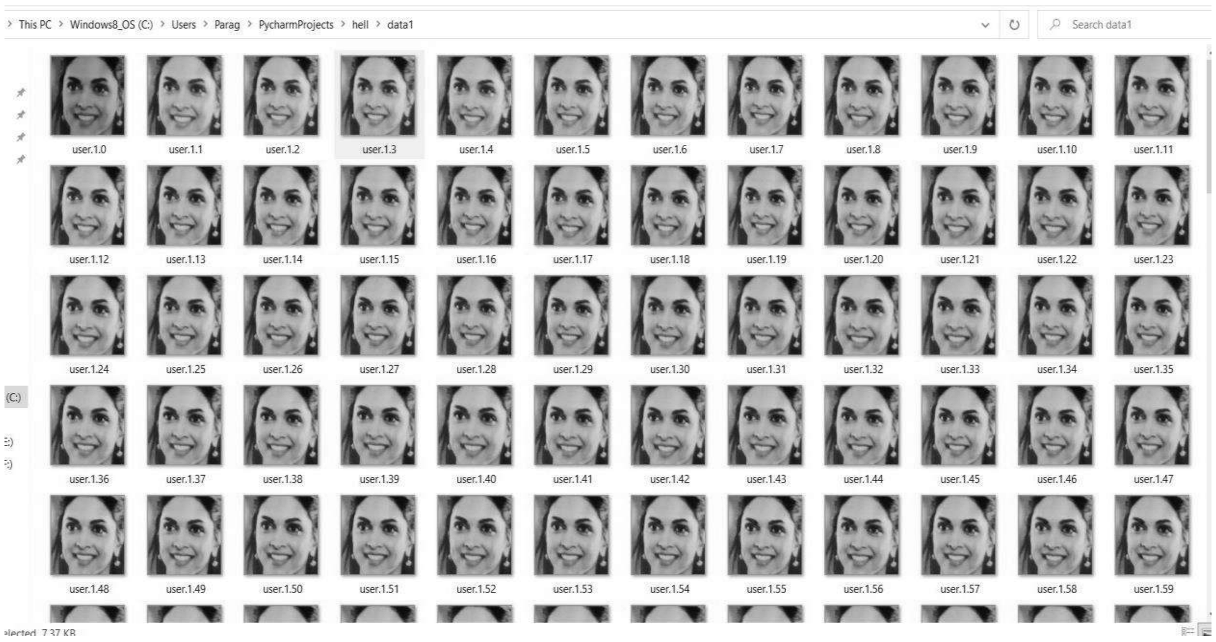

This PC > Windows8_OS (C:) > Users > Parag > PycharmProjects > hell > data1
Search data1
user.1.0 user.1.1 user.1.2 user.1.3 user.1.4 user.1.5 user.1.6 user.1.7 user.1.8 user.1.9 user.1.10 user.1.11
user.1.12 user.1.13 user.1.14 user.1.15 user.1.16 user.1.17 user.1.18 user.1.19 user.1.20 user.1.21 user.1.22 user.1.23
user.1.24 user.1.25 user.1.26 user.1.27 user.1.28 user.1.29 user.1.30 user.1.31 user.1.32 user.1.33 user.1.34 user.1.35
user.1.36 user.1.37 user.1.38 user.1.39 user.1.40 user.1.41 user.1.42 user.1.43 user.1.44 user.1.45 user.1.46 user.1.47
user.1.48 user.1.49 user.1.50 user.1.51 user.1.52 user.1.53 user.1.54 user.1.55 user.1.56 user.1.57 user.1.58 user.1.59
elected 7.37 KB

Fig 7: Step3 Output after running Program code for generating dataset to train classifier.

NOTE: Change the user id to next number so as to train as in this case I have changes the user id to 2 and executed the program. You can see the dataset is filled with other persons photograph.

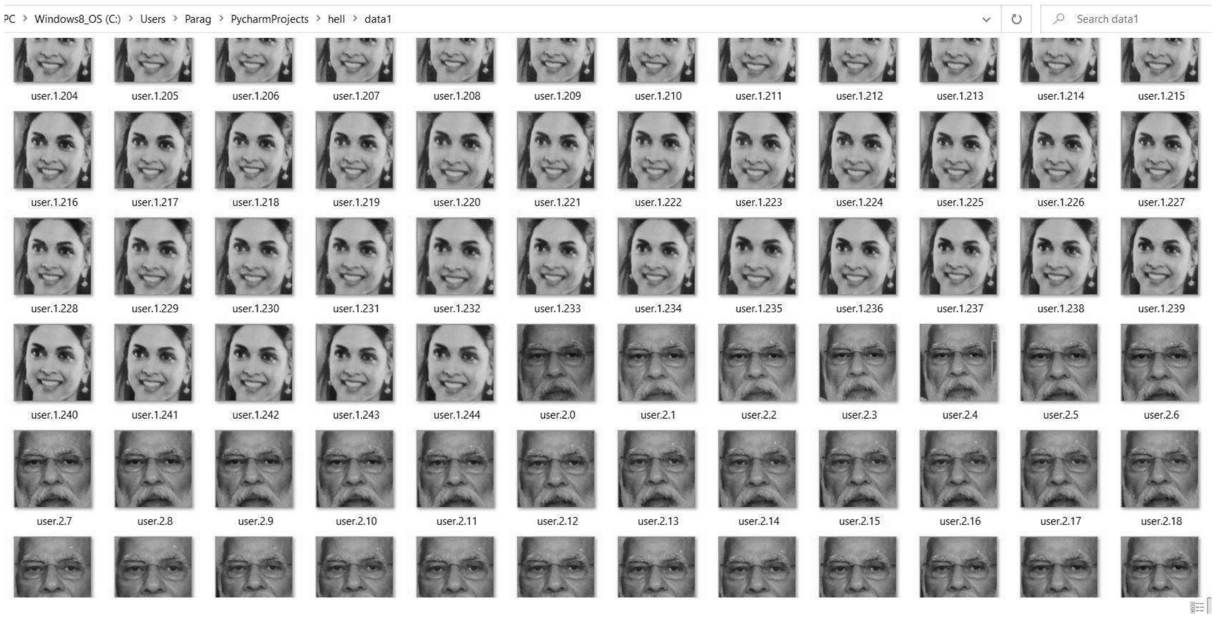
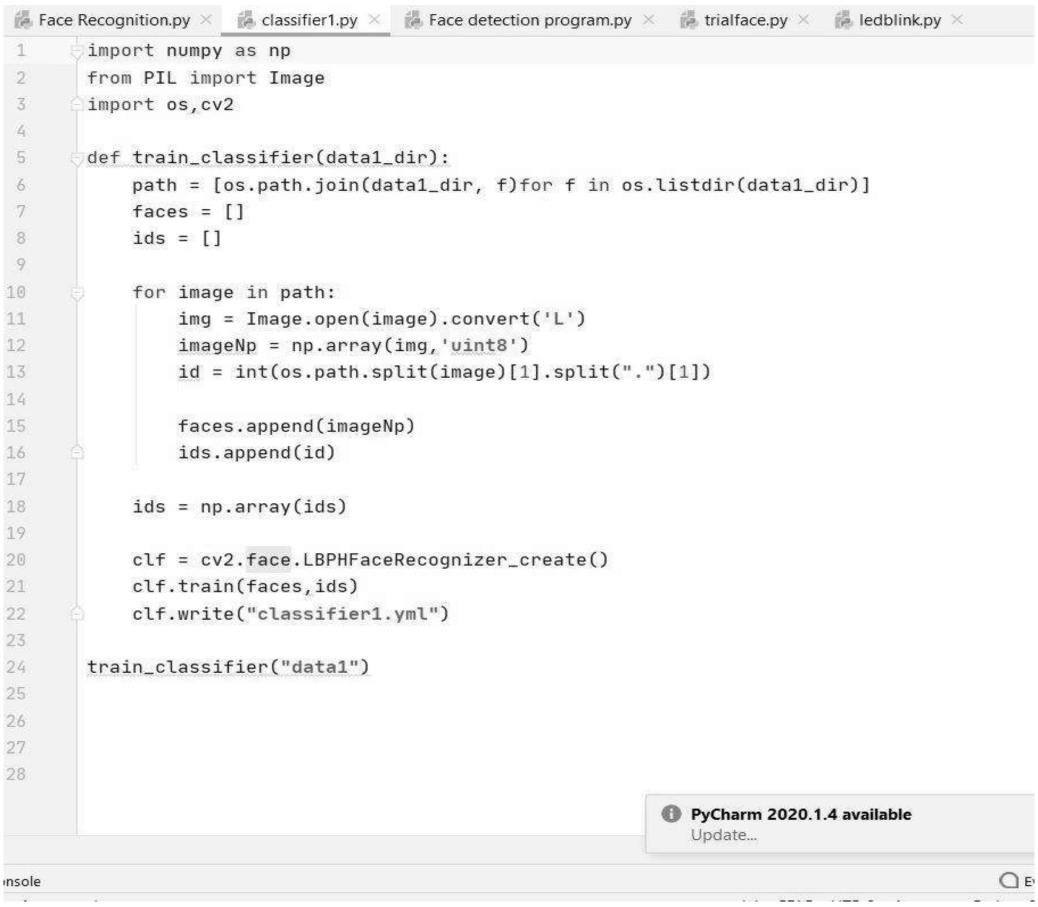

PC > Windows8_OS (C:) > Users > Parag > PycharmProjects > hell > data1
Search data1
user.1.204 user.1.205 user.1.206 user.1.207 user.1.208 user.1.209 user.1.210 user.1.211 user.1.212 user.1.213 user.1.214 user.1.215
user.1.216 user.1.217 user.1.218 user.1.219 user.1.220 user.1.221 user.1.222 user.1.223 user.1.224 user.1.225 user.1.226 user.1.227
user.1.228 user.1.229 user.1.230 user.1.231 user.1.232 user.1.233 user.1.234 user.1.235 user.1.236 user.1.237 user.1.238 user.1.239
user.1.240 user.1.241 user.1.242 user.1.243 user.1.244 user.2.0 user.2.1 user.2.2 user.2.3 user.2.4 user.2.5 user.2.6
user.2.7 user.2.8 user.2.9 user.2.10 user.2.11 user.2.12 user.2.13 user.2.14 user.2.15 user.2.16 user.2.17 user.2.18

Fig 8: Step3 Output after changing user id = 2 , & running Program code for generating dataset to train classifier.

Step 4:Training Classifier to Recognize a person.

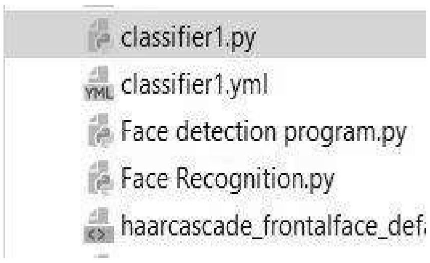
Start writing a new python program in PyCharm name it as classifier1py.



```python
import numpy as np
from PIL import Image
import os,cv2

def train_classifier(data1_dir):
    path = [os.path.join(data1_dir, f)for f in os.listdir(data1_dir)]
    faces = []
    ids = []

    for image in path:
        img = Image.open(image).convert('L')
        imageNp = np.array(img,'uint8')
        id = int(os.path.split(image)[1].split(".")[1])

        faces.append(imageNp)
        ids.append(id)

    ids = np.array(ids)

    clf = cv2.face.LBPHFaceRecognizer_create()
    clf.train(faces,ids)
    clf.write("classifier1.yml")

train_classifier("data1")
```

Fig 9: Step4 Program code for training classifier to recognize a person.

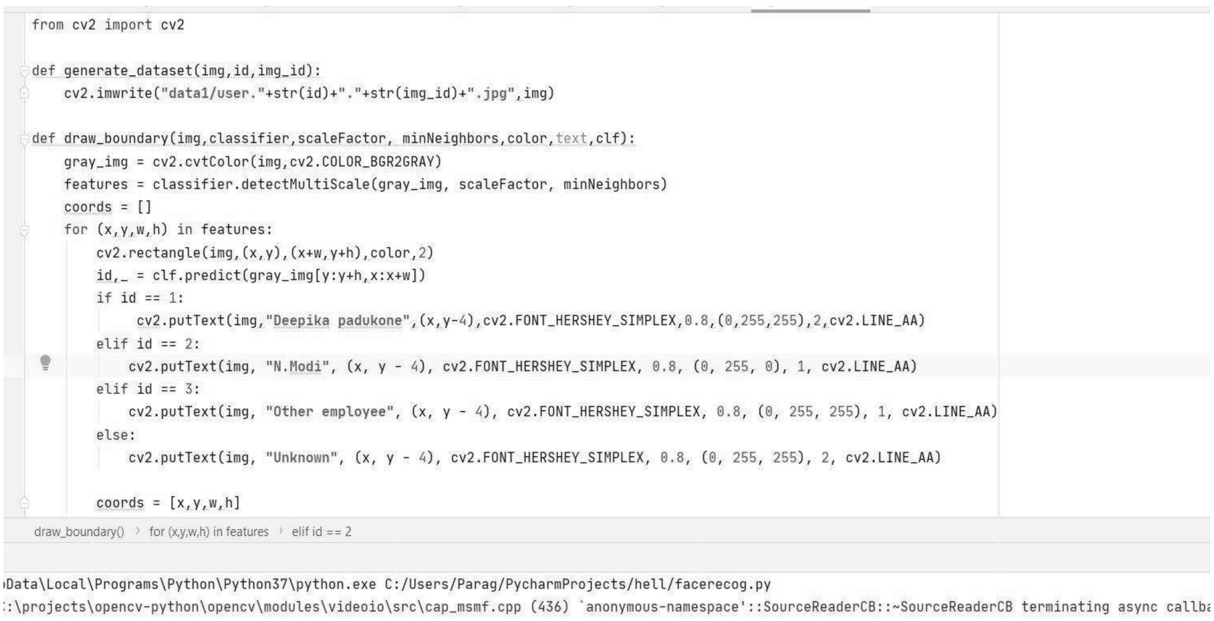
After you finish executing the classifier1.py file you will see another file named Classifier1.yml file.



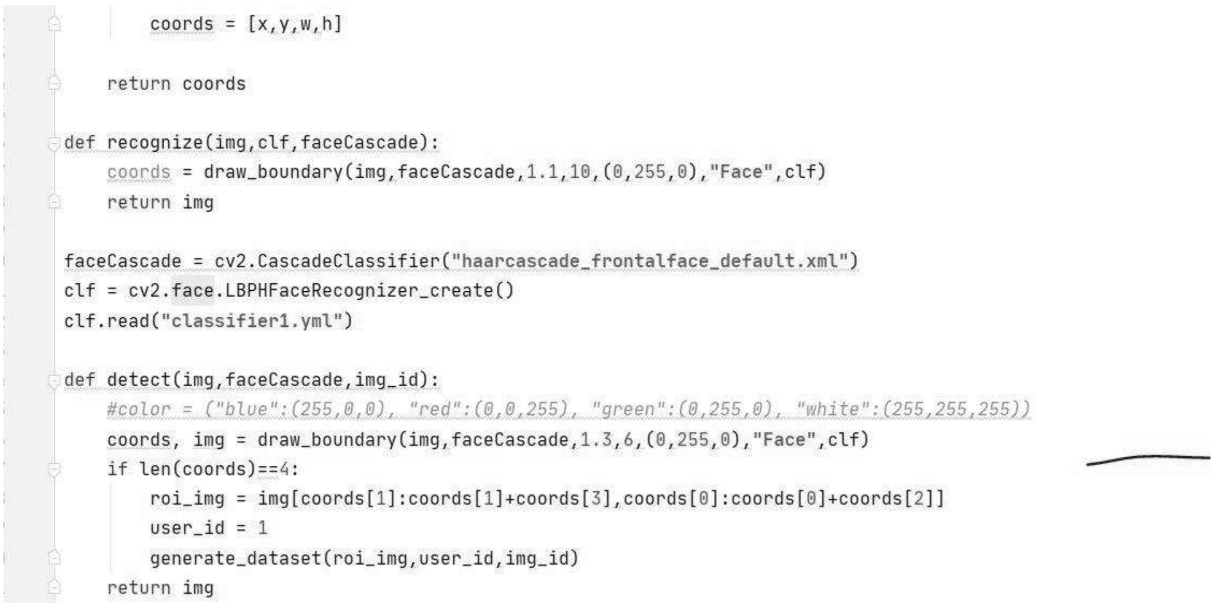
```
classifier1.py
classifier1.yml
Face detection program.py
Face Recognition.py
haarcascade_frontalface_def
```

Fig 10: Step4 Output after running Program code for training classifier to recognize a person.

Step 5:Recognizing a Person by Face.


```
from cv2 import cv2

def generate_dataset(img,id,img_id):
    cv2.imwrite("data1/user."+str(id)+"."+str(img_id)+".jpg",img)

def draw_boundary(img,classifier,scaleFactor, minNeighbors,color,text,clf):
    gray_img = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    features = classifier.detectMultiScale(gray_img, scaleFactor, minNeighbors)
    coords = []
    for (x,y,w,h) in features:
        cv2.rectangle(img,(x,y),(x+w,y+h),color,2)
        id,_ = clf.predict(gray_img[y:y+h,x:x+w])
        if id == 1:
             cv2.putText(img,"Deepika padukone",(x,y-4),cv2.FONT_HERSHEY_SIMPLEX,0.8,(0,255,255),2,cv2.LINE_AA)
        elif id == 2:
            cv2.putText(img, "N.Modi", (x, y - 4), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 0), 1, cv2.LINE_AA)
        elif id == 3:
            cv2.putText(img, "Other employee", (x, y - 4), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 255), 1, cv2.LINE_AA)
        else:
            cv2.putText(img, "Unknown", (x, y - 4), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 255, 255), 2, cv2.LINE_AA)

        coords = [x,y,w,h]
```

```
)Data\Local\Programs\Python\Python37\python.exe C:/Users/Parag/PycharmProjects/hell/facerecog.py
:\projects\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (436) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async callba
```


```
        coords = [x,y,w,h]

    return coords

def recognize(img,clf,faceCascade):
    coords = draw_boundary(img,faceCascade,1.1,10,(0,255,0),"Face",clf)
    return img

faceCascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
clf = cv2.face.LBPHFaceRecognizer_create()
clf.read("classifier1.yml")

def detect(img,faceCascade,img_id):
    #color = ("blue":(255,0,0), "red":(0,0,255), "green":(0,255,0), "white":(255,255,255))
    coords, img = draw_boundary(img,faceCascade,1.3,6,(0,255,0),"Face",clf)
    if len(coords)==4:
        roi_img = img[coords[1]:coords[1]+coords[3],coords[0]:coords[0]+coords[2]]
        user_id = 1
        generate_dataset(roi_img,user_id,img_id)
    return img
```