Progress Report Guideline:

|  |  |
| --- | --- |
| Course Code | DAB402 |
| Team Number | 3 |
| Team Members | Amanpreet Kaur (0798150)  PrabhKirat Singh(0790727)  Ramandeep kaur(793411) |
| Instructor | Francisco |

* **Intro to project: 1.**Facial Recognition System for attendance is a technology capable of matching a human face from a digital image or a video frame against a database of faces, typically employed to authenticate users through ID verification services, works by pinpointing and measuring facial features from a given image.

**2.** Traditionally, students’ attendance records are taken manually by teachers throughroll calling in the class, or the students are required to physically sign the attendance sheet each time for the attendance of each class, in order to replace traditional attendance, we propose a more convenient and intelligent attendance system which is implemented using facial recognition.

Eigenfaces: Eigenfaces uses principal of component Analysis. It is an appearance based approach which captures the variation in a collection of face images. It then uses this data to compare different face images in a holistic manner.

Fisherfaces: This method is an enhancement of the Eigenface method. It uses Fisher’s Linear Discriminant Analysis(FLDA or LDA) for the dimensionality reduction.

Local Binary Patterns Histograms: Local Binary Patterns Histogram is a simple yet very efficient texture operator which labels the pixels pf an image by thresholding the neighborhood of each pixel and considers the result as a binary number and then visualize the results as histograms.

**FLOWCHART OF PROJECT:**

**Diagram

Description automatically generated**

* **The status of the project**

**:** While starting our project we defined objectives related to project-specified milestones to achieve the objectives and we completed most probably all the milestones. Also, the defined goals have been achieved within a specific time. So we consider this project has been completed successfully within the desired timeline. Although, we continue working on our project in order to add some new features which make the project more attractive and helpful in taking online attendance. Face recognition is a computer application that is capable of detecting, tracking, identifying or verifying human faces from an image or video captured using digital camera. A form of biometrics. System compares the scans to record stored in central or local databases or even on a smart card.

* **The milestones achieved:** Milestones helped a lot in the completion of the project in a timely and effective manner. We specific points within a project’s life cycle to measure the progress and worked accordingly to that we complete our goals accordingly.

Milestones are as follows:

* 1. **Data collection and Aggregation**: It is really a very crucial and starting step of the project. we started by collecting data and refining the data which is being used by our team in our project.
  2. **Planning and defining:** The second step of the project is to plan how we can start and define the algorithm used in the project. We planned to use different algorithms of machine learning for while and choose the best algorithm for our project. We plan to use CNN first and then KNN. If we succeed with CNN we decided not to go with KNN but if CNN is not successful then we decided to KNN. But unfortunately, we found some issues with CNN and jump to LBPH for face recognition. Now we are using LBPH for face recognition , in this algorithm we find ways to implement recognizer on faces. We also used HAAR Cascade Classifier to implement the color collection of images.
  3. **Model implementation, training, and evaluation:** It was the most important and time-demanding stage. Here we try to implement the various algorithm and checked the accuracy for each. We tried CNN, Eigenfaces, and LBPH algorithm for Face Recognition for a while. But we started working on LBPH and HaarCascade classifier. We made classes for taking images and training images and implementing them.

200 training images generated by the algorithm using

haarcascade\_frontalface\_default.xml from haar cascades, so 200 histograms

Crops images to 200\*200 pixel values and reduces RGB images to grayscale with

8bits per pixel.

During testing , the algorithm again creates histogram from the test image , and it compares that with the training histograms to try to get a match of the image.

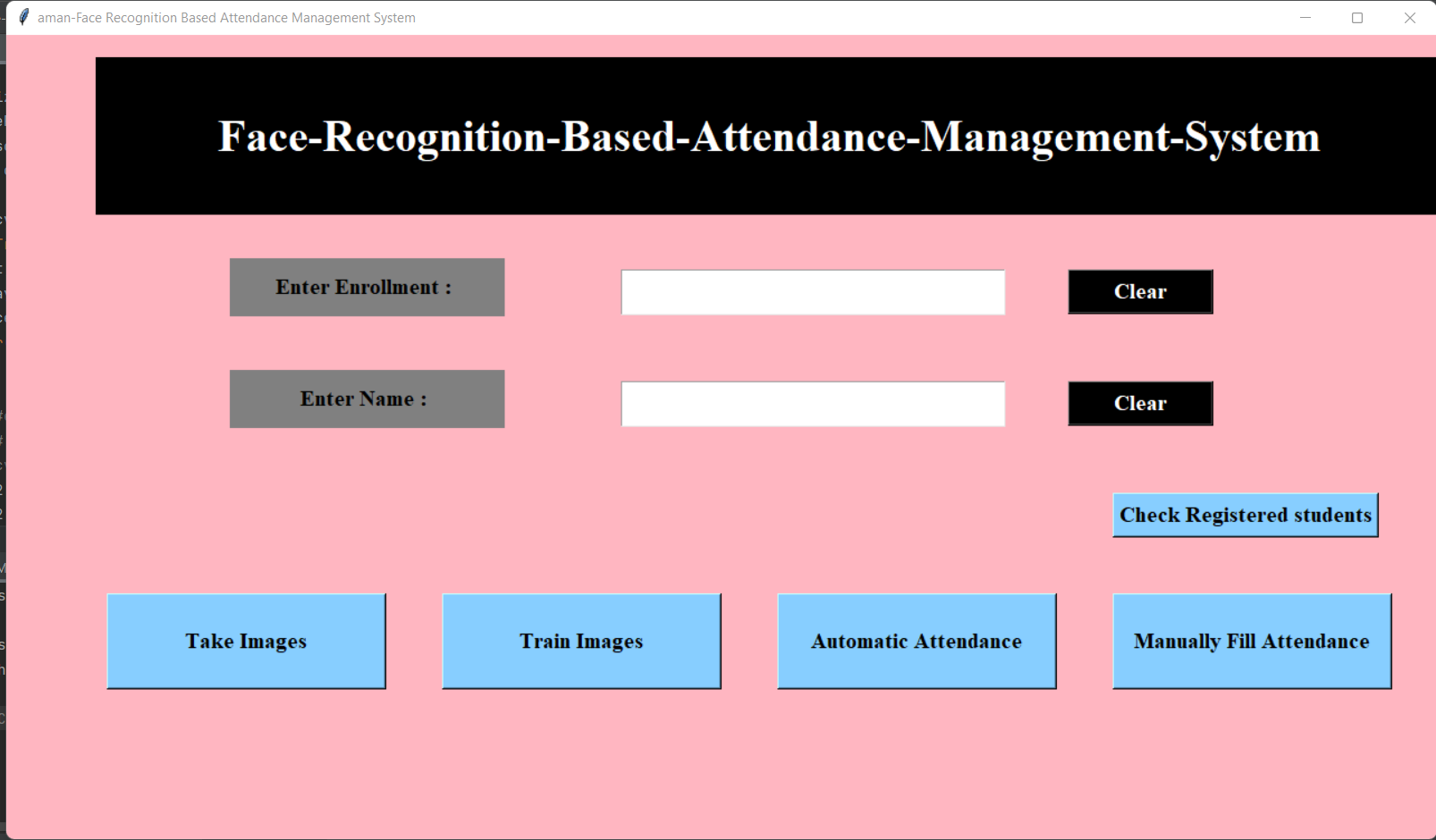
A picture containing application

Description automatically generated

4.Testing and Resulting : in this part of our project we decided to test the accuracy of our model . it detects the images of a person and shows weather this person is registered in our database or not.

**Coding of the project and implementation :**

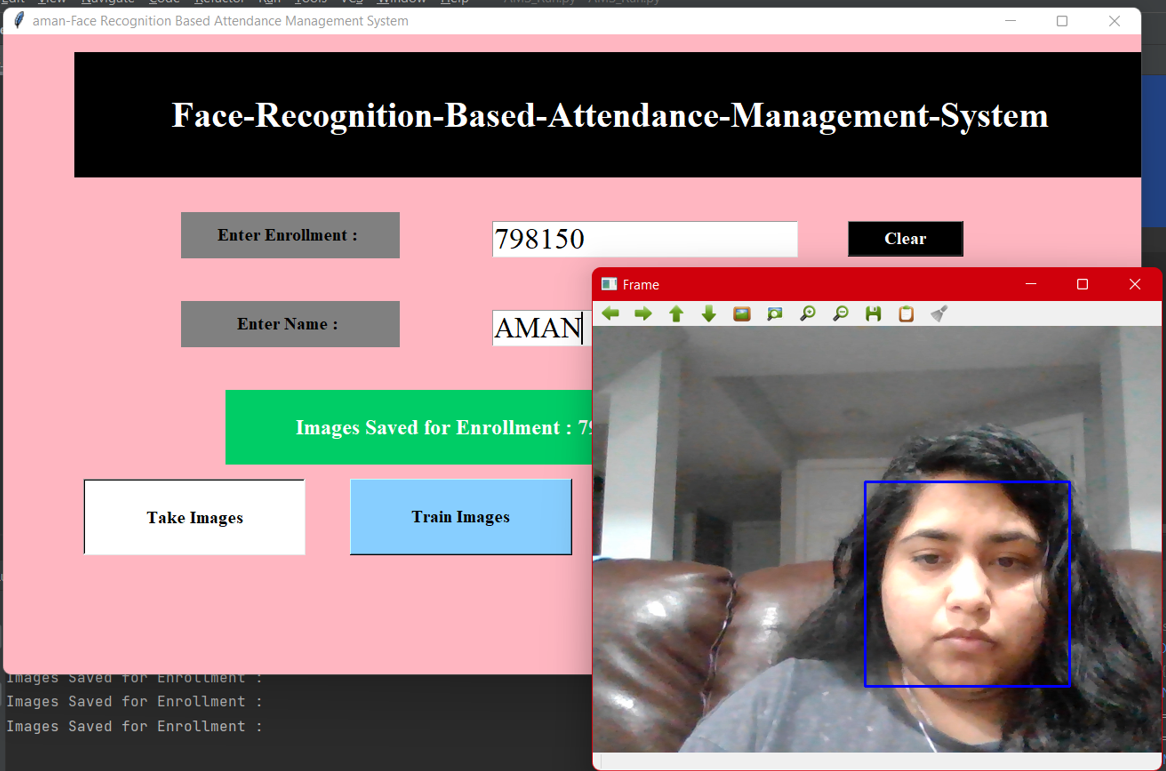
**First page:**



This is the first page of the project in which the whole framework is described.

import tkinter as tk  
from tkinter import \*  
import cv2  
import csv  
import os  
import numpy as np  
from PIL import Image, ImageTk  
import pandas as pd  
import datetime  
import time  
  
# Window is our Main frame of system  
window = tk.Tk()  
window.title("aman-Face Recognition Based Attendance Management System")  
  
window.geometry('1280x720')  
  
window.configure(background='light pink')

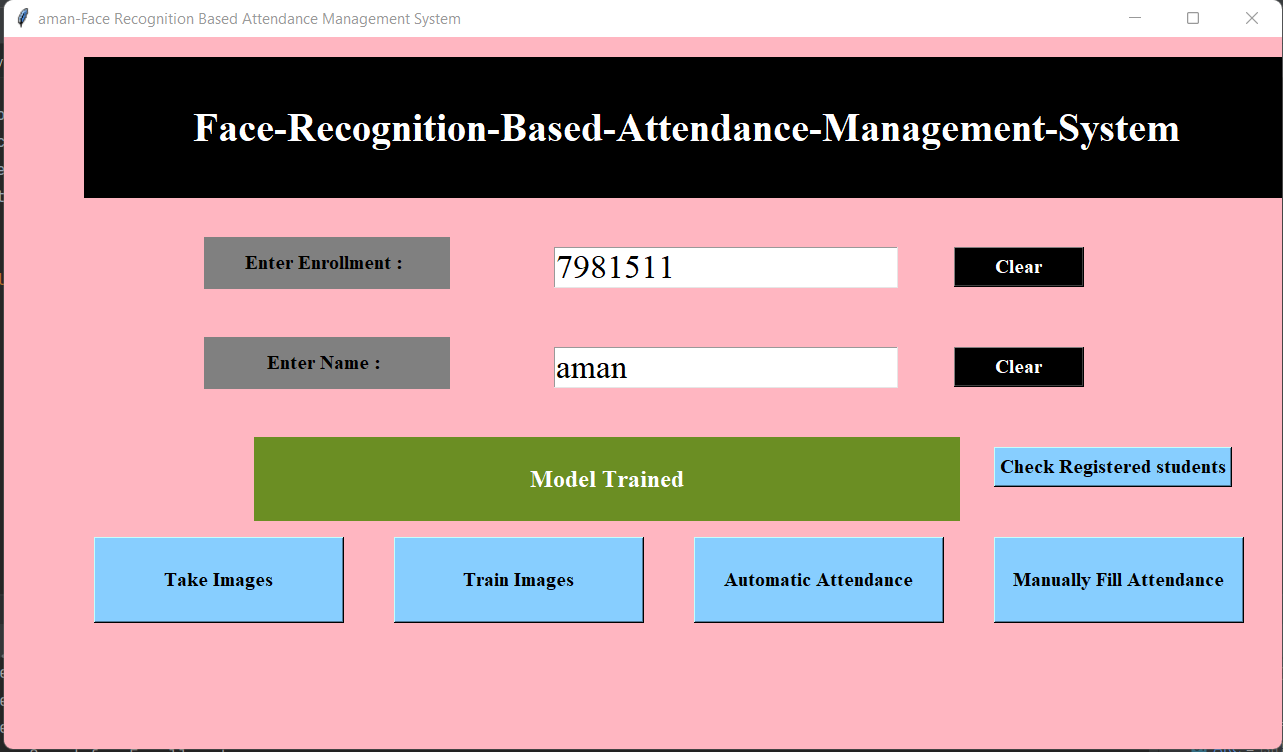
**STUDENT REGISTRATION:**



The button take image is used to capture the image of a person through video capturing by using video camera. And saved the picture in training image folder which is create to save images as database.

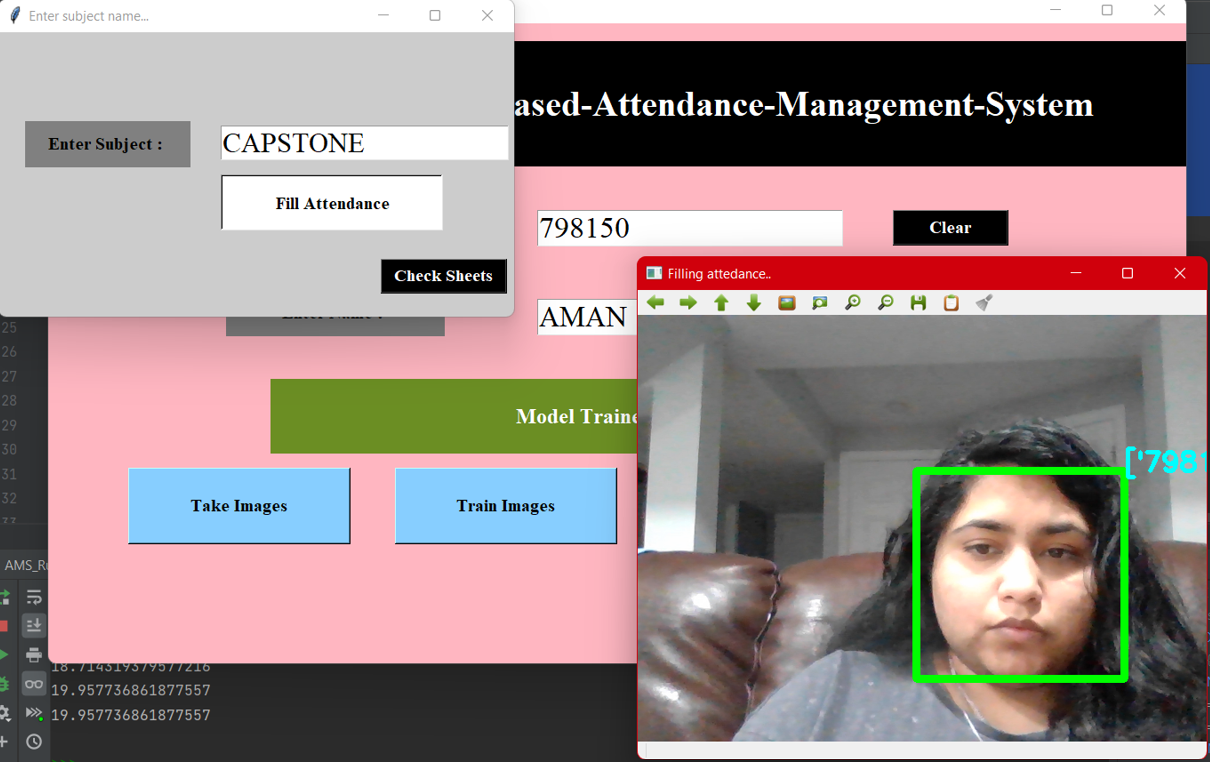
def take\_img():  
 l1 = txt.get()  
 l2 = txt2.get()  
 if l1 == '':  
 err\_screen()  
 elif l2 == '':  
 err\_screen()  
 else:  
 try:  
 cam = cv2.VideoCapture(0)  
 detector = cv2.CascadeClassifier(  
 'haarcascade\_frontalface\_default.xml')  
 Enrollment = txt.get()  
 Name = txt2.get()  
 sampleNum = 0  
 while (True):  
 ret, img = cam.read()  
 gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)  
 faces = detector.detectMultiScale(gray, 1.3, 5)  
 for (x, y, w, h) in faces:  
 cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)  
 # incrementing sample number  
 sampleNum = sampleNum + 1  
 # saving the captured face in the dataset folder  
 cv2.imwrite("TrainingImage/ " + Name + "." + Enrollment + '.' + str(sampleNum) + ".jpg",  
 gray)  
 print("Images Saved for Enrollment :")  
 cv2.imshow('Frame', img)  
 # wait for 100 miliseconds  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
 #  
 # # break if the sample number is morethan 100  
 elif sampleNum > 20:  
 break  
  
  
 cam.release()  
 cv2.destroyAllWindows()  
 ts = time.time()  
 Date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')  
 Time = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')  
 row = [Enrollment, Name, Date, Time]  
 with open('StudentDetails\StudentDetails.csv', 'a+') as csvFile:  
 writer = csv.writer(csvFile, delimiter=',')  
 writer.writerow(row)  
 csvFile.close()  
 res = "Images Saved for Enrollment : " + Enrollment + " Name : " + Name  
 Notification.configure(  
 text=res, bg="SpringGreen3", width=50, font=('times', 18, 'bold'))  
 Notification.place(x=250, y=400)  
 except FileExistsError as F:  
 f = 'Student Data already exists'  
 Notification.configure(text=f, bg="Red", width=21)  
 Notification.place(x=450, y=400)

**TRAIN MODEL:**

****

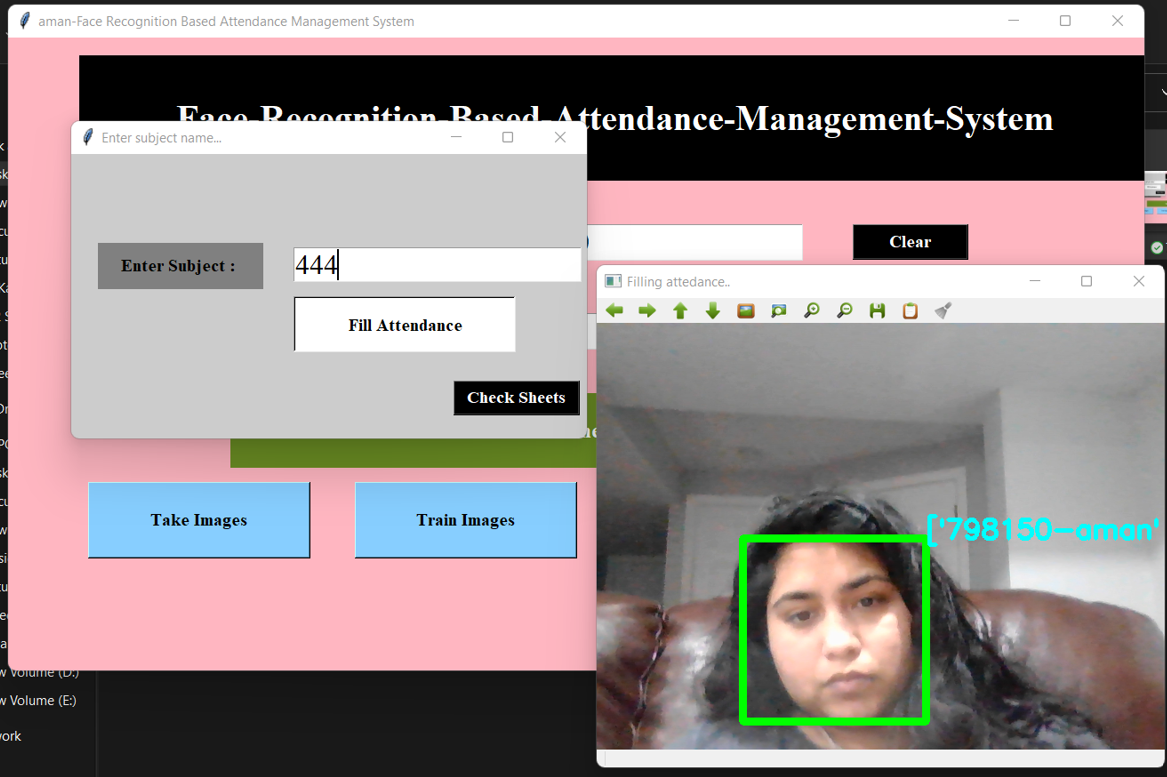
def trainimg():  
 recognizer = cv2.face.LBPHFaceRecognizer\_create()  
 global detector  
 detector = cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")  
 try:  
 global faces, Id  
 faces, Id = getImagesAndLabels("TrainingImage")  
 except Exception as e:  
 l = 'please make "TrainingImage" folder & put Images'  
 Notification.configure(text=l, bg="SpringGreen3",  
 width=50, font=('times', 18, 'bold'))  
 Notification.place(x=350, y=400)  
  
 recognizer.train(faces, np.array(Id))  
 try:  
 recognizer.save("TrainingImageLabel\Trainner.yml")  
 except Exception as e:  
 q = 'Please make "TrainingImageLabel" folder'  
 Notification.configure(text=q, bg="SpringGreen3",  
 width=50, font=('times', 18, 'bold'))  
 Notification.place(x=350, y=400)  
  
 res = "Model Trained" # +",".join(str(f) for f in Id)  
 Notification.configure(text=res, bg="olive drab",  
 width=50, font=('times', 18, 'bold'))  
 Notification.place(x=250, y=400)  
  
  
def getImagesAndLabels(path):  
 imagePaths = [os.path.join(path, f) for f in os.listdir(path)]  
 # create empth face list  
 faceSamples = []  
 # create empty ID list  
 Ids = []  
 # now looping through all the image paths and loading the Ids and the images  
 for imagePath in imagePaths:  
 # loading the image and converting it to gray scale  
 pilImage = Image.open(imagePath).convert('L')  
 # Now we are converting the PIL image into numpy array  
 imageNp = np.array(pilImage, 'uint8')  
 # getting the Id from the image  
  
 Id = int(os.path.split(imagePath)[-1].split(".")[1])  
 # extract the face from the training image sample  
 faces = detector.detectMultiScale(imageNp)  
 # If a face is there then append that in the list as well as Id of it  
 for (x, y, w, h) in faces:  
 faceSamples.append(imageNp[y:y + h, x:x + w])  
 Ids.append(Id)  
 return faceSamples, Ids  
  
  
window.grid\_rowconfigure(0, weight=1)  
window.grid\_columnconfigure(0, weight=1)

**CHECKING ATTENDANCE AUTOMATICALLY:**

****

def getImagesAndLabels(path):  
 imagePaths = [os.path.join(path, f) for f in os.listdir(path)]  
 # create empth face list  
 faceSamples = []  
 # create empty ID list  
 Ids = []  
 # now looping through all the image paths and loading the Ids and the images  
 for imagePath in imagePaths:  
 # loading the image and converting it to gray scale  
 pilImage = Image.open(imagePath).convert('L')  
 # Now we are converting the PIL image into numpy array  
 imageNp = np.array(pilImage, 'uint8')  
 # getting the Id from the image  
  
 Id = int(os.path.split(imagePath)[-1].split(".")[1])  
 # extract the face from the training image sample  
 faces = detector.detectMultiScale(imageNp)  
 # If a face is there then append that in the list as well as Id of it  
 for (x, y, w, h) in faces:  
 faceSamples.append(imageNp[y:y + h, x:x + w])  
 Ids.append(Id)  
 return faceSamples, Ids

**TESTING :**

****

import cv2  
import numpy as np  
  
recognizer = cv2.face.LBPHFaceRecognizer\_create()  
  
recognizer.read('TrainingImageLabel/trainner.yml')  
cascadePath = "haarcascade\_frontalface\_default.xml"  
faceCascade = cv2.CascadeClassifier(cascadePath)  
font = cv2.FONT\_HERSHEY\_SIMPLEX  
  
cam = cv2.VideoCapture(0)  
while True:  
 ret, im = cam.read()  
 gray = cv2.cvtColor(im, cv2.COLOR\_BGR2GRAY)  
 faces = faceCascade.detectMultiScale(gray, 1.2, 5)  
 for(x, y, w, h) in faces:  
 Id, conf = recognizer.predict(gray[y:y+h, x:x+w])  
  
 # #else:  
 # # Id = "Unknown"  
 # cv2.rectangle(im, (x-22,y-90), (x+w+22, y-22), (0,255,0), -1)  
 cv2.rectangle(im, (x, y), (x + w, y + h), (0, 260, 0), 7)  
 cv2.putText(im, str(Id), (x, y-40), font, 2, (255, 255, 255), 3)  
  
 # cv2.putText(im, str(Id), (x + h, y), font, 1, (0, 260, 0), 2)  
 cv2.imshow('im', im)  
 if cv2.waitKey(10) & 0xFF == ord('q'):  
 break  
cam.release()  
cv2.destroyAllWindows()

**CONCLUSION:**

The proposed facial recognition based auto attendance system is many times more efficient than it’s fingerprint or RFID based counterparts as no manual intervention is required. ► Saves time as the system works for several people simultaneously. ► The program is relatively lightweight and can be run on inexpensive hardware