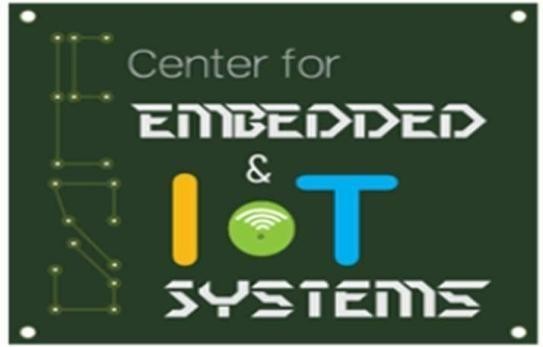
# SMART ATTENDANCE DISPLAY SYSTEM



A project report submitted in partial fulfilment of requirements for the course on Distributed IoT Systems

CHARISHMA. V (20K41A0410)

SHIVA PRIYA.P (19K41A04A9)

NIHAL RAJ.D (19K41A0497)

RAHUL.B (19K41A0495)

Under the guidance of

**Mr. P. Ramchandar Rao**

Assistant Professor

Member, Center for Embedded Systems and IoT Department of ECE



Warangal Urban, Telangana, India -506371

**S R ENGINEERING COLLEGE**

Ananthasagar, Hasanparthy, Warangal



**CERTIFICATE**

This is to certify that the course project entitled **“SMART ATTENDANCE DISPLAY SYSTEM”** is the bonafied work carried out by V. Charishma (20K41A0410), P.Shiva Priya (19K41A04A9), D.Nihal (19K41A0497),

B.Rahul (19K41A0495), in the partial fulfilment of the requirement for the award of course **Distributed IoT Systems** during the academic year 2021-2022 under the guidance and supervision.

**Mr. P. Ramchandar Rao**

Assistant Professor & Center for Embedded Systems and IoT

Department of ECE

### Chapter no Title Page.No

1. INTRODUCTION 5
   1. 1.1 [Literature review 5](#_TOC_250017)
2. METHODOLOGY 6
   * 1. 2.1 [Existing Product 1 6](#_TOC_250016)
     2. 2.1.2 [Limitations 6](#_TOC_250015)

[2.2 Existing Product 2 6](#_TOC_250014)

* + 1. 2.2.1 [Existing Product 3 8](#_TOC_250013)
    2. 2.2.2 [Limitations 8](#_TOC_250012)
  1. 2.3 [Proposed Method 8](#_TOC_250011)
     1. 2.4.1[Block Diagram 9](#_TOC_250010)
     2. 2.4.2 Hardware & Software 9
     3. Components
     4. 2.5 Components Description 9
        1. 2.5.1 [Esp32 Cam 9](#_TOC_250009)
        2. 2.5.2 [Ftdi Module 9](#_TOC_250008)

2.6 [Working Model 10](#_TOC_250007)

* + 1. 2.7 [Working 10](#_TOC_250006)
    2. 2.8 [Flowchart 11](#_TOC_250005)
    3. 2.9 Algorithm(steps followed) 12
    4. **3 RESULTS AND LIMITATIONS 14**
    5. 3.1 [Results 14](#_TOC_250004)
  1. 3.2 [Limitations 15](#_TOC_250003)
  2. **4 CONCLUSION AND FUTURESCOPE 16**
  3. 4.1 [Conclusion 16](#_TOC_250002)
  4. 4.2 [Futurescope 16](#_TOC_250001)

**5 REFERENCES 18**

* 1. 5.1 [REFERENCES 19](#_TOC_250000)
  2. **6**  **APPENDIX** **20**

6.1Project code 20

**ABSTRACT**

This technology is used for smart" Attendance recognition and detection in classroom" generally teachers note the attendance orally and mark on sheet, as technology is developed there must be a change in education system as well. Our product makes use of an camera component which recognizes the faces of the students and mark the attendance itself so, this makes the work faster and even simpler . In order to improve the advancement in attendance system we have introduced this technique which reduces the work of teachers in classroom.

# CHAPTER-1 INTRODUCTION

Attendance is a mandatory in every organization. Maintaining the attendance is time consuming and a huge task. We are planning to to take attendance in schools and educational institutions in a smart way using IOT

. General method is a time-consuming process and there might be some chances to miss some roll calls or student may give fake attendance and some times the records might miss. These are the problems arise in traditional method. So we are going to take attendance using automatic attendance system. Here when the teacher wanted to take the attendance as the face information of the students are stored in the software so when the student face is detected it recognizes and marks the attendance and stores it in the database and all the records are maintained. So that the time is not wasted, also no fake attendances are given.

## 1.2 LITERATURE REVIEW

This system involves micro controllers, machine learning and artificial intelligence. This system directly detects the face and marks the attendance stores in database with faculties permission.

Existing attendance marking and systems are:

1. Attendance system using finger print: Using biometric sensor the finger print of the student is sensed and the attendance is marked the drawback is hacking and time consuming.
2. Attendance system using iris detection: Iris of the student is scanned and marked it generally requires close proximity which causes discomfort and it is costly.
3. Attendance system using RFID: Every student is provided with individual electronic cards which is costlier can be used by any one and we might loose the card sometimes.
4. Attendance system using facial recognition: The student face is detected recognized and marked,it is a non time consuming process,no fake attendance can be given and it is secure.

# CHAPTER-2

## METHODOLOGY

## 2 EXISTING METHODOLOGY

### 2.1 EXISTING PRODUCT 1:

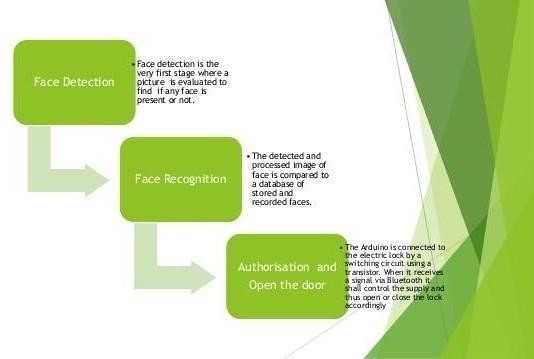
Security is at most concern for anyone nowadays, whether it's data security or security of their own home. With the advancement of technology and the increasing use of IoT, digital door locks have become very common these days. Digital lock doesn’t require any physical key but it uses RFID, fingerprint, Face ID, pin, passwords, etc. to control the door lock. In past, we have developed many digital door lock applications using these various technologies. In this tutorial we build a **Face ID controlled Digital Door lock system using ESP32-CAM**.

Fig.1:Flow chart of Face ID controlled Digital Door lock system using ESP32- CAM.

### 2.1.2 LIMITATIONS:

1. Face cannot be identified during night time.
2. Unwanted face recognition happens all the time when people are present in front of the camera
3. Every time user has to stand at particular position in order to get aligned his with the cameras.
4. Every time door gets locked even it is not necessary.

**2.2 EXISTING PRODUCT 2:**

The main heavy program will be at the server-side that is our computer, or one can even use raspberry-pi as a server. In this attendance system, we will not just **detect** the person but also **store** the information of the person detected in a Microsoft **Excel File**. Moreover, the duration of time they have stayed in the frame is also recorded into an excel sheet. The script for Face Recognition is written in the python programming language, thus we will also have to **install Python** and its required **Libraries**. The ESP32 Camera will capture image & store the information in Excel file.

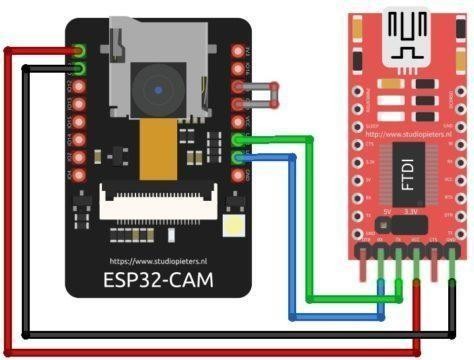


Fig.2: Schematic diagram of ESP32-cam module.

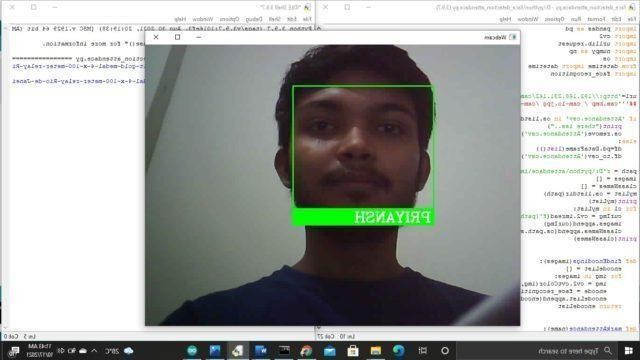


Fig.3: Student face is recognized using ESP32-cam.

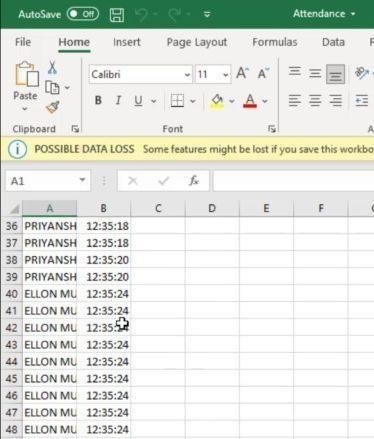


Fig.4: Students attendance information generated in Excel sheet.

### 2.3 EXISTING PRODUCT 3:

Face recognition video streaming online web server:

It is an web server used with esp32 cam, majorly used for streaming the videos online ,which are used for the communications between far distances. Generally it is an online communication.



Fig.5: ESP32-cam module Fig.6: Online video stream using ESP32- cam

### 2.3.1 LIMITATIONS:

1. It takes more memory for storing the videos.
2. It needs more amount of power consumption for the working.
3. It uses high internet bandwidth.

## 2.4 PROPOSED METHOD

### 2.4.1 BLOCK DIAGRAM

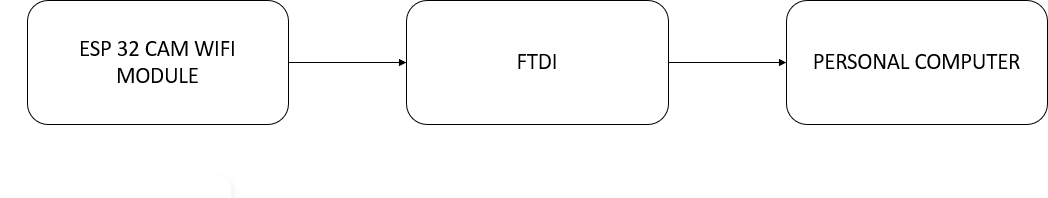


Fig no:7 Block diagram of proposed method

### 2.4.2 HARDWARE AND SOFTWARE COMPONENTS:

1. ESP32-cam module
2. FTDI module
3. Arduino IDE
4. Python 3.7.9

### 2.5 COMPONENTS DISCRIPTION:

### 2.5.1 ESP32 CAM :

The ESP32 is a very small camera module with the ESP32-S chip that costs approximately $10. Besides the OV2640 camera, and several GPIOs to connect peripherals, it also features a micro SD card slot that can be useful to store images taken with the camera or to store files to serve to clients.

* The smallest 802.11b/g/n Wi-Fi BT SoC module
* Low power 32-bit CPU, can also serve the application processor
* Up to 160MHz clock speed, summary computing power up to 600 DMIPS
* Built-in 520 KB SRAM, external 4MPSRAM
* Supports UART/SPI/I2C/PWM/ADC/DAC
* Support OV2640 and OV7670 cameras, built-in flash lamp.
* Support image Wi-Fi upload.

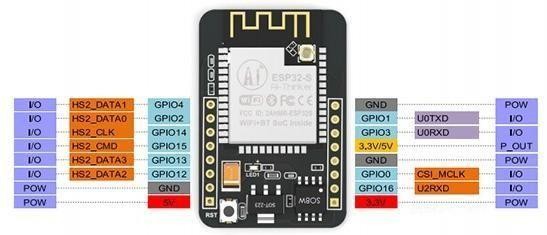


Fig.9: ESP32-cam module pinout

### 2.5.2 FTDI MODULE:

Fig.10: FTDI module

1. Included USB transceiver, without external circuit device
2. Includes a clock circuit and power-on reset circuit
3. With 3.3V and 5V dual power output
4. With three LEDs: power indicator, data reception indicator, the data transmission indicator, working status.
5. Meet the USB2.0 specification requirements
6. SUSPEND pin supports USB suspend state
7. With self-recovery fuse. In the event of the accidental short circuit, it can effectively protect your computer USB port and Down-loader
8. With reset signal output, etc. directly to the Arduino board Pro-mini download.

## 2.6 WORKING MODEL

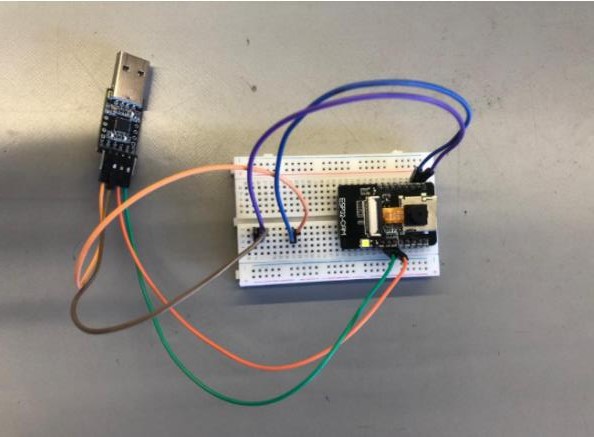


Fig.11: Working model of attendance display system using ESP32-cam

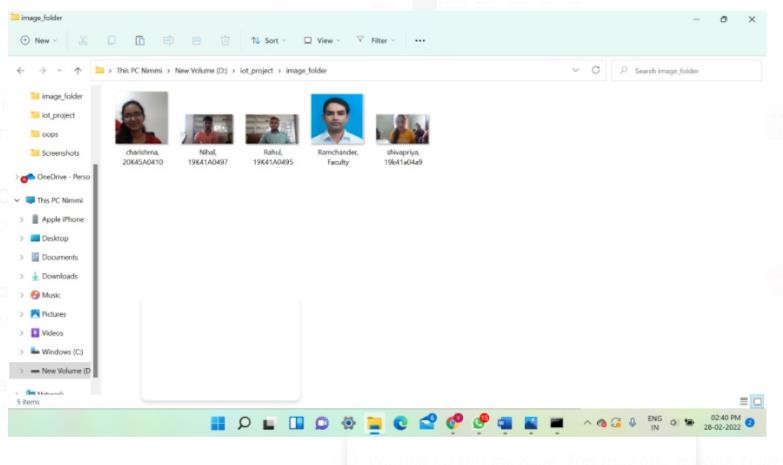


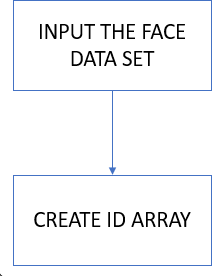
Fig.12: photos stored in the image folder with name and roll number.

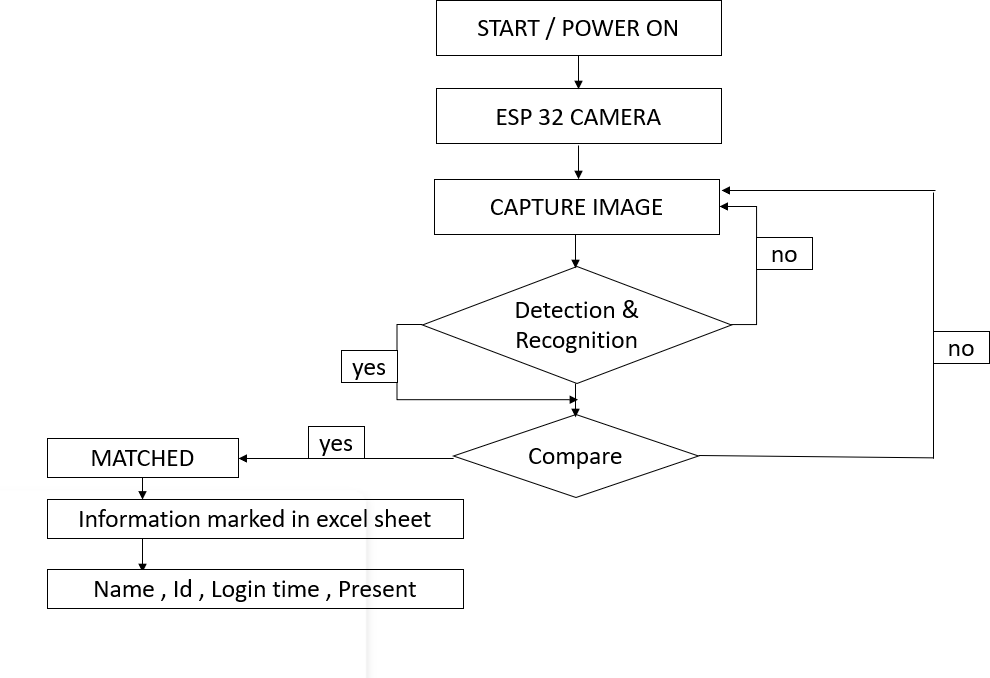
### 2.7 WORKING:

We used python to run the esp32 cam module, using Arduino ide software we uploaded the code into the module. we have stored the photos of the students with their name and roll number details in a folder.

The working of our project can be explained as, firstly after switching on the power button of the hardware every component gets activated and start working. The ESP 32 cam present in it starts detecting the faces which are in front of camera, and if the detected faces are matched with the already existing FACE DATASET, it will generate attendance in Excel-sheet as per the student details available with respect to their photos. Each and every time when you want to take the attendance we have to clear the previously generated attendance sheet because once a face is detected the name is marked on the sheet so, when the same face is detected again by the cam it will not generate the attendance again because of previous mark.so, we have to store the previously generated attendance data into other sheet every time when we want to mark attendance for the next time.

### 2.8 FLOWCHART:





**2.9 ALGORITHM:**

(steps followed to build the face recognition system)

1. Install Libraries:

pip install dlib # installing dlib

pip install face recognition # installing face recognition

pip install opencv # installing opencv

import os

from datetime import datetime

1. Import Libraries:

import cv2

import numpy as np

import face\_recognition

1. Loading images: 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)

# After importing libraries you need to load an image. face\_recognition library loads images in the form of BGR, in order to print the image you should convert it into RGB using OpenCV.

1. Find face location and draw bounding boxes:

# print(faceDis

matchIndex = np.argmin(faceDis)

if matches[matchIndex]:

name = classNames[matchIndex].upper()

# 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, 1, (255, 255, 255), 2)

markAttendance(name)

cv2.imshow('Webcam', img)

key=cv2.waitKey(5)

if key==ord('q'):

break

1. Train an image for face recognition:

#cap = cv2.VideoCapture(0)

while True:

#success, img = cap.read()

img\_resp=urllib.request.urlopen(url)

imgnp=np.array(bytearray(img\_resp.read()),dtype=np.uint8)

img=cv2.imdecode(imgnp,-1)

# 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)

1. Building a face recognition system:

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

def markAttendance(name):

with open("Attendance.csv", 'r+') as f:

myDataList = f.readlines()

nameList = []

dateList = []

for line in myDataList:

entry = line.split(',')

nameList.append(entry[0])

name\_pin = name.split(", ")

if name\_pin[0] not in nameList:

now = datetime.now()

dtString = now.strftime('%H:%M:%S')

date = now.strftime('%d/%m/%y')

mark\_at = "Present"

f.writelines(f'\n{name\_pin[0]}, {name\_pin[1]}, {date}, {dtString}, {mark\_at}')

encodeListKnown = findEncodings(images)

print('Encoding Complete')

#Resize the image by 1/4 only for the recognition part. output frame will be of the original size.

#Resizing improves the Frame per Second.

#Face\_recognition. face locations() is called on the resized image(imgS) .for face bounding box coordinates must be multiplied by 4 in order to overlay on the output frame.

#Face\_recognition. distance () returns an array of the distance of the test image with all images present in our train directory.

#The index of the minimum face distance will be the matching face.

After finding the matching name we call the markAttendance function.

Draw bounding box using cv2.rectangle().

We put the matching name on the output frame using cv2.putText().

**CHAPTER-3**

**RESULTS AND LIMITATIONS**

* 1. **3. RESULTS**

1. Detection of single student showing their name, ID on the screen.

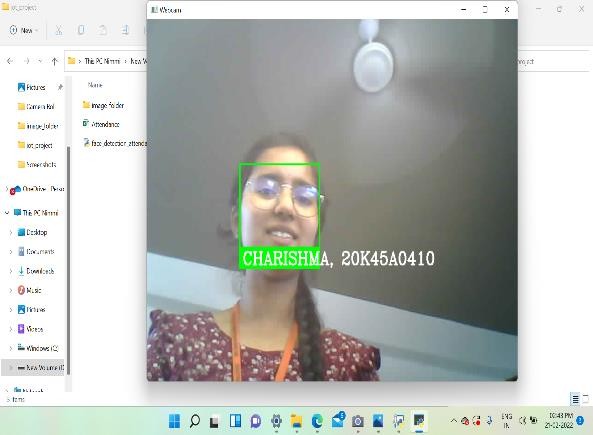
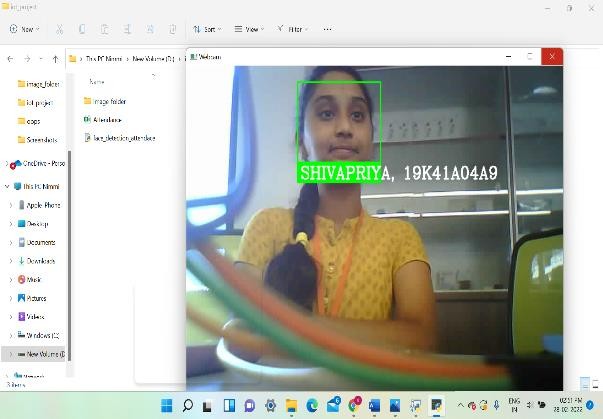


Fig.13: ESP32 cam detecting the faces Fig.14: ESP32 cam detecting the faces

1. Detection of two students at a time and showing their name, ID on the screen.

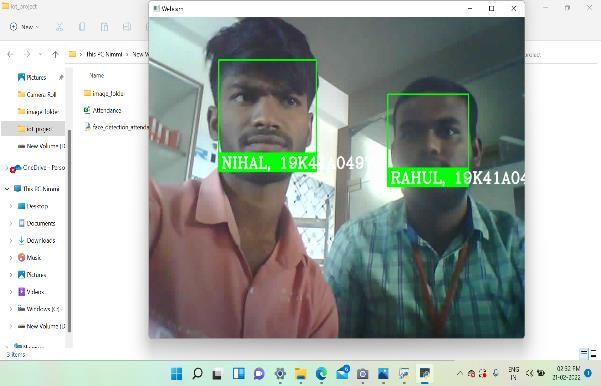
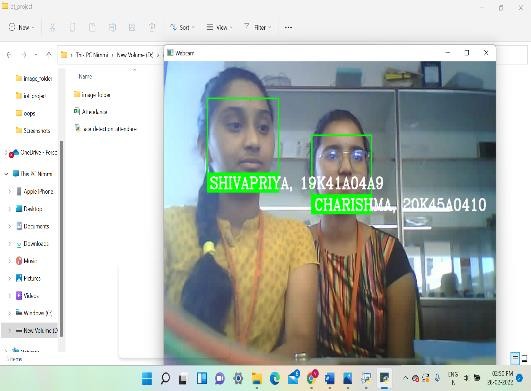
 

Fig.15: ESP32 cam detecting Fig.16: ESP32 cam detecting the faces the faces

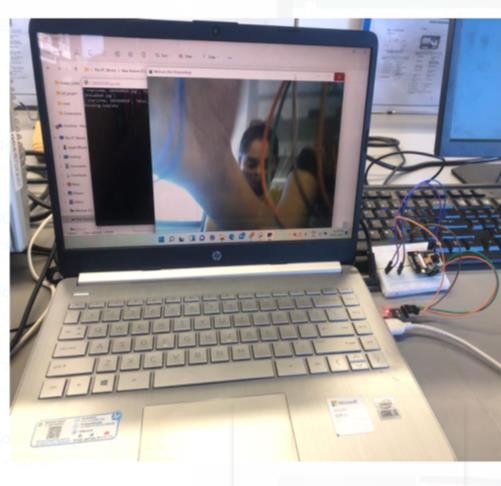
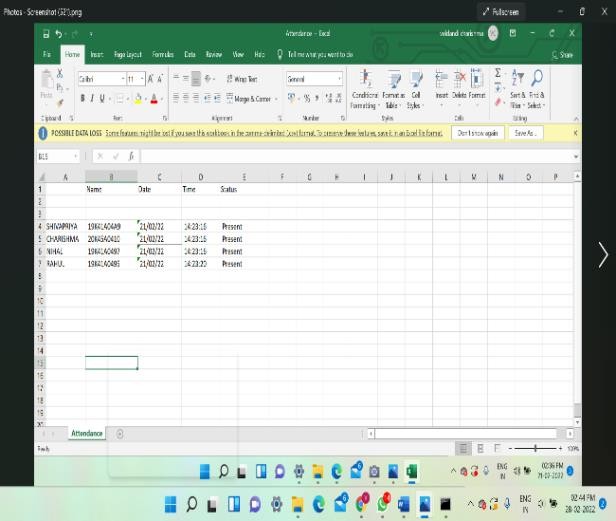
 

Fig.17: ESP-32 cam module connected Fig.18: Attendance generated in personal computer. Excel-sheet

## LIMITATIONS:

1. The whole process must be started again for taking at the attendance every time.
2. Excel sheet has to be refreshed every time to store new data again.
3. Absentee’s students list is not given.

# CHAPTER-4

## CONCLUSION & FUTURE SCOPE

* 1. **4.1 CONCLUSION:**

As we know that maintaining the attendance is time consuming and a huge task. To overcome the problems, arise, we are going to display attendance using automatic attendance system. Here the images of students are stored in the software so when the student face is detected it recognizes and marks the attendance and stores it in the database and all the records are maintained. So that the time is not wasted, also no fake attendances are given.

## 4.2 FUTURE SCOPE:

1. Biometric can be implemented.
2. There can be a chance of the detecting more number of faces
3. Displaying the student’s status in the excel sheet who are absent.
4. Regenerating the new attendance for every particular instance of time without starting it again.

**CHAPTER-5**

**5.1 REFERENCES**

1. B. Kavinmathi and S. Hemalatha, "Attendance System for Face Recognition using GSM module", *4th International Conference on Signal Processing and Integrated Networks”*, 2018.
2. Yohei Kawaguchi and Tetsuo Shoji, "Face Recognition-based Lecture Attendance System", *“3rd AERU…”*, 2005.
3. Ketan N. Mahajan and Nagaraj V. Dharwadkar, "Classroom attendance system using surveillance camera", *International Conference on Computer Systems Electronics and Control”*, 2017.
4. Shubhobrata Bhattacharya, Gowtham Sandeep Nainala, Prosenjit Das and Aurobinda Routray, "Smart Attendance Monitoring System (SAMS): A Face Recognition based Attendance System for Classroom Environment", *IEEE 18th International Conference on Advanced Learning Technologies*, 2018.
5. E. Varadharajan, R. Dharani and S. Jeevitha, Automatic attendance management system using face detection, 2017.
6. Guo Jing-Ming, "Complexity reduced face detection using probability-based face mask prefiltering and pixel-based hierarchical-feature Ada boosting", *Signal Processing Letters*, 2011.

Show in Context

1. K. Senthamil Selvil, P. Chitrakala, A. Antony and S Jenitha, "face recognition based attendance marking system", *International Journal of Computer Science and Mobile Computing*, 2014.

Show in Context

1. Chen and Joy Iong Zong, "Smart Security System for Suspicious Activity Detection in Volatile Areas", *Journal of Information Technology 2*, 2020.
2. Jacob and I. Jeena, "Capsule network based biometric recognition system", *Journal of Artificial Intelligence 1*, 2019.
3. Kirtiraj Kadam, Manasi Jadhav, Shivam Mulay and Tushar Indalkar, "Attendance Monitoring System Using Image Processing and Machine Learning", *International Journal of Advance Engineering and Research Development*, 2017.
4. Rajat Kumar Chauhan, Vivekanand Pandey and M Lokanath, "Smart Attendance System Using CNN", *International Journal of Pure and Applied Mathematics*, 2018.
5. Mayank Yadav and Anmol Aggarwal, "Motion based attendance system in real time environment for multimedia application", 2018.
6. Wei Wu, Chuanchang Liu and Zhiyuan Su, "Novel Real-time Face Recognition from Video Streams", *International Conference on Computer Systems Electronics and Control*, 2017.
7. Changxing Ding and Dacheng Tao, "Trunk-Branch Ensemble Convolutional Neural Networks for Video-Based Face Recognition", *IEEE transactions on pattern analysis and machine intelligence*, 2018.
8. Aziza Ahmedi and Dr Suvarna Nandyal, "An Automatic Attendance System Using Image processing", *The International Journal of Engineering and Science*, 2015.

### APPENDIX

**PROJECT CODE:**

import pandas as pd import cv2

import urllib.request import numpy as np import os

from datetime import datetime import face\_recognition

path = r'D:\iot\_project\image\_folder' [url='http://192.168.90.63/cam](http://192.168.90.63/cam-hi.jpg%27)-[hi.jpg'](http://192.168.90.63/cam-hi.jpg%27)

'''cam.bmp / cam-lo.jpg /cam-hi.jpg / cam.mjpeg '''

'''if 'Attendance.csv' in os.listdir(os.path.join(os.getcwd(),'attendace')): print("there iss..")

os.remove("Attendance.csv") else:

df=pd.DataFrame(list()) df.to\_csv("Attendance.csv")'''

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

def markAttendance(name):

with open("Attendance.csv", 'r+') as f: myDataList = f.readlines() nameList = []

dateList = []

for line in myDataList: entry = line.split(',')

nameList.append(entry[0]) name\_pin = name.split(", ")

if name\_pin[0] not in nameList: now = datetime.now()

dtString = now.strftime('%H:%M:%S') date = now.strftime('%d/%m/%y') mark\_at = "Present"

f.writelines(f'\n{name\_pin[0]}, {name\_pin[1]}, {date}, {dtString},

{mark\_at}')

encodeListKnown = findEncodings(images) print('Encoding Complete')

#cap = cv2.VideoCapture(0) while True:

#success, img = cap.read()

img\_resp=urllib.request.urlopen(url) imgnp=np.array(bytearray(img\_resp.read()),dtype=np.uint8) img=cv2.imdecode(imgnp,-1)

# 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() # 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, 1, (255, 255, 255), 2)

markAttendance(name)

cv2.imshow('Webcam', img) key=cv2.waitKey(5)

if key==ord('q'): break

cv2.destroyAllWindows() cv2.imread