

SPRINT 4

Date	16 November 2022
Team ID	PNT2022TMID21337
Project Name	Project – Smart Farmer-IoT Enabled Smart Farming Application

CONFIGURATION OF NODE-RED TO SEND COMMANDS TO IBM CLOUD

Here we add two buttons in UI

1 -> for motor on

2 -> for motor off

We used a function node to analyse the data received and assign command to each number.

The python code for the analysis is:

```
if status == "motoron":  
    print ("motor is on")  
elif status == "motoroff":  
    print ("motor is off")  
else :  
    print ("please send proper  
command")
```

Code:

```
import time  
import sys  
import ibmiotf.application  
import ibmiotf.device  
import random  
  
#Provide your IBM Watson Device  
Credentials  
organization = "obbnyv"  
deviceType = "raspberrypi"
```

```

deviceId = "123456789"
authMethod = "token"
authToken = "12345678910"

# 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
```

```
    ph=random.randint(3,9)
```

```
    moisture=random.randint(0,110)
```

```
    temperature=random.randint(-
20,125)
```

```
    Humid=random.randint(0,100)
```

```
    data = {'ph':ph, 'moisture':
moisture, 'temperature' : temperature,
'Humid': Humid }
```

```
    #print data
```

```
    def myOnPublishCallback():
```

```
        print ("Published Soil Moisture
= %s %% " %moisture,"Temperature =
%s C" % temperature, "Humidity = %s
%% " % Humid, "ph: %s "%ph, "to
IBM Watson")
```

```
    success =
deviceCli.publishEvent("IoTSensor",
"json", data, qos=0,
on_publish=myOnPublishCallback)
```

```
    if not success:
```

```
        print("Not connected to IoTTF")
```

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

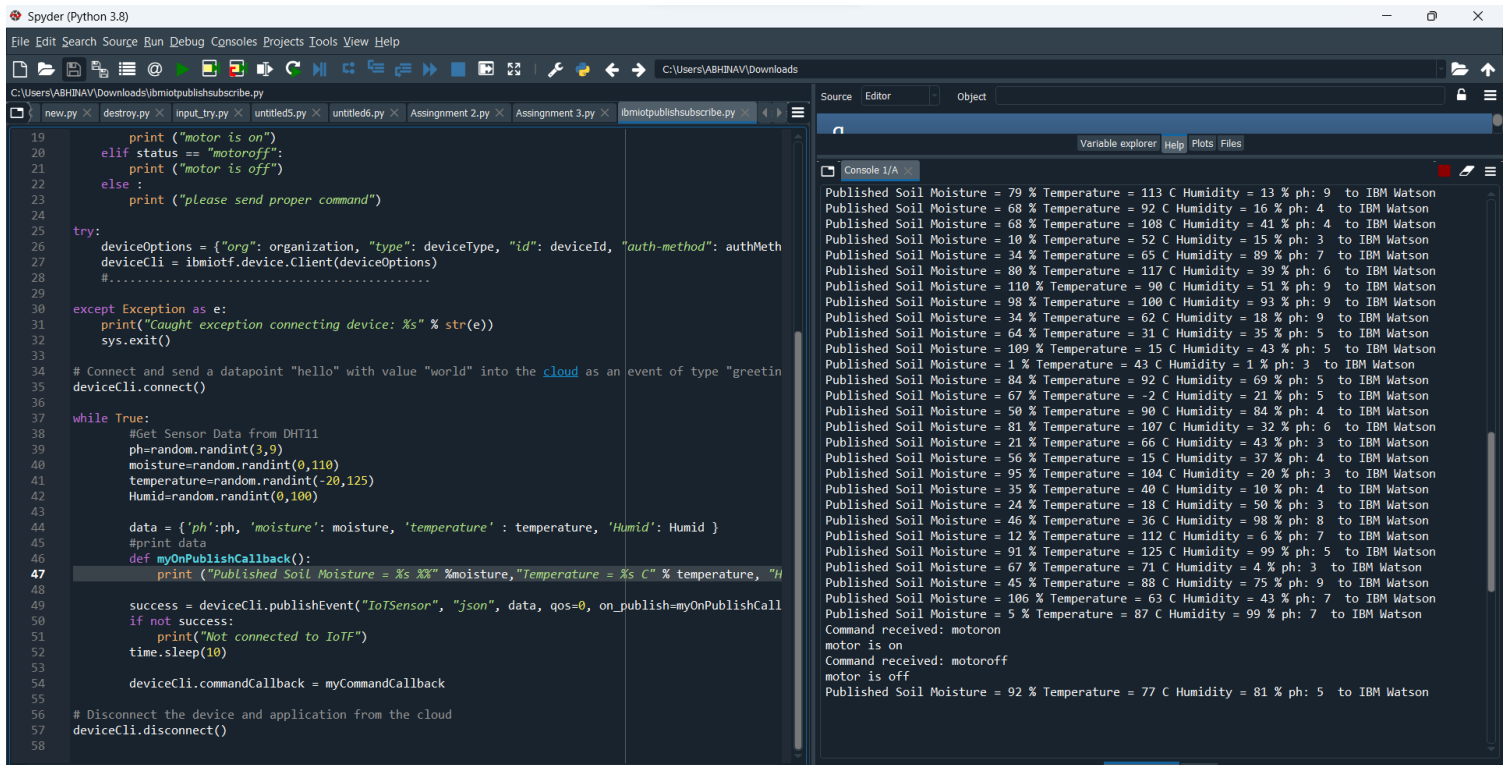
```
deviceCli.commandCallback =
```

myCommandCallback

Disconnect the device and application
from the cloud

deviceCli.disconnect()

Output:



The screenshot shows the Spyder Python IDE interface. The left pane displays a Python script with the following code:

```
19 print ("motor is on")
20 elif status == "motoroff":
21     print ("motor is off")
22 else :
23     print ("please send proper command")
24
25
26 try:
27     deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMeth
28     deviceCli = ibmiotf.device.Client(deviceOptions)
29     #.....
30 except Exception as e:
31     print("Caught exception connecting device: %s" % str(e))
32     sys.exit()
33
34 # Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greetin
35 deviceCli.connect()
36
37 while True:
38     #Get Sensor Data from DHT11
39     ph=random.randint(3,9)
40     moisture=random.randint(0,110)
41     temperature=random.randint(-20,125)
42     Humid=random.randint(0,100)
43
44     data = {'ph':ph, 'moisture': moisture, 'temperature': temperature, 'Humid': Humid }
45     #print data
46     def myOnPublishCallback():
47         print ("Published Soil Moisture = %s %%" %moisture,"Temperature = %s C" % temperature, "H
48
49     success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCall
50     if not success:
51         print("Not connected to IoTF")
52         time.sleep(10)
53
54     deviceCli.commandCallback = myCommandCallback
55
56 # Disconnect the device and application from the cloud
57 deviceCli.disconnect()
58
```

The right pane shows the console output, which includes the following text:

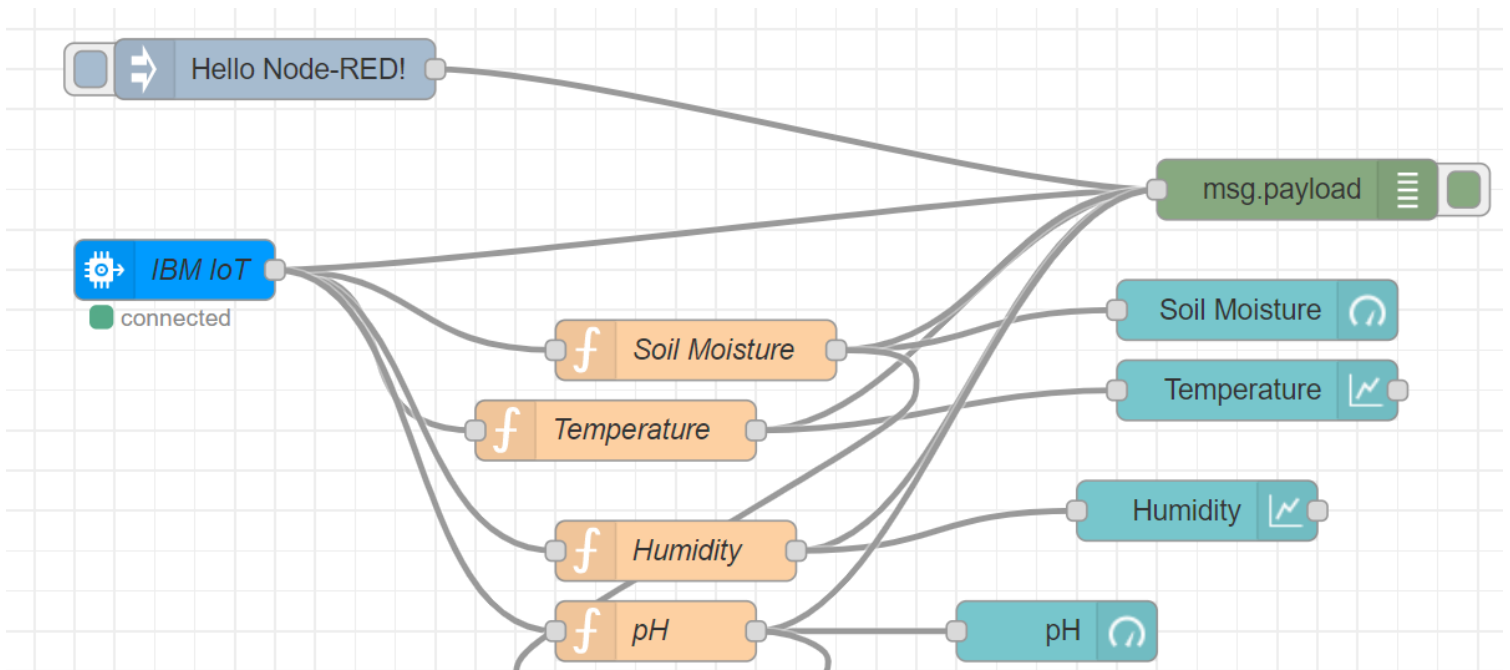
```
Published Soil Moisture = 79 % Temperature = 113 C Humidity = 13 % ph: 9 to IBM Watson
Published Soil Moisture = 68 % Temperature = 92 C Humidity = 16 % ph: 4 to IBM Watson
Published Soil Moisture = 68 % Temperature = 108 C Humidity = 41 % ph: 4 to IBM Watson
Published Soil Moisture = 10 % Temperature = 52 C Humidity = 15 % ph: 3 to IBM Watson
Published Soil Moisture = 34 % Temperature = 65 C Humidity = 89 % ph: 7 to IBM Watson
Published Soil Moisture = 80 % Temperature = 117 C Humidity = 39 % ph: 6 to IBM Watson
Published Soil Moisture = 110 % Temperature = 90 C Humidity = 51 % ph: 9 to IBM Watson
Published Soil Moisture = 98 % Temperature = 100 C Humidity = 93 % ph: 9 to IBM Watson
Published Soil Moisture = 34 % Temperature = 62 C Humidity = 18 % ph: 9 to IBM Watson
Published Soil Moisture = 64 % Temperature = 31 C Humidity = 35 % ph: 5 to IBM Watson
Published Soil Moisture = 109 % Temperature = 15 C Humidity = 43 % ph: 5 to IBM Watson
Published Soil Moisture = 1 % Temperature = 43 C Humidity = 1 % ph: 3 to IBM Watson
Published Soil Moisture = 84 % Temperature = 92 C Humidity = 69 % ph: 5 to IBM Watson
Published Soil Moisture = 67 % Temperature = -2 C Humidity = 21 % ph: 5 to IBM Watson
Published Soil Moisture = 50 % Temperature = 90 C Humidity = 84 % ph: 4 to IBM Watson
Published Soil Moisture = 81 % Temperature = 107 C Humidity = 32 % ph: 6 to IBM Watson
Published Soil Moisture = 21 % Temperature = 66 C Humidity = 43 % ph: 3 to IBM Watson
Published Soil Moisture = 56 % Temperature = 15 C Humidity = 37 % ph: 4 to IBM Watson
Published Soil Moisture = 95 % Temperature = 104 C Humidity = 20 % ph: 3 to IBM Watson
Published Soil Moisture = 35 % Temperature = 40 C Humidity = 10 % ph: 4 to IBM Watson
Published Soil Moisture = 24 % Temperature = 18 C Humidity = 50 % ph: 3 to IBM Watson
Published Soil Moisture = 46 % Temperature = 36 C Humidity = 98 % ph: 8 to IBM Watson
Published Soil Moisture = 12 % Temperature = 112 C Humidity = 6 % ph: 7 to IBM Watson
Published Soil Moisture = 91 % Temperature = 125 C Humidity = 99 % ph: 5 to IBM Watson
Published Soil Moisture = 67 % Temperature = 71 C Humidity = 4 % ph: 3 to IBM Watson
Published Soil Moisture = 45 % Temperature = 88 C Humidity = 75 % ph: 9 to IBM Watson
Published Soil Moisture = 106 % Temperature = 63 C Humidity = 43 % ph: 7 to IBM Watson
Published Soil Moisture = 5 % Temperature = 87 C Humidity = 99 % ph: 7 to IBM Watson
Command received: motoron
motor is on
Command received: motoroff
motor is off
Published Soil Moisture = 92 % Temperature = 77 C Humidity = 81 % ph: 5 to IBM Watson
```

Adjusting User Interface

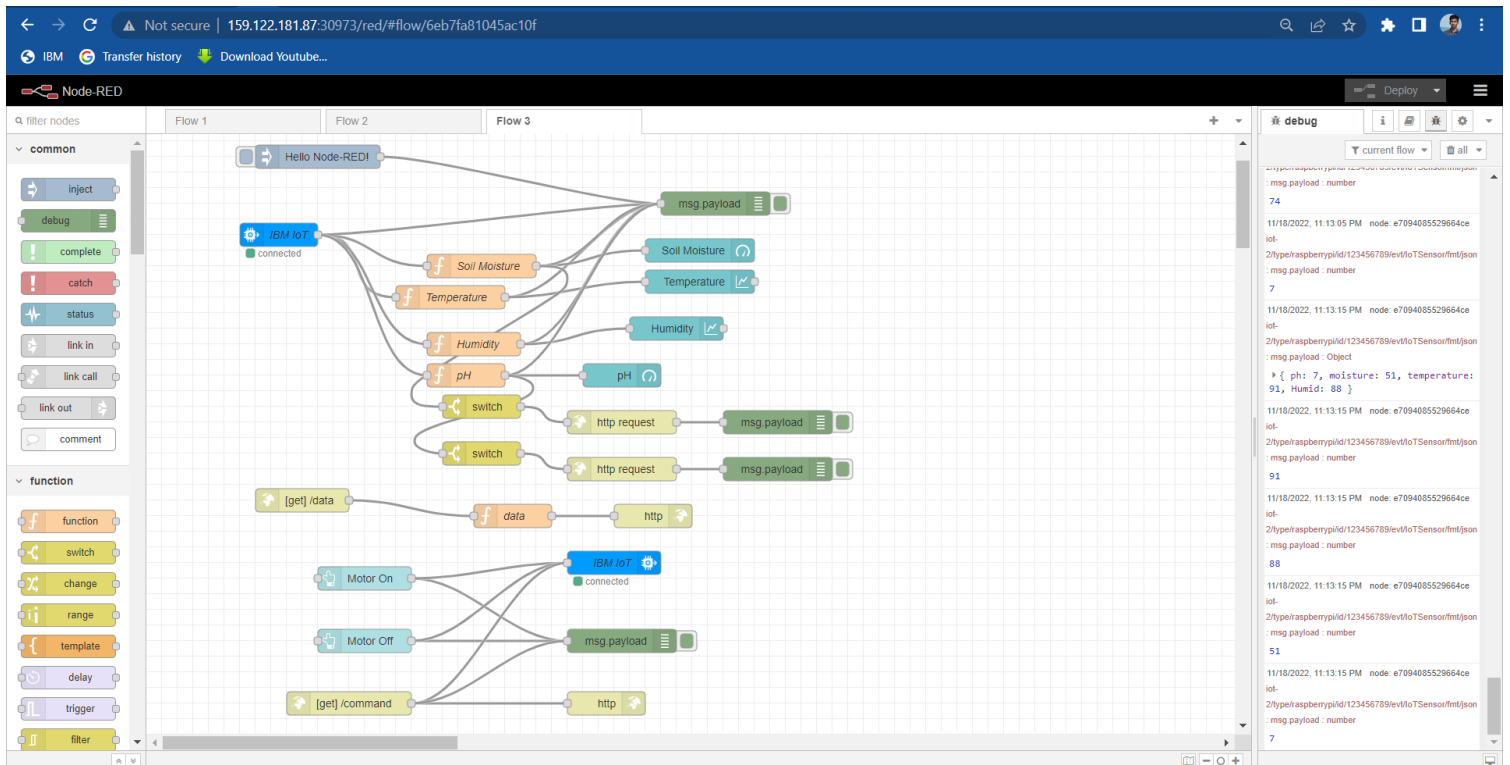
In order to display the parsed JSON data a Node-Red dashboard is created.

Here we are using Gauges, text and button nodes to display in the UI and helps to monitor the parameters and control the farm equipment.

Below images we started to create the flow 1



COMPLETE PROGRAM FLOW:



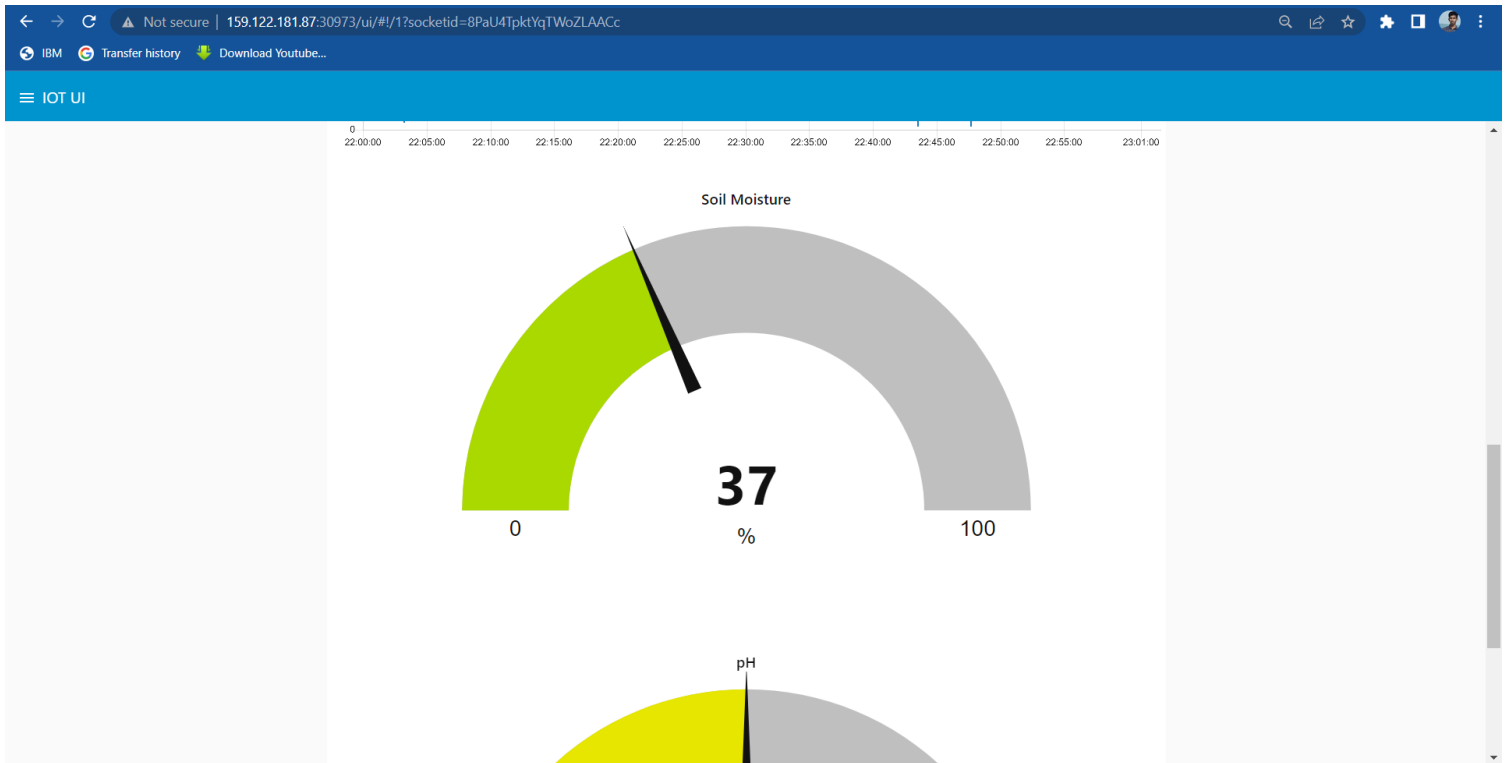
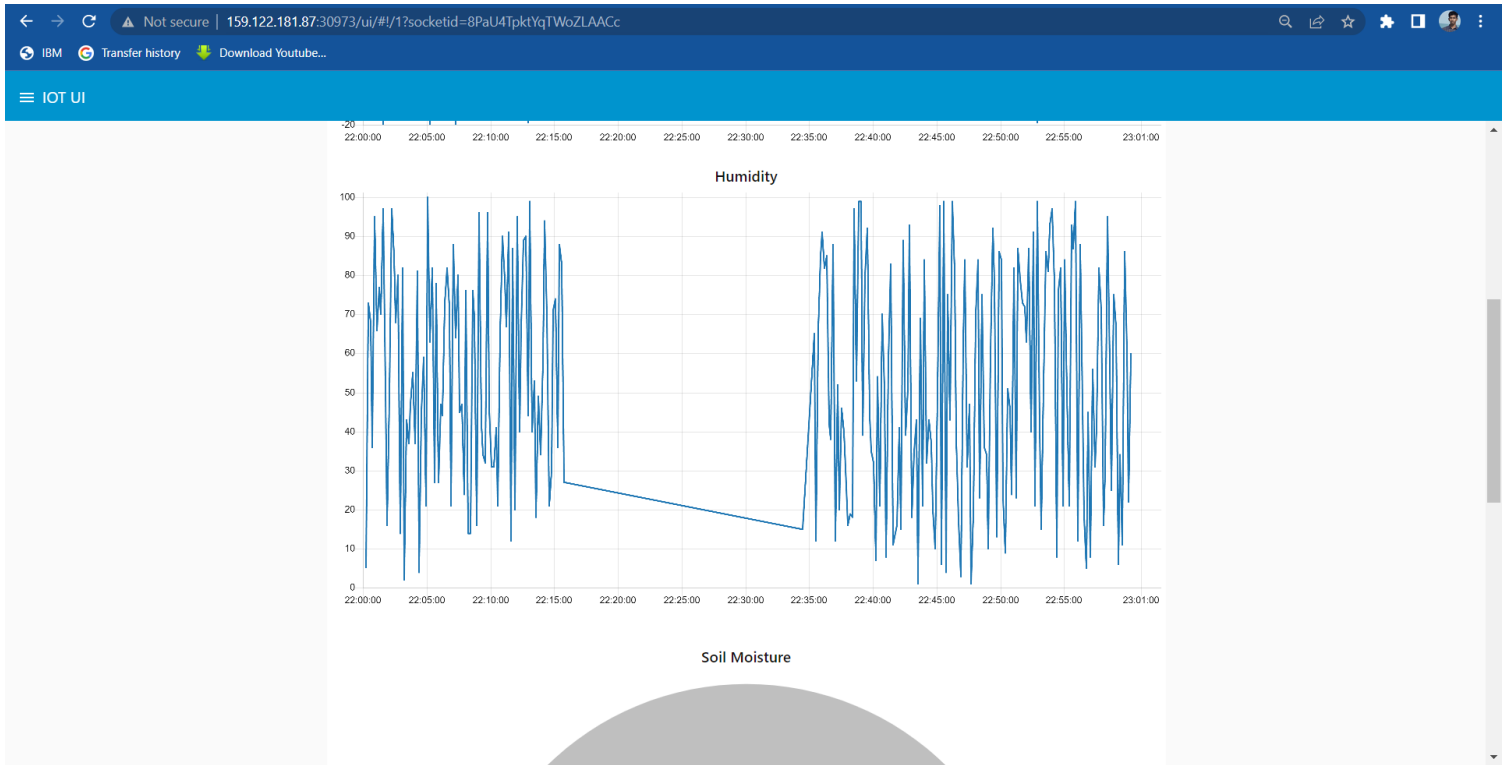
HTML Response:

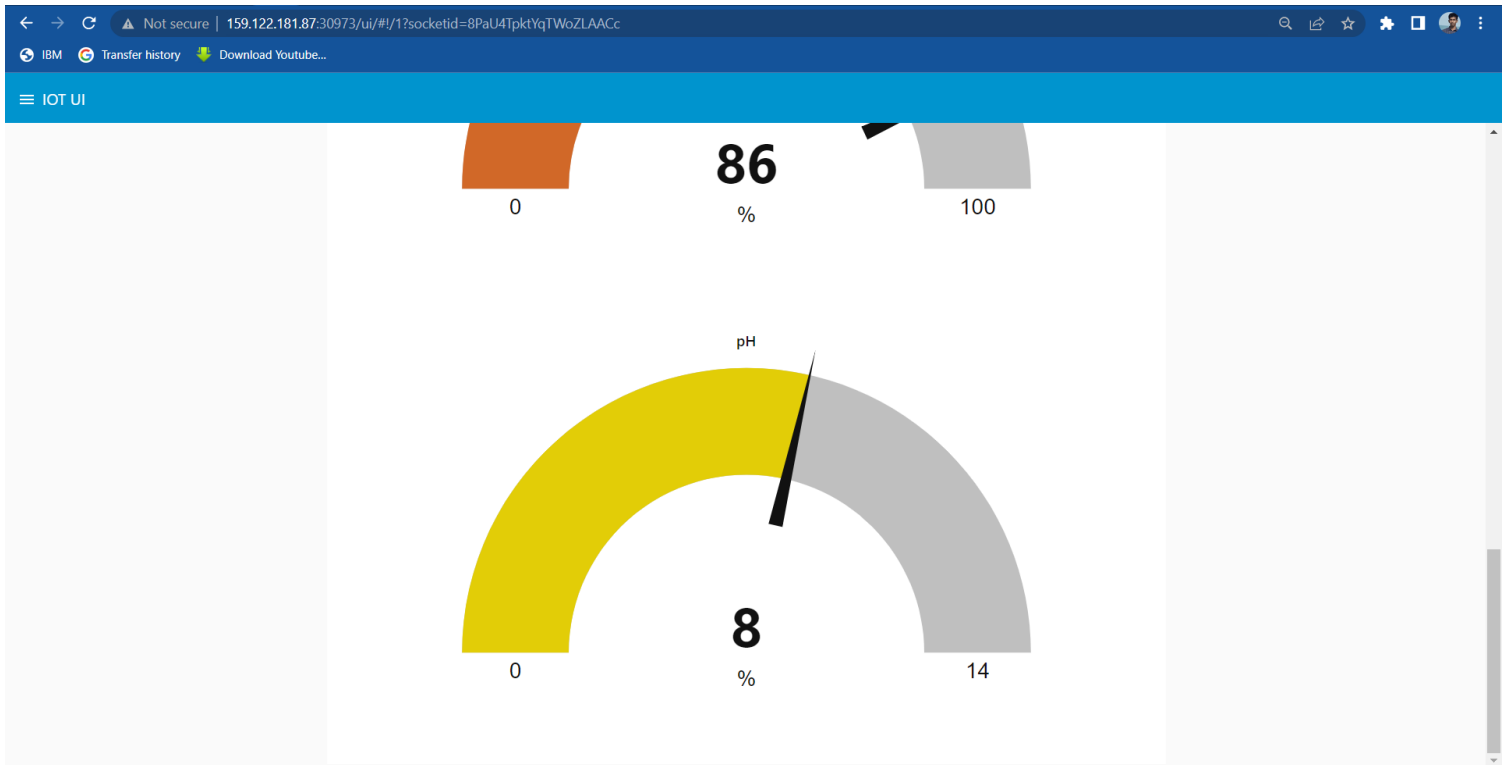


```
{"pH":5,"Soil":109,"Temperature":15,"Humid":43}
```

UI Dashboard:



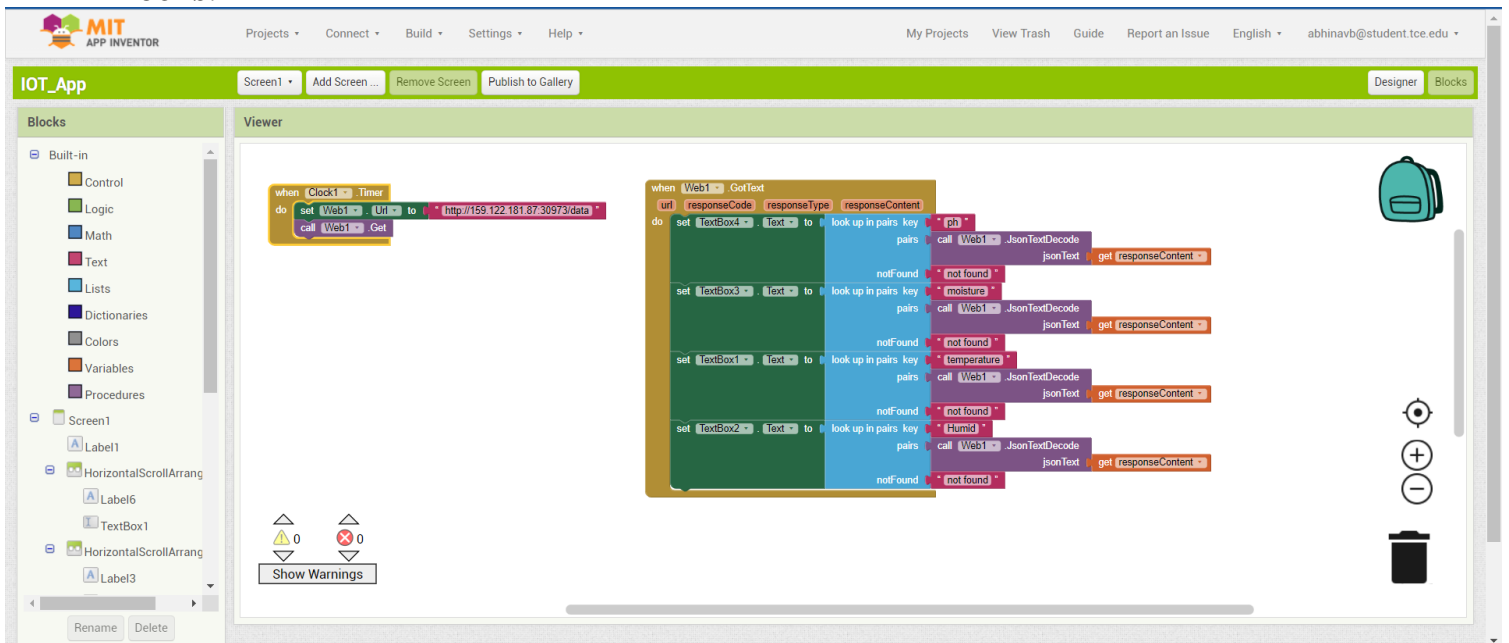


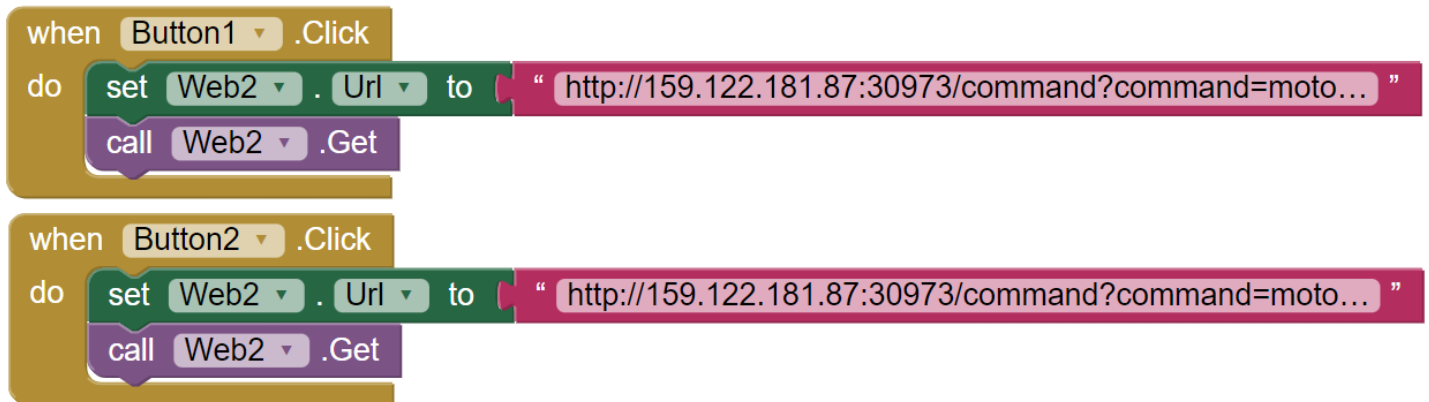


MIT App Inventor:

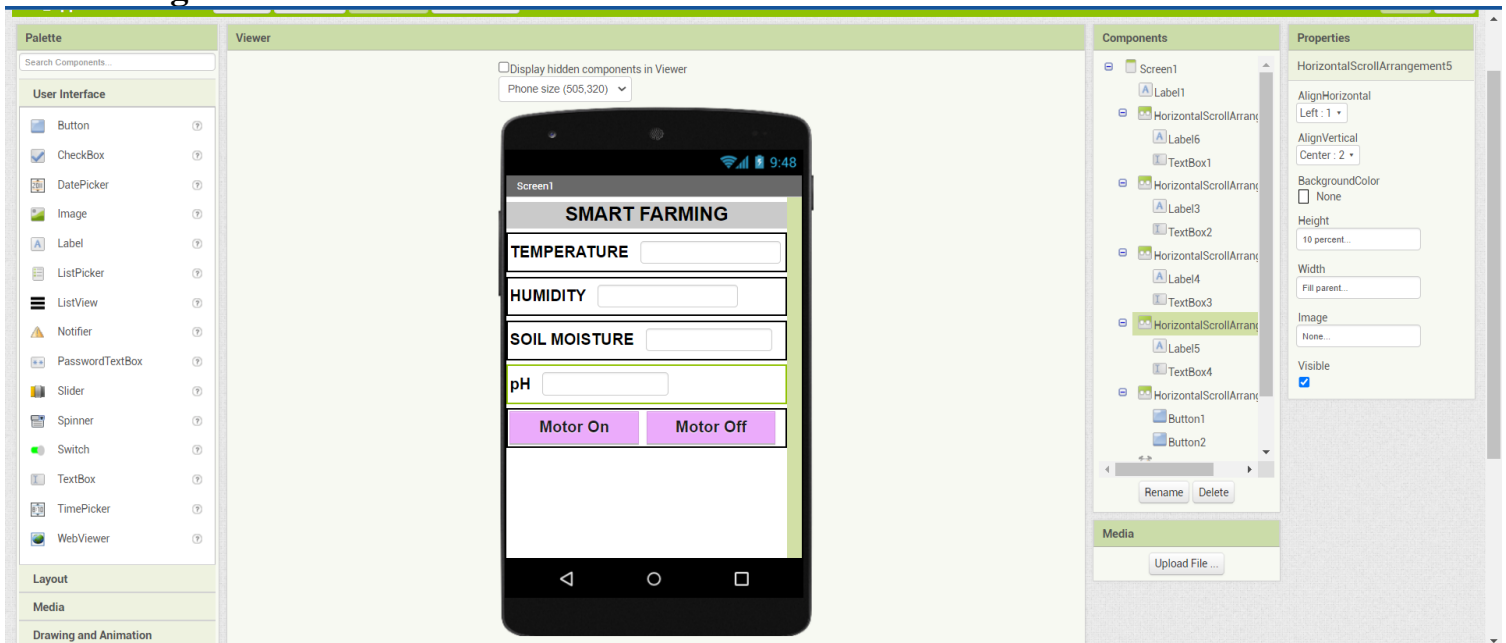
MIT App Inventor is an intuitive, visual programming environment that allows everyone even children to build fully functional apps for smartphones and tablets. Those new to MIT App Inventor can have a simple first app up and running in less than 30 minutes. And what's more, our blocks-based tool facilitates the creation of complex, high-impact apps in significantly less time than traditional programming environments.

Blocks:





Designer:



IBM Watson:

← → ↺

obbnvy.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Transfer history Download Youtube...

IBM Watson IoT Platform

abhinavb@student.tce.edu
ID: obbnvy

⋮

⚙️

👤

📊

🔍

🔧

Browse

Action

Device Types

Interfaces

Add Device +

🔍 Search by Device ID

Device Simulator ☒

🔍

<input type="checkbox"/>	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location								
▼ <input type="checkbox"/>	123456789	Connected	raspberrypi	Device	Oct 29, 2022 12:54 PM	→ ...								
<div><div>Identity</div><div>Device Information</div><div>Recent Events</div><div>State</div><div>Logs</div><div>×</div></div> <div>The recent events listed show the live stream of data that is coming and going from this device.</div> <table><thead><tr><th>Event</th><th>Value</th><th>Format</th><th>Last Received</th></tr></thead><tbody><tr><td>IoTSensor</td><td>{"ph":9,"moisture":28,"temperature":33,"Humid"...</td><td>json</td><td>a few seconds ago</td></tr></tbody></table>							Event	Value	Format	Last Received	IoTSensor	{"ph":9,"moisture":28,"temperature":33,"Humid"...	json	a few seconds ago
Event	Value	Format	Last Received											
IoTSensor	{"ph":9,"moisture":28,"temperature":33,"Humid"...	json	a few seconds ago											

Items per page 50 | 1–1 of 1 item

0 Simulations running