

Face Recognition Project Report

As a project work for Course

PYTHON PROGRAMMING (INT 213)

Submitted by:

Name	Registration Number	Roll Number	Semester/ Course code
Marthala Satya Anush Kumar Reddy	12014518	RK20BGB66	3 rd /INT213
P Kiran Kumar Reddy	12014385	RK20BGB68	3 rd /INT213

Submitted to:

Dr. Sagar Pande Sir



L OVELY
P ROFESSIONAL
U NIVERSITY

Lovely Professional University,
Jalandhar, Punjab(India).

Table Of Contents

1.Abstract	3
2.Introduction	4
3. Team members with roles	5
4.Modules	6-9
5.Working of code	10
6.Output of code	11
7.Source Code	12-13
8.Conclusion	14
9.References	14-15

Abstract:

The primary objective of this project is to implement what we've have learnt throughout our course of Python programming and use that to develop a Face Recognition with all required functionalities. Face recognition is a method of identifying or verifying the identity of an individual using their face. The human face is central to our identity. It plays an essential role in day-to-day interactions, communication and other routine chores. To build fully automated systems that analyse the information contained in face images, resilient and impeccable face detection algorithms are required, and many algorithms are in use. Partial face occlusion is one of the most challenging problems in face recognition. A face recognition system can confront faces in the real-world applications occluded with the use of accessories, such as mask, scarf or shades, hands on the face, the objects a person carries, and the external sources.

ACKNOWLEDGEMENT:-

I would like to thank my python sir – Dr. Sagar Pande Sir for giving opportunity to prove our talent on this project. Many thanks to my friends and seniors as well, who spent countless hours to listen and provide feedbacks.

INTRODUCTION:

To recognise a face, it is first important that we detect/locate a face in an image/video. There are various facial detection softwares that can detect a Human face in an image. We extract a human face and then move on to the next step.

The next step is to extract features from a face using a face embedding model. A face embedding is a vector that represents the features extracted from the face and we can use these vectors to recognise faces. Note that face embedding for the same face may be really close in the vector space, Whereas the face embedding of two different faces may be really far away. We get a face embedding after passing the image through a face embedding model.

We have face embedding for each face in the system. Whenever we pass a new face to the system, it calculates its face embedding and compares it with the ones we already have. The face recognised, if its face embedding closely matches any other face embedding in the database.

Team Members:

Team Leaders:-

1. Marthala Satya Anush Kumar Reddy

Contributions:

1. Coding(Joined)
2. OpenCV(joined)

2. P. Kiran Kumar Reddy

Contributions:

1. Coding(Joined)
2. OpenCV(joined)

Modules:

- **OpenCV**
- **Opencv-contrib**
- **NumPy**
- **Face_recognition**

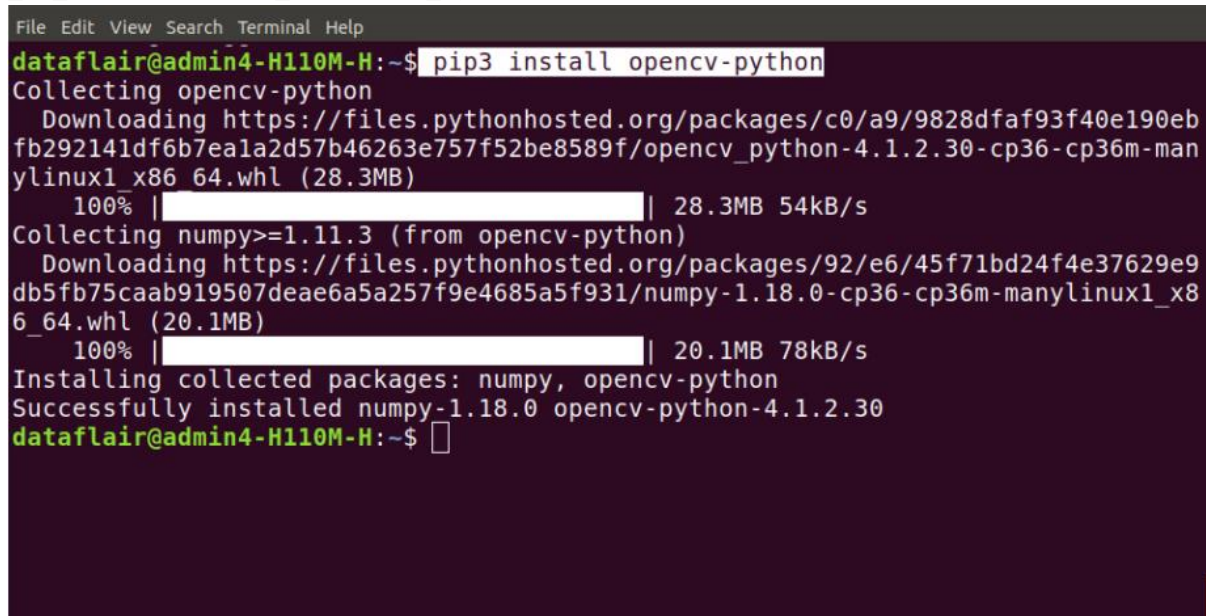
The real advantage of Python is that we don't need to write every single detail for our use, we can just import the required modules and modify them accordingly. For mine Face recognition project, we used these modules.

1.OpenCV:

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as NumPy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e. whatever operations one can do in NumPy can be combined with OpenCV.

This OpenCV tutorial will help you learn the Image-processing from Basics to Advance, like operations on Images, Videos using a huge set of OpenCV-programs and projects.

pip install opencv-python in the terminal.

A terminal window with a dark background and light-colored text. The prompt is 'dataflair@admin4-H110M-H:~\$'. The command 'pip3 install opencv-python' is entered. The output shows the collection of 'opencv-python' and 'numpy' packages, their download progress (100%), and the final installation status: 'Successfully installed numpy-1.18.0 opencv-python-4.1.2.30'.

```
File Edit View Search Terminal Help
dataflair@admin4-H110M-H:~$ pip3 install opencv-python
Collecting opencv-python
  Downloading https://files.pythonhosted.org/packages/c0/a9/9828dfaf93f40e190ebfb292141df6b7eala2d57b46263e757f52be8589f/opencv_python-4.1.2.30-cp36-cp36m-manylinux1_x86_64.whl (28.3MB)
    100% |████████████████████████████████████████| 28.3MB 54kB/s
Collecting numpy>=1.11.3 (from opencv-python)
  Downloading https://files.pythonhosted.org/packages/92/e6/45f71bd24f4e37629e9db5fb75caab919507deae6a5a257f9e4685a5f931/numpy-1.18.0-cp36-cp36m-manylinux1_x86_64.whl (20.1MB)
    100% |████████████████████████████████████████| 20.1MB 78kB/s
Installing collected packages: numpy, opencv-python
Successfully installed numpy-1.18.0 opencv-python-4.1.2.30
dataflair@admin4-H110M-H:~$
```

2.OpenCV-contrib:

OpenCV contrib is a specialized module present in the Python programming language, which is exclusively needed for the system to run SURF feature descriptions alongside the OpenCV module present in the open-source library. It must be noted that deserve algorithm is subjected to be encryption with copy rights, and hence it should only be used for learning and development processes or personal use and should not be used for any production purposes unless the copyright status of the module is completely understood and addressed by the user. OpenCV provides for pre-compiled Java jars and all files on their website.

However, if you use the pre-bundled OpenCV jar files and DLL files, you will get errors when working through the programs that are developed around the OpenCV tutorials. This is especially seen when dealing with the org.feature packages that will not be found and will prevent the quote from being compiled.

pip install opencv-contrib-python in the terminal

```
(cv) isla-nublar:~ adrianrosebrock$ pip install opencv-contrib-python
Collecting opencv-contrib-python
  Downloading https://files.pythonhosted.org/packages/ea/7a/e9503b1cfb68335b9371f52e330f79478bd7b46c1c5aee72bf0e0e1e1be3/opencv_contrib_python-3.4.3.18-cp37-cp37m-macosx_10_6_x86_64.macosx_10_9_intel.macosx_10_9_x86_64.macosx_10_10_intel.macosx_10_10_x86_64.whl (47.9MB)
    100% |#####| 48.0MB 1.3MB/s
Requirement already satisfied: numpy>=1.14.5 in ./virtualenvs/cv/lib/python3.7/site-packages (from opencv-contrib-python) (1.15.1)
Installing collected packages: opencv-contrib-python
Successfully installed opencv-contrib-python-3.4.3.18
(cv) isla-nublar:~ adrianrosebrock$ python
Python 3.7.0 (default, Jun 29 2018, 20:13:13)
[Clang 9.1.0 (clang-902.0.39.2)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.4.3'
>>>
```

3.NumPy:

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.

In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

pip install numpy in the terminal


```
Microsoft Windows [Version 10.0.18362.535]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\acer>pip install numpy
Collecting numpy
  Downloading https://files.pythonhosted.org/packages/ce/ad/2e88f36b56f64f70c081b32fa5512dacedf12005ccb0c2d300d44dcc121/numpy-1.17.4-cp37-cp37m-win32.whl (10.7MB)
    | 10.7MB 467kB/s
Installing collected packages: numpy
Successfully installed numpy-1.17.4

C:\Users\acer>
```

4.Face_recognition:

The face_recognition library, created by Adam Geitgey, wraps around dlib's facial recognition functionality, and this library is super easy to work with and we will be using this in our code.

Remember to install dlib library first before you install face_recognition.

```
C:\Windows\System32\cmd.exe
C:\>pip install face_recognition
Collecting face_recognition
  Downloading https://files.pythonhosted.org/packages/3f/ed/ad9a20842f373d4633fc8b49109b623597d6f193d3bbbf7780a5ee0eef2/face_recognition-1.2.3-py3-none-any.whl
Collecting face-recognition-models>0.3.0 (from face_recognition)
  Downloading https://files.pythonhosted.org/packages/cf/3b/4fd8c534f6c0d1b00ce0973d01331525538045084c73c153ee6df20224cf/face_recognition_models-0.3.0.tar.gz (100.1MB)
    | 100.2MB 49kB/s
Collecting dlib>19.7 (from face_recognition)
  Using cached https://files.pythonhosted.org/packages/05/57/e0a0caa3c09a27f00bc78da39c423e2553f402a3705adc619176a3a24b36/dlib-19.17.0.tar.gz
Requirement already satisfied: Click>=6.0 in c:\python 3.7\lib\site-packages (from face_recognition) (7.0)
Requirement already satisfied: numpy in c:\python 3.7\lib\site-packages (from face_recognition) (1.16.4)
Requirement already satisfied: Pillow in c:\python 3.7\lib\site-packages (from face_recognition) (6.1.0)
Installing collected packages: face-recognition-models, dlib, face-recognition
Successfully installed face-recognition-models-0.3.0 dlib-19.17.0 face-recognition-1.2.3
```

Working of Code:

You might be good at recognizing faces. You probably find it a cinch to identify the face of a family member, friend, or acquaintance. You're familiar with their facial features — their eyes, nose, mouth — and how they come together.

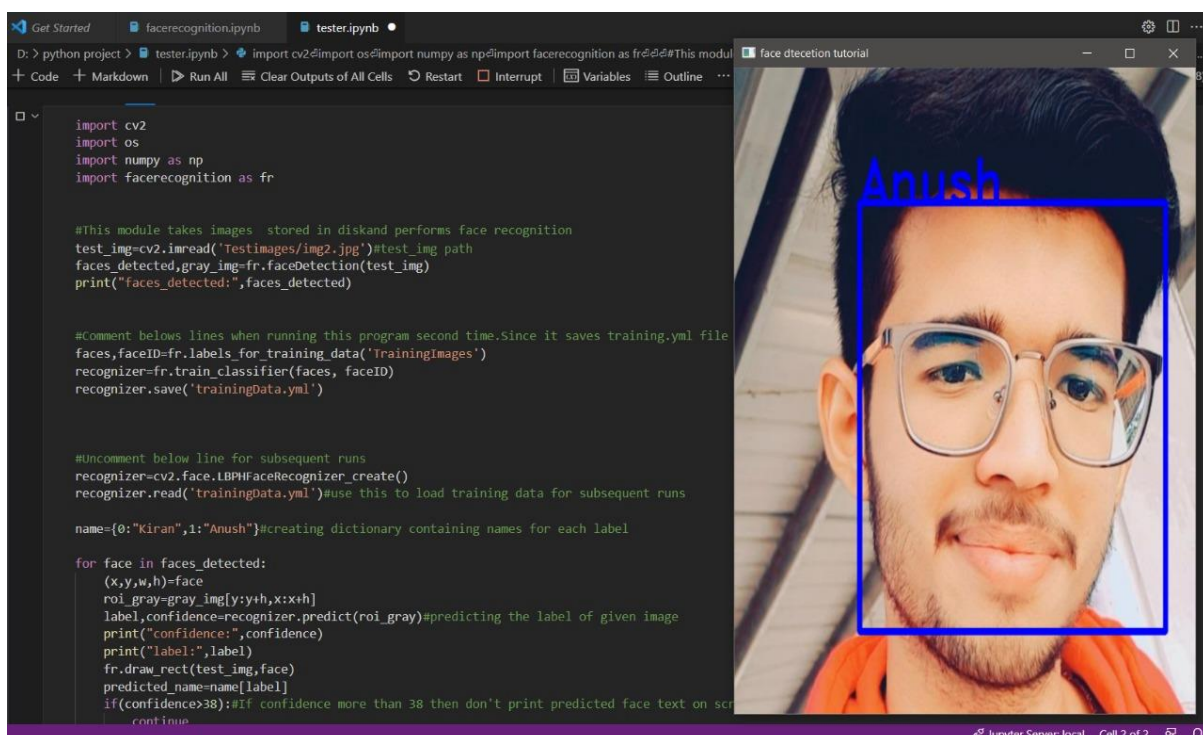
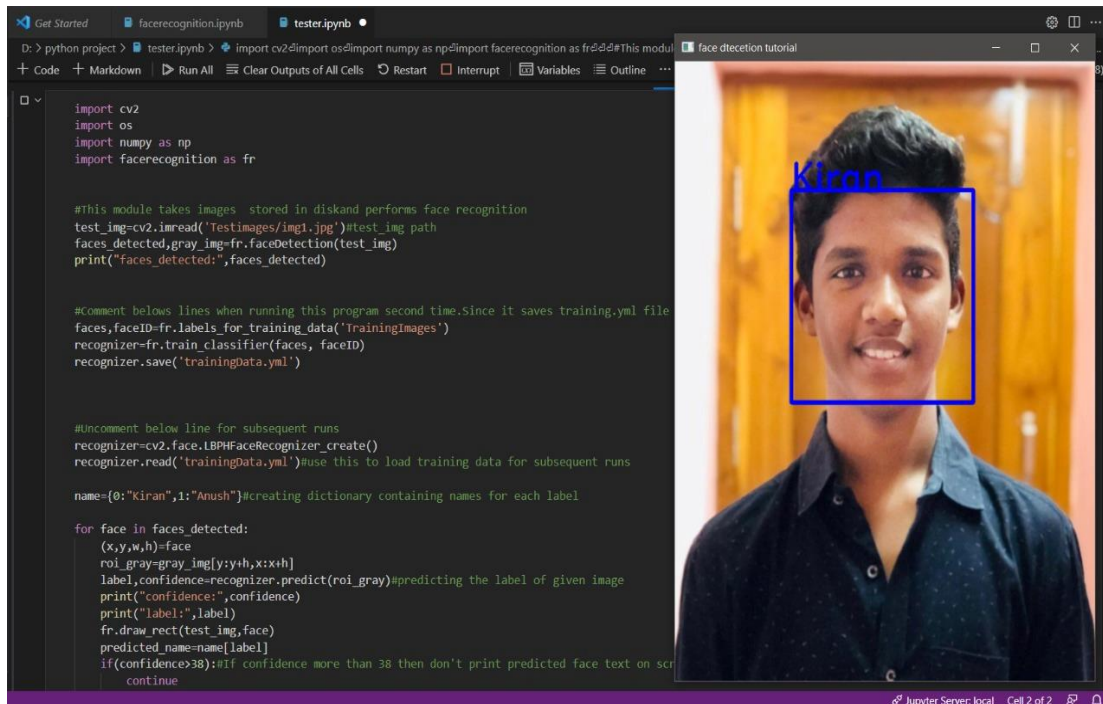
That's how a facial recognition system works, but on a grand, algorithmic scale. Where you see a face, recognition technology sees data. That data can be stored and accessed. For instance, half of all American adults have their images stored in one or more facial-recognition databases that law enforcement agencies can search, according to a Georgetown University study.

So how does facial recognition work? Technologies vary, but here are the basic steps:

Step 1. A picture of your face is captured from a photo or video. Your face might appear alone or in a crowd. Your image may show you looking straight ahead or nearly in profile.

Step 2. Facial recognition software reads the geometry of your face. Key factors include the distance between your eyes and the distance from forehead to chin. The software identifies facial landmarks — one system identifies 68 of them — that are key to distinguishing your face. The result: your facial signature.

Output Screen shots:



Source Code:

```
Get Started | facerecognition.ipynb | tester.ipynb
D:\> python project > facerecognition.ipynb > import cv2import osimport numpy as np@d@d#This module contains all common functions that are called in tester.py file@d@d#Given an image below function returns rect
+ Code + Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | ... | base (Python 3.8.8)

import cv2
import os
import numpy as np

#This module contains all common functions that are called in tester.py file

#Given an image below function returns rectangle for face detected alongwith gray scale image
def faceDetection(test_img):
    gray_img=cv2.cvtColor(test_img,cv2.COLOR_BGR2GRAY)#convert color image to grayscale
    face_haar_cascade=cv2.CascadeClassifier('C:/Users/HP/anaconda3/Lib/site-packages/cv2/data/haarcascade_frontalface_default.xml')#Load haar classifier
    faces=face_haar_cascade.detectMultiScale(gray_img,scaleFactor=1.32,minNeighbors=5)#detectMultiScale returns rectangles

    return faces,gray_img

#Given a directory below function returns part of gray_img which is face alongwith its label/ID
def labels_for_training_data(directory):
    faces=[]
    faceID=[]

    for path,subdirname,filenames in os.walk(directory):
        for filename in filenames:
            if filename.startswith("."):
                print("Skipping system file")#Skipping files that startwith .
                continue

            id=os.path.basename(path)#fetching subdirectory names
            img_path=os.path.join(path,filename)#fetching image path
            print("img_path:",img_path)
            print("id:",id)
            test_img=cv2.imread(img_path)#loading each image one by one
            if test_img is None:
                print("Image not loaded properly")
                continue
```

```
Get Started | facerecognition.ipynb | tester.ipynb
D:\> python project > facerecognition.ipynb > import cv2import osimport numpy as np@d@d#This module contains all common functions that are called in tester.py file@d@d#Given an image below function returns rect
+ Code + Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | ... | base (Python 3.8.8)

img_path=os.path.join(path,filename)#fetching image path
print("img_path:",img_path)
print("id:",id)
test_img=cv2.imread(img_path)#loading each image one by one
if test_img is None:
    print("Image not loaded properly")
    continue

faces_rect,gray_img=faceDetection(test_img)#Calling faceDetection function to return faces detected in particular image
if len(faces_rect)!=1:
    continue #Since we are assuming only single person images are being fed to classifier
(x,y,w,h)=faces_rect[0]
roi_gray=gray_img[y:y+w,x:x+h]#cropping region of interest i.e. face area from grayscale image
faces.append(roi_gray)
faceID.append(int(id))

return faces,faceID

#Below function trains haar classifier and takes faces,faceID returned by previous function as its arguments
def train_classifier(faces,faceID):
    face_recognizer=cv2.face.LBPHFaceRecognizer_create()
    face_recognizer.train(faces,np.array(faceID))
    return face_recognizer

#Below function draws bounding boxes around detected face in image
def draw_rect(test_img,face):
    (x,y,w,h)=face
    cv2.rectangle(test_img,(x,y),(x+w,y+h),(255,0,0),thickness=3)

#Below function writes name of person for detected label
def put_text(test_img,text,x,y):
    cv2.putText(test_img,text,(x,y),cv2.FONT_HERSHEY_SIMPLEX,2,(255,0,0),2)
```

```

Get Started facerecognition.ipynb tester.ipynb
D: > python project > tester.ipynb > import cv2import osimport numpy as npimport facerecognition as fr#This module takes images stored in diskand performs face recognition#test_img=cv2.imread('Test...
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt Variables Outline ... base (Python 3.8.8)

pip install opencv-contrib-python
[1] ✓ 3.8s Python
... Requirement already satisfied: opencv-contrib-python in c:\users\hp\anaconda3\lib\site-packages (4.5.4.58)
Requirement already satisfied: numpy>=1.17.3 in c:\users\hp\anaconda3\lib\site-packages (from opencv-contrib-python) (1.20.1)
Note: you may need to restart the kernel to use updated packages.

import cv2
import os
import numpy as np
import facerecognition as fr

#This module takes images stored in diskand performs face recognition
test_img=cv2.imread('TestImages/img1.jpg')#test_img path
faces_detected,gray_img=fr.faceDetection(test_img)
print("faces_detected:",faces_detected)

#Comment belows lines when running this program second time.Since it saves training.yml file in directory
faces,faceID=fr.labels_for_training_data('TrainingImages')
recognizer=fr.train_classifier(faces, faceID)
recognizer.save('trainingData.yml')

#Uncomment below line for subsequent runs
recognizer=cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainingData.yml')#use this to load training data for subsequent runs

name={0:"Kiran",1:"Anush"}#creating dictionary containing names for each label

```

```

Get Started facerecognition.ipynb tester.ipynb
D: > python project > tester.ipynb > import cv2import osimport numpy as npimport facerecognition as fr#This module takes images stored in diskand performs face recognition#test_img=cv2.imread('Test...
+ Code + Markdown | Run All Clear Outputs of All Cells Restart Interrupt Variables Outline ... base (Python 3.8.8)

recognizer=fr.train_classifier(faces, faceID)
recognizer.save('trainingData.yml')

#Uncomment below line for subsequent runs
recognizer=cv2.face.LBPHFaceRecognizer_create()
recognizer.read('trainingData.yml')#use this to load training data for subsequent runs

name={0:"Kiran",1:"Anush"}#creating dictionary containing names for each label

for face in faces_detected:
    (x,y,w,h)=face
    roi_gray=gray_img[y:y+h,x:x+h]
    label,confidence=recognizer.predict(roi_gray)#predicting the label of given image
    print("confidence:",confidence)
    print("label:",label)
    fr.draw_rect(test_img,face)
    predicted_name=name[label]
    if(confidence>30):#If confidence more than 30 then don't print predicted face text on screen
        continue
    fr.put_text(test_img,predicted_name,x,y)

resized_img=cv2.resize(test_img,(500,700))
cv2.imshow("face dtetection tutorial",resized_img)
cv2.waitKey(0)#waits indefinitely until a key is pressed
cv2.destroyAllWindows

[2] ✓ 20.7s Python
... faces_detected: [[211 200 330 330]]
confidence: 30.569901072822326
label: 0
<function destroyAllWindows>

```


Conclusion:

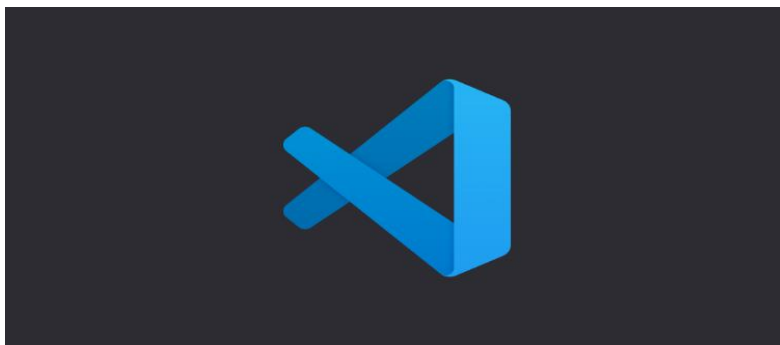
Face recognition is an emerging technology that can provide many benefits. Face recognition can save resources and time, and even generate new income streams, for companies that implement it right.

A face detection and recognition system would certainly speed up the process of checking student attendance in comparison to other biometrics authentication methods and in the right circumstances it would be able to match their accuracy. Nowadays there are a wide variety of software, whether it is a Face API like Microsoft's or a library like OpenCV, that makes face detection and recognition accessible and reliable and is constantly improving. Each software imposes various restrictions, such as the limited number of calls you can make to Microsoft's Face API. However, using more than one software can reduce these restrictions and lead to better results.

References:

To conduct this project the following tools have been used:

- Visual studio code: For Compiling + Library Management + Executing code.



- Stack Overflow: For Doubt clearing.



- Udemy & Coursera: For Python Learning.

