INDEX

1. INTRODUCTION

Motivation

The motivation behind this project is to simplify the means by which attendance is taken during lectures and how much time it takes. The use of ID cards or manually calling out attendance and writing it down on sheets is not productive and efficient. This system will detect the number of faces on the class and will also identify them from the store database. With the face detection and recognition system in place, it will be easy to tell if a student is actually present in the classroom or not.

Related Works

A final year project at Kingston University London, The system will be presented an image either via camera or from memory and it must detect the number of faces on it automatically. After identifying faces, the system should crop the faces from the image and store them in memory for image recognition which will be done in the second step. The system should be able to automatically count the number of faces detected on the image. The key algorithms are Viola-Jones for face detection and Hidden Markov Model with SVD.

Another final year project at Universiti Tunku, The approach performs face recognition-based student attendance system. This method is also similar to others and begins with the input of an image either loaded from memory or from camera. Then it pre-processes the facial features and extracts it followed by subjective selecting and then the recognition of the facial images from known database. Both LBP and PCA feature extraction methods are studied in detail and computed in this approach in order to make comparisons. LBP is enhanced in this approach to reduce the illumination effect. An algorithm to combine enhanced LBP and PCA is also designed for subjective selection in order to increase the accuracy.

This is a project done by students as a final year project at University of Nairobi, The system will comprise of two modules. The first module a.k.a face detector is a mobile component, which is basically a camera that captures student faces and stores them in a file using computer vision face detection algorithms and face extraction techniques. The second module is a desktop application that does face recognition of the captured images (faces) in the file, marks the students register and then stores the results in a database for future analysis.

Our Work

Our aim is to comes out with solution to problems faced by all previous systems. Our project is mainly based on video face recognization using dlib and face recognization library by using python programming language. Also we are using some important tools of OpenCV library and numpy of python to improves image processing tecniques used in our project. Our main objective is when a student comes in front of camera it automatic recognize him or her and if students data matches with our data base it will show their attendance percentage on

output window and also attendance will be marked on google sheet. And also when a particular student comes twise in front of camera it will show attendance has been already marked.

2. DESIGN METHODOLOGY

➢ Basics Of Library used in project

Computer Vision

Computer vision is a process by which we can understand the images and videos how they are stored and how we can manipulate and retrieve data from them. Computer Vision is the base or mostly used for Artificial Intelligence. Computer-Vision is playing a major role in self-driving cars, robotics as well as in photo correction apps.

OpenCV

Computer vision is a process by which we can understand the images and videos how they are stored and how we can manipulate and retrieve data from them. Computer Vision is the base or mostly used for Artificial Intelligence. Computer-Vision is playing a major role in self-driving cars, robotics as well as in photo correction apps.

Applications of OpenCV

- face recognition.
- Automated inspection and surveillance.
- number of people – count (foot traffic in a mall, etc).
- Vehicle counting on highways along with their speeds.
- Interactive art installations.
- Anomaly (defect) detection in the manufacturing process (the odd defective products).
- Street view image stitching.
- Video/image search and retrieval.
- Robot and driver-less car navigation and control.
- object recognition.
- Medical image analysis.
- Movies – 3D structure from motion.
- TV Channels advertisement recognition.

OpenCV Functionality

- Image/video I/O, processing, display (core, imgproc, highgui)
- Object/feature detection (objdetect, features2d, nonfree)
- Geometry-based monocular or stereo computer vision (calib3d, stitching, videostab)
- Computational photography (photo, video, superres)
- Machine learning & clustering (ml, flann)

- CUDA acceleration (gpu)

OpenCV is one of the most popular computer vision libraries. If you want to start your journey in the field of computer vision, then a thorough understanding of the concepts of OpenCV is of paramount importance.

We will try to introduce the most basic and important concepts of OpenCV in an intuitive manner.

Some Basic Operations in OpenCV

- Reading an image
- Extracting the RGB values of a pixel
- Extracting the Region of Interest (ROI)
- Resizing the Image
- Rotating the Image
- Drawing a Rectangle
- Displaying text

Face Recognition

Recognize and manipulate faces from Python or from the command line with the world's simplest face recognition library. Built using dlib's state-of-the-art face recognitionbuilt with deep learning. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark. This also provides a simple face_recognition command line tool that lets you do face recognition on a folder of images from the command line.

When you install face_recognition, you get a simple command-line programcalled face_recognition that you can use to recognize faces in aphotograph or folder full for photographs. First, you need to provide a folder with one picture of each person you already know. There should be one image file for each person with the files named according to who is in the picture

Then in you simply run the command face_recognition, passing in the folder of known people and the folder (or single image) with unknown people and it tells you who is in each image.

There's one line in the output for each face. The data is comma-separated with the filename and the name of the person found. An unknown_person is a face in the image that didn't match anyone in your folder of known people.

If you are getting multiple matches for the same person, it might be that the people in your photos look very similar and a lower tolerance value is needed to make face comparisons more strict.

➢ Flow chart

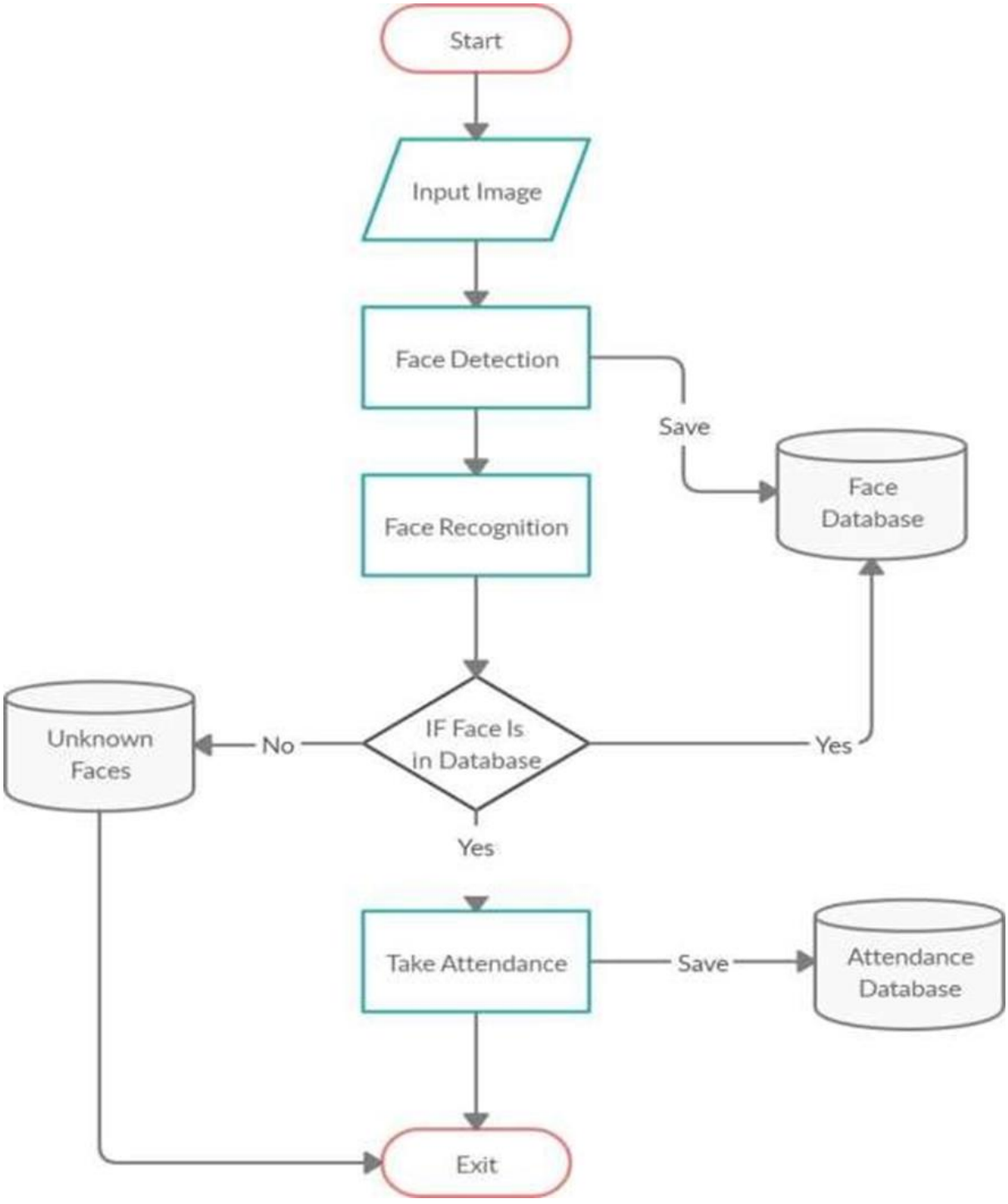

Fig. 1 Deep learning as a subset of machine learning which is a part of Artificial Intelligence

Results and Output Window

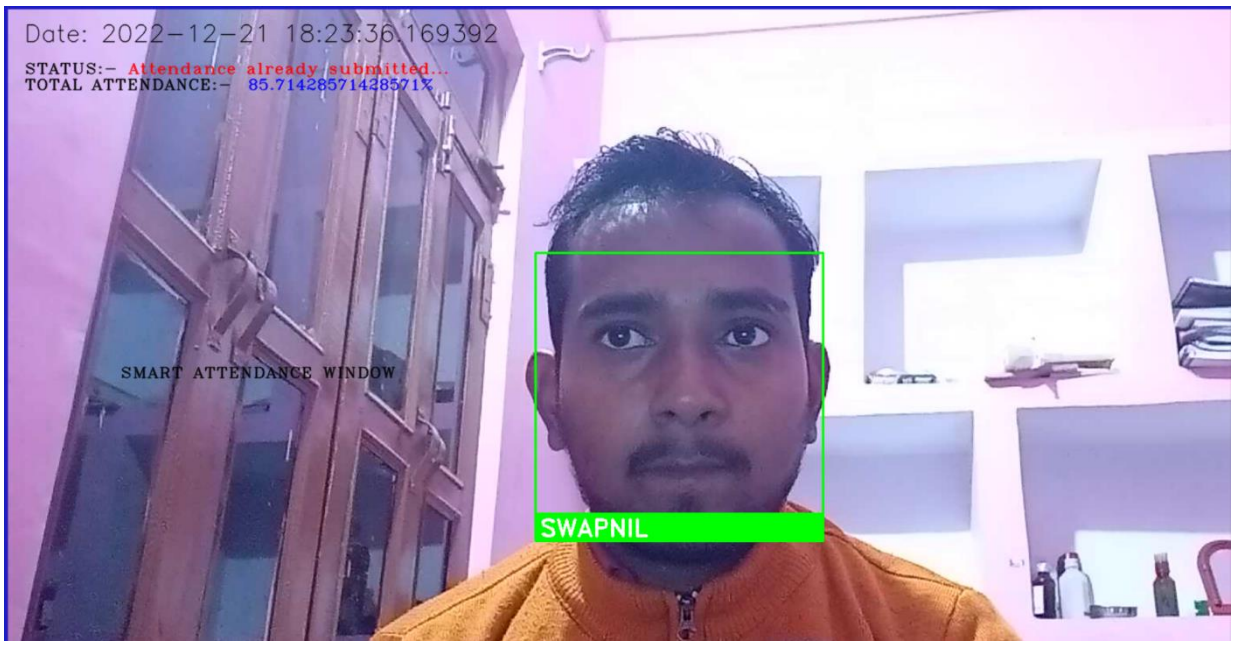

Fig. 2 Output Window is showing a student Swapnil who's data matches with stored data.

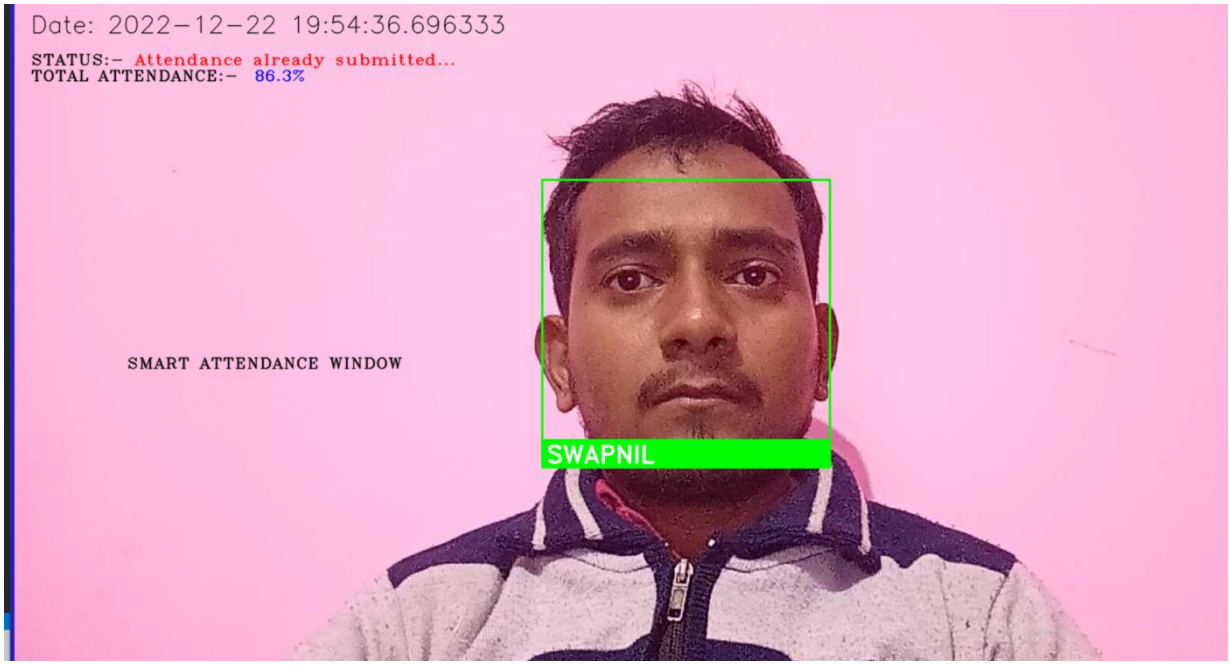

Fig. 3 Output Window of Swapnil next day

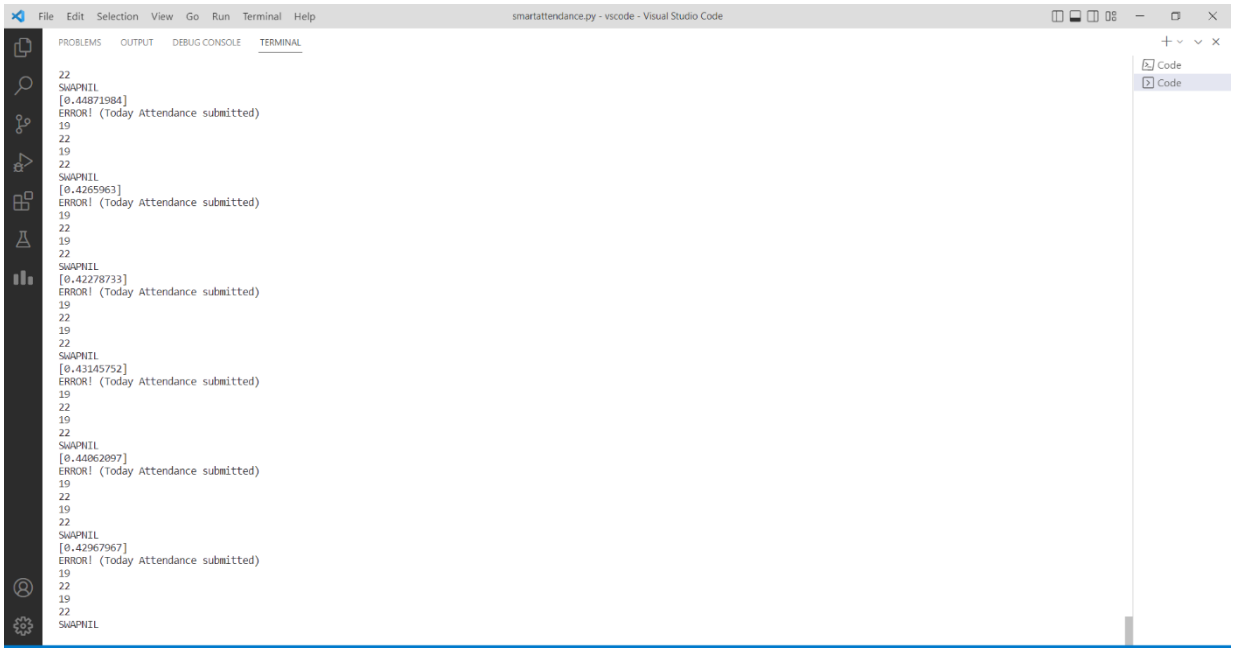

Fig. 4 Turminal Window result of Swapnil

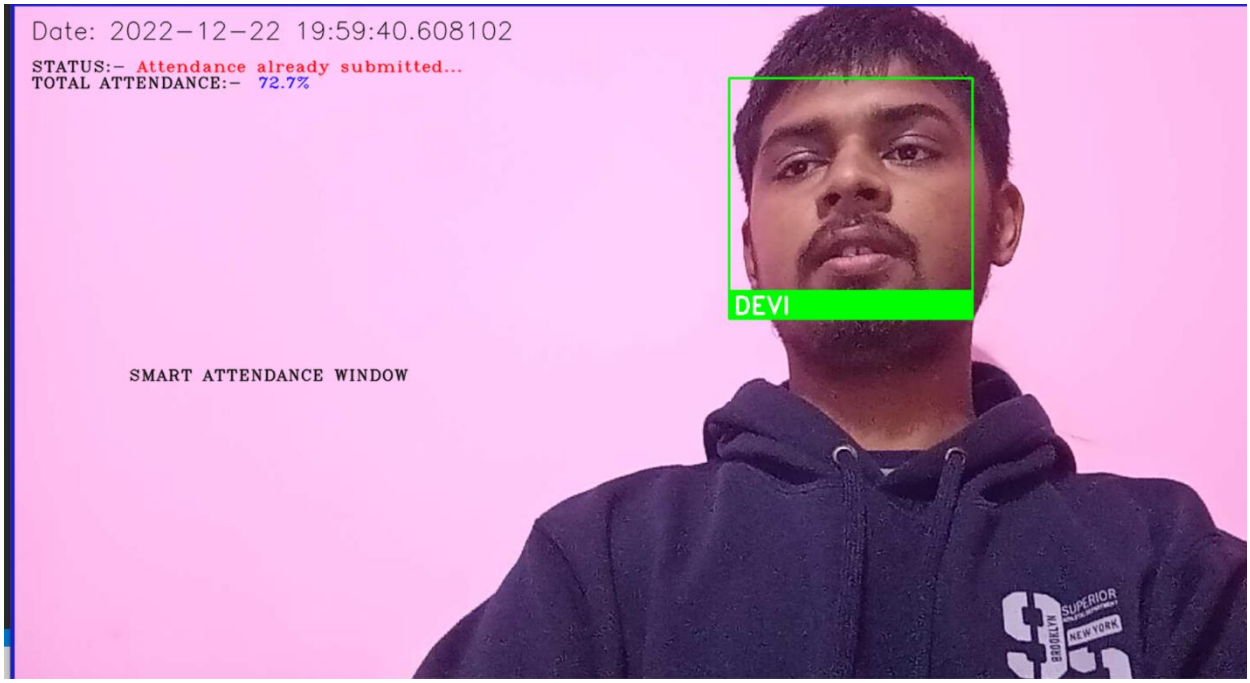

Fig. 5 Output Window of a second student Devi

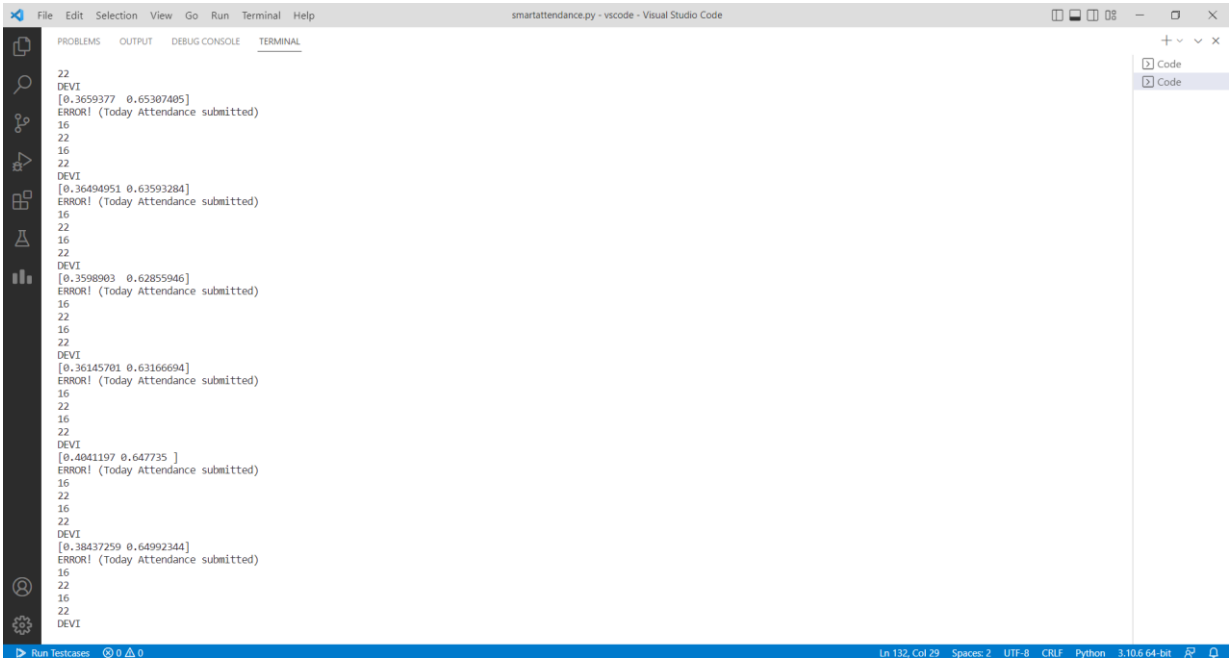

Fig. 6 4 Turminal Window result of Devi

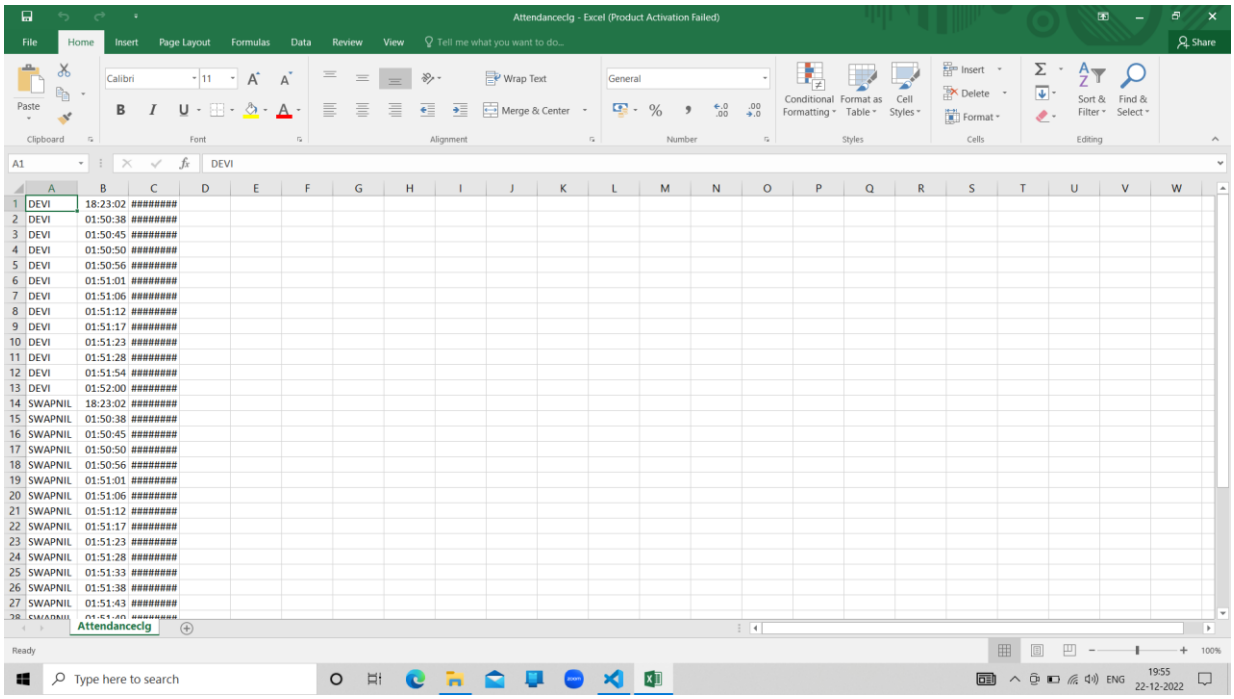

Fig. 7 Google Sheet where per day data of students get stored

Fig. 8 Output Window of a third student who's data is not matched with stored data

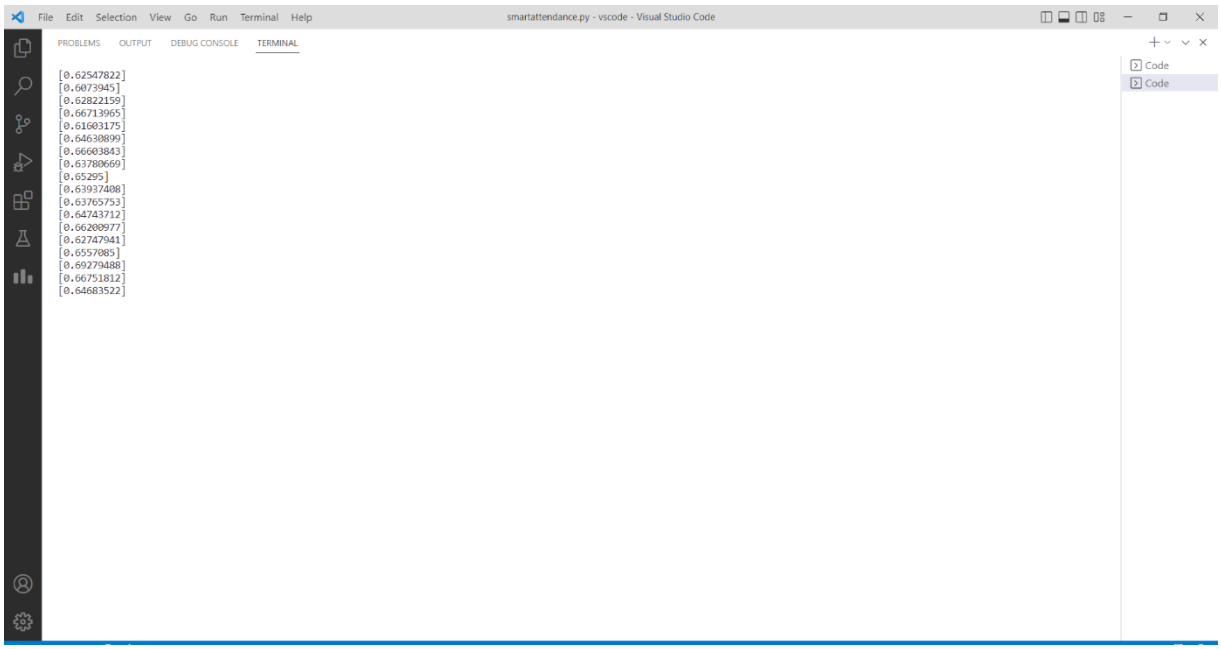

Fig. 9 Turminal Window result of Unknown syudent

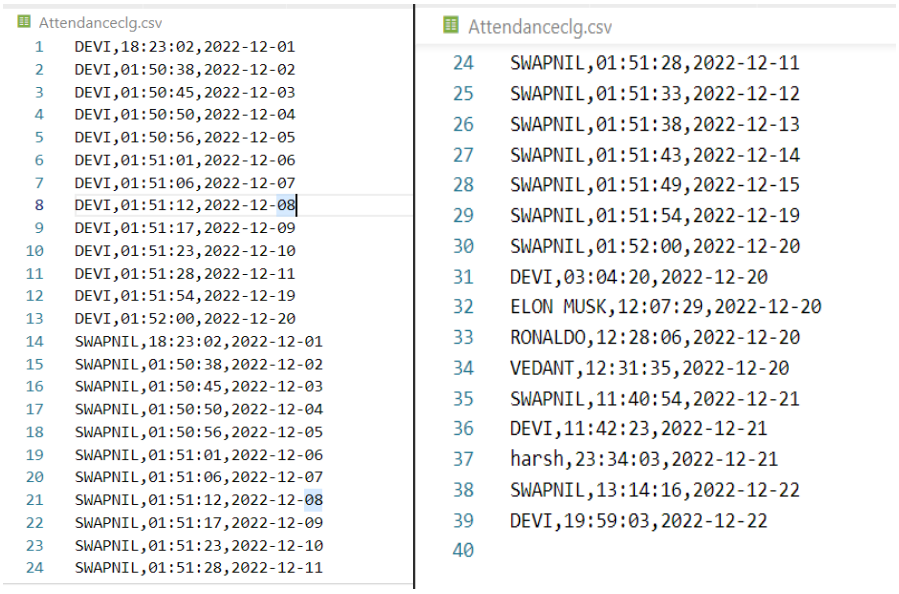

Fig. 10 CSV file of stored data of students with date and time

If we calculate the percentage of student DEVI from data given in csv file is (16/22)X100 = 72.7% , and of student SWAPNIL is (19/22)X100 = 86.3%

4. CONCLUSION & FUTURE SCOPE

CONCLUSION

Number of modules are available in our project to achieve incredible number of tasks. The best thing about our project is that you can view the flow of data from one block to other and have more freedom to make changes according to your requirements. The Automatic Class Attendance System implemented in this project would be much more difficult if it was not implemented on our project. The objective of class attendance system is to automate the time consuming and error prone attendance system. There are always limitations of every system. One can only have fixed number of students and provide less freedom to have interclass attendance system. This means the attendance system for one class can't be used for attendance system of other class. One must change programming to do this.

FUTURE SCOPE

The system can be made more flexible and scalable using these recommendations. Please note that the system implemented here is just a prototype of idea presented via this project. The recommendations are as follows:

- The system can be extended to more number of students with freedom to change list of students according to class changes.
- The system can be made more flexible to allow updating of templates in case student incurs significant amount of change in his facial features.
- The system can also be extended to allow better face recognition algorithm in which even rotational features of face can be detected efficiently.

5. REFERENCES

1) **https://github.com/askitlouder#:~:text=https%3A//www.youtube.com/channel/UCpWIxCtUo2IHh9d3fiyMaJw/playlists - ask it loud , youtube sorce**

2) **https://www.computervision.zone/about-me/ - CV Zone, Murtaza Hassan**

3) **https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78 - Modern Face Recognition with Deep Learning**

4) **https://addepto.com/blog/the-use-of-opencv-in-image-processing-5-examples/ - The use of opencv in image processing**

6. APPENDIX

➢ Code used in project

```
import cv2
import numpy as np
import face_recognition
import os
import time
from datetime import datetime
from datetime import date

path = "E:\ImagesAttendance"
images = []
classNames = []
myList = os.listdir(path)
print(myList)
for cl in myList:
    curImg = cv2.imread(f'{path}/{cl}')
    images.append(curImg)
    classNames.append(os.path.splitext(cl)[0])
print(classNames)


def findEncodings(images):
  encodeList = []
  for img in images:
     img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
     encode = face_recognition.face_encodings(img)[0]
     encodeList.append(encode)
  return encodeList

global nameList
#ATTENDANCE BLOCK//////////////////////////////////
def markAttendance(name):
  with open('Attendanceclg.csv','r+') as f:
     myDataList = f.readlines()
     global nameList
     nameList = []
     for line in myDataList:
       entry = line.split('-')
       ent= entry[0]
       entry2 = ent.split(',')
```

```
        global ct
        ct=entry2[0]+","+entry[2]  #entry2 contain date
        nameList.append(ct)
      today= date.today()
      string=str(today)
      s1=string[8:10]
      name2=name+","+s1+'\n'
      global sh
      if name2 not in nameList:
        sh = True
        time.sleep(4)
        now = datetime.now()
        dtString = now.strftime('%H:%M:%S')
        print(today)
        today= date.today()
        string1=str(today)
        f.writelines(f'{name},{dtString},{string1}\n')
        print ("Attendance has been Submitted sucessfully")
        return(True)
      else :
        sh = False
        print ("ERROR! (Today Attendance submitted)")
        return(False)


  #PECENTAGE BLOCK//////////////////
def Attendanceper(name):
    datelist=[]
    namelist=[]
    for line in nameList:
      split=line.split('\n')
      split1=split[0]
      split2=split1.split(',')
      split3=split2[1]
      splitn=split2[0]
      namelist.append(splitn)
      datelist.append(split3)
    z=len(datelist)-1
    v=int(datelist[z])-int(datelist[0])+1
    x=name
    def countX(namelist,x):
        return namelist.count(x)
    m=countX(namelist,x)
    print(m)
    print(v)
    global Per
    per = float(m/v)*100
    return per
```

```
encodeListKnown = findEncodings(images)
print('Encoding Complete')
camera = "http://10.153.193.186:8080/video"
#camera = "http://192.168.166.137:8080/video"
cap = cv2.VideoCapture(0)
cap.set(cv2.CAP_PROP_FPS,30)
cap.open(camera)
#print("check==",cap.isOpened())
#fourcc = cv2.VideoWriter_fourcc(*'XVID')
#out = cv2.VideoWriter("E:\\ouyput.avi",fourcc,20.0,(640,480))

while True:
    success, img = cap.read()
    #img = captureScreen()
    imgS = cv2.resize(img,(0,0),None,0.25,0.25)
    imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)

    facesCurFrame = face_recognition.face_locations(imgS)
    encodesCurFrame = face_recognition.face_encodings(imgS,facesCurFrame)

    for encodeFace,faceLoc in zip(encodesCurFrame,facesCurFrame):
     matches = face_recognition.compare_faces(encodeListKnown,encodeFace)
     faceDis = face_recognition.face_distance(encodeListKnown,encodeFace)
     print(faceDis)
     matchIndex = np.argmin(faceDis)


     if matches[matchIndex]:
          name = classNames[matchIndex].upper()
          y1,x2,y2,x1 = faceLoc
          y1, x2, y2, x1 = y1*4,x2*4,y2*4,x1*4
          cv2.rectangle(img,(x1,y1),(x2,y2),(0,255,0),2)
          cv2.rectangle(img,(x1,y2-35),(x2,y2),(0,255,0),cv2.FILLED)
          cv2.putText(img,name,(x1+6,y2-
6),cv2.FONT_HERSHEY_DUPLEX,1,(255,255,255),2)
          date_data = "Date: "+str(datetime.now())
          cv2.putText(img,date_data,(20,40),cv2.FONT_HERSHEY_SIMPLEX,1,(0,0,0),
1,cv2.LINE_AA)
          cv2.putText(img,"SMART ATTENDANCE
WINDOW",(140,460),cv2.FONT_HERSHEY_COMPLEX_SMALL,1,(0,0,0),1,cv2.LINE_AA)
          cv2.putText(img,"STATUS:-
",(20,80),cv2.FONT_HERSHEY_COMPLEX_SMALL,1,(0,0,0),1,cv2.LINE_AA)
          cv2.putText(img,"TOTAL ATTENDANCE:-
",(20,100),cv2.FONT_HERSHEY_COMPLEX_SMALL,1,(0,0,0),1,cv2.LINE_AA)
          markAttendance(name)
          Attendanceper(name)
          num = Attendanceper(name)
```

```
            act=str(num)+"%"
            cv2.putText(img,act,(300,100),cv2.FONT_HERSHEY_COMPLEX_SMALL,1,(255,0
,0),1,cv2.LINE_AA)
            if sh :
              cv2.putText(img,"Attendance
Submitted...",(150,80),cv2.FONT_HERSHEY_COMPLEX_SMALL,1,(37,120,5),1,cv2.LINE_A
A)

            else :
              cv2.putText(img,"Attendance already
submitted...",(150,80),cv2.FONT_HERSHEY_COMPLEX_SMALL,1,(0,0,255),1,cv2.LINE_AA
)

            print(name)

      else:
            y1,x2,y2,x1 = faceLoc
            y1, x2, y2, x1 = y1*4,x2*4,y2*4,x1*4
            cv2.rectangle(img,(x1,y1),(x2,y2),(0,0,255),2)
            cv2.rectangle(img,(x1,y2-35),(x2,y2),(0,0,255),cv2.FILLED)
            cv2.putText(img,"NOTmatch",(x1+6,y2-
6),cv2.FONT_HERSHEY_COMPLEX,1,(255,255,255),2)
      img=cv2.copyMakeBorder(img,5,5,5,5,cv2.BORDER_CONSTANT,value=[196,29,29])

      cv2.imshow('Webcam',img)
      if cv2.waitKey(25)==27:
            break

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