**GAS LEAKAGE .PY**

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM Watson Device Credentials

organization = "pi0ywk"

deviceType = "Gas\_Geakage\_Detector"

deviceId = "Udayakpr007"

authMethod = "token"

authToken = "8148922991"

# Initialize GPIO

def myCommandCallback(cmd):

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

status=cmd.data['command']

if status == "alarmon":

print ("Alarm is on please all Evacuate Fans On")

elif status == "alarmoff":

print ("Alarm is off and Fans Off")

elif status == "sprinkleron":

print ("Sprinkler is On Evacuate Faster")

elif status == "sprinkleroff":

print("Sprinkler is Off")

else:

print("Please send proper command")

#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 random function

temp=random.randint(0,120)

Humid=random.randint(0,100)

gas=random.randint(0,1500)

data={'temp':temp,'Humid':Humid,'gas':gas}

#print data

def myOnPublishCallback():

print (" Published Temperature = %s C" % temp, "Humidity = %s %%" % Humid, "Gas\_Level = %s ppm" %gas, "to IBM Watson")

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

if not success:

print("\n Not connected to IoTF")

if temp>60 :

print("\n Fire Detected due to gas Leak ! Alarm ON! Sprinkler ON! Call The Fire Police \n")

elif gas>350:

print("\n Gas is Leaking \n")

time.sleep(10)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()