

|  |  |
| --- | --- |
|  | PNT2022TMID18163 |
|  | Industry specific intelligent fire management system |



import time import sys

import ibmiotf.application import ibmiotf.device import random

#Provide your IBM Watson Device Credentials organization = "4chhfb”

deviceType = "NodeMCU" deviceId = "12345" authMethod="use-token-auth"

authToken = "201903074”

# Initialize GPIO

def myCommandCallback(cmd):

print("Command received: %s" % [cmd.data](http://cmd.data/)['command']) status=[cmd.data](http://cmd.data/)['command']

if status=="lighton": print ("led is on") else :

print ("led is off") #print(cmd)

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()

while True:

#Get Sensor Data from DHT22,DHT11,

Temp=random.randint(-20,120) Humidity=random.randint(0,120) Flame=random.randint(0,100) Gas=random.randint(0,80)

data = {'Temp' :Temp ,'Humidity' : Humidity,'Flame' : Flame,'Gas' : Gas}

def myOnPublishCallback():

if Flame > 100:

data = {'Flame' : Flame}

print ("Temperature =%s c" % Temp ,"Humidity =%s u" % Humidity,"Flame =%s ir" % Flame ,"Gas

=%s ppm" % Gas )

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

if not success:

print("Not connected to IoTF") time.sleep(1)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud deviceCli.disconnect()















