## **FACE IDENTITY PROGRAM**

Step 1: Access to webcam

Step 2: Face identification.

Step 3: Data collection

Step 4: Training

Step 5: Face recognition

Step 6: Programming Arduino

## **Step 1: Access to Webcam**

If Opencv is installed on your computer, you are ready. Open the command prompt and type "pip install opencv".

For any further issues, [click here](https://stackoverflow.com/questions/23708898/pip-is-not-recognized-as-an-internal-or-external-command#:~:text=You%20need%20to%20add%20the,added%20to%20your%20PATH%20variable.).

Code Name => AccessTo\_webcam.py

## **Step 2: Face Identification.**

OpenCV provides a training method or pre-trained models called a Cascade Classifier. The pre-trained models are located in the data folder in the OpenCV installation. I'm providing that file. Just download it and place it in your project folder—the folder where the "AccessTo\_webcam.py" file is stored. If you still need to create one then do it.

Code Name => Face\_identifaction.py

Haarcascade XML File Name => haarcascade\_frontalface\_default.xml

## **Step 3: Data Collection**

Data collection is rather the easiest step in this project. create a folder named "image\_data" in your main project folder. Inside the "image\_data" folder, add additional folders with the person's name, where we will store the data.

Create your folders and name them. I recommend collecting about 20 images per person. You can also add more images, but see that data collected for all the persons contains the same number of images. It helps to provide accuracy.

## **Step 4: Training**

In brief, we will go through all the folders and images in the "image\_data" folder and create a dictionary containing the label ID and the corresponding name. Simultaneously, we will load the image to detect the face in every image called "Region of Interest" and create a ".yml" file containing that information.

Code Name => face\_trainer.py

## **Step 5: Face Recognition**

We will load our trained models into the Python file, Access our webcam, identify Faces in the video stream and do a comparison or prediction between the current face that is identified in the video stream, and the model that was trained. if the data is matched, then we say that the person is recognised it is just that simple...

Code Name => face\_recognise.py

## **Step 6: Programming Arduino**

To do now…