First

#include "config.h"

#include <Adafruit\_SSD1306.h>

#include <Wire.h>

#include <Adafruit\_Sensor.h>

#define SOUNDVELOCITY 0.034

#define CMTOINCH 0.393701

const int trig = 12;

const int echo = 14;

const int buz = 13;

const float thres = 0.93;

Adafruit\_SSD1306 oled(128, 32, &Wire);

long duration;

float distancecm;

float distanceinch;

void setup() {

Serial.begin(115200);

pinMode(trig, OUTPUT);

pinMode(echo, INPUT);

pinMode(buz, OUTPUT);

oled.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

oled.display();

oled.setTextSize(1);

oled.setColor(WHITE);

}

void loop() {

oled.clearDisplay();

// Trigger the ultrasonic sensor

digitalWrite(trig, LOW);

delayMicroseconds(2);

digitalWrite(trig, HIGH);

delayMicroseconds(10);

digitalWrite(trig, LOW);

// Read the echo duration

duration = pulseIn(echo, HIGH);

// Calculate distances

distancecm = duration \* SOUNDVELOCITY / 2;

distanceinch = distancecm \* CMTOINCH;

// Print data to serial monitor

Serial.print("Duration: ");

Serial.println(duration);

Serial.print("Distance cm: ");

Serial.println(distancecm);

Serial.print("Distance inch: ");

Serial.println(distanceinch);

// Trigger buzzer if distance is less than the threshold

if (distancecm < thres) {

tone(buz, 1000);

delay(1000);

noTone(buz);

}

// Display data on OLED

oled.clearDisplay();

oled.setCursor(0, 0);

oled.print("Dis cm: ");

oled.println(distancecm);

oled.print("Dis inch: ");

oled.println(distanceinch);

oled.display();

delay(5000);

}

**Second**

#include "config.h"

#include <Adafruit\_SSD1306.h>

#include <Wire.h>

#include <Adafruit\_Sensor.h>

#include <DHT.h>

#define DHTTYPE DHT11

#define DHTPIN 4

Adafruit\_SSD1306 oled(128, 32, &Wire);

void setup() {

Serial.begin(115200);

oled.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

oled.display();

oled.setTextSize(1);

oled.setTextColor(WHITE);

oled.clearDisplay();

}

void loop() {

float humi = dht.readHumidity();

float temp = dht.readTemperature();

Serial.print("Temperature: ");

Serial.println(temp);

Serial.print("Humidity: ");

Serial.println(humi);

oled.clearDisplay();

oled.setCursor(0, 0);

oled.print("Temp: ");

oled.println(temp);

oled.print("Humi: ");

oled.println(humi);

oled.display();

delay(5000);

}

**Third**

import Adafruit\_DHT

import matplotlib.pyplot as plt

import time

# Initialize the DHT11 sensor

sensor = Adafruit\_DHT.DHT11

pin = 4

a = 10

# Initialize lists to store the collected data

temperature\_data = []

humidity\_data = []

# Collect data from the sensor at regular intervals

while a>0:

humidity, temperature = Adafruit\_DHT.read(sensor, pin)

temperature\_data.append(temperature)

humidity\_data.append(humidity)

time.sleep(5)

a -= 1

# Use matplotlib to visualize the collected data

plt.plot(temperature\_data, 'r')

plt.xlabel('No of Iterations')

plt.ylabel('Temperature')

plt.title('Temperature Graph')

plt.show()

plt.bar(range(len(humidity\_data)), humidity\_data)

plt.xlabel('No of Iterations')

plt.ylabel('Humidity')

plt.title('Humidity Graph')

plt.show()

**fourth**

import requests

import time

import Adafruit\_DHT

api\_key = "J6WDF39BE27DX1H”

pin = 4

sensor = Adafruit\_DHT.DHT11

i = 10

while i:

humidity, temperature = Adafruit\_DHT.read(sensor, pin)

payload = {'api\_key': api\_key, 'field1': str(temperature), 'field2': str(humidity)}

req = requests.post("https://api.thingspeak.com/update?-api\_key=J6WDF39BE27DX1H&field1=0", params=payload)

if req.status\_code == requests.codes.ok:

print("Data successfully updated")

else:

print("Error in the code")

i = i - 1

time.sleep(10)

**fiveth**

from tkinter import \*

from gpiozero import AngularServo

from time import sleep

root = Tk()

root.geometry("600x150")

def rotate(angle\_var):

Servo = AngularServo(17, min\_angle=-90, max\_angle=90)

angle = angle\_var.get()

while True:

servo.angle = angle

sleep(0.02)

VI = DoubleVar()

TitleLabel = Label(text="Enter angle of rotation using slide: ")

TitleLabel.config(font=("helvetica", 15))

TitleLabel.config(fg="#DE3163")

Slider = Scale(root, variable=VI, from\_=-90, to=90, length=720, tickinterval=15, orient=HORIZONTAL)

Submitbtn = Button(root, text="Click to submit", command=lambda: rotate(VI), bg="yellow")

TitleLabel.pack(anchor=CENTER)

**Sixth**

import cv2

from pushbullet import Pushbullet

from time import sleep

cam=cv2.VideoCapture(0)

img=0

def capture():

print("Capturing image")

ret,img = cam.read()

k=cv2.waitKey(10000)

print("Image Caputred successfully")

cv2.imwrite('The addres', img) #change image address and put what you have used

cam.release()

cv2.destroyAllWindows()

print("Sending")

def send():

pb=Pushbullet("\_\_api key from website(generate token)\_\_\_\_")

dev=pb.get\_device("\_\_deivce name from app\_\_\_") #the name which is shown in the app

with open('The address', 'rb') as pic: #change image address and put what you have used

file\_data=pb.upload\_file(pic, 'picture.jpg')

push=pb.push\_file(\*\*file\_data)

value=push['active']

capture()

send()

**seven**

**import cv2**

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

**image = 0**

**def CaptureImage():**

**global image**

**print("Capturing Image :")**

**ret, image = cam.read()**

**k = cv2.waitKey(10000)**

**print("Image Capture successfully")**

cv2.imwrite('/home/pi/pictures/testimage.jpg',image)

cam.release()

cv2.destroyAllWindows()

def AnnotateImage():

print("print Annotation Completed..")

ImageCopy = cv2.imread('/home/pi/Picture/testimage.jpg',image)

ImageDimension = ImageCopy.shape

print("Dimension of the Image :",str(ImageDimension))

Start = (0,340)

End = (100,480)

cv2.rectangle(ImageCopy , Start,End,(0,0,255),thickness = 3, lineType = cv2.LINE\_8)

cv2.putText(ImageCopy, "pen\_stand", (0,330), cv2.FONT\_HERSHEY\_SIMPLEX, 0.5,(255,0,0),1)

cv2.imshow('AnnotatedImage', ImageCopy)

cv2.waitKey(10000)

CaptureImage()

AnnotateImage()

**Eighth**

#client

import socket

s = socket.socket()

port = 12346

s.connect(('127.0.0.1'),port))

print(s.recv(1024).decode())

s.close()

#server

import socket

s = socket.socket()

port = 12346

s.bind(('',port))

print("socket blinded to %s" %(port))

s.listen(5)

print("server is ready to accept the connection")

while True:

c,addr = s.accept()

print("got connection from ",addr)

c.send('Thank you for connecting'.encode())

c.close()

break