*Internet Of Things*

*Project*

*Automated Attendance System Based On*  *Face Recognition(OpenCV)*

By Team 12

1) Battu Priyanka

2)Venkatasiri Suvva

3)Deekshit Kumar Reddy Chevuru

4)Bala sankararao Pulimi

***1) INTRODUCTION***

**1.1 Overview**

Attendance system is a necessary tool for taking attendance in any environment ,where attendance is critical.However the current method that institutions uses is faculty passes an attendance sheet (or) make roll calls and mark the attendance of the students,which some times disturbs the discipline of the class and this sheet further goes to the admin department, which is updated to an excel sheet .This process is quite hectic and time-consuming.So,this project is aimed at developing a less instrusive, cost effective and more efficient automated student attendance management system using face recognition from visual recognition through open cv.

**1.2 Purpose**

The basic developing of an automatic attendance management system is to computarize the standard method of taking attendance . Automated attendance system of face detection and recognition is to present face recognition in real time environment for educational institutes (or) an organisation to see and mark attendance of their students and employees on a daily basis to keep track of their presence.This project is developed for deploying an easy and a secure way of taking down attendance.We can use this in colleges,companies etc. In this,the device is equipped with a camera.Whenever the camera detects a face it captures the image.The captured image will be given to face recognisation model to recognise the person.

when the match is found the attendance of that particular person is stored and the attendance will be marked.We can also intimate the persons,who are absent by sending messages through respective mobile numbers.Our approach focuses on cost effective,more efficient and less intrusive and easy deployment in low computational environment .Our system achieved a testing accuray of 90% ,we can improve effeciency by training the model with more and more pics .

***2) LITERATURE SURVEY***

**2.1 Existing problem**

The existing system requires manual work.In addition to this,there is also a chance of forgery (one person signing the persence of the other one).More calculatons and man power is required to calculate the attendance.It is difficult to maintain a register in manual systems.

2.2 **Proposed Solution**

* The proposed solution is marking attendance of the persons via face id.
* It will detect the faces via wire less camera/web cam and then recognise the faces.
* After recognition it will mark the attendance of the recognised students and update the attendance record.

***3) THEORITICAL ANALYSIS***

***3.1* Block diagram**

Ibm Cloud,

Iot platform,

Node-red,

visual recog

Logitech Camera

Raspberry pi

Mit App

***3.2 Hardware/software designation***

We developed this application by using some software designations

1)IBM Cloud

(i). iot platform

(ii). visual Recognizer

(iii). Node-red

2)Python software

3)OpenCv

4)Fast2 SMS service

5)MIT APP inventer

***4) EXPERIMENTAL INVESTIGATIONS***

While performing this experiment (or) project, we will investigate about the face recognisation and the code we had written for the automated attendance display.By comparing the faces which were detected and the images which were trained by the visual recognition, a message will be sent to the parents of students who were absent to the class by using "Fast2SMS" service.

***5) FLOW CHART***

start

stop

retrieve data in mit and display

by node-red create http request

send data to devise in ibm iot platform

sending sms to parents using fast2Sms

web cam to capture image

passing image to ibm visual recog

if student

increment no of days present

present

absent

***6) RESULT***

By using this project we had captured the images and sent them for visual recognisation ,from the output of visual recognisaion we found the students who were present and absent to the class,for the students who were present their attendance is updated and displayed in app provided to admin,to the students who were absent a message had been sent to their parents about their absence for the class.

***7) ADVANTAGES & DISADVANTAGES***

**7.1 Advantages**

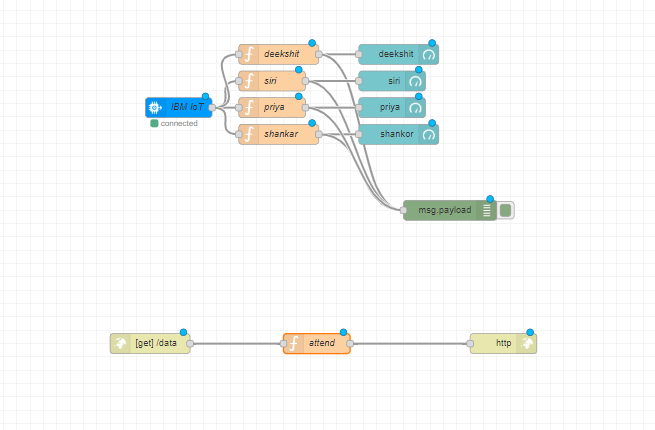
* The integration process is easy and flawless.
* High accuracy allows avoiding false identification.
* Facial recognition system is full automated.
* Time fraud will be excluded.
* Man power is less.

**7.2 Disadvantages**

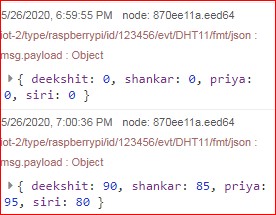
* Difficulties with data processing and storing.
* Troubles with images size and quality.
* Strong influence of camera angle.

***8) APPLICATIONS***

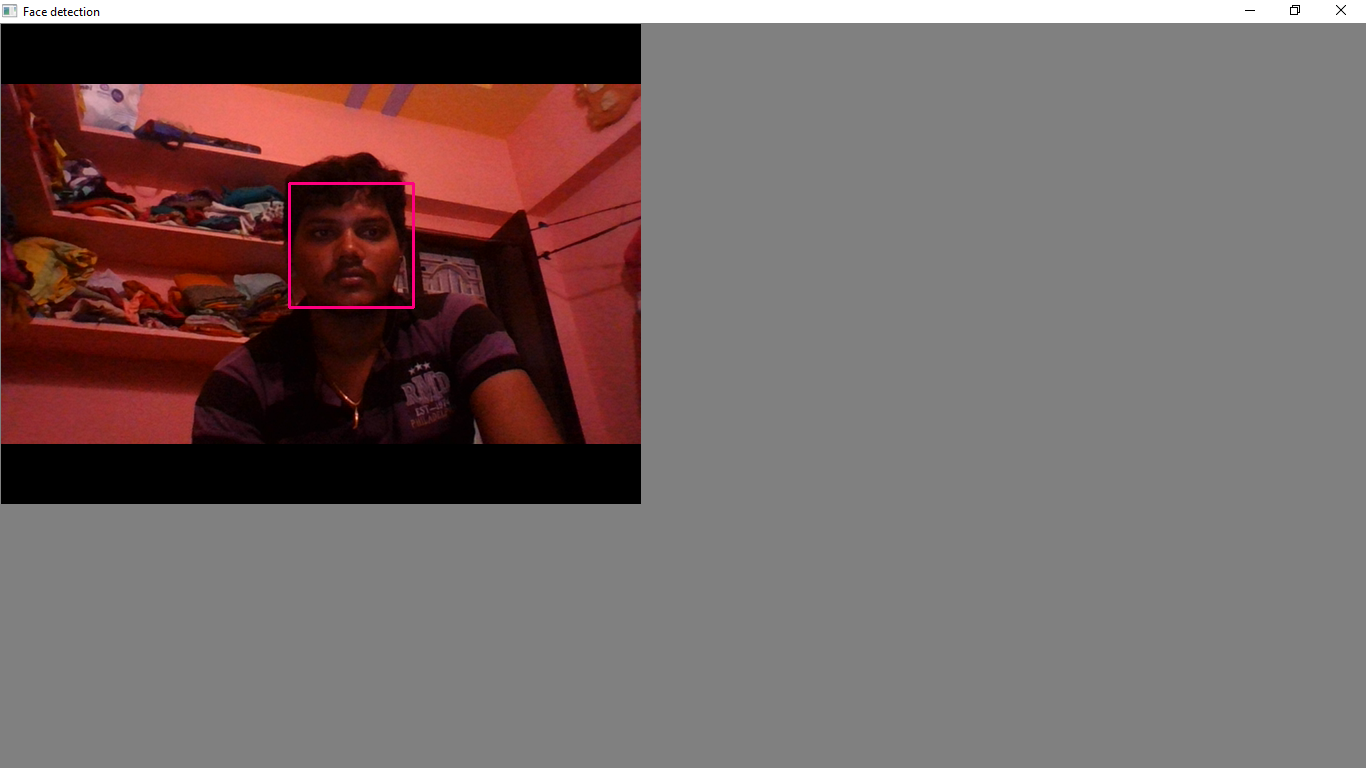
* Can record data who are entering into an instution or organisation.
* Can record attendance flawlessly.



***Fig:Node-red flow***

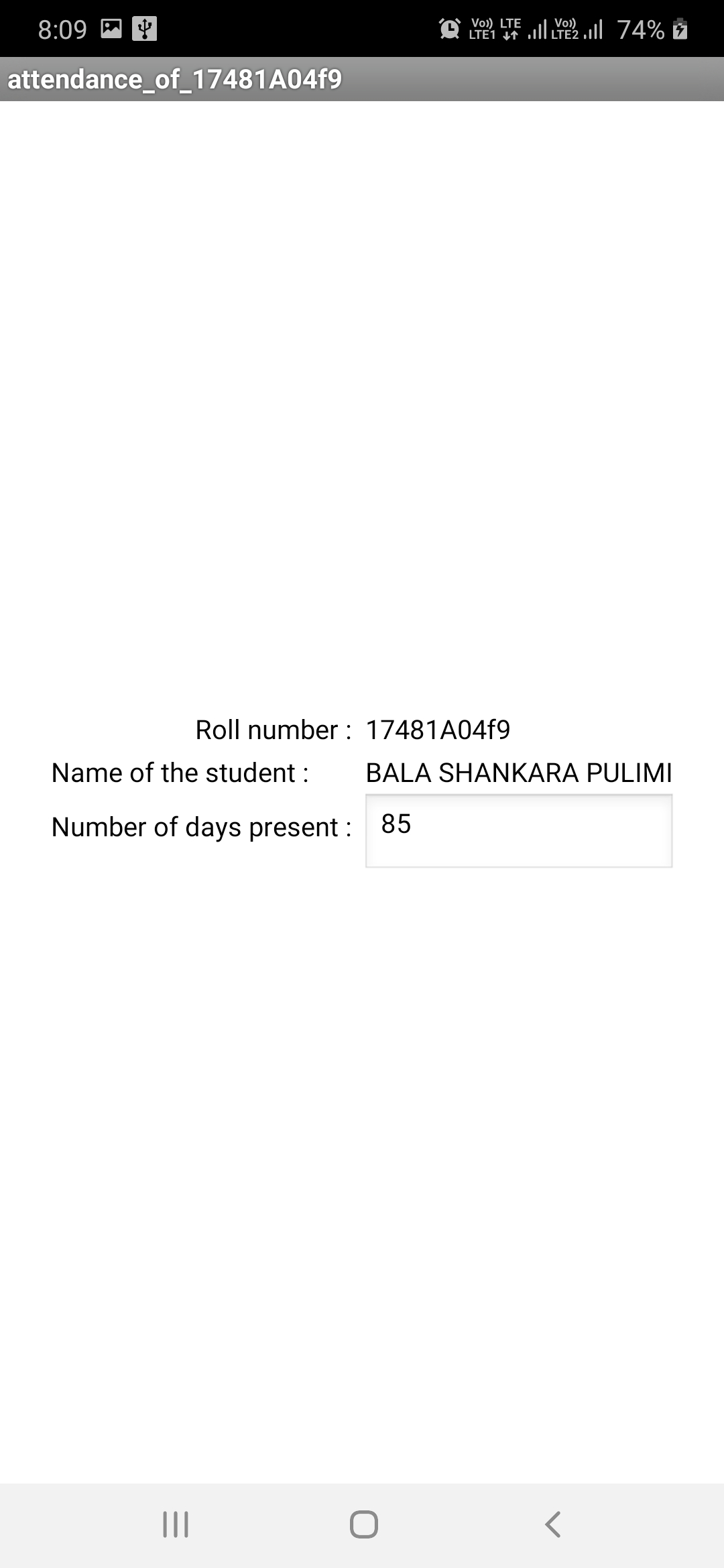


***Fig:Getting data from device to node-red***



***Fig: Detecting and capturing image***

***Fig:App student Roll no page***



***Fig:Student Attendance details***

***9) Conclusion***

Face recognition technologies have been associated generally with very costly top secure applications. Today the core technologies have evolved and the cost of equipments is going down dramatically due to the integration and the increasing of face recognition technology are now cost effective ,relable and highly accurate.

***10) Future scope***

The system we have developed has successfully able to acomplish the task of marking the attendance in the classroom automatically and output obtained in an excel sheet as desired in real time .Another important aspect where we can work is towards creating an online data base of the attendance and its automatic updating ,keeping in mind growing popularity of internet of things.

***11) Bibilography***

* <https://deekshit7.eu-gb.mybluemix.net/red/#flow/c0ca49bc.b76048>
* <https://cloud.ibm.com/>
* <https://github.com/>
* <https://ai2.appinventor.mit.edu/#5533567421906944>

***12) Appendix***

***import cv2***

***import requests***

***import time***

***import sys***

***import ibmiotf.application***

***import ibmiotf.device***

***import random***

***import numpy as np***

***import datetime***

***import cv2***

***import numpy as np***

***import datetime***

***import time***

***import json***

***from watson\_developer\_cloud import VisualRecognitionV3***

***from ibm\_watson import TextToSpeechV1***

***from ibm\_cloud\_sdk\_core.authenticators import IAMAuthenticator***

***def imgcapture():***

***face\_classifier=cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")***

***eye\_classifier=cv2.CascadeClassifier("haarcascade\_eye.xml")***

***#It will read the first frame/image of the video***

***video=cv2.VideoCapture(0)***

***pr=set()***

***while True:***

***#capture the first frame***

***check,frame=video.read()***

***gray=cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)***

***#detect the faces from the video using detectMultiScale function***

***faces=face\_classifier.detectMultiScale(gray,1.3,5)***

***#print("face detected")***

***eyes=eye\_classifier.detectMultiScale(gray,1.3,5)***

***#drawing rectangle boundries for the detected face***

***for(x,y,w,h) in faces:***

***cv2.rectangle(frame, (x,y), (x+w,y+h), (127,0,255), 2)***

***cv2.imshow('Face detection', frame)***

***picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M-%S")***

***cv2.imwrite(picname+'.jpg',frame)***

***pr.add(picname)***

***#drawing rectangle boundries for the detected eyes***

***for(ex,ey,ew,eh) in eyes:***

***cv2.rectangle(frame, (ex,ey), (ex+ew,ey+eh), (127,0,255), 2)***

***cv2.imshow('Face detection', frame)***

***#waitKey(1)- for every 1 millisecond new frame will be captured***

***Key=cv2.waitKey(1)***

***if Key==ord('q'):***

***#release the camera***

***video.release()***

***#destroy all windows***

***cv2.destroyAllWindows()***

***break***

***return pr***

***def visualrecog(pr):***

***r=set()***

***for i in pr:***

***visual\_recognition = VisualRecognitionV3(***

***'2018-03-19',***

***iam\_apikey='9Mh1HmR5OvtAGwybq3sLW5ywdmpajUnMDvak529p1D5P')***

***with open(f'./{i}.jpg', 'rb') as images\_file:***

***classes1 = visual\_recognition.classify(***

***images\_file,***

***threshold='0.6',***

***classifier\_ids='project\_1557605068').get\_result()***

***print(json.dumps(classes1, indent=2))***

***w=classes1['images'][0]['classifiers'][0]['classes'][0]['class']***

***r.add(w)***

***return r***

***def message(a):***

***for i in a:***

***p=requests.get(f'https://www.fast2sms.com/dev/bulk?authorization=FjVvsaXLCQurTmJWE5o9ZR31IPhDY6KetNbnMyl2SHqz8kiB4OGKjBpeUilDonZWCOLzgHYthRk61bIw&sender\_id=FSTSMS&message=your%20ward%20{i}%20is%20absent&language=english&route=p&numbers=7330605911,9959653574')***

***#h=requests.get('https://www.fast2sms.com/dev/bulk?authorization=FjVvsaXLCQurTmJWE5o9ZR31IPhDY6KetNbnMyl2SHqz8kiB4OGKjBpeUilDonZWCOLzgHYthRk61bIw&sender\_id=FSTSMS&message=hello%20how%20are%20you&language=english&route=p&numbers=7330605911')***

***print(p.status\_code)***

***def iotplat(d,s,p,si):***

***print("entered")***

***#Provide your IBM Watson Device Credentials***

***organization = "82mqmc"***

***deviceType = "raspberrypi"***

***deviceId = "123456"***

***authMethod = "token"***

***authToken = "7330605911"***

***# Initialize GPIO***

***def myCommandCallback(cmd):***

***print("Command received: %s" % cmd.data)***

***try:***

***deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}***

***deviceCli = ibmiotf.device.Client(deviceOptions)***

***#..............................................***

***except Exception as e:***

***print("Caught exception connecting device: %s" % str(e))***

***sys.exit()***

***# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times***

***deviceCli.connect()***

***#a={"deekshit","siri","priya","shankar","vamsi"}***

***#hum=random.randint(10,50)***

***#print(hum)***

***#temp = 5***

***m=d***

***q=s***

***r=p***

***t=si***

***print(m)***

***print(q)***

***print(r)***

***print(t)***

***#Send Temperature & Humidity to IBM Watson***

***data = {'deekshit' : m, 'shankar' : q, 'priya' : r, 'siri' : t }***

***#print (data)***

***def myOnPublishCallback():***

***print("Published deekshit = %s " % m)***

***print("Published shankar = %s " % q)***

***print("Published priya = %s " % r)***

***print("Published siri = %s " % t)***

***success = deviceCli.publishEvent("DHT11", "json", data, qos=0, on\_publish=myOnPublishCallback)***

***if not success:***

***print("Not connected to IoTF")***

***time.sleep(2)***

***deviceCli.commandCallback = myCommandCallback***

***# Disconnect the device and application from the cloud***

***#deviceCli.disconnect()***

***d=0***

***s=0***

***p=0***

***si=0***

***for x in range(0,1):***

***pr=imgcapture()***

***print(pr)***

***r=visualrecog(pr)***

***print(r)***

***t={"deekshit","priya","shankar","siri"}***

***a=t.difference(r)***

***message(a)***

***if 'deekshit' in r:***

***d=d+1***

***if 'shankar' in r:***

***s=s+1***

***if 'priya' in r:***

***p=p+1***

***if 'siri' in r:***

***si=si+1***

***print(d)***

***print(s)***

***print(p)***

***print(si)***

***iotplat(d,s,p,si)***

***print("over")***