PROJECT REPORT

on

Attendance Management System using Face Recognition

REPORT SUBMITTED

to

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

for the

PBL OF COMPUTER FUNDAMENTAL

in

ENGINEERING AND APPLIED SCIENCE DEPARTMENT

by

Class: FY Division: P Batch: P2

Name	Roll No	PRN	Contact
Sanskriti Takale	1632	22210577	9699980279
Sai Nangare	1639	22210801	7249517720
Amruta Patil	1642	22210865	8999624843
Akanksha Sarvade	1644	22210945	8329521498
Shejal Rajgade	1646	22210977	8010580827

Batch Teacher: Mrs. Kalpana Pardeshi

INDEX

ABSTRACT

Today, almost the whole world is connected to the Internet. All the digital devices are connected to the Internet which infuses work easier for the people. Nowadays, many of the devices are being developed using the Internet of Things (IoT), computing, image processing, and machine learning.

This attendance management system had been developed to appraise the attendance of the student and recognition of the student faces for marking up the attendance.

This system is enacted to form a classroom attendance system that uses the concept of face recognition as today's manual attendance systems become more time-consuming and cumbersome to keep up properly.

A database of all the students with their details is stored in a CSV file, and attendance is recorded when the face that is recognized by the system is available in the database.

The system is designed and developed in python language. It has its own face recognition method and listening features using the Local Binary Pattern Histogram (LBPH) Algorithm within the project of the OpenCV library.

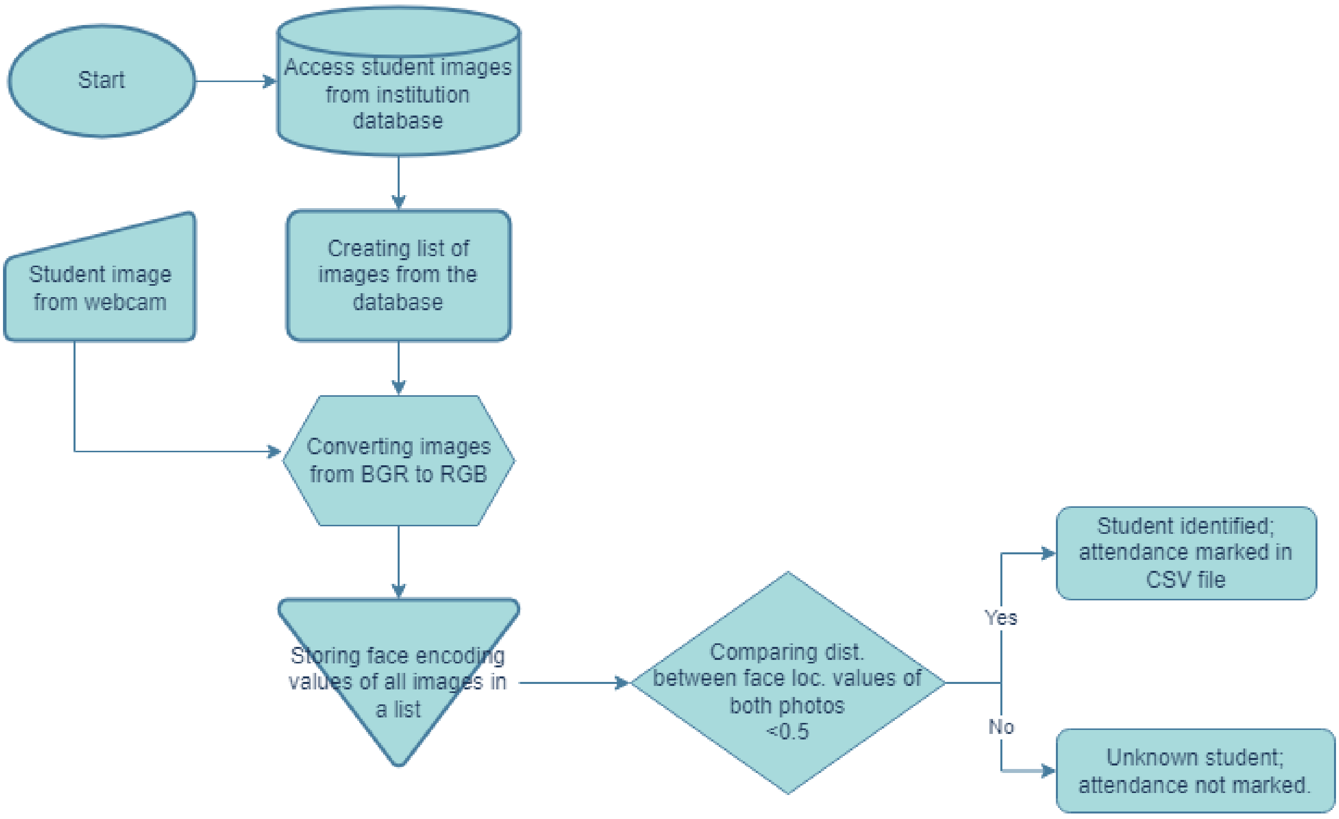
INTRODUCTION

- Face Recognition is a biometric method of identifying an individual by comparing live capture or digital image data with the stored record for that person.
- Face Recognition Attendance System is marking of attendance based on this technology.
- It provides an automated attendance system that is practical. reliable and eliminates disturbance and time-loss of traditional attendance systems.
- This system can accurately evaluate student's performance depending on their recorded attendance rate.

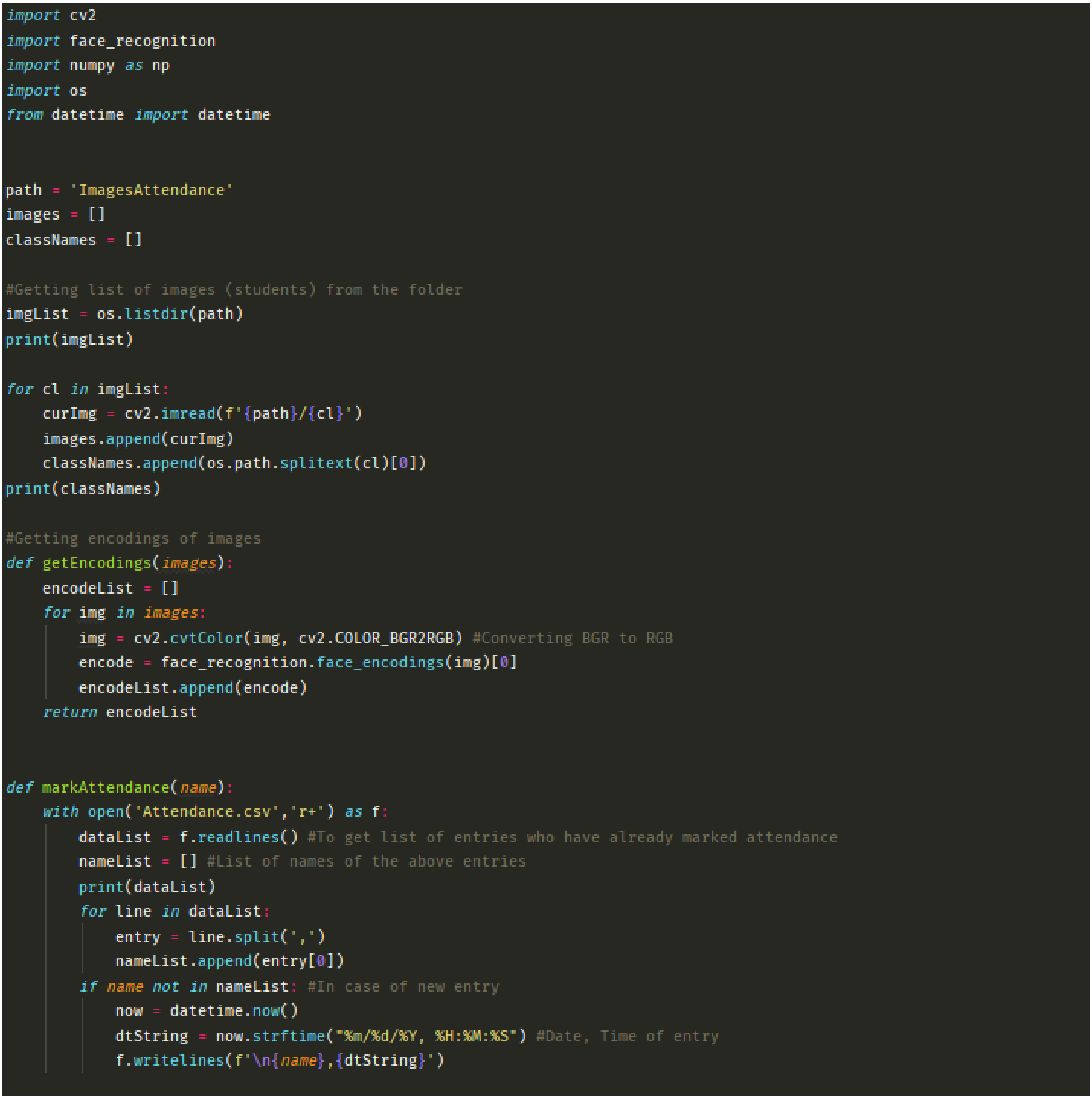
EXPERIMENTAL DETAILS

- Algorithm
  1. Create a list of images of students from the database.
  2. Get encodings of the images, convert them from BGR to RGB.
  3. Capture an image of the student using webcam and obtain encoding of the same.
  4. Compare encoding of image obtained via webcam with encoding of image available in database. Find distance between the two images.
  5. If distance between the two images is <0.5, append name of student, date and time of attendance in a csv file.

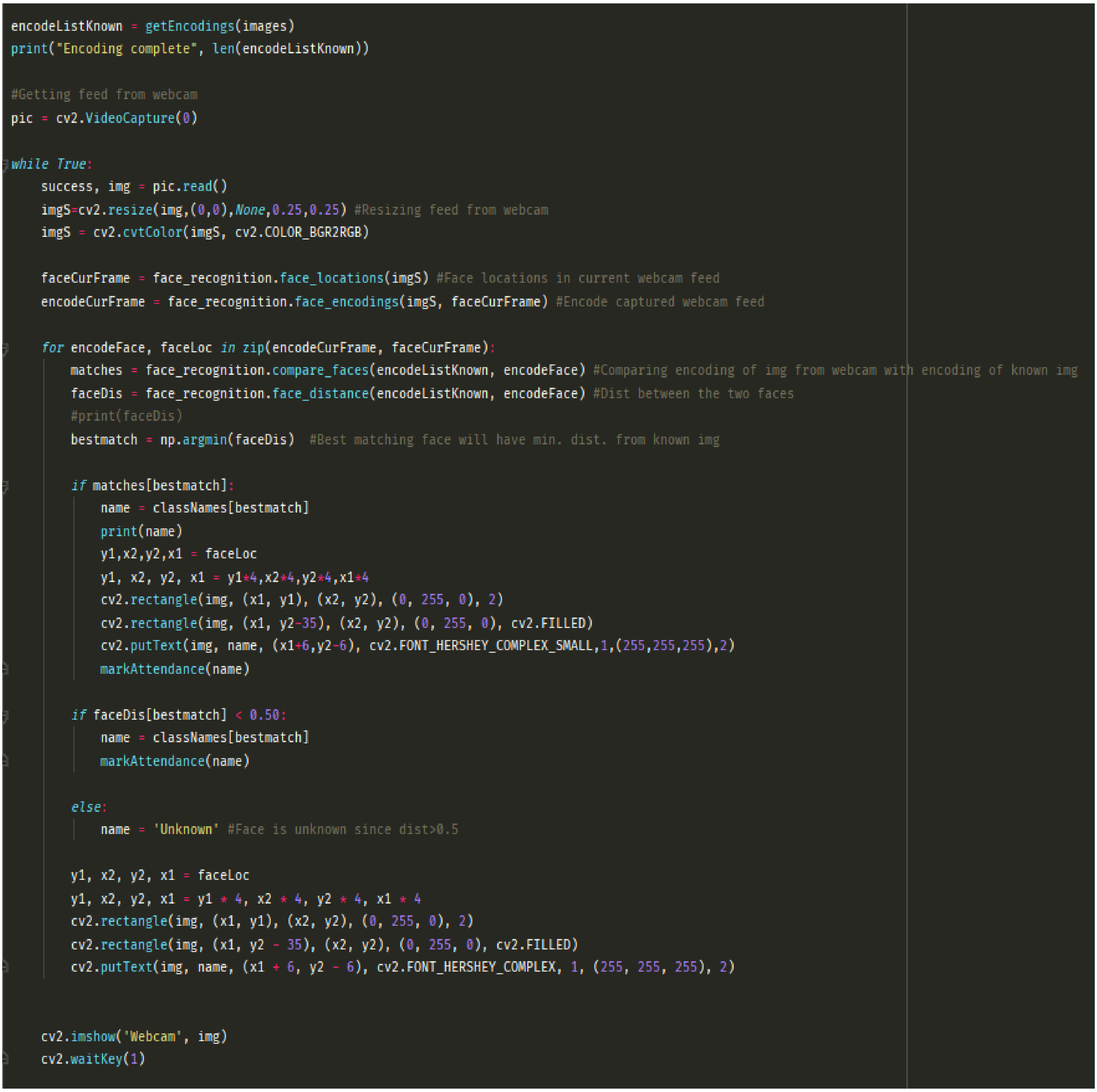
- Flowchart


Start
Access student images from institution database
Student image from webcam
Creating list of images from the database
Converting images from BGR to RGB
Storing face encoding values of all images in a list
Comparing dist. between face loc. values of both photos <0.5
Yes
No
Student identified; attendance marked in CSV file
Unknown student; attendance not marked.

- Program

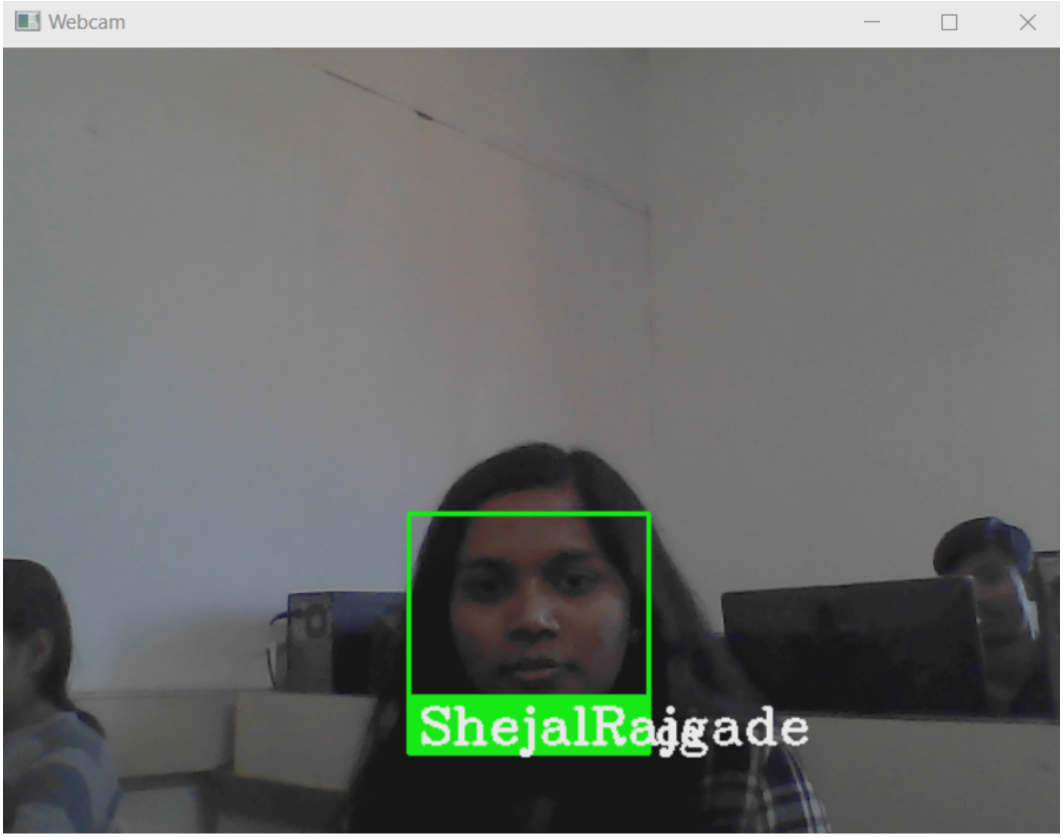


```
import cv2
import face_recognition
import numpy as np
import os
from datetime import datetime


path = 'ImagesAttendance'
images = []
classNames = []

#Getting list of images (students) from the folder
imgList = os.listdir(path)
print(imgList)

for cl in imgList:
    curImg = cv2.imread(f'{path}/{cl}')
    images.append(curImg)
    classNames.append(os.path.splitext(cl)[0])
print(classNames)

#Getting encodings of images
def getEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB) #Converting BGR to RGB
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    return encodeList


def markAttendance(name):
    with open('Attendance.csv','r+') as f:
        dataList = f.readlines() #To get list of entries who have already marked attendance
        nameList = [] #List of names of the above entries
        print(dataList)
        for line in dataList:
            entry = line.split(',')
            nameList.append(entry[0])
        if name not in nameList: #In case of new entry
            now = datetime.now()
            dtString = now.strftime("%m/%d/%Y, %H:%M:%S") #Date, Time of entry
            f.writelines(f'\n{name},{dtString}')
```

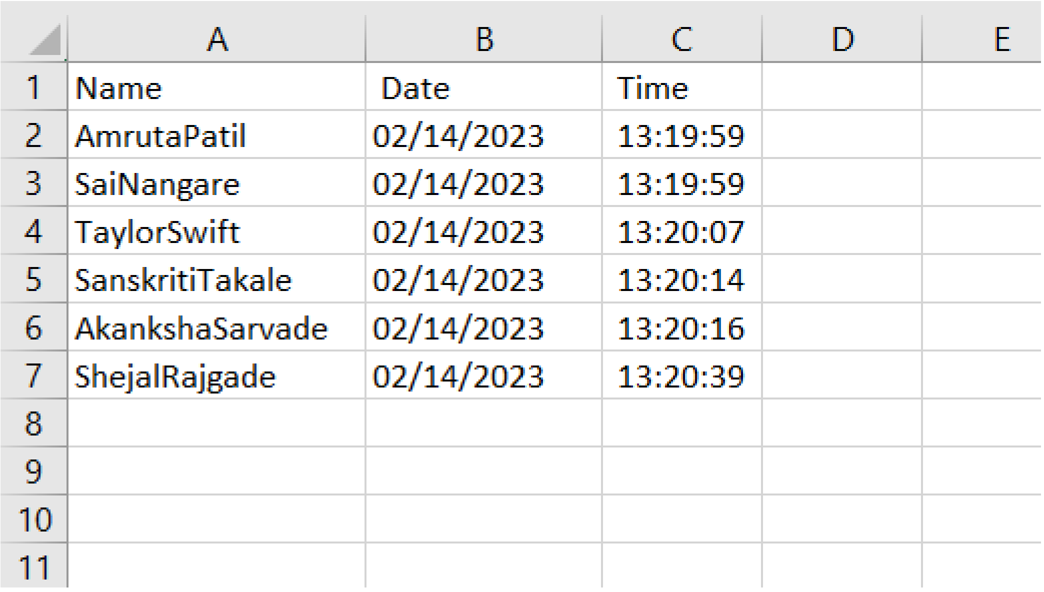


```python
encodeListKnown = getEncodings(images)
print("Encoding complete", len(encodeListKnown))

#Getting feed from webcam
pic = cv2.VideoCapture(0)

while True:
    success, img = pic.read()
    imgS=cv2.resize(img,(0,0),None,0.25,0.25) #Resizing feed from webcam
    imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)

    faceCurFrame = face_recognition.face_locations(imgS) #Face locations in current webcam feed
    encodeCurFrame = face_recognition.face_encodings(imgS, faceCurFrame) #Encode captured webcam feed

    for encodeFace, faceLoc in zip(encodeCurFrame, faceCurFrame):
        matches = face_recognition.compare_faces(encodeListKnown, encodeFace) #Comparing encoding of img from webcam with encoding of known img
        faceDis = face_recognition.face_distance(encodeListKnown, encodeFace) #Dist between the two faces
        #print(faceDis)
        bestmatch = np.argmin(faceDis)  #Best matching face will have min. dist. from known img

        if matches[bestmatch]:
            name = classNames[bestmatch]
            print(name)
            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_COMPLEX_SMALL,1,(255,255,255),2)
            markAttendance(name)

        if faceDis[bestmatch] < 0.50:
            name = classNames[bestmatch]
            markAttendance(name)

        else:
            name = 'Unknown' #Face is unknown since dist>0.5

        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_COMPLEX, 1, (255, 255, 255), 2)


    cv2.imshow('Webcam', img)
    cv2.waitKey(1)
```

- Example:



- CSV file log:



	A	B	C	D	E
1	Name	Date	Time		
2	AmrutaPatil	02/14/2023	13:19:59		
3	SaiNangare	02/14/2023	13:19:59		
4	TaylorSwift	02/14/2023	13:20:07		
5	SanskritiTakale	02/14/2023	13:20:14		
6	AkankshaSarvade	02/14/2023	13:20:16		
7	ShejalRajgade	02/14/2023	13:20:39		
8					
9					
10					
11					

RESULTS

1. Image of student is captured via a webcam and encoding of the same is compared with encoding of image of student available in the database.
2. On confirming student details, the CSV file gets appended with Name of student, Date and Time of attendance.
3. The obtained CSV file is the final attendance sheet for the given class.

CONCLUSION

Thus, we have successfully created a program to mark attendance using face_recognition library in python. Using this library we can identify images captured with the webcam, recognise and compare it with images available in our database and mark the attendance of the student.

This is a highly efficient and time-conserving way of recording attendance with no room for errors.

REFERENCES

1. FACE RECOGNITION + ATTENDANCE PROJECT | OpenCV Python | Computer Vision
   https://www.computervision.zone/topic/attendance-project/
2. Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning
   https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78
3. Face Recognition: Understanding LBPH Algorithm
   https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b