

## **SPRINT – 4**

Date	15 NOVEMBER 2022
Team ID	PNT2022TMID04663
Project Name	Smart Farmer-IoT Enabled smartFarming Application

### **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 Credentials
organization = "yy3qcm"
deviceType = "ibm22"
deviceId = "123"
authMethod = "token"
authToken = "12345678"

# 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("enter crt command")
S
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(0,100)
    Humid=random.randint(0,100)
    moisture=random.randint(0,100)
    data = { 'temp' : temp, 'Humid': Humid , 'moisture' : moisture }
    #print data
    def myOnPublishCallback():
        print ("Published Temperature = %s C" % temp, "Humidity = %s %" % Humid,"moisture = %s
%%" % moisture, "to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
    if not success:
        print("Not connected to IoT")
    time.sleep(3)
    deviceCli.commandCallback = myCommandCallback

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

```

- DATA SEND FROM PYTHON PROGRAM :

The image shows two windows. The left window is a text editor showing a Python script named `ibmcode.py`. The script imports `time`, `sys`, `ibmiotf.application`, `ibmiotf.device`, and `random`. It defines credentials for an IBM Watson IoT device and a function `myCommandCallback` that prints received commands. The right window is a Python 3.7.0 Shell showing the execution of the script. It displays the connection status and a series of temperature, humidity, and moisture readings published to IBM Watson.

```

ibmcode.py - C:\Users\Nithin.R\AppData\Local\Programs\Python\Python37\ibmcode.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 = "yy3qcm"
deviceType = "ibm22"
deviceId = "123"
authMethod = "token"
authToken = "12345678"

# 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("enter crt command")

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 infor
mation.
>>>
RESTART: C:\Users\Nithin.R\AppData\Local\Programs\Python
\Python37\ibmcode.py
2022-11-19 00:43:23,676 ibmiotf.device.Client INFO
Connected successfully: d:yy3qcm:ibm22:123
Published Temperature = 20 C Humidity = 68 % moisture = 1
5 % to IBM Watson
Published Temperature = 45 C Humidity = 28 % moisture = 3
6 % to IBM Watson
Published Temperature = 100 C Humidity = 56 % moisture =
39 % to IBM Watson
Published Temperature = 34 C Humidity = 57 % moisture = 8
3 % to IBM Watson
Published Temperature = 17 C Humidity = 76 % moisture = 9
0 % to IBM Watson
Published Temperature = 22 C Humidity = 33 % moisture = 9
8 % to IBM Watson
Published Temperature = 37 C Humidity = 81 % moisture = 8
4 % to IBM Watson
Published Temperature = 43 C Humidity = 31 % moisture = 6
6 % to IBM Watson
Published Temperature = 75 C Humidity = 83 % moisture = 4
2 % to IBM Watson

```

- DATA RECEIVED IN IBM CLOUD :

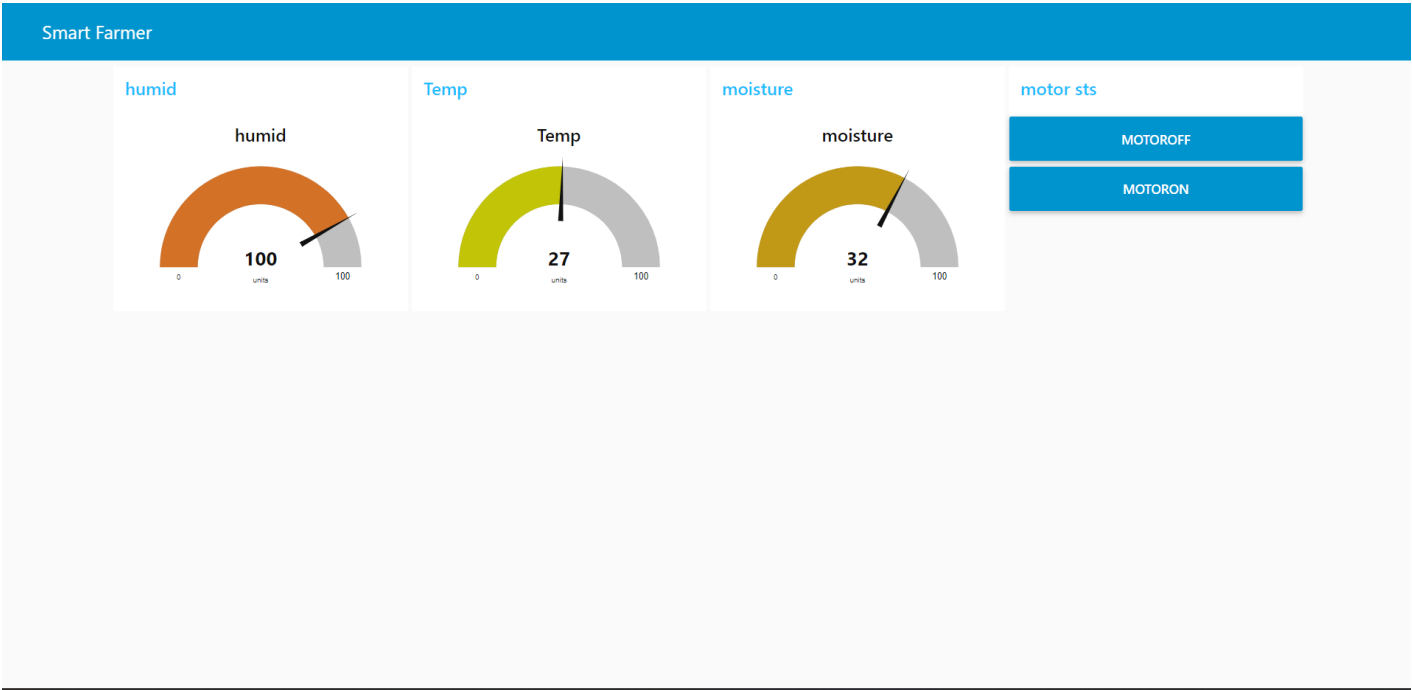
The image shows the IBM Watson IoT Platform interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A search bar is present with the text 'Search by Device ID'. The main content area displays a table of devices. The selected device is '123', which is 'Connected' and of type 'ibm22'. Below the table, the 'Recent Events' tab is active, showing a stream of data events from the device.

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123	Connected	ibm22	Device	Nov 17, 2022 3:03 PM	

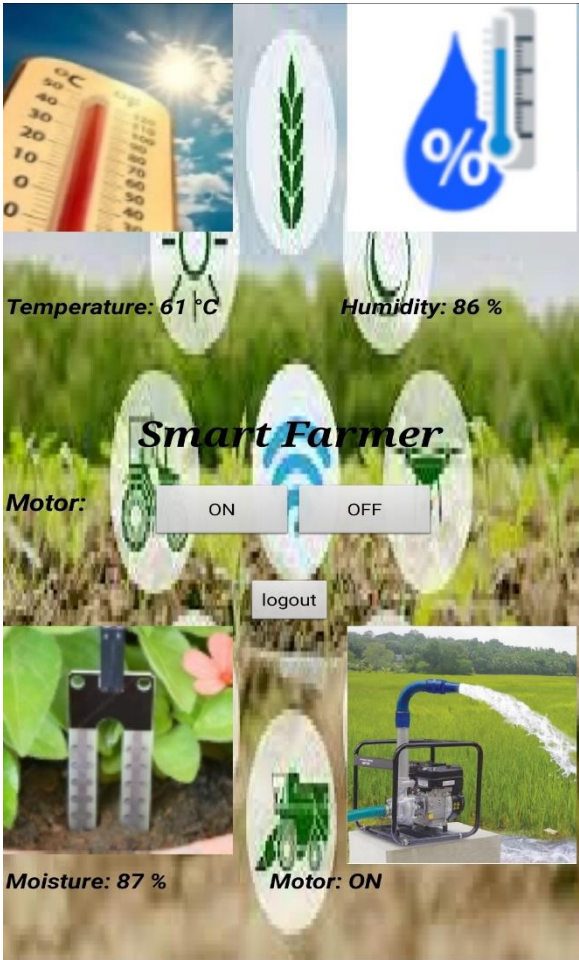
  

Event	Value	Format	Last Received
IoTSensor	{"temp":48,"Humid":88,"moisture":31}	json	a few seconds ago
IoTSensor	{"temp":42,"Humid":53,"moisture":69}	json	a few seconds ago
IoTSensor	{"temp":40,"Humid":32,"moisture":64}	json	a few seconds ago
IoTSensor	{"temp":79,"Humid":29,"moisture":39}	json	a few seconds ago
IoTSensor	{"temp":81,"Humid":19,"moisture":92}	json	a few seconds ago

- DATA RECEIVED IN NODE – RED DASHBOARD (WEB UI)



- DATA RECEIVED IN MOBILE APP



- COMMAND RECEIVED FROM WEB UI AND MOBILE APP

- MOTOR ON COMMAND



\*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: D:\IBM PROJECT\python 3.7\ibmiotpython.py =====

2022-11-14 14:22:24,419 ibmiotf.device.Client INFO Connected successfully: d:p2cfk6:SMART:15

Published Temperature = 68 C Humidity = 66 % Soil Moisture = 78 % to IBM Watson

Published Temperature = 16 C Humidity = 85 % Soil Moisture = 39 % to IBM Watson

Command received: motoron

motor is on

Published Temperature = 39 C Humidity = 32 % Soil Moisture = 75 % to IBM Watson

Command received: motoron

motor is on

Published Temperature = 48 C Humidity = 21 % Soil Moisture = 5 % to IBM Watson

#### ○ MOTOR OFF COMMAND



```
*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: D:\IBM PROJECT\python 3.7\ibmiotpython.py =====
2022-11-14 14:22:24,419 ibmiotf.device.Client INFO Connected successfully: d:p2cfk6:SMART:15
Published Temperature = 68 C Humidity = 66 % Soil Moisture = 78 % to IBM Watson
Published Temperature = 16 C Humidity = 85 % Soil Moisture = 39 % to IBM Watson
Command received: motoron
motor is on
Published Temperature = 39 C Humidity = 32 % Soil Moisture = 75 % to IBM Watson
Command received: motoron
motor is on
Published Temperature = 48 C Humidity = 21 % Soil Moisture = 5 % to IBM Watson
Published Temperature = 9 C Humidity = 29 % Soil Moisture = 44 % to IBM Watson
Published Temperature = 85 C Humidity = 64 % Soil Moisture = 17 % to IBM Watson
Command received: motoroff
motor is off
Published Temperature = 12 C Humidity = 43 % Soil Moisture = 94 % to IBM Watson
Command received: motoroff
motor is off
Published Temperature = 72 C Humidity = 86 % Soil Moisture = 0 % to IBM Watson
Published Temperature = 100 C Humidity = 95 % Soil Moisture = 90 % to IBM Watson
|
```

## ADVANTAGES:

- Less labour cost.
- Field can be monitored the environment parameters and controlled the motor remotely.
- Better standards of living.
- Farmers can also monitor and control the farm field by Web UI.
- Increase in convenience to farmers.

## DISADVANTAGES:

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

## CONCLUSION:

Thus, the objective of the project is to implement an IOT system in order to help farmers to control the motor function and monitor the environment parameters like temperature, humidity and soil moisture of their farms has been implemented successfully.