DEVELOP A PYTHON CODE

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random

#IBM Watson Device Credentials
organization = "re4wy2" #replace with org ID
deviceType = "abcd"
deviceId = "12"
authMethod = "token"
authToken = "12345678"

#Receives Command fro Node-RED
def myCommandCallback(cmd): #gets data from ibm cloud to python
    print("Command receive: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print("motor is on")
    elif status=="motoroff":
        print ("motor is off")
    else :
        print("please send proper command")
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
deviceCli.connect()

while True:
    #Get sensor data from DHT11

    temp=random.randint(20,30)
    humidity=random.randint(40,60)
    randomNumber=random.randint(21,80)
    data = {'temp':temp, 'humidity':humidity,
'randomNumber':randomNumber}
    #print data
    def myOnPublishCallback():
        print("Published Temperature = %s" % temp, "Humidity = %s" %
humidity, "soilmoisture = %s" % randomNumber, "to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish = myOnPublishCallback())
    if not success:
        print("NOT CONNECTED TO IoTF")
    time.sleep(5)
    deviceCli.commandCallback = myCommandCallback
#disconect the device and application from the cloud
deviceCli.disconnect()
```