**REALTIME WEATHER BASED SMART SPRINKLER SYSTEM FOR GOLF COURSE**

**1.INTRODUCTION**

Continous monitoring of weather conditions and observing the soil moisture and turning on the motors automatically using

->The aim of the project is to monitor and storage moisture information.

->Alert is generated if the soil moisture is low.

->To turn on and turn off motor automatically using mobile app.

**2.LITERATURE SURVEY**

The problem is when the soil moisture is below the threshold value alert must be generated and the water sprinklers(motors) are turned are controlled using mobile app.

The solution for above problem is creating an iot application in which an application is developed which is connected with ibm iot platform .Data is exchanged between iot platform and appication.

**3.THEORITICAL ANALYSIS**

**3.1 BLOCK DIAGRAM**

OBJECT STORAGE

DATABASE

NODERED

IBM WATSON CLOUD PLATFORM

LAPTOP

(RASPBERRYPI)

MQTT

HTTP

SMARTWEATHER MONITORING

->In this project we generate sensor values using python code.To generate temperature ,pressure,humidity values we use openweathermap.com, these values are send to python code .

->Sensor values are sent to ibmiot platform.

->Now data is available in ibm platform we need to build application to get data .we use nodered to build application.

->In noderred different nodes(ibmiot ,function,guage,ibmout) are used to build flows .when this nodered is connected to ibm cloud platform data is visible in nodered.

->We build mobile application by using mitappinventor.

->In mit app inventor in the blocks section we use different blocks to connect to web and to give command.

**5. FLOW CHART**

INSTALL PYTHON IDLE

CREATE IBM ACCOUNT

CREATE NODE-RED APPLICATION

CREATE IBM WATSON IOT PLATFORM

CREATE A WEATHER API

CREATE MIT APP INVENTOR ACCOUNT

CREATING A CODE SNIPPET FOR DHT11 SENSOR TO MEASURE TEMPERATURE AND HUMIDITY

CREATE A CODE SNIPPET FOR SOIL MOISTURE SENSOR

PUBLISHING DATA TO IBM CLOUD

CREATE A NODE RED FLOW TO GET DATA FROM DEVIDE

USE DASHBOARD NODES FOR CREATING UI

CREATING HTTP REQUESTS TO COMMUNICATE WITH MOBILE

DESIGNING UI TO DISPLAY VALUES

CONFIGURE THE APPLICATION TO RECEIVE DATA FROM CLOUD

CONFIGURE THE MOBILE APP FOR CONTROLLING MOTOR USING BUTTONS

**Result:**

Continous monitoring of weather and soil moisture is done.

alert is generated if the soil moisture is low.Motor is operated automatically using mobile app.

**Advantages and disadvantages:**

Advantage is motor can be controlled automatically using mobile app without going to the field.

Disadvantage is if network is not good we may loss the data and we canot get correct data.

**Applications:**

Agriculture.

sports.

gardening.

**Conclusion:**

This project is very usefull in agriculture and it is time saving.

**Future scope :**

Government may intiate such type of applications in agriculture.

**APPENDIX:**

**A.SOURCE CODE**

import time  
import sys  
import ibmiotf.application  
import ibmiotf.device  
import random  
import requests  
import json  
#Provide your IBM Watson Device Credentials  
organization = "zajnh3"  
deviceType = "raspberrypi"  
deviceId = "1234567"  
authMethod = "token"  
authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd):  
 print("Command received: %s" % cmd.data)  
 print(type(cmd.data))  
 i=cmd.data['command']  
 if i=='motoron':  
 print("motor is on")  
 elif i=='motoroff':  
 print("motor is off")

try:  
 deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}  
 deviceCli = ibmiotf.device.Client(deviceOptions)#.............................................#making of client  
   
except Exception as e:  
 print("Caught exception connecting device: %s" % str(e))#to print errors  
 sys.exit()

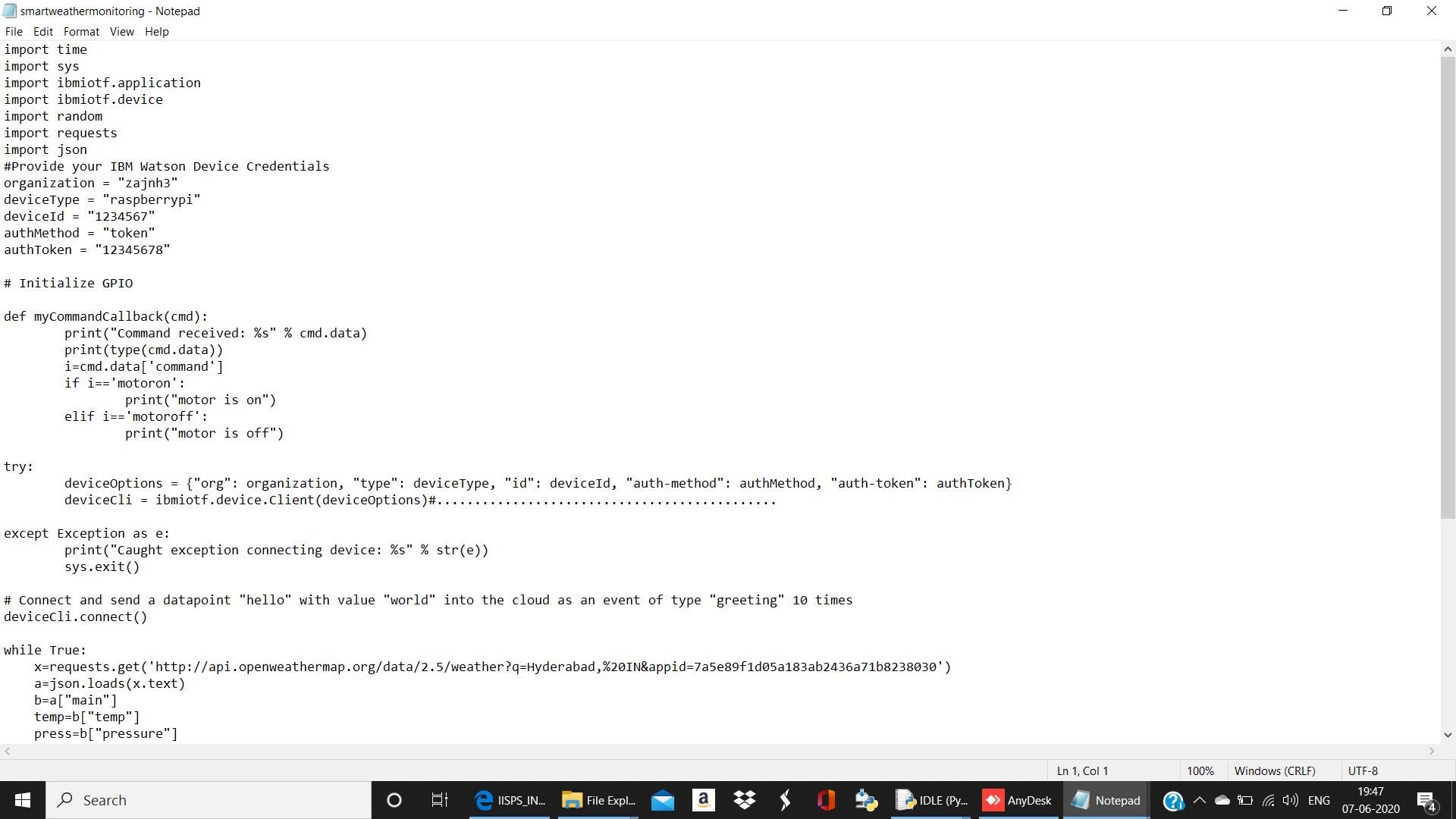
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times  
deviceCli.connect()

while True:  
 x=requests.get('http://api.openweathermap.org/data/2.5/weather?q=Hyderabad,%20IN&appid=7a5e89f1d05a183ab2436a71b8238030')  
 a=json.loads(x.text)  
 b=a["main"]  
 temp=b["temp"]  
 press=b["pressure"]  
 hum=b["humidity"]  
 soilm=random.randint(0,100)  
 #Send Temperature & Humidity to IBM Watson  
 data = { 'Temperature' : temp, 'Humidity': hum, 'Pressure': press, 'soilmoisture' : soilm }  
 #print (data)  
 def myOnPublishCallback():  
 print ("Published Temperature = %s C" % temp, "Humidity = %s %%" % hum, "Pressure = %s " % press, "Soilmoisture = %s %%" % soilm, "to IBM Watson")

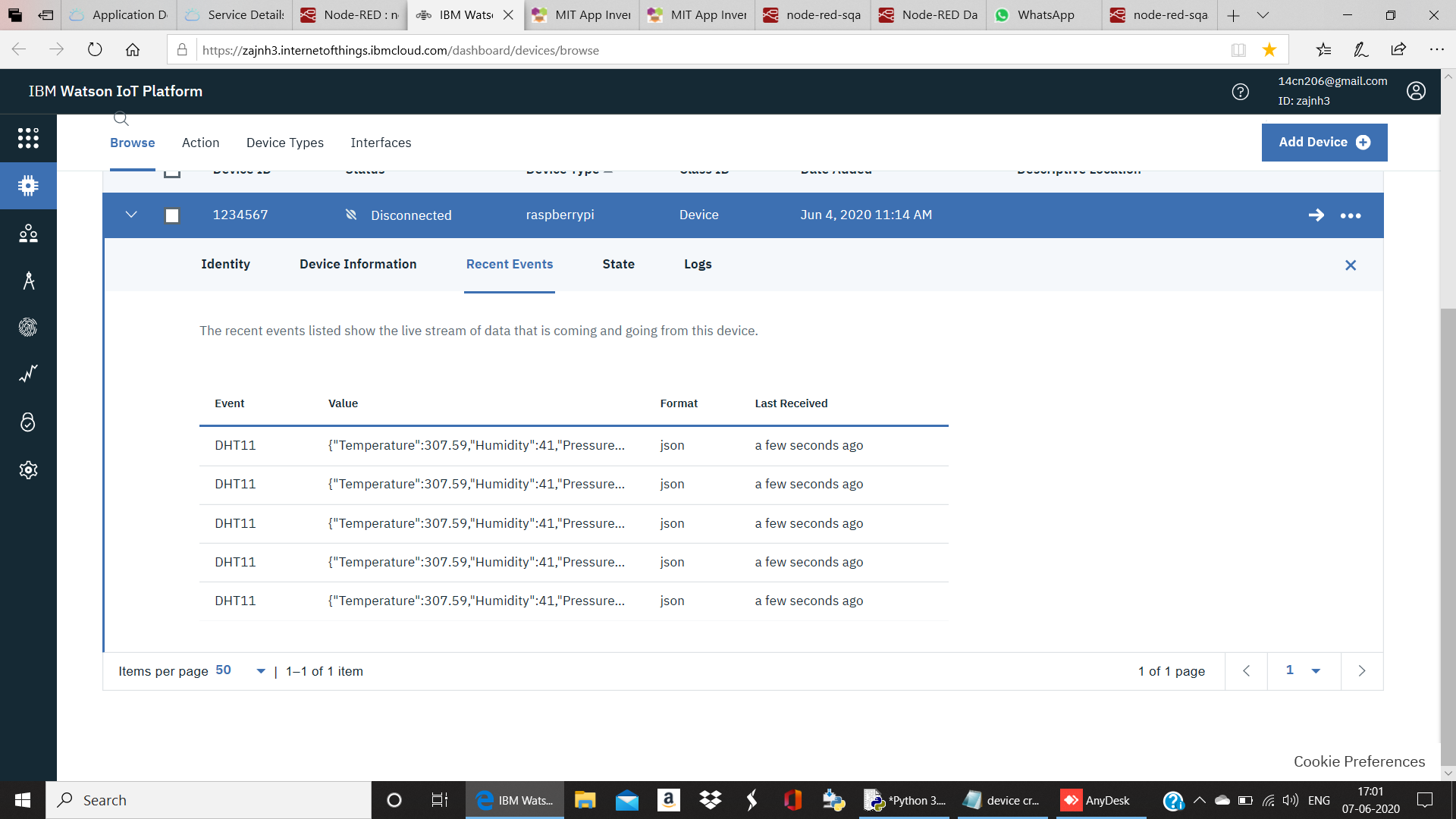
success = deviceCli.publishEvent("DHT11", "json", data, qos=0, on\_publish=myOnPublishCallback)  
 if not success:  
 print("Not connected to IoTF")  
 time.sleep(2)  
   
 deviceCli.commandCallback = myCommandCallback

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

The values of temperature,humidity,pressure,soil moisture are generated virtually through python code as shown below.



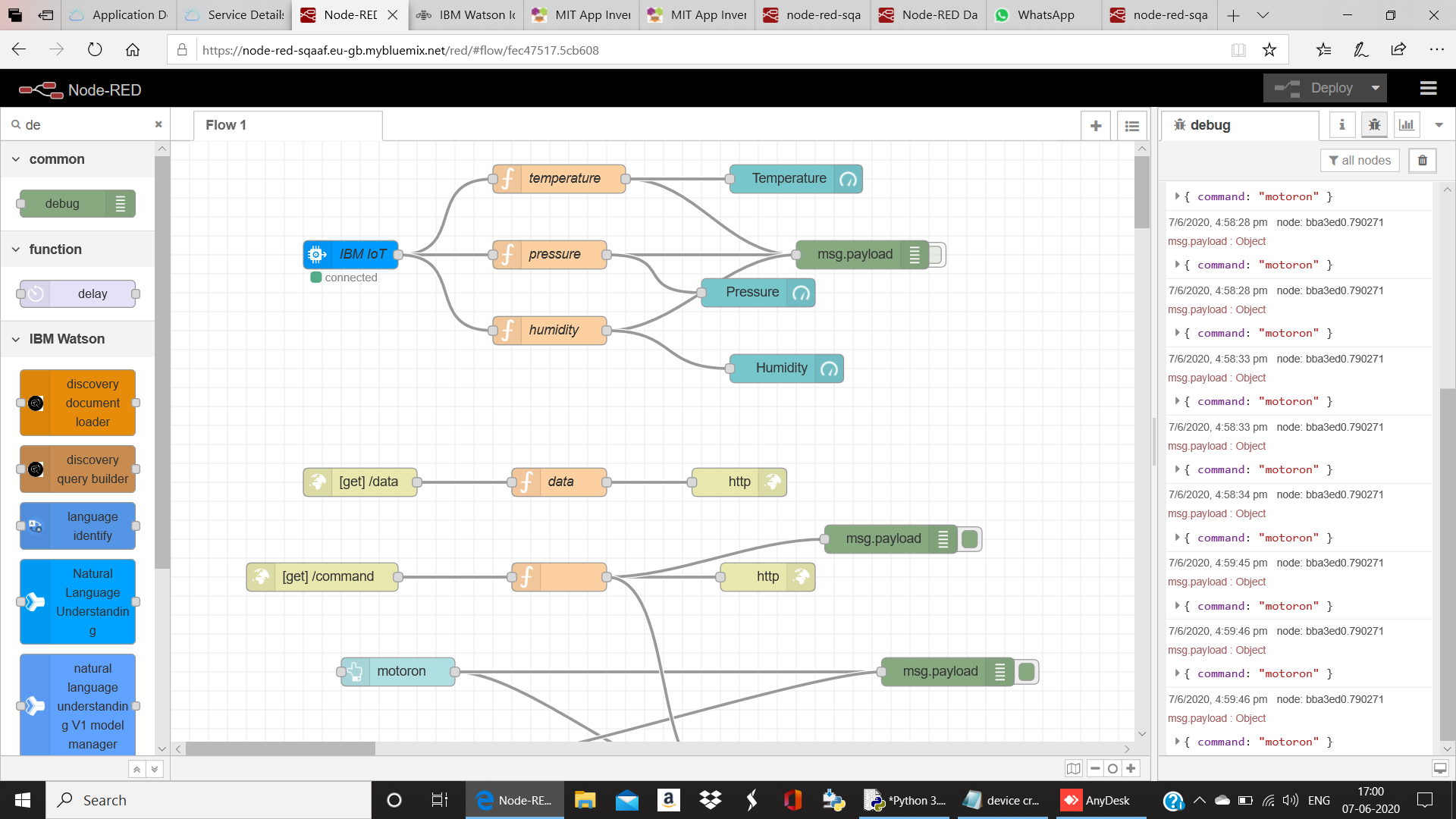


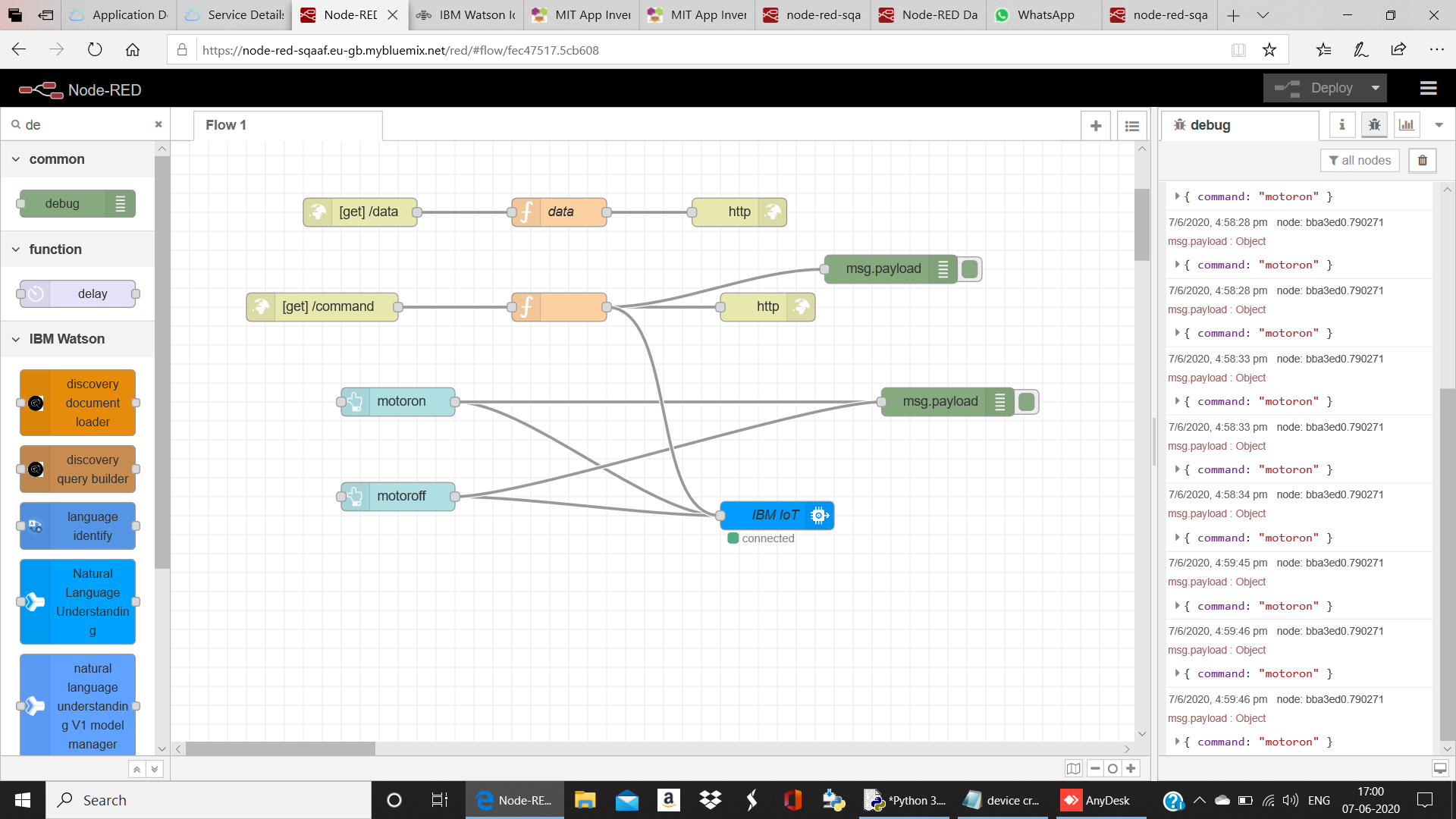
The sensor values are send to cloud platform and the data can be seen in ibm cloud 

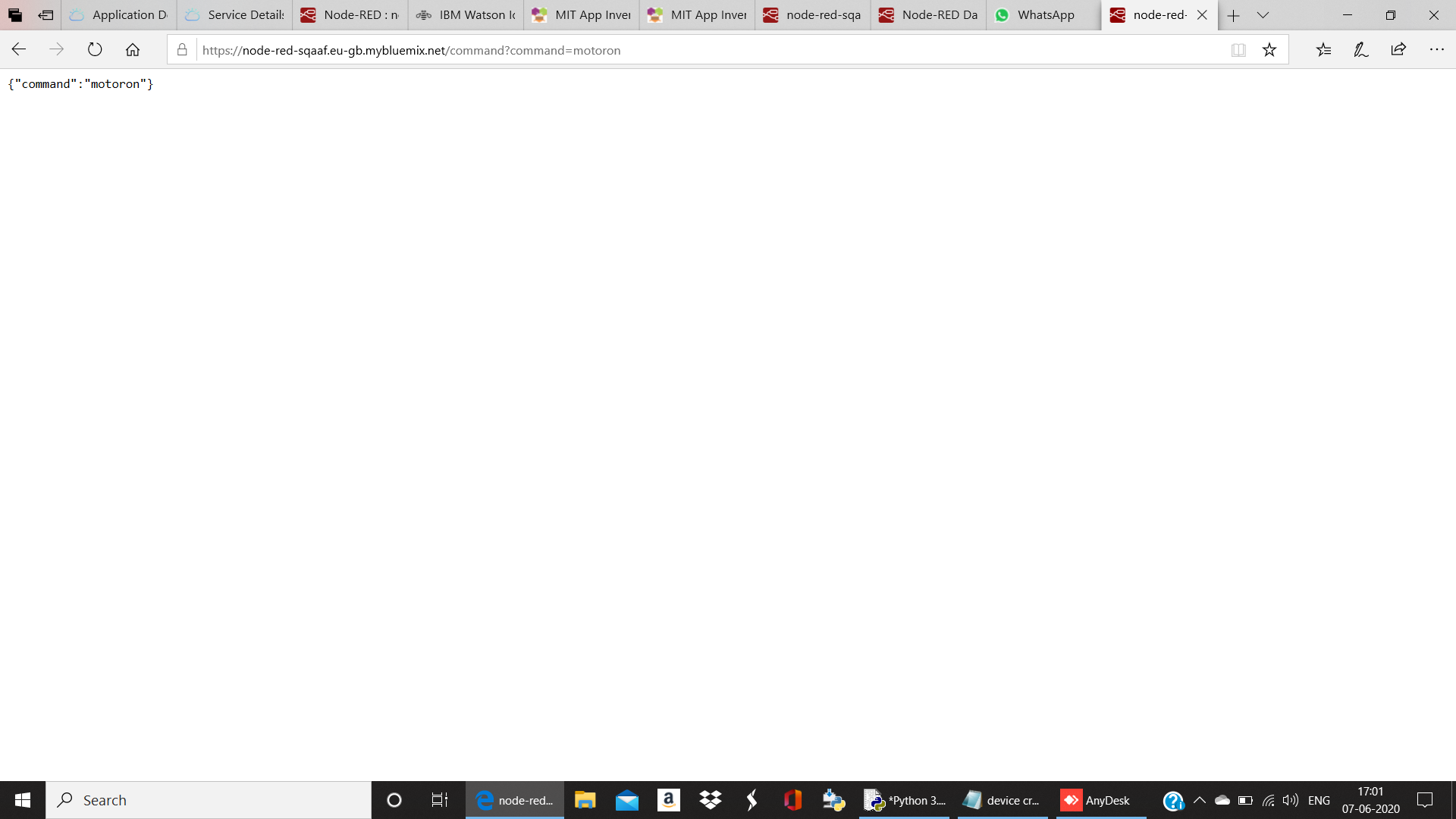
To send data from cloud to application we need to build application.

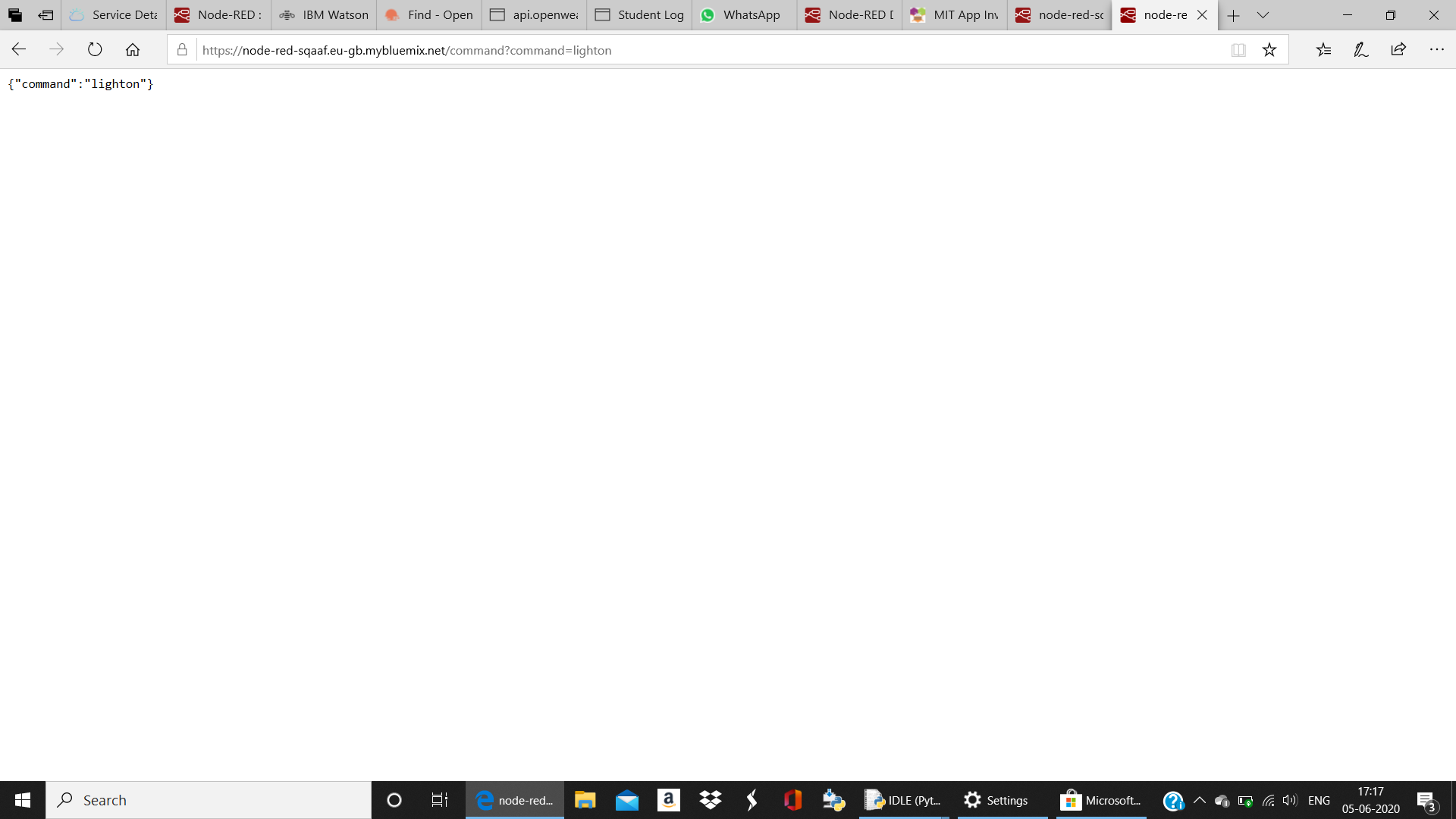
To build application we use nodered.For our project we build below flows.

Data from cloud can be seen in nodered.

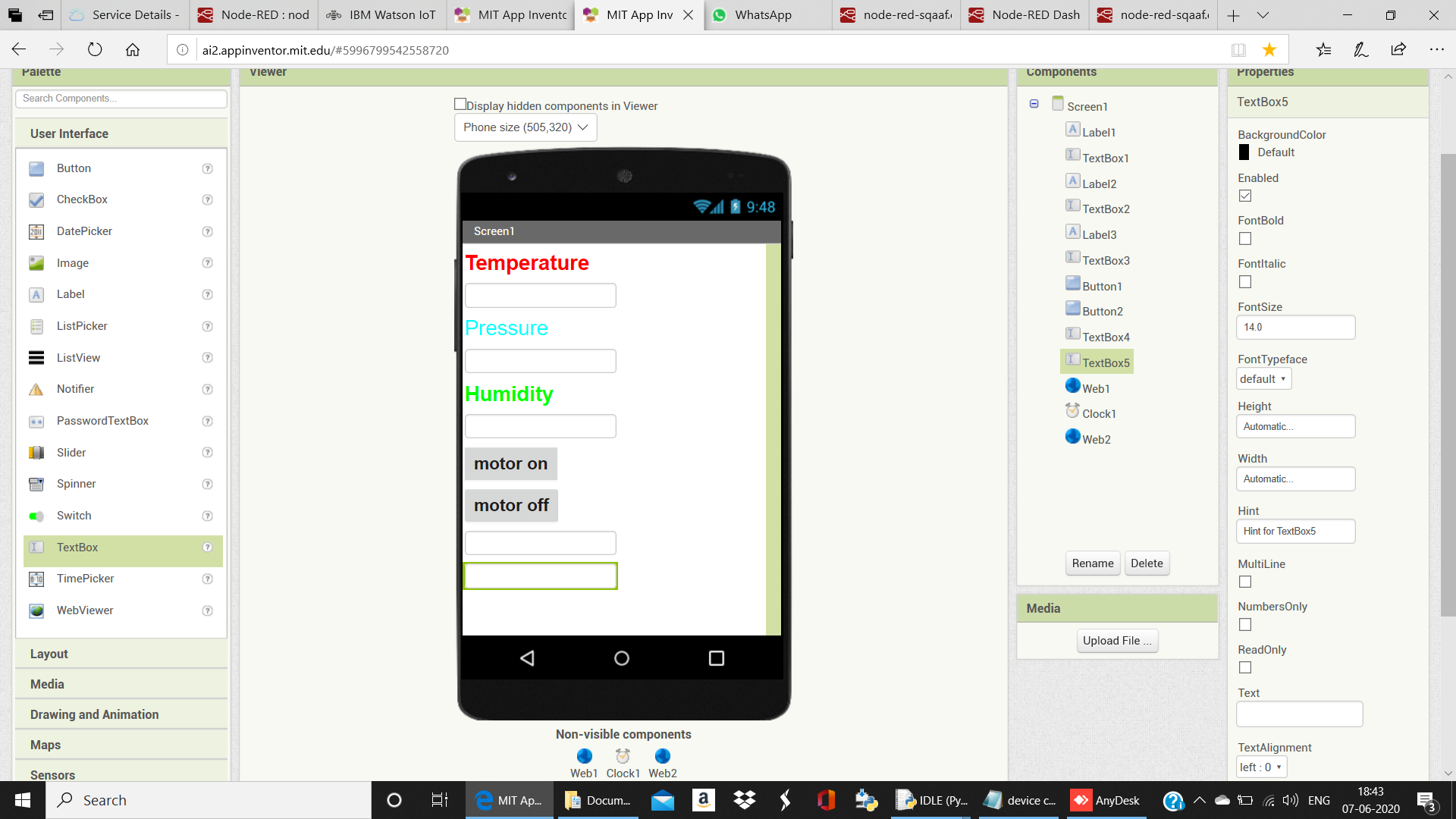


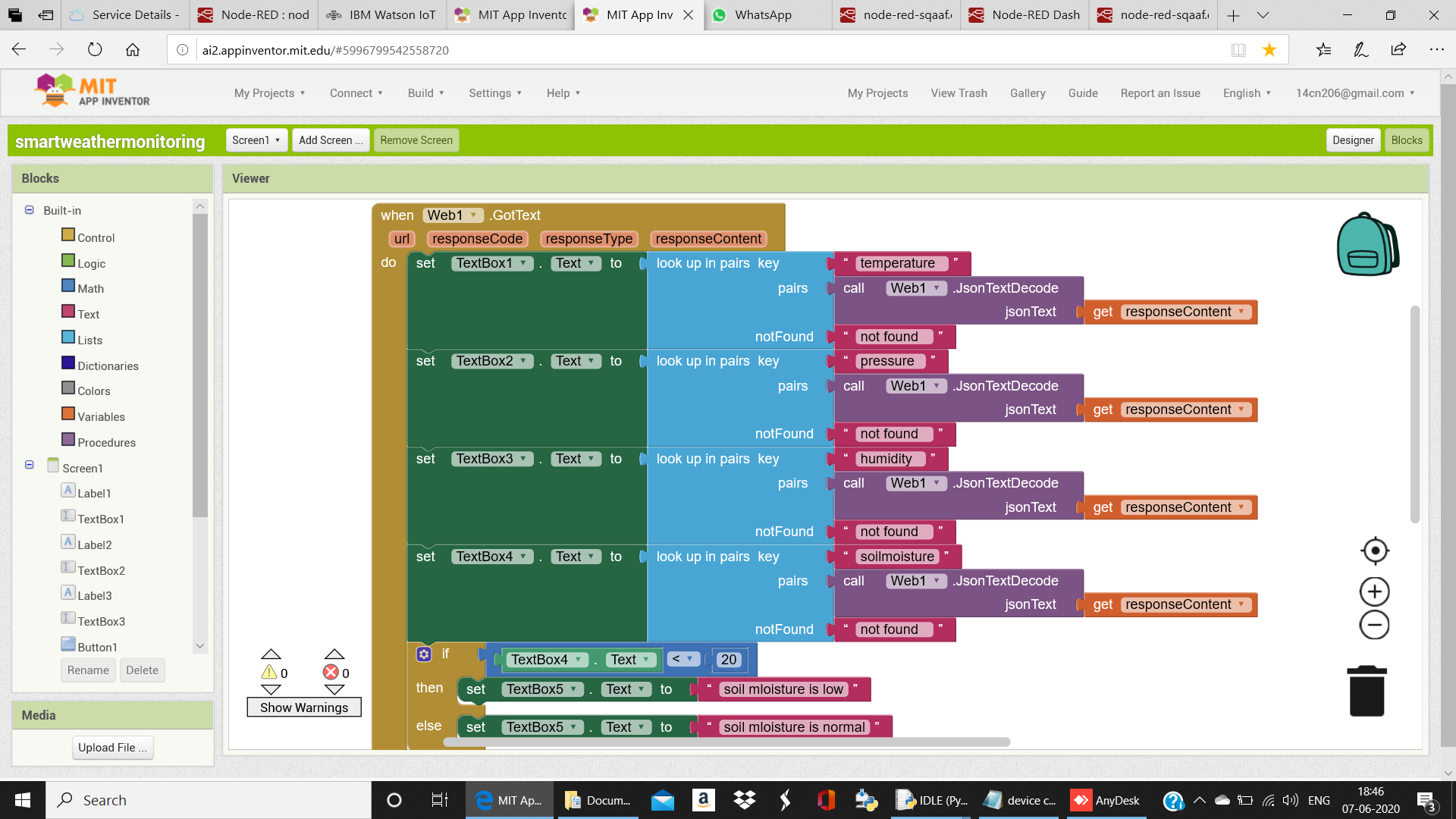


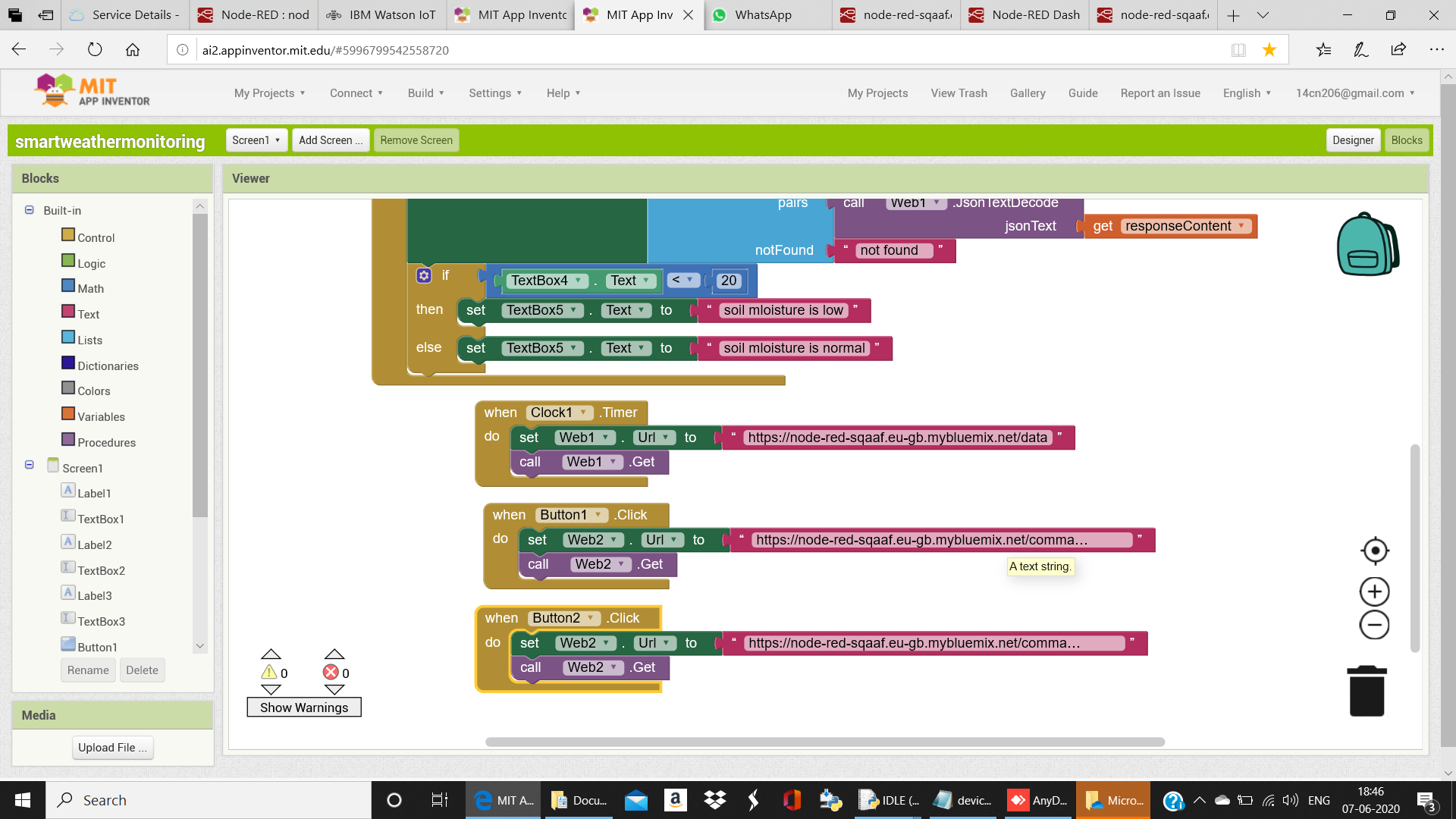




now to send command and to send data to mobile app we use mit app inventor.







Data can be seen in mobile application and we can switchon motor using mobile.Trigger is generated is moisture value is below threshold value.

