```
import cv2
import os
import numpy as np
from PIL import Image
import pickle

BASE_DIR = os.path.dirname(os.path.abspath(__file__))
image_dir = os.path.join(BASE_DIR, "images")

face_cascade = cv2.CascadeClassifier('cascades/data/haarcascade_frontalface_alt2.xml')
recognizer = cv2.face.LBPHFaceRecognizer_create()

current_id = 0
label_ids = {}
y_labels = []
x_train = []

for root, dirs, files in os.walk(image_dir):
	for file in files:
		if file.endswith("png") or file.endswith("jpg"):
			path = os.path.join(root, file)
			label = os.path.basename(root).replace(" ", "-").lower()
			if not label in label_ids:
				label_ids[label] = current_id
				current_id += 1
			id_ = label_ids[label]

			pil_image = Image.open(path).convert("L")
			size = (550, 550)
			final_image = pil_image.resize(size, Image.ANTIALIAS)
			image_array = np.array(final_image, "uint8")

			faces = face_cascade.detectMultiScale(image_array, scaleFactor=1.5,
minNeighbors=5)

			for (x,y,w,h) in faces:
				roi = image_array[y:y+h, x:x+w]
				x_train.append(roi)
				y_labels.append(id_)


with open("pickles/face-labels.pickle", 'wb') as f:
	pickle.dump(label_ids, f)

recognizer.train(x_train, np.array(y_labels))
recognizer.save("trainner.yml")
```

```
face_cascade = cv2.CascadeClassifier('cascades/data/haarcascade_frontalface_alt2.xml')
eye_cascade = cv2.CascadeClassifier('cascades/data/haarcascade_eye.xml')
smile_cascade = cv2.CascadeClassifier('cascades/data/haarcascade_smile.xml')
```

```
recognizer = cv2.face.LBPHFaceRecognizer_create()
recognizer.read("trainner.yml")

labels = {"person_name": 1}
with open("pickles/face-labels.pickle", 'rb') as f:
          og_labels = pickle.load(f)
          labels = {v:k for k,v in og_labels.items()}

cap = cv2.VideoCapture(0)

while(True):
    ret, frame = cap.read()
    gray  = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, scaleFactor=1.5, minNeighbors=5)
    for (x, y, w, h) in faces:

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


     id_, conf = recognizer.predict(roi_gray)
     if conf>=4 and conf <= 85:

      font = cv2.FONT_HERSHEY_SIMPLEX
      name = labels[id_]
      color = (255, 255, 255)
      stroke = 2
      cv2.putText(frame, name, (x,y), font, 1, color, stroke, cv2.LINE_AA)

     img_item = "7.png"
     cv2.imwrite(img_item, roi_color)

     color = (255, 0, 0)
     stroke = 2
     end_cord_x = x + w
     end_cord_y = y + h
     cv2.rectangle(frame, (x, y), (end_cord_x, end_cord_y), color, stroke)

    cv2.imshow('frame',frame)
    if cv2.waitKey(20) & 0xFF == ord('q'):
        break


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