

## **SPRINT DELIVERY – 4**

|                     |  |
|---------------------|--|
| <b>Team ID</b>      | PNT2022TMID20250                         |
| <b>Project Name</b> | IoT Enabled Smart<br>Farming Application |
| <b>Date</b>         | 10 November 2022                         |

## 5.5 Receiving commands from IBM cloud using Python program

```
import time import
```

```
sys
```

```
import ibmiotf.application
```

```
import ibmiotf.device import
```

```
random
```

```
#Provide your IBM Watson Device
```

```
Credentialsorganization = "157uf3"
```

```
deviceType = "abcd" deviceId = "7654321"
```

```
authMethod = "token" authToken =
```

```
"87654321"
```

```
# Initialize GPIO
```

```
def myCommandCallback(cmd): print("Command
```

```
received: %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 "hello" with value "world" into the cloud as an  
event of type "greeting" 10 times deviceCli.connect()
```

while True:

```
    #Get Sensor Data from  
DHT11
```

```
temp=random.randint(90,110)
```

```
Humid=random.randint(60,100)
```

```
Mois=random. Randint(20,120)
```

```
    data = { 'temp' : temp, 'Humid': Humid ,  
‘Mois’: Mois}
```

```
    #print data    def  
myOnPublishCallback(  
):
```

```
    print ("Published Temperature = %s C" % temp, "Humidity = %s %%"  
%Humid, “Moisture =%s deg c” % Mois “to IBM Watson")
```

```
    success = deviceCli.publishEvent("IoTSensor", "json", data,  
qos=0,on_publish=myOnPublishCallback)    if not success:  
        print("Not connected to IoTF")
```

```
time.sleep(10)
```

```
    deviceCli.commandCallback = myCommandCallback #  
Disconnect the device and application from the cloud  
deviceCli.disconnect()
```

```
ibmiotpublishsubscribe.py - C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py (3.7.0)
File Edit Format Run Options Window Help

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

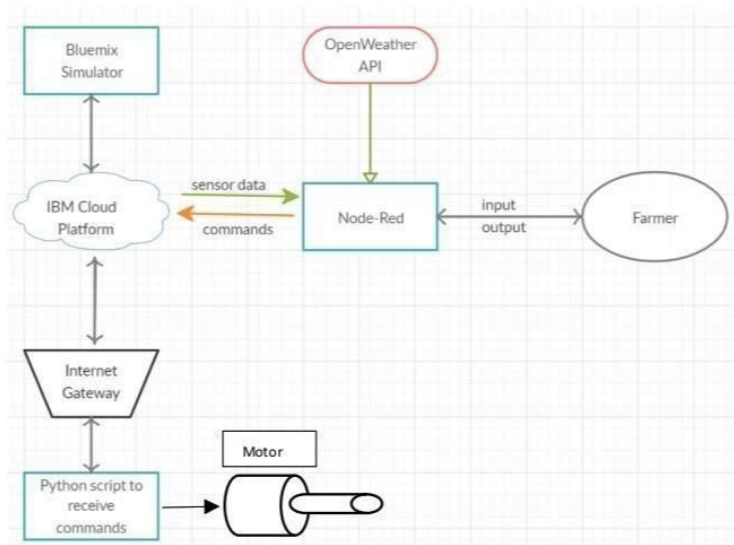
#Provide your IBM Watson Device Credentials
organization = "157uf3"
deviceType = "abcd"
deviceId = "7654321"
authMethod = "token"
authToken = "87654321"

# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %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": authMe
deviceCli = ibmiotf.device.Client(deviceOptions)
#.....
Ln: 22 Col: 21
```

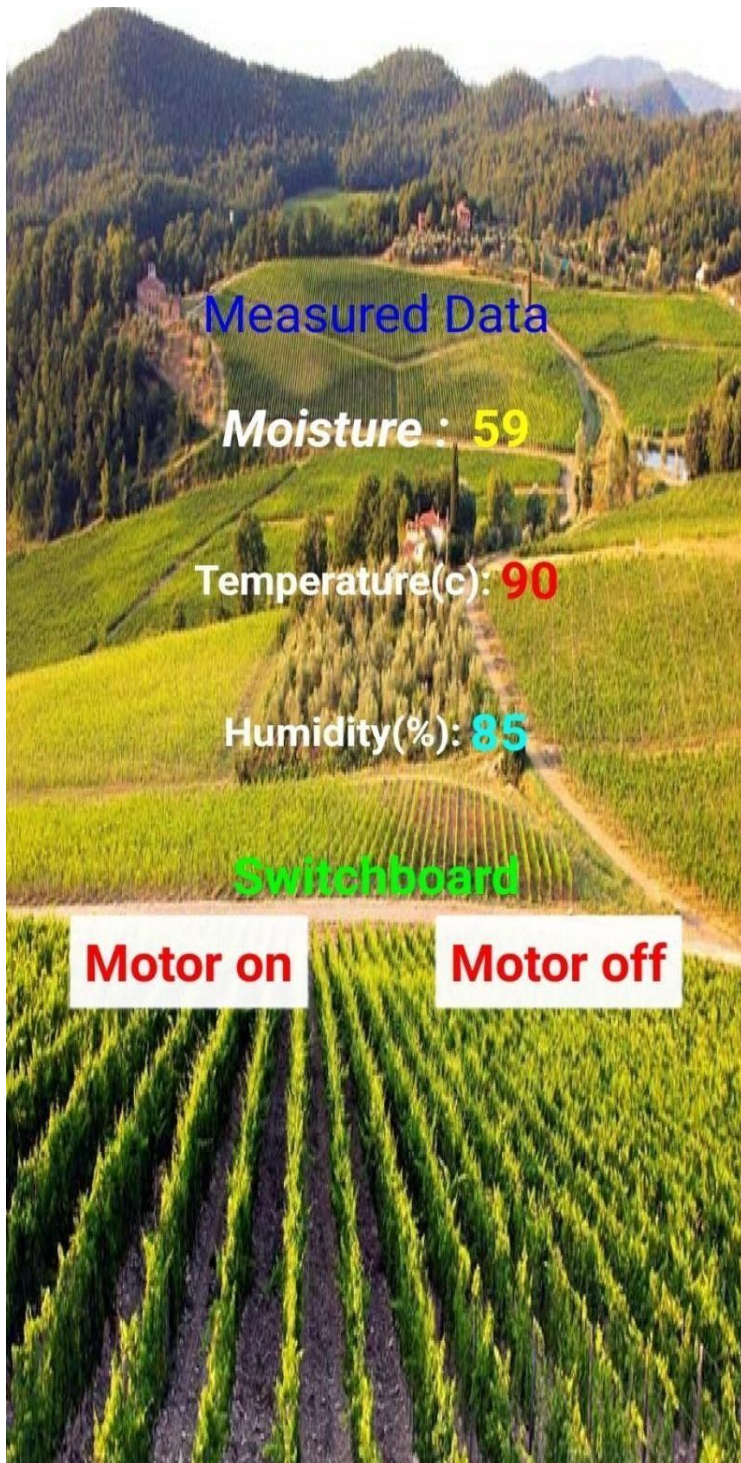
```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py =====
2022-11-07 20:01:24,074 ibmiotf.device.Client INFO Connected successfully: d:157uf3:abcd:7654321
Published Moisture = 90 deg C Temperature = 96 C Humidity = 76 % to IBM Watson
Published Moisture = 102 deg C Temperature = 110 C Humidity = 68 % to IBM Watson
Published Moisture = 45 deg C Temperature = 99 C Humidity = 100 % to IBM Watson
Command received: motoron
motor is on
Published Moisture = 77 deg C Temperature = 91 C Humidity = 85 % to IBM Watson
Published Moisture = 73 deg C Temperature = 94 C Humidity = 86 % to IBM Watson
Command received: motoroff
motor is off
Published Moisture = 101 deg C Temperature = 104 C Humidity = 87 % to IBM Watson
```

## 6. Flow Chart



## 7. Observations & Results

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ELCOT\Downloads\ibmiotpublishsubscribe.py =====
2022-11-07 20:01:24,074 ibmiotf.device.Client INFO Connected successfully: d:157uf3:abcd:7654321
Published Moisture = 90 deg C Temperature = 96 C Humidity = 76 % to IBM Watson
Published Moisture = 102 deg C Temperature = 110 C Humidity = 68 % to IBM Watson
Published Moisture = 45 deg C Temperature = 99 C Humidity = 100 % to IBM Watson
Command received: motoron
motor is on
Published Moisture = 77 deg C Temperature = 91 C Humidity = 85 % to IBM Watson
Published Moisture = 73 deg C Temperature = 94 C Humidity = 86 % to IBM Watson
Command received: motoroff
motor is off
Published Moisture = 101 deg C Temperature = 104 C Humidity = 87 % to IBM Watson
```



Measured Data

Moisture : 59

Temperature(c): 90

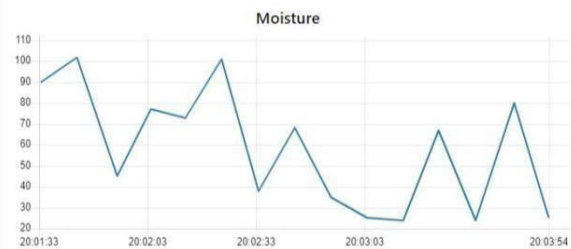
Humidity(%): 85

Switchboard

Motor on

Motor off

### Farming Measure Data



### Switchboard

MOTOR SWITCH ON

MOTOR SWITCH OFF

## 8. Advantages & Disadvantages Advantages:

- Farms can be monitored and controlled remotely.
- Increase in convenience to farmers.
- Less labor cost.
- Better standards of living.

## Disadvantages:

- Lack of internet/connectivity issues.
- Added cost of internet and internet gateway infrastructure.
- Farmers wanted to adapt the use of Mobile App.

## 9. Conclusion

Thus the objective of the project to implement an IOT system in order to help farmers to control and monitor their farms has been implemented successfully.