

# REPORT

**SUBMITTED BY - Mayank Gupta**

**EMAIL - [guptamayankwork@gmail.com](mailto:guptamayankwork@gmail.com)**

# **PROJECT NAME : SMART AGRICULTURE SYSTEM BASED ON IOT**

**DATE :15-06-2020**

## **PROJECT OVERVIEW**

SMART AGRICULTURE SYSTEM IS COMMERCIALY SCALABLE METHOD WHICH IS BEING BUILT TO REDUCE THE EFFORTS OF FARMERES.THIS PROJECT WILL SOLVE THE MOST IMPORTANT PROBLEM THE FARMERS ARE FACING THAT IS WATERING THE CROPS AT RIGHT TIME AND ACCORDING TO THE REAL TIME FIELD CONDITIONS.

THERE ARE MANY THINGS TO TAKE CARE OF, WHILE WATERING OF CROPS.IF WE WATER THE CROPS TO MUCH, THE CROPS CAN GET DAMAGED DUE TOP WATER LOGGING.IF WE WATER THE CROPS ON A RAINY DAY ,THE EXTRA RAINWATER MAY TAMPER THE GROWTH OF CROPS.IF IN ANY CASE THE FARMER FORGET TO WETER THE CROPS FOR ONE DAY OR TWO BECAUSE OF ANY REASON,IT CAN AGAIN MAKE THE SOIL TWO DRY FOR THE PLANTS TO GET THEIR DAILY NUTRITION LEVEL.

CONSIDERING MANY FACTS ,THE PROJECT IS MAINLY FOCUSED ON AUTOMATING THE PROCESS OF WATER PUMP AND PROVIDING THE FARMERS WITH AN APPLICATION WHERE THEY CAN SEE THE STATUS OF THE PUMP,MOISTURE OF THE SOIL,HUMIDITY ,TEMPERATURE AS WELL AS WEATHER FORECASTING SO THE FARMER CAN MANY ARRANGEMENTS ACCORDINGLY.

THE ULTIMATE GOAL OF THE PROJECT IS TO INCREASE THE QUALITY OF YIELD BY TAKING CARE OF IRRIGATION PROCESS AND STARING THE REVOLUTIONARY ERA OF FARMERS BY BRINGING TECHNOLOGY INTO THE DOMAIN.

## PROJECT SCOPE

-WE ARE DEVELOPING AN UI FOR THE FARMERS TO HELP THEM CONTROL THE REAL FIELD PARAMETERS SUCH AS TEMPERATURE,HUMIDITY AND SOIL MOISTURE IN ORDER TO IRRIGATE THE FIELD ACCORDINGLY.

-THE UI IS BUILD USING NODE RED.

-THE IBM CLOUD IS USED TO STORE DATA FROM IBM WATSON IOT SIMULATOR WHICH IS KIND OF VIRTUAL SENSOR.

-THIS UI WILL HELP THE FARMERS CONTROL THE IRRIGATION ACCORDING TO THE FIELD PARAMETERS AND THEY CAN DO SO FROM ANYWHERE.

## USE OF IBM WATSON

The screenshot displays the IBM Watson IoT Platform interface. At the top, the header includes the platform name, a user profile for 'guptamayankwork@gmail.com' with ID 'nxg6u2', and a navigation bar with 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area features a 'Browse' tab, an 'Add Device' button, and a table of devices. The table has columns for Device ID, Status, Device Type, Class ID, and Date Added. A single device is listed: 'Arduino' with status 'Disconnected', type 'IOTDevice', class 'Device', and added on 'Jun 9, 2020 1:07 AM'. Below the table, a status bar indicates '1 Simulation running'.

| Device ID | Status       | Device Type | Class ID | Date Added          |
|-----------|--------------|-------------|----------|---------------------|
| Arduino   | Disconnected | IOTDevice   | Device   | Jun 9, 2020 1:07 AM |

1 Simulation running

IBM Watson IoT Platform

guptamayankwork@gmail.com  
ID: nxg6u2

Browse

Action

Device Types

Interfaces

Add Device

Arduino

Disconnected

IOTDevice

Device

Jun 9, 2020 1:07 AM

→ ...

Identity

Device Information

Recent Events

State

Logs

Device ID

Device Type

Date Added

Added By

Connection Status

Arduino

IOTDevice

Jun 9, 2020 1:07 AM

guptamayankwork@gmail.com

Disconnected

Last Connected: Jun 14, 2020 2:12 AM

Client Address: 118.91.184.127 SecureToken

Duration: 36 minutes

Data Transferred: 354 B

Items per page: 50 | 1-1 of 1 item

1 Simulation running

nces

IISPS INT 2244 S...docx

IISPS INT 1205 S...docx

IISPS INT 1205 S...pdf

Show all

IBM Watson IoT Platform

guptamayankwork@gmail.com  
ID: nxg6u2

Delete

1 item selected

Cancel

Device ID

Status

Device Type

Class ID

Date Added

✓

✓

Arduino

Disconnected

IOTDevice

Device

Jun 9, 2020 1:07 AM

→ ...

Identity

Device Information

Recent Events

State

Logs

The recent events listed show the live stream of data that is coming and going from this device.

| Event   | Value  | Format | Last Received     |
|---------|--|--------|-------------------|
| event_1 | {"Temperature":91,"humidity":93,"distance":76} | json   | a few seconds ago |
| event_1 | {"Temperature":53,"humidity":84,"distance":56} | json   | a few seconds ago |
| event_1 | {"Temperature":92,"humidity":52,"distance":22} | json   | a few seconds ago |
| event_1 | {"Temperature":49,"humidity":66,"distance":63} |        |                   |
| event_1 | {"Temperature":90,"humidity":50,"distance":30} |        |                   |

1 Simulation running

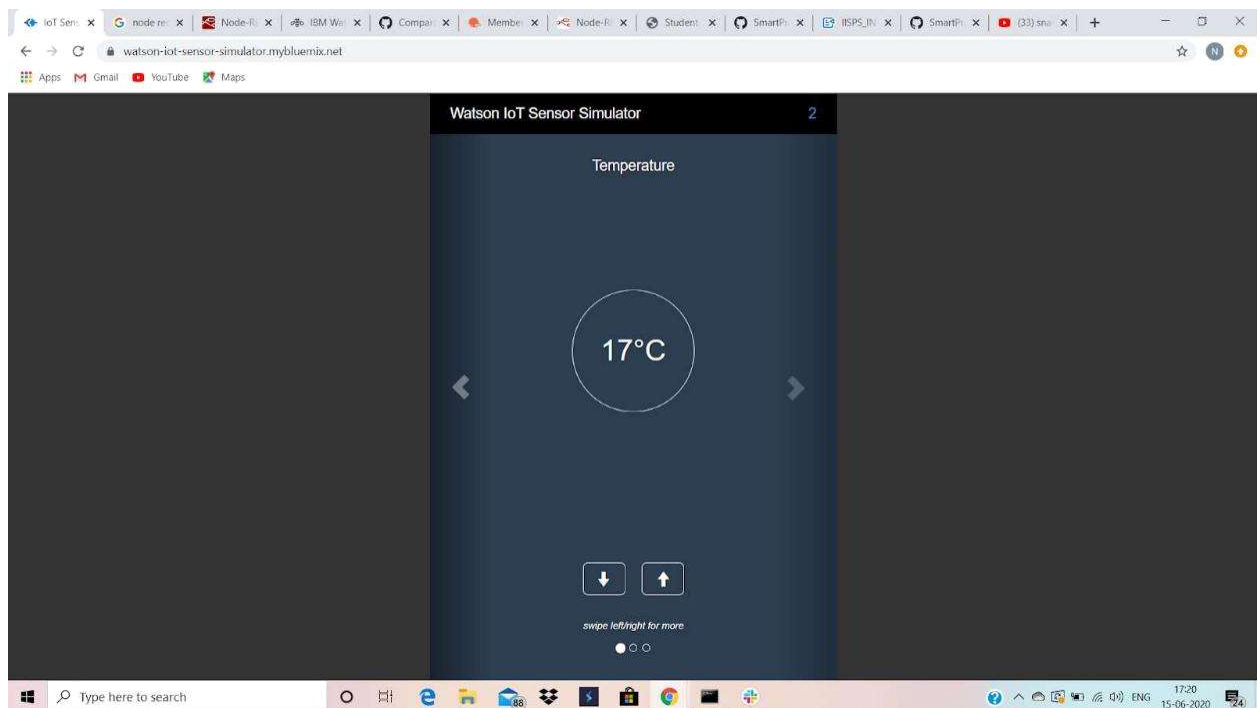
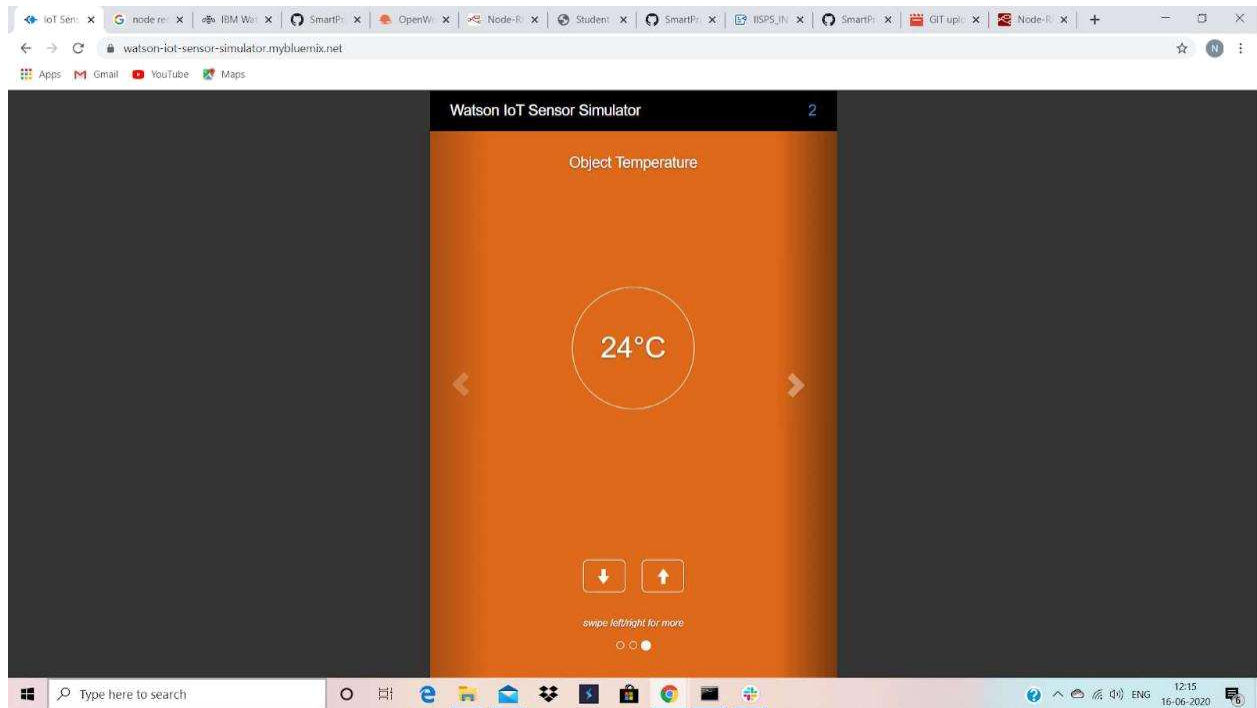
nces

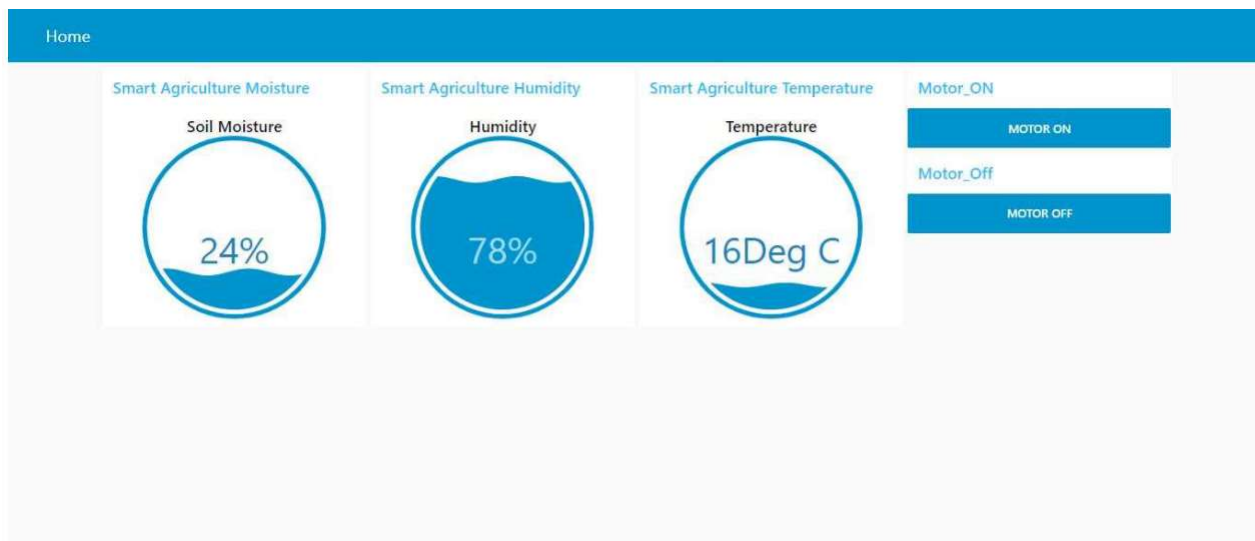
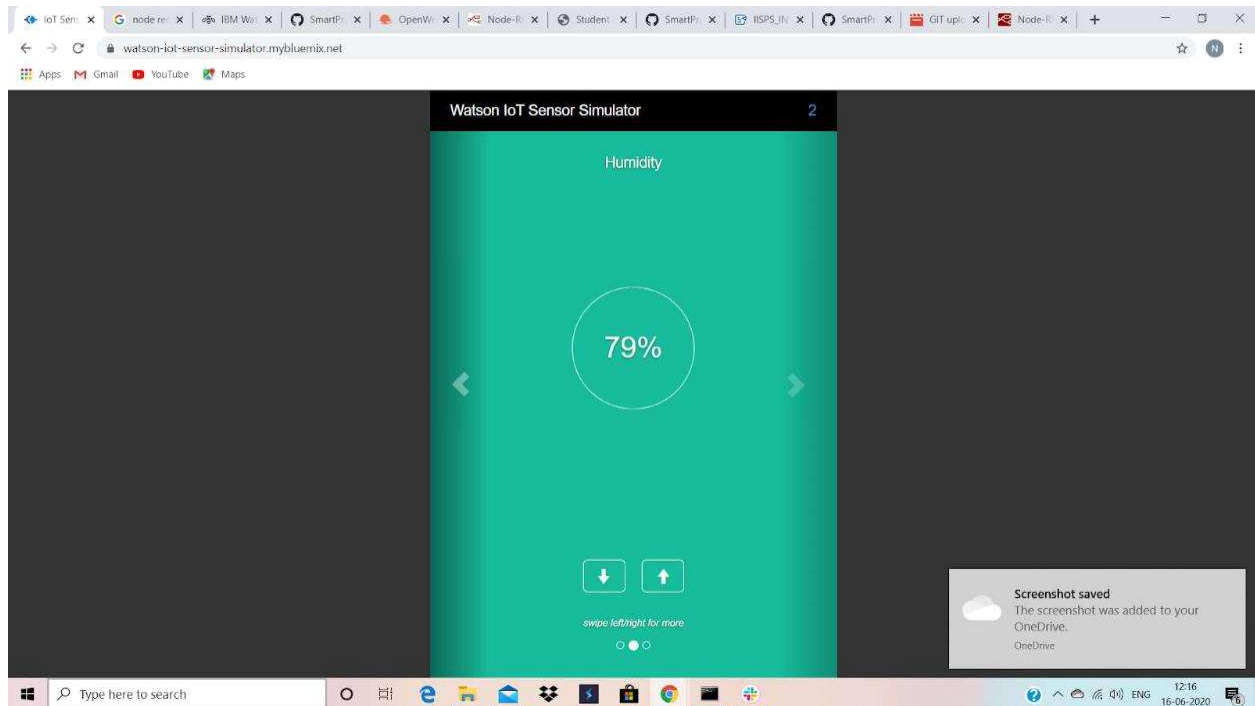
IISPS INT 2244 S...docx

IISPS INT 1205 S...docx

IISPS INT 1205 S...pdf

Show all

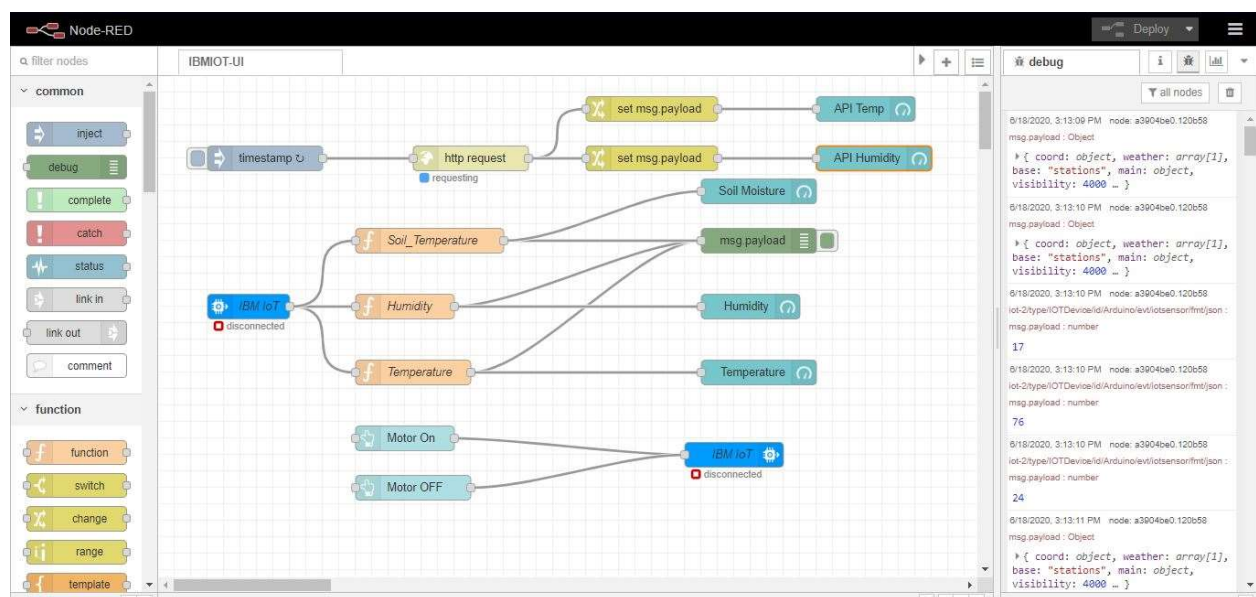




HERE WE HAVE ALL WORKING COMPONENTS SUCH AS THE IOT SENSOR.THE DEVICE WHICH CONNECTS TO THE SENSOR AND UPLOADS DATA INTO THE IBM CLOUD.THE MOTOR WHICH TAKES INPUT FROM THE WEB APP AND THEN UPLOADS INPUT VIA PYTHON CODE ON CLOUD.

THE BOTTOM CORNER OF THE IMAGE SHOWS SOME CARDS WHICH DISPLAYS THE DATA IN A VISUALLY APPEALING WAY THAN JUST NUMBERS SUCH AS LINE GRAPHS OR GAUGE WHICH MAKES IT EASY TO UNDERSTAND.

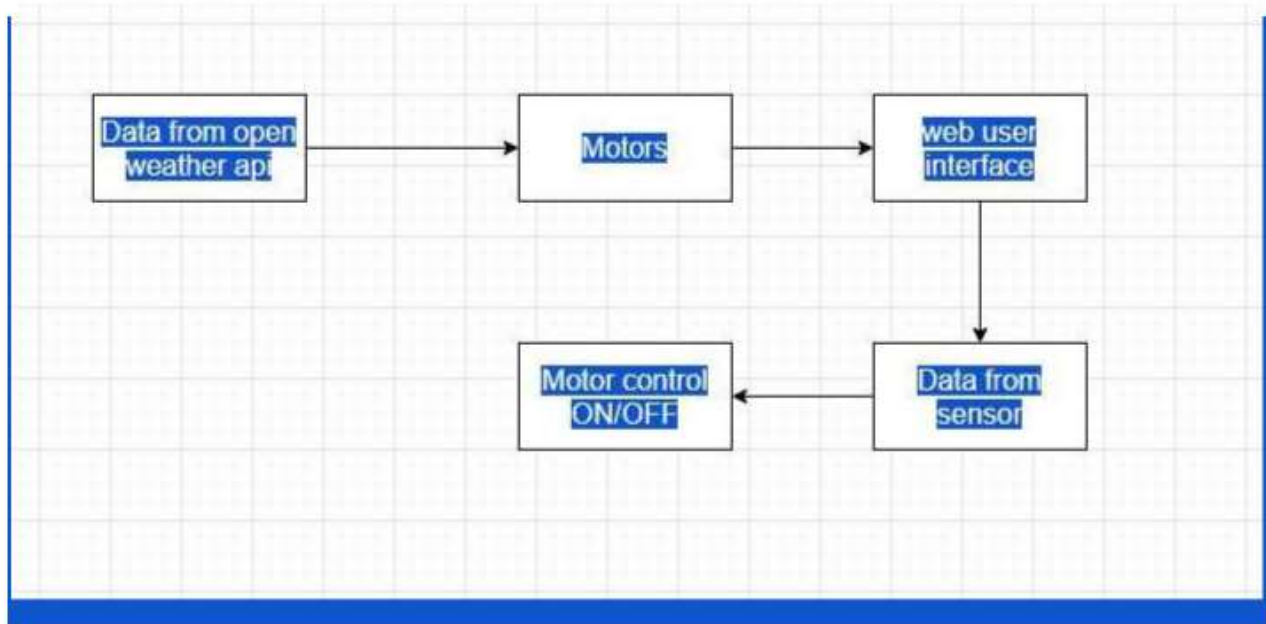
## FUNCTIONING OF THE APP



IN THIS FLOW WE HAVE IBM NODES TO GET THE IOT SENSOR DATA FROM IBM CLOUD AND PUSH THE MOTOR ON/OFF COMMANDS FROM THE WEB DASHBOARD FROM THE WEB DASHBOARD BACK TO THE IBM CLOUD.

HERE WE ARE USING HTTP REQUEST WHICH I AM USING TO GET CURRENT WEATHER DATA AND DISPLAY ON THE DASHBOARD.

## THEORY BLOCK DIAGRAM



THIS BLOCK DIAGRAM IS SHOWING THEORITICAL COMPUTATION OF WHAT ARE WE BASICALLY DOING IN THIS PROJECT.

SO, HERE WE ARE BASICALLY MONITORING THE WEATHER CONDITIONS AND COLLECTING DATA FROM OPEN WEATHER API.

THE REAL TIME FIELD CONDITIONS ARE MONITORED VIA SENSOR AND THE DATA IS COLLECTED VIA SENSOR.

AND ACCORDINGLY THE MOTOR IS CONTROLLED BY ON/OFF BUTTONS.

ALL THESE DATAS ARE SEND TO THE WEB USER INTERFACE AND THE MOTOR IS CONTROLLED ACCORDINGLY.



## NODE-RED ON COMMAND PROMPT

```
node-red
Microsoft Windows [Version 10.0.18362.900]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\nidhi singh>node-red
15 Jun 19:59:51 - [info]

Welcome to Node-RED
=====

15 Jun 19:59:51 - [info] Node-RED version: v1.0.6
15 Jun 19:59:51 - [info] Node.js version: v12.17.0
15 Jun 19:59:51 - [info] Windows_NT 10.0.18362 x64 LE
15 Jun 20:00:00 - [info] Loading palette nodes
15 Jun 20:00:27 - [info] Dashboard version 2.22.1 started at /ui
15 Jun 20:00:27 - [info] Settings file : \Users\nidhi singh\.node-red\settings.js
15 Jun 20:00:27 - [info] Context store : 'default' [module=memory]
15 Jun 20:00:27 - [info] User directory : \Users\nidhi singh\.node-red
15 Jun 20:00:27 - [warn] Projects disabled : editorTheme.projects.enabled=false
15 Jun 20:00:27 - [info] Flows file : \Users\nidhi singh\.node-red\flows_LAPTOP-MNRFJBMS.json
15 Jun 20:00:28 - [warn]

-----
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
-----

15 Jun 20:00:28 - [info] Starting flows
15 Jun 20:00:28 - [info] Started flows
15 Jun 20:00:28 - [info] Server now running at http://127.0.0.1:1880/
15 Jun 20:00:34 - [info] Stopping flows
15 Jun 20:00:34 - [info] Stopped flows
15 Jun 20:00:34 - [info] Starting flows
15 Jun 20:00:34 - [info] Started flows
[BaseClient:connect] Iotfclient is offline. Retrying connection
[BaseClient:connect] Iotfclient is offline. Retrying connection
[BaseClient:connect] Iotfclient is offline. Retrying connection
[BaseClient:connect] Iotfclient is offline. Retrying connection
[BaseClient:onError] Connection Error :: Error: getaddrinfo ENOTFOUND loc9vd.messaging.internetofthings.ibmcloud.com
16 Jun 00:23:33 - [error] [ibmiot out:IBM IoT] Error: getaddrinfo ENOTFOUND loc9vd.messaging.internetofthings.ibmcloud.com
[BaseClient:onError] Connection Error :: Error: getaddrinfo ENOTFOUND loc9vd.messaging.internetofthings.ibmcloud.com
16 Jun 00:23:33 - [error] [ibmiot in:IBM IoT] Error: getaddrinfo ENOTFOUND loc9vd.messaging.internetofthings.ibmcloud.com
[BaseClient:connect] Iotfclient is offline. Retrying connection
[BaseClient:connect] Iotfclient is offline. Retrying connection
```

## PROJECT SCHEDULE

| WEEK 1                             | WEEK 2                       | WEEK 3                                    | WEEK 4                                    |
|------------------------------------|------------------------------|---|---|
| planned my project                 | created device in IBM cloud  | started working with node-red             | started to prepare reports on zoho writer |
| set up the development environment | installed node red locally   | configured IBM iot sensor and node-red UI | pushed my files to github repository      |
| created accounts in IBM cloud      | installed the required nodes | downloaded python idle and ran the code   | recorded the feedback video               |

## PYTHON CODE

subscribebm.py - C:\Users\mayan\Desktop\ibmsubscribe-master\subscribebm.py (3.7.4)

File Edit Format Run Options Window Help

```
import time
import sys
import ibmiotf.application # to install pip install ibmiotf
import ibmiotf.device

#Provide your IBM Watson Device Credentials
organization = "nxg6u2" #replace the ORG ID
deviceType = "IOTDevice"#replace the Device type wi
deviceId = "Arduino"#replace Device ID
authMethod = "token"
authToken = "123456789" #Replace the authtoken

def myCommandCallback(cmd): # function for Callback
    print("Command received: %s" % cmd.data)
    if cmd.data['command']=='motoron':
        print("Motor ON IS RECEIVED")

    elif cmd.data['command']=='motoroff':
        print("Motor OFF IS RECEIVED")

    if cmd.command == "setInterval":

        if 'interval' not in cmd.data:
            print("Error - command is missing required information: 'interval'")
        else:
            interval = cmd.data['interval']

    elif cmd.command == "print":
        if 'message' not in cmd.data:
            print("Error - command is missing required information: 'message'")
        else:
            output=cmd.data['message']
            print(output)

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()
```

subscribebm.py - C:\Users\nidhi singh\Desktop\ibmsubscribe-master\subscribebm.py (3.0.3)

File Edit Format Run Options Window Help

```
import time
import sys
import ibmiotf.application # to install pip install ibmiotf
import ibmiotf.device

#Provide your IBM Watson Device Credentials:
organization = "lsc9yd" #replace the ORG ID
deviceType = "motor"#replace the Device type wi
deviceId = "2"#replace Device ID
authMethod = "token"
authToken = "12345678" #Replace the authtoken

def myCommandCallback(cmd): # function for callback
    print("Command received: %s" % cmd.data)
    if cmd.data['command']=='lighton':
        print("MOTOR ON IS RECEIVED")

    elif cmd.data['command']=='lightoff':
        print("MOTOR OFF IS RECEIVED")

    if cmd.command == "setInterval":

        if 'interval' not in cmd.data:
            print("Error - command is missing required information: 'interval'")
        else:
            interval = cmd.data['interval']

    elif cmd.command == "print":
        if 'message' not in cmd.data:
            print("Error - command is missing required information: 'message'")
        else:
            output=cmd.data['message']
            print(output)

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:
```

deviceCli.send\_datapoint("hello", "world", "greeting", 10)

In: 23 Col: 32

Type here to search

1845  
16-06-2020

```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help
Motor OFF IS RECEIVED
Command received: {'command': 'motoroff'}
Motor OFF IS RECEIVED
Command received: {'command': 'motoroff'}
Motor OFF IS RECEIVED
Command received: {'command': 'motoroff'}
Motor OFF IS RECEIVED
2020-06-17 17:42:40,212 ibmiotf.device.Client ERROR Unexpected disconnec
ct from the IBM Watson IoT Platform: 1
2020-06-17 17:42:42,144 ibmiotf.device.Client INFO Connected successfu
lly: d:nxg6u2:IOTDevice:Arduino
Command received: {'command': 'motoron'}
Motor ON IS RECEIVED
Command received: {'command': 'motoron'}
Motor ON IS RECEIVED
Command received: {'command': 'motoron'}
Motor ON IS RECEIVED
2020-06-17 17:42:43,759 ibmiotf.device.Client ERROR Unexpected disconnec
ct from the IBM Watson IoT Platform: 1
2020-06-17 17:42:45,737 ibmiotf.device.Client INFO Connected successfu
lly: d:nxg6u2:IOTDevice:Arduino
2020-06-17 17:42:47,377 ibmiotf.device.Client ERROR Unexpected disconnec
ct from the IBM Watson IoT Platform: 1
2020-06-17 17:42:49,234 ibmiotf.device.Client INFO Connected successfu
```

WHEN THE MOTOR COMMANDS ARE GIVEN ON THE NODE-RED DASHBOARD ,WE GET SIGNAL IN THE PYTHON PROGRAM.

## BIBLIOGRAPHY

- 1.<https://cloud.ibm.com/>
- 2.<https://watson-iot-sensor-simulator.mybluemix.net/>
- 3.<https://loc9vd.internetofthings.ibmcloud.com/dashboard/devices/browse>
- 4.<http://localhost:1880/ui/#!/0?socketid=cxfzbqCJL5U8bODKAAAc>

5. <http://localhost:1880/#flow/23ee611b.2e0ebe>

6. <https://github.com/SmartPracticeschool/IISPS-INT-2244-Smart-Agriculture-system-based-on-IoT>