In [3]:

```
import cv2 #This is the OpenCV library, used for image processing and c
import numpy as np
import face_recognition
import os
from datetime import datetime

# the directory variable stores the path where images and attendance re
directory = r'C:\WORK\Facial recognition'
if not os.path.exists(directory):
    os.makedirs(directory)

path = os.path.join(directory, 'images')#The subdirectory where face im
images = []#A list to hold the loaded 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):#A function that converts each image in the i
    encodeList = []#The encoding represents the features of a face that
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    return encodeList

def markAttendance(name):#unction that logs the name and the current ti
    file_path = os.path.join(directory, 'Attendance.csv')

    # Check if the file exists
    if not os.path.exists(file_path):
        with open(file_path, 'w') as f:
            f.write('Name,Time\n')  # Write the header if the file is r

    # Open the file in append mode
    with open(file_path, 'r+') as f:
        myDataList = f.readlines()
        nameList = []
        for line in myDataList:
            entry = line.split(',')
            nameList.append(entry[0])
        if name not in nameList:
            now = datetime.now()
            dtString = now.strftime('%Y-%m-%d %H:%M:%S')
            f.writelines(f'{name},{dtString}\n')

encodeListKnown = findEncodings(images)#Stores the encodings of the knc
print('Encoding Complete')

cap = cv2.VideoCapture(0)#Captures video from the webcam

while True:
    success, img = cap.read()

    if not success or img is None:
        print("Failed to capture image. Exiting...")
        break
```

```
    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, facesCurFra

    for encodeFace, faceLoc in zip(encodesCurFrame, facesCurFrame):#For
        matches = face_recognition.compare_faces(encodeListKnown, encod
        faceDis = face_recognition.face_distance(encodeListKnown, encod
        matchIndex = np.argmin(faceDis)
# Finds the index of the known face with the smallest distance to the c
        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), cv
            cv2.putText(img, name, (x1 + 6, y2 - 6), cv2.FONT_HERSHEY_C
            markAttendance(name)

    cv2.imshow('Webcam', img)#Displays the current frame with the recog
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

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

```
['DR.manjunatha.jpg', 'Mr.Abhimanyu.jpg', 'Mr.alex.jpg', 'Mr.allen.jp
g', 'Mr.deepak.jpg', 'Mr.divin.jpg', 'Mr.imad.jpg', 'mr.joseph.jpg',
'Mr.labeeb.jpg', 'Mr.williams.jpg', 'reji.jpg']
['DR.manjunatha', 'Mr.Abhimanyu', 'Mr.alex', 'Mr.allen', 'Mr.deepak',
'Mr.divin', 'Mr.imad', 'mr.joseph', 'Mr.labeeb', 'Mr.williams', 'rej
i']
Encoding Complete
```

In [ ]:

In [ ]: