SPRINT 1

import random

import ibmiotf.application

import ibmiotf.device

from time import sleep

import sys

#IBM Watson Device Credentials.

organization = "op701j"

deviceType = "Lokesh"

deviceId = "Lokesh89"

authMethod = "token"

authToken = "1223334444"

def myCommandCallback(cmd):

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

status=cmd.data['command']

if status=="sprinkler\_on":

print ("sprinkler is ON")

else :

print ("sprinkler 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()

#Connecting to IBM watson.

deviceCli.connect()

while True:

#Getting values from sensors.

temp\_sensor = round( random.uniform(0,80),2)

PH\_sensor = round(random.uniform(1,14),3)

camera = ["Detected","Not Detected","Not Detected","Not Detected","Not Detected","Not Detected",]

camera\_reading = random.choice(camera)

flame = ["Detected","Not Detected","Not Detected","Not Detected","Not Detected","Not Detected",]

flame\_reading = random.choice(flame)

moist\_level = round(random.uniform(0,100),2)

water\_level = round(random.uniform(0,30),2)

#storing the sensor data to send in json format to cloud.

temp\_data = { 'Temperature' : temp\_sensor }

PH\_data = { 'PH Level' : PH\_sensor }

camera\_data = { 'Animal attack' : camera\_reading}

flame\_data = { 'Flame' : flame\_reading }

moist\_data = { 'Moisture Level' : moist\_level}

water\_data = { 'Water Level' : water\_level}

# publishing Sensor data to IBM Watson for every 5-10 seconds.

success = deviceCli.publishEvent("Temperature sensor", "json", temp\_data, qos=0)

sleep(1)

if success:

print (" ............................publish ok............................. ")

print ("Published Temperature = %s C" % temp\_sensor, "to IBM Watson")

success = deviceCli.publishEvent("PH sensor", "json", PH\_data, qos=0)

sleep(1)

if success:

print ("Published PH Level = %s" % PH\_sensor, "to IBM Watson")

success = deviceCli.publishEvent("camera", "json", camera\_data, qos=0)

sleep(1)

if success:

print ("Published Animal attack %s " % camera\_reading, "to IBM Watson")

success = deviceCli.publishEvent("Flame sensor", "json", flame\_data, qos=0)

sleep(1)

if success:

print ("Published Flame %s " % flame\_reading, "to IBM Watson")

success = deviceCli.publishEvent("Moisture sensor", "json", moist\_data, qos=0)

sleep(1)

if success:

print ("Published Moisture Level = %s " % moist\_level, "to IBM Watson")

success = deviceCli.publishEvent("Water sensor", "json", water\_data, qos=0)

sleep(1)

if success:

print ("Published Water Level = %s cm" % water\_level, "to IBM Watson")

print ("")

#Automation to control sprinklers by present temperature an to send alert message to IBM Watson.

if (temp\_sensor > 35):

print("sprinkler-1 is ON")

success = deviceCli.publishEvent("Alert1", "json",{ 'alert1' : "Temperature(%s) is high, sprinkerlers are turned ON" %temp\_sensor }

, qos=0)

sleep(1)

if success:

print( 'Published alert1 : ', "Temperature(%s) is high, sprinkerlers are turned ON" %temp\_sensor,"to IBM Watson")

print("")

else:

print("sprinkler-1 is OFF")

print("")

#To send alert message if farmer uses the unsafe fertilizer to crops.

if (PH\_sensor > 7.5 or PH\_sensor < 5.5):

success = deviceCli.publishEvent("Alert2", "json",{ 'alert2' : "Fertilizer PH level(%s) is not safe,use other fertilizer" %PH\_sensor } ,

qos=0)

sleep(1)

if success:

print('Published alert2 : ' , "Fertilizer PH level(%s) is not safe,use other fertilizer" %PH\_sensor,"to IBM Watson")

print("")

#To send alert message to farmer that animal attack on crops.

if (camera\_reading == "Detected"):

success = deviceCli.publishEvent("Alert3", "json", { 'alert3' : "Animal attack on crops detected" }, qos=0)

sleep(1)

if success:

print('Published alert3 : ' , "Animal attack on crops detected","to IBM Watson","to IBM Watson")

print("")

#To send alert message if flame detected on crop land and turn ON the splinkers to take immediate action.

if (flame\_reading == "Detected"):

print("sprinkler-2 is ON")

success = deviceCli.publishEvent("Alert4", "json", { 'alert4' : "Flame is detected crops are in danger,sprinklers turned ON" }, qos=0)

sleep(1)

if success:

print( 'Published alert4 : ' , "Flame is detected crops are in danger,sprinklers turned ON","to IBM Watson")

#To send alert message if Moisture level is LOW and to Turn ON Motor-1 for irrigation.

if (moist\_level < 20):

print("Motor-1 is ON")

success = deviceCli.publishEvent("Alert5", "json", { 'alert5' : "Moisture level(%s) is low, Irrigation started" %moist\_level }, qos=0)

sleep(1)

if success:

print('Published alert5 : ' , "Moisture level(%s) is low, Irrigation started" %moist\_level,"to IBM Watson" )

print("")

#To send alert message if Water level is HIGH and to Turn ON Motor-2 to take water out.

if (water\_level > 20):

print("Motor-2 is ON")

success = deviceCli.publishEvent("Alert6", "json", { 'alert6' : "Water level(%s) is high, so motor is ON to take water out "

%water\_level }, qos=0)

sleep(1)

if success:

print('Published alert6 : ' , "water level(%s) is high, so motor is ON to take water out " %water\_level,"to IBM Watson" )

print("")

#command recived by farmer

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()

OUTPUT:

