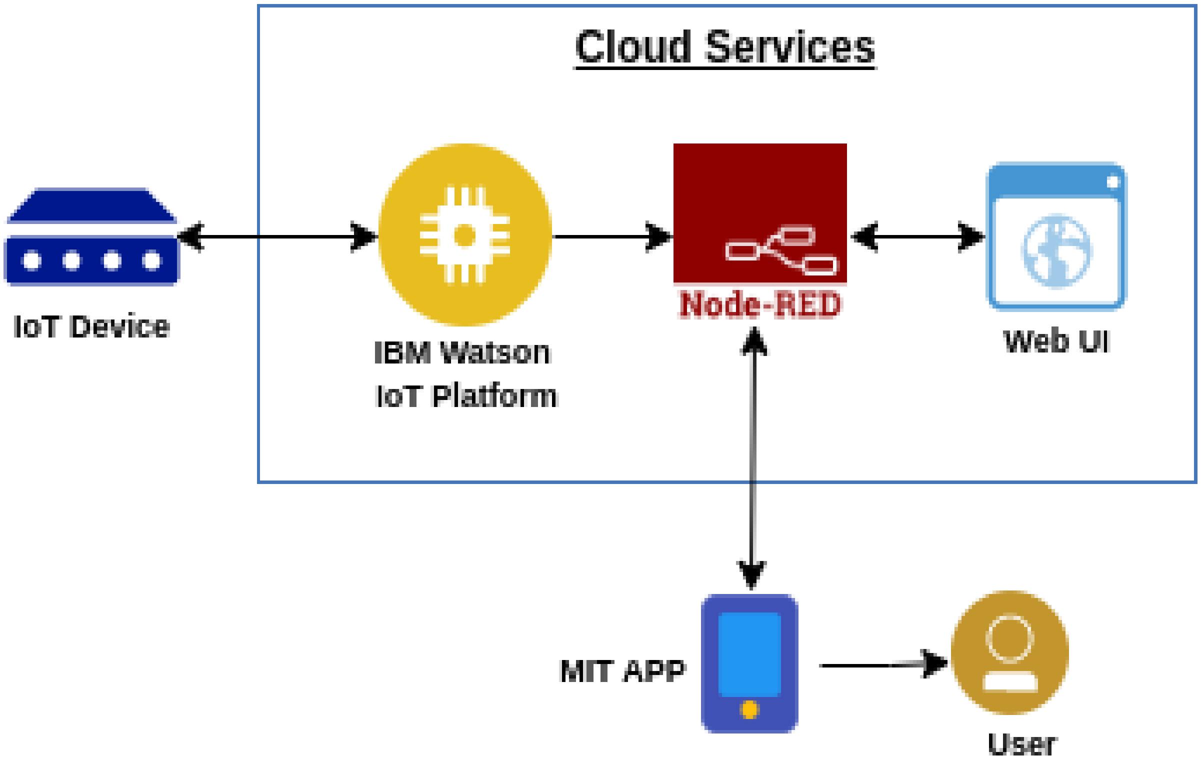
Final Deliverables

Team ID	**PNT2022TMID21342**
Project Name	**SmartFarmer - IoT Enabled Smart Farming Application**
Team Members	**Akash B(917719D008)** **Krishna Prasanna V G (917719D041)** **Lingeshwaran K (917719D044)** **Gunal L (917719D025)**

OBJECTIVE:

- Sometimes elderly people forget to keep track of whether the agriculture field is irrigated or not.
- They are all also unable to visit the agriculture field due to some reasons.
- And it is difficult for farmers to find that whether every corner of the field is well irrigated or not.
- An app is built for the user (farmers) which enables him/her to keeps the track of field condition parameters like soil moisture content, temperature, humidity of the field can be viewed and analyzed.
- If the field is not irrigated recently the field parameters goes low and user(farmer) can view it in his/her mobile phone. Further steps of turning ON the motor can be done from his/her house itself.

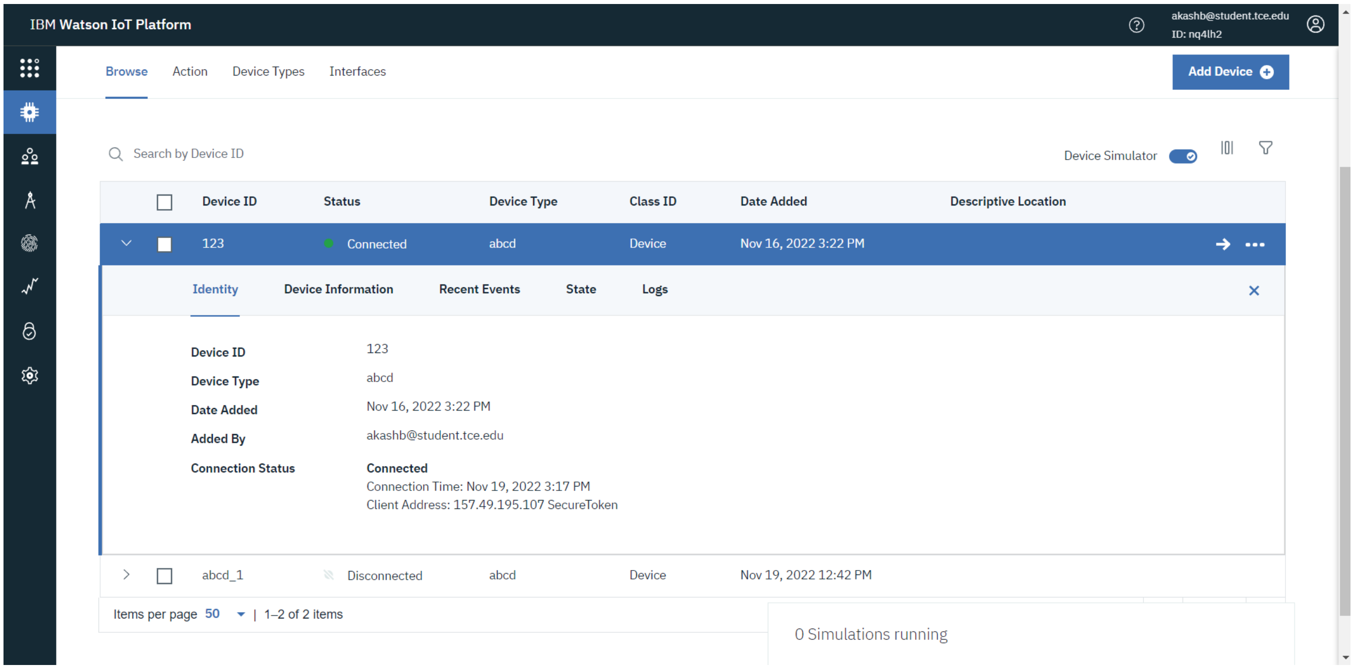
FLOW OF THE PROJECT:



WAYS ACHIEVES THE PROJECT FLOW:

Step1:

Create an IBM Watson Device and note down the credentials, after that create a App “Standard App” and node down the API key and Token.



Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123	Connected	abcd	Device	Nov 16, 2022 3:22 PM	
abcd_1	Disconnected	abcd	Device	Nov 19, 2022 12:42 PM	

Device ID 123

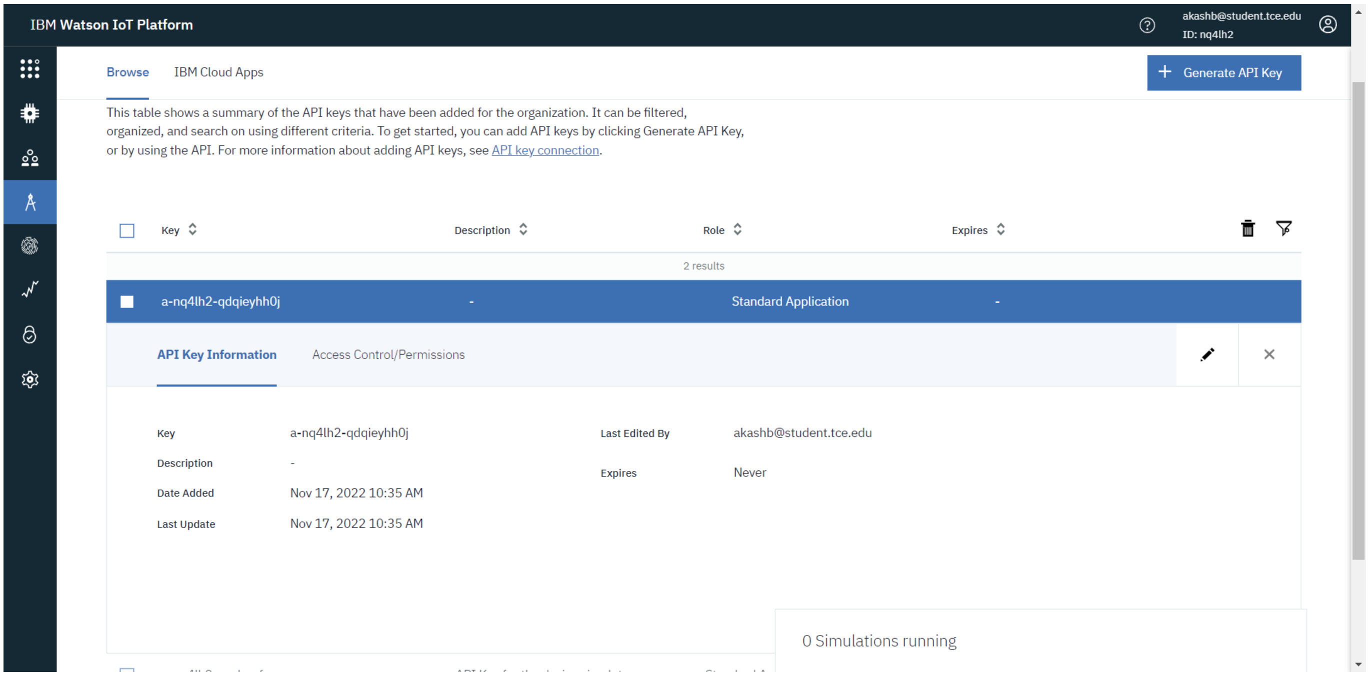
Device Type abcd

Date Added Nov 16, 2022 3:22 PM

Added By akashb@student.tce.edu

Connection Status Connected
Connection Time: Nov 19, 2022 3:17 PM
Client Address: 157.49.195.107 SecureToken

Items per page 50 | 1–2 of 2 items



This table shows a summary of the API keys that have been added for the organization. It can be filtered, organized, and search on using different criteria. To get started, you can add API keys by clicking Generate API Key, or by using the API. For more information about adding API keys, see API key connection.

Key	Description	Role	Expires
a-nq4lh2-qdqieyhh0j	-	Standard Application	-

2 results

Key a-nq4lh2-qdqieyhh0j

Description -

Date Added Nov 17, 2022 10:35 AM

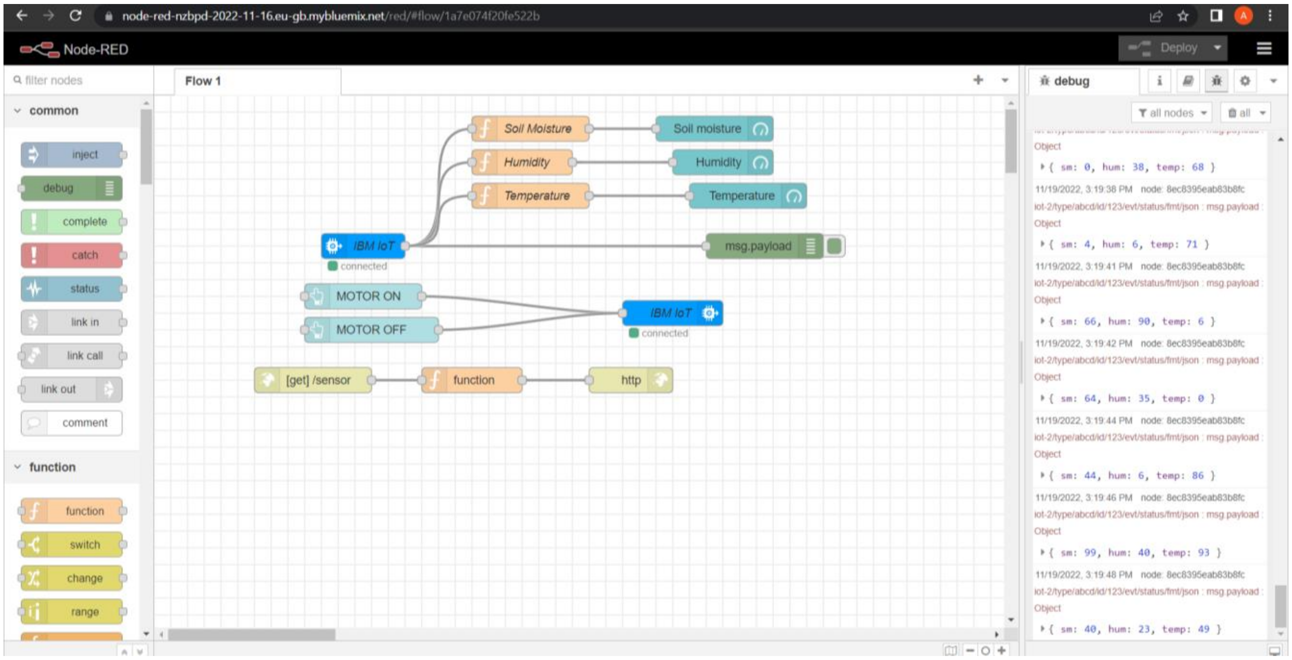
Last Update Nov 17, 2022 10:35 AM

Last Edited By akashb@student.tce.edu

Expires Never

Step 2:

Go to node red flow editor and create nodes for the project.



```
Object
{ sm: 0, hum: 38, temp: 68 }
11/19/2022, 3:19:38 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
{ sm: 4, hum: 6, temp: 71 }
11/19/2022, 3:19:41 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
{ sm: 66, hum: 90, temp: 6 }
11/19/2022, 3:19:42 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
{ sm: 64, hum: 35, temp: 0 }
11/19/2022, 3:19:44 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
{ sm: 44, hum: 6, temp: 86 }
11/19/2022, 3:19:46 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
{ sm: 99, hum: 40, temp: 93 }
11/19/2022, 3:19:48 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
{ sm: 40, hum: 23, temp: 49 }
```

Nodes we user for this project:

1. Function Node
2. Button Node
3. IBM IoT IN
4. IBM IoT Out
5. Inject Node
6. Debug Node

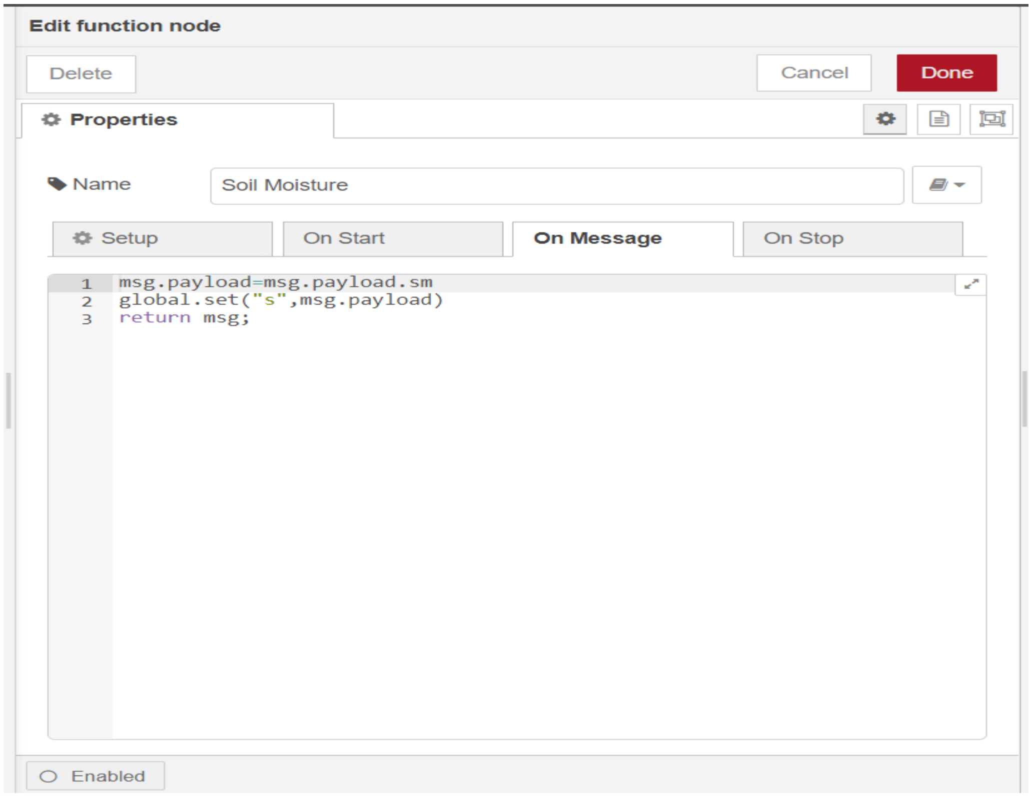
1. Function Node:

We created four different function nodes for four different functions. Drag “Function Node” below the function nodes.

Function:

- Soil Moisture
- Humidity
- Temperature
- function

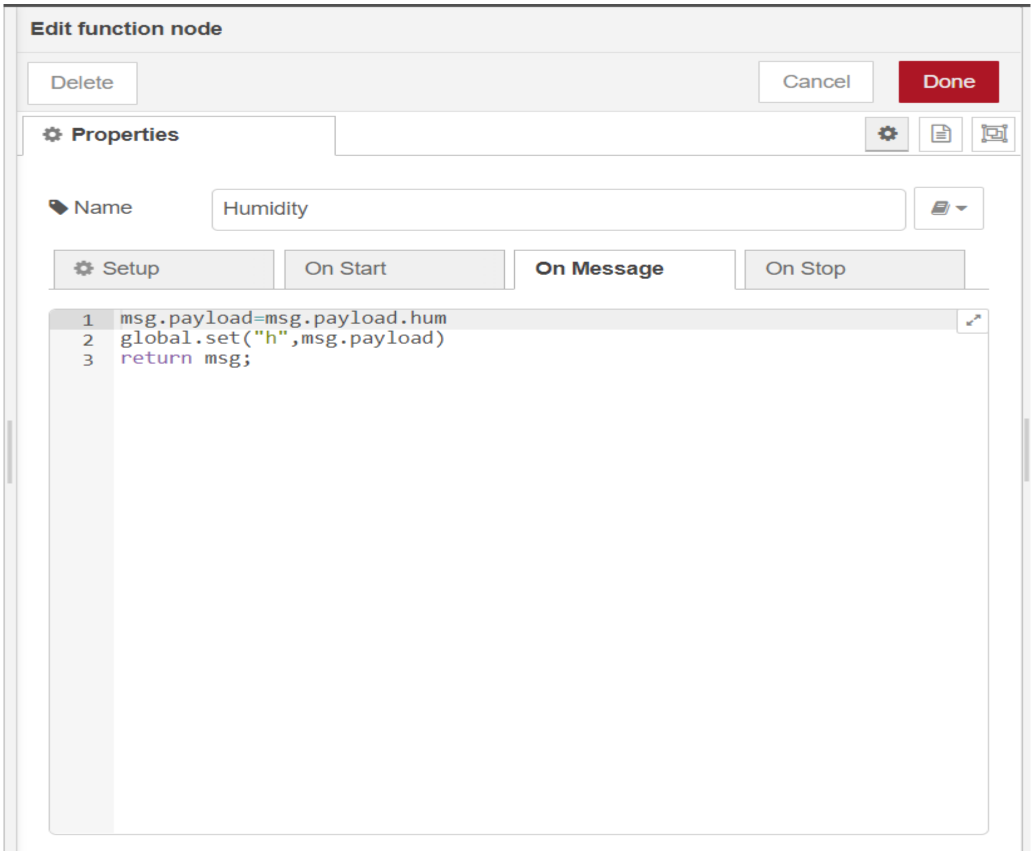
1.1 Soil Moisture:



Name: Soil Moisture

```
msg.payload=msg.payload.sm
global.set("s",msg.payload)
return msg;
```

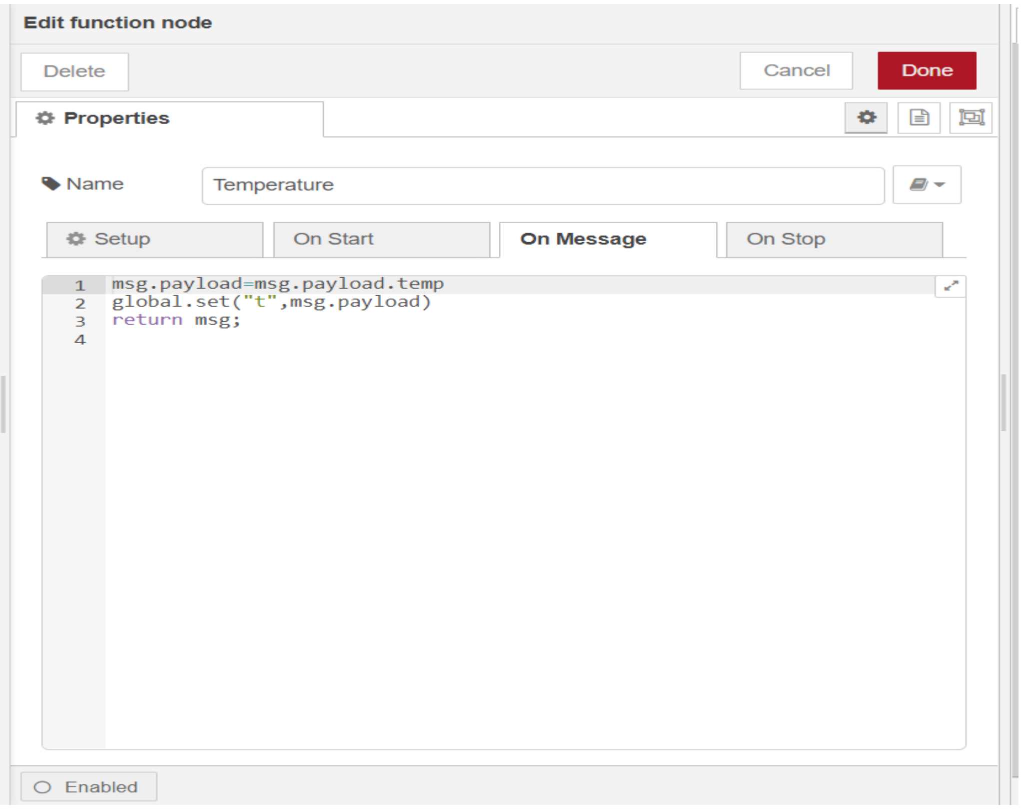
1.2 Humidity:



Name: Humidity

```
msg.payload=msg.payload.hum
global.set("h",msg.payload)
return msg;
```

1.3 Temperature:



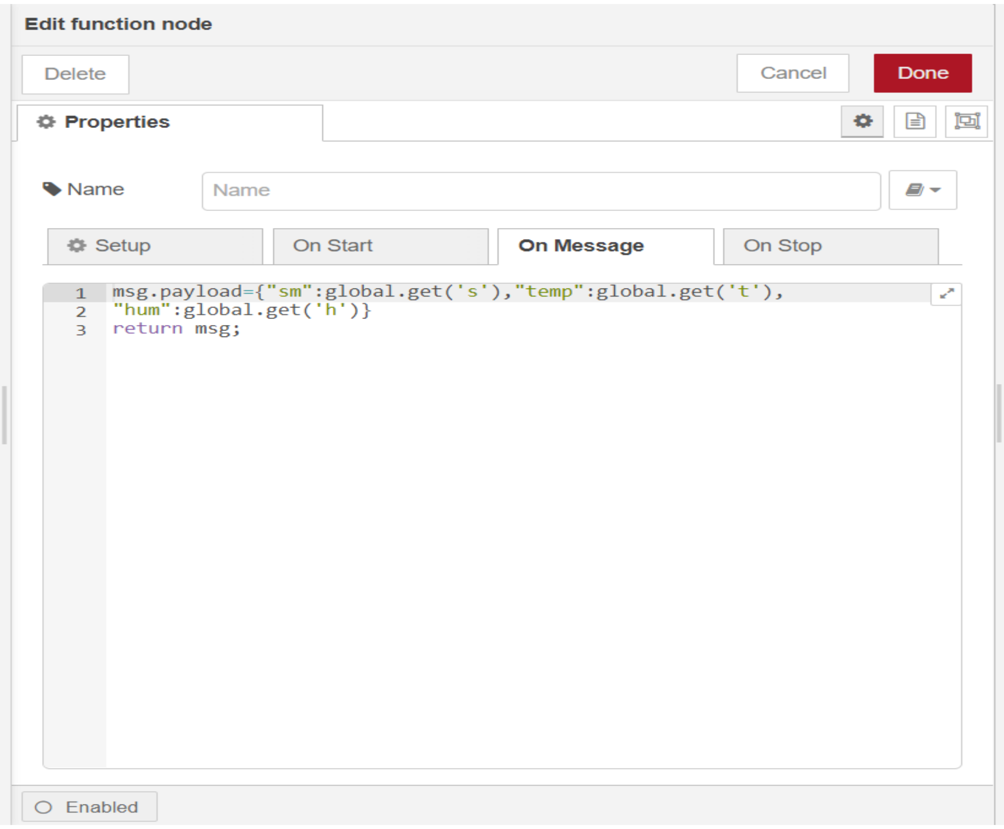
Name: Temperature

```
msg.payload=msg.payload.temp
global.set("t",msg.payload)
return msg;

```

1.4 Function:

To send the field parameters to mobile application.

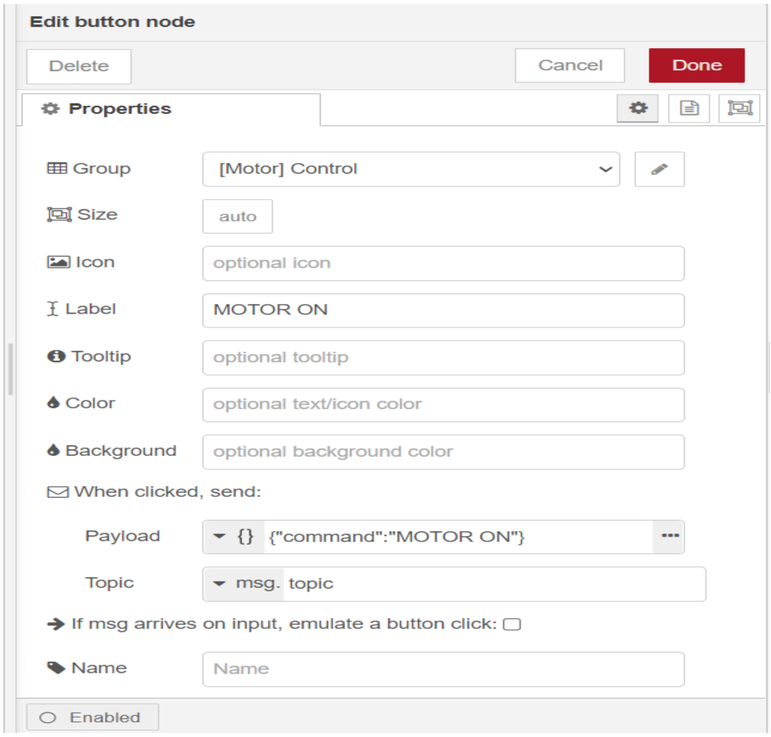


```
msg.payload={"sm":global.get('s'),"temp":global.get('t'),
"hum":global.get('h')}
return msg;
```

2. Button Node:

2.1 MOTOR ON Node:

Drag “Button node” from dashboard nodes and name the node as “MOTOR ON”. Button is use to turn on the supply of the motor for irrigating the field.



Group [Motor] Control

Size auto

Icon optional icon

Label MOTOR ON

Tooltip optional tooltip

Color optional text/icon color

Background optional background color

When clicked, send:

Payload {} {"command":"MOTOR ON"}

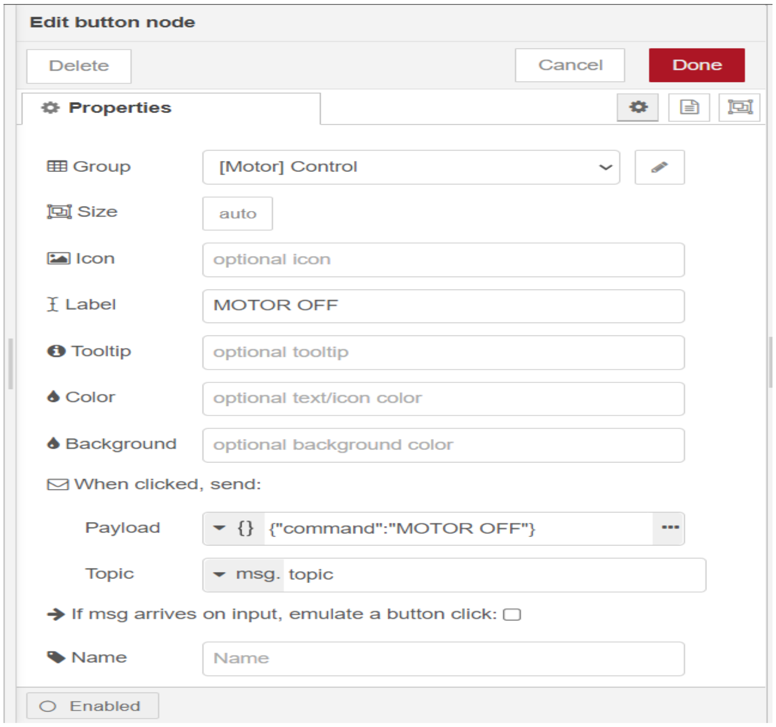
Topic msg. topic

If msg arrives on input, emulate a button click: □

Name Name

2.2 MOTOR OFF Node:

Drag “Button node” from dashboard nodes and name the node as “MOTOR OFF”. Button is use to turn off the supply of the motor after irrigating the field.



Group [Motor] Control

Size auto

Icon optional icon

Label MOTOR OFF

Tooltip optional tooltip

Color optional text/icon color

Background optional background color

When clicked, send:

Payload {} {"command":"MOTOR OFF"}

Topic msg. topic

If msg arrives on input, emulate a button click: □

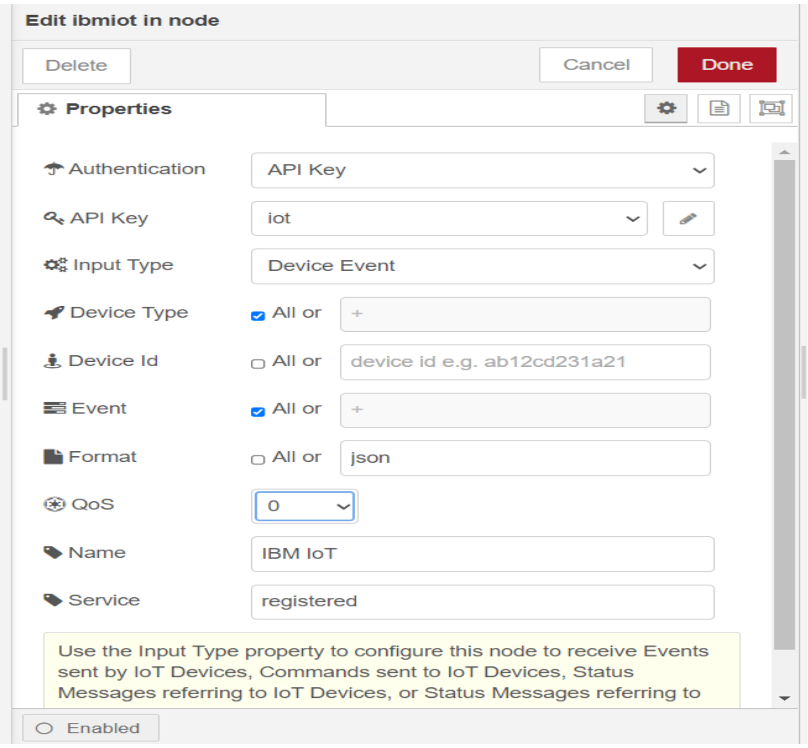
Name Name

3. IBM IoT In Node:

Drag “IBM IoT In Node” from the Input Nodes, which is used to get thesoil moisture, humidity and temperature readings from IBM IoT platform.

Enter the following details,

- IBM IoT App API Key
- IBM IoT App Token
- IBM IoT Device Type as ALL
- IBM IoT Event as ALL
- Format as json
- QoS as 0
- Data as data



Edit ibmiot in node

Authentication	API Key
API Key	iot
Input Type	Device Event
Device Type	☑ All or
Device Id	□ All or device id e.g. ab12cd231a21
Event	☑ All or
Format	□ All or json
QoS	0
Name	IBM IoT
Service	registered

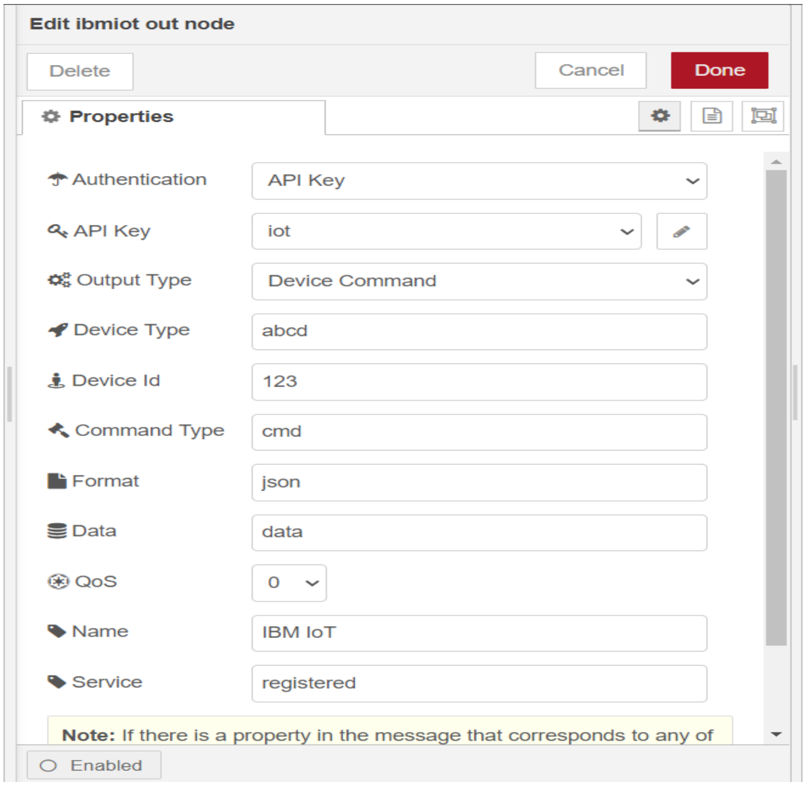
Use the Input Type property to configure this node to receive Events sent by IoT Devices, Commands sent to IoT Devices, Status Messages referring to IoT Devices, or Status Messages referring to

4. IBM IoT Out Node:

Drag “IBM IoT Out Node” from the Output Nodes, which is used to send the soil moisture, humidity and temperature readings of the field to the device.

Enter the following details,

- IBM IoT App API Key
- IBM IoT App Token
- IBM IoT Device Type
- IBM IoT Device ID
- Output Type as Command
- Command Type as cmd
- Format as json
- Data as data
-



Edit ibmiot out node

Authentication: API Key
API Key: iot
Output Type: Device Command
Device Type: abcd
Device Id: 123
Command Type: cmd
Format: json
Data: data
QoS: 0
Name: IBM IoT
Service: registered

Note: If there is a property in the message that corresponds to any of

5. Debug node:

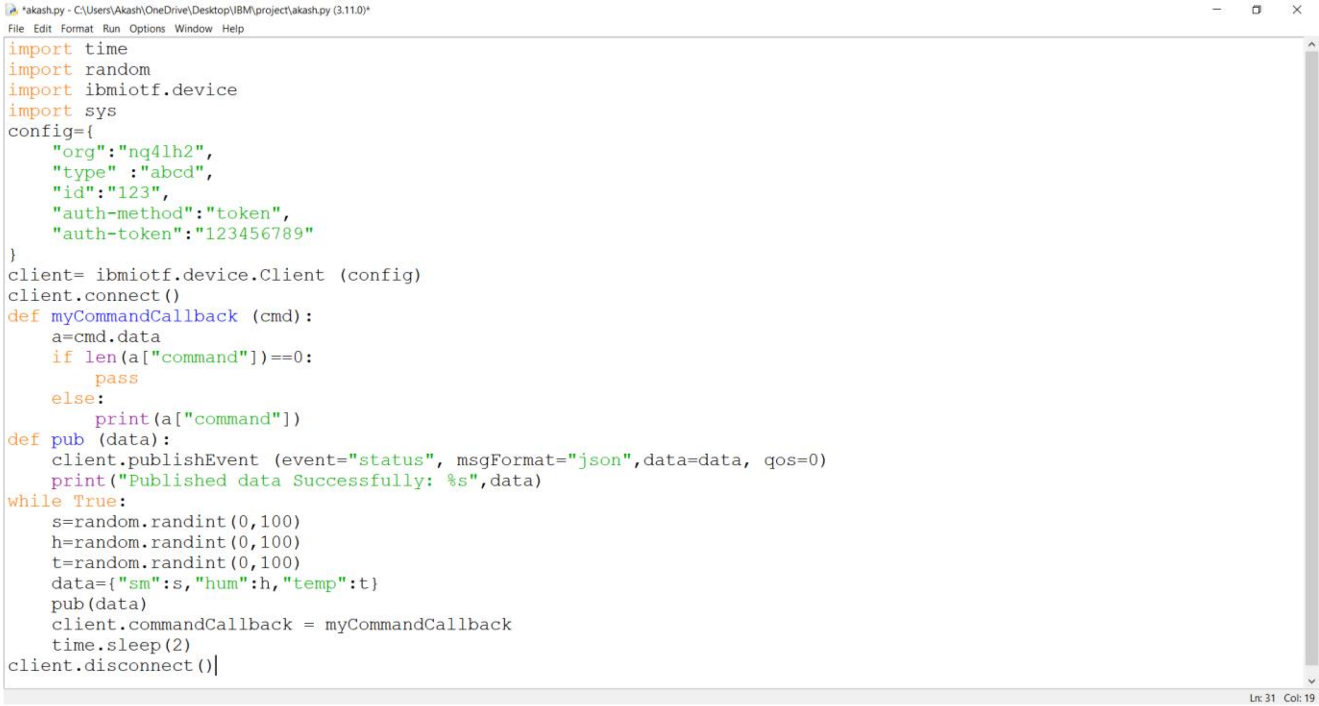
Drag “Debug Node” rom Common Nodes. Which is used to view the payloads.

Step 3:

Write a python script to connect with IBM IoT device and receiving the readings of the agriculture field parameters.

1. Library Used in code:
    a. time
    b. random
    c. ibmiotf.device
    d. sys

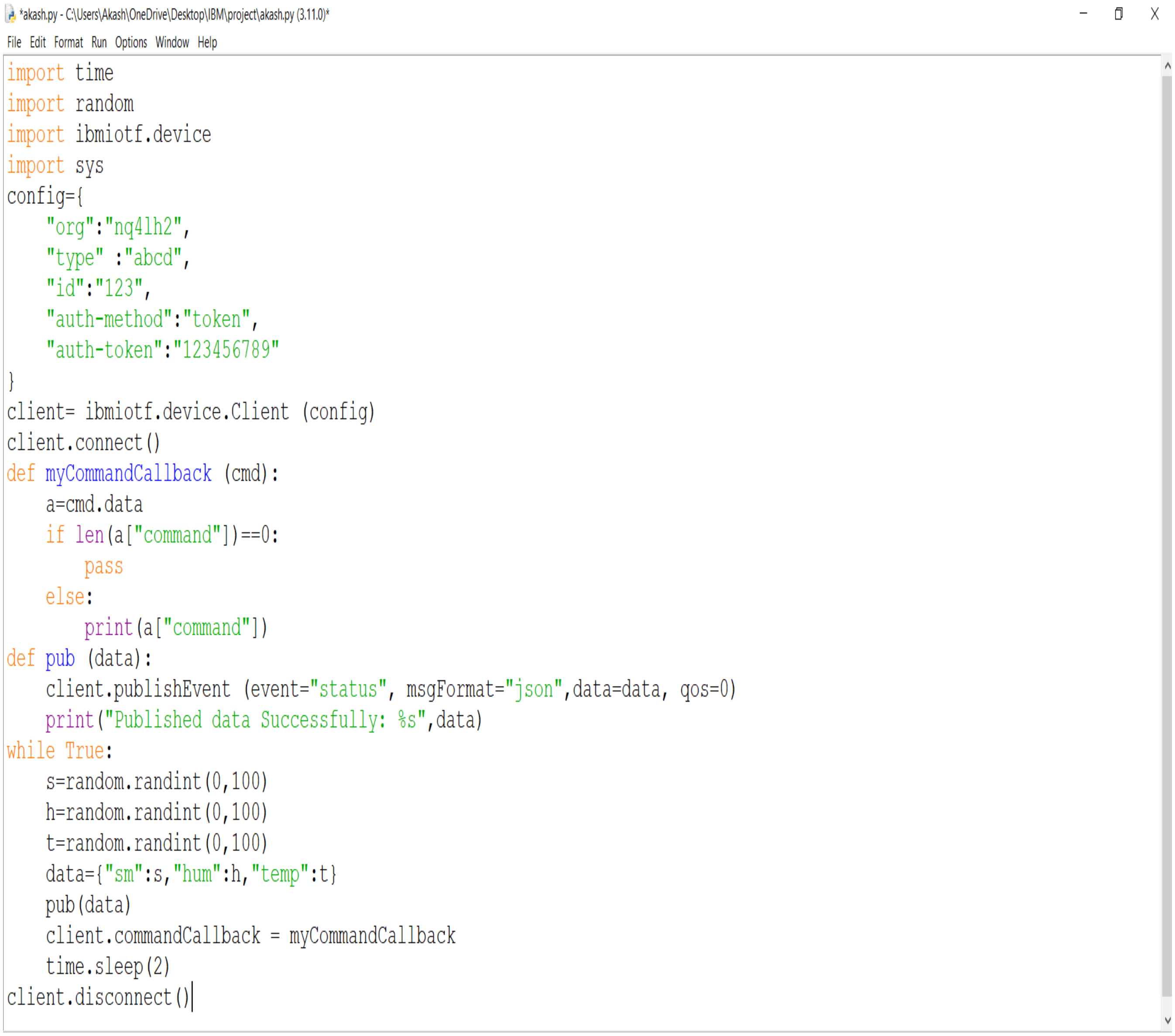
2. Code for Connect with IBM Watson IoT device


akash.py - C:\Users\Akash\OneDrive\Desktop\IBM\project\akash.py (3.11.0)
File Edit Format Run Options Window Help
Ln: 31 Col: 19

```python
import time
import random
import ibmiotf.device
import sys
config={
    "org":"nq4lh2",
    "type" :"abcd",
    "id":"123",
    "auth-method":"token",
    "auth-token":"123456789"
}
client= ibmiotf.device.Client (config)
client.connect()
def myCommandCallback (cmd):
    a=cmd.data
    if len(a["command"])==0:
        pass
    else:
        print(a["command"])
def pub (data):
    client.publishEvent (event="status", msgFormat="json",data=data, qos=0)
    print("Published data Successfully: %s",data)
while True:
    s=random.randint(0,100)
    h=random.randint(0,100)
    t=random.randint(0,100)
    data={"sm":s,"hum":h,"temp":t}
    pub(data)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
```

Step 4:

Complete code for Project:

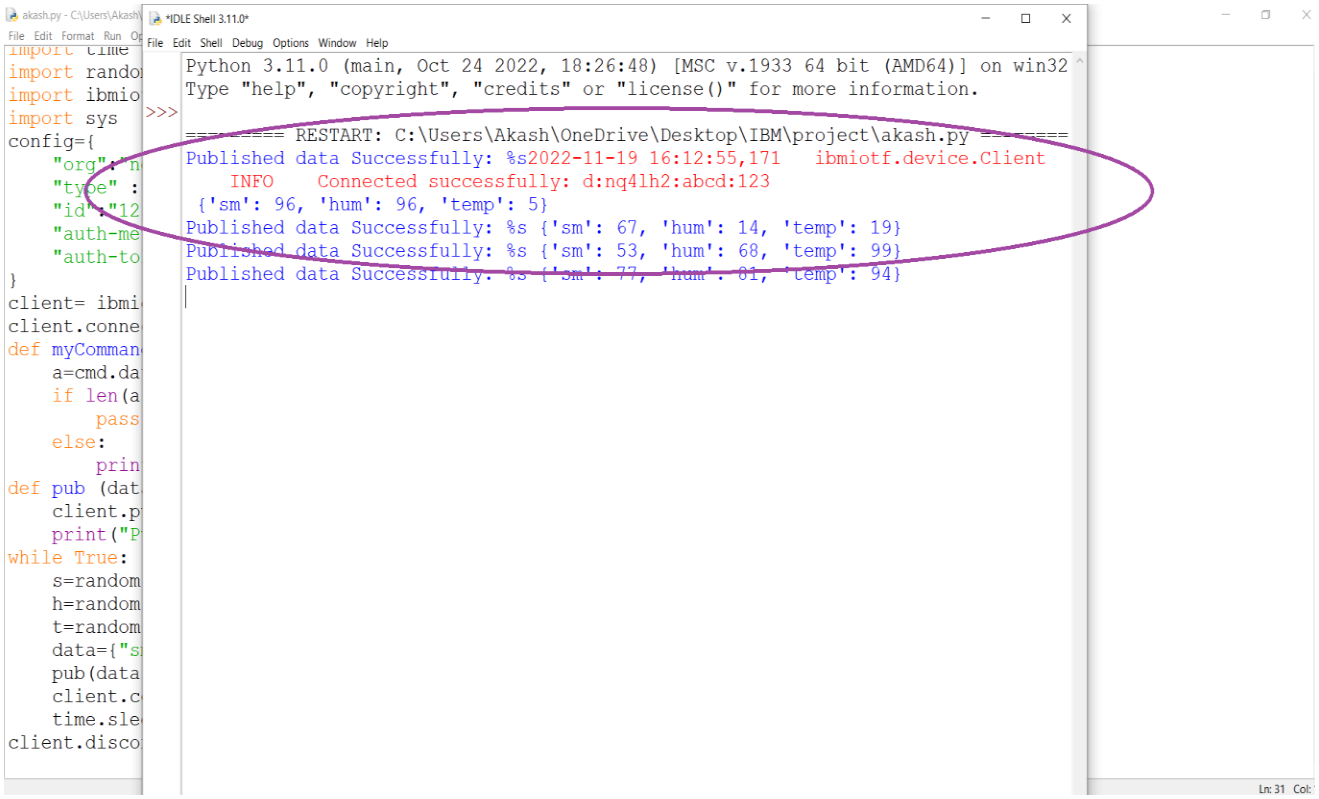


akash.py - C:\Users\Akash\OneDrive\Desktop\IBM\project\akash.py (3.11.0)
File Edit Format Run Options Window Help

```
import time
import random
import ibmiotf.device
import sys
config={
    "org":"nq4lh2",
    "type" :"abcd",
    "id":"123",
    "auth-method":"token",
    "auth-token":"123456789"
}
client= ibmiotf.device.Client (config)
client.connect()
def myCommandCallback (cmd):
    a=cmd.data
    if len(a["command"])==0:
        pass
    else:
        print(a["command"])
def pub (data):
    client.publishEvent (event="status", msgFormat="json",data=data, qos=0)
    print("Published data Successfuly: %s",data)
while True:
    s=random.randint(0,100)
    h=random.randint(0,100)
    t=random.randint(0,100)
    data={"sm":s,"hum":h,"temp":t}
    pub(data)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
```

RESULT:

1. Connect with the device.

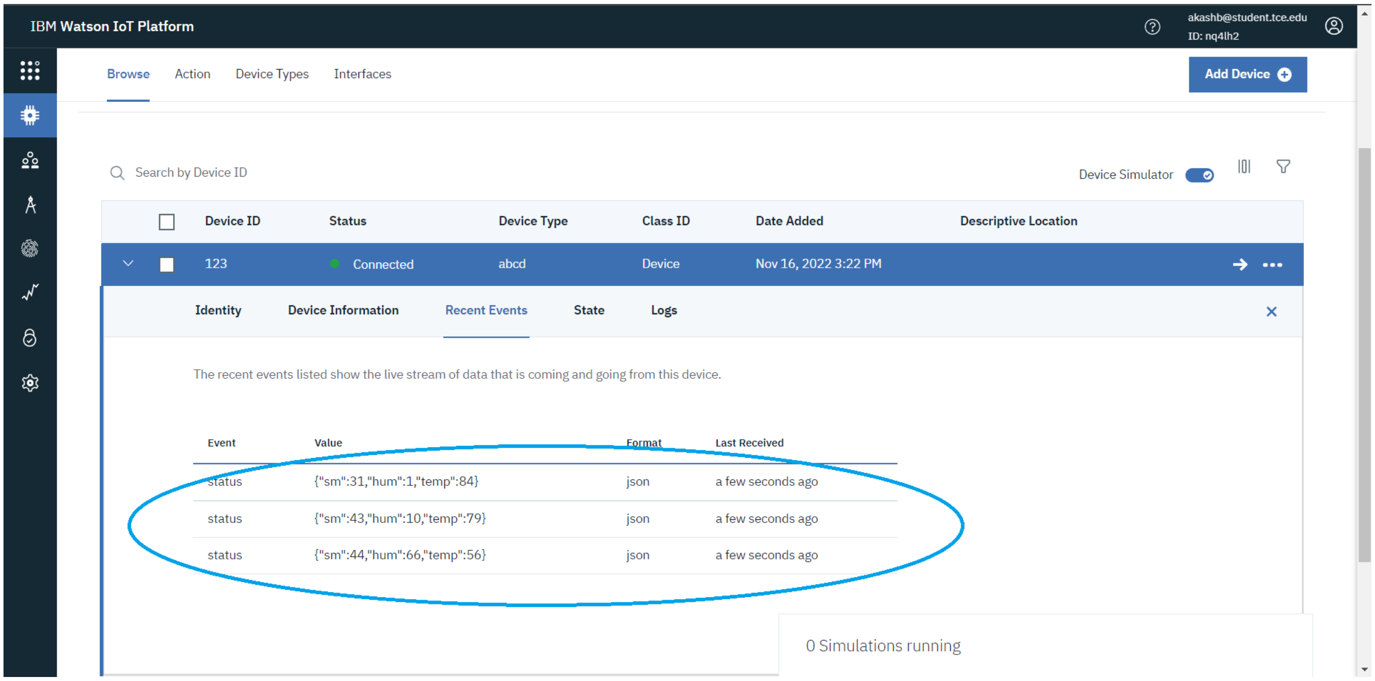


akash.py - C:\Users\Akash\
File Edit Format Run Op
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
Ln: 31 Col:

```
import time
import rando
import ibmio
import sys
config={
    "org":"n
    "type" :
    "id":"12
    "auth-me
    "auth-to
}
client= ibmi
client.conne
def myComman
    a=cmd.da
    if len(a
        pass
    else:
        prin
def pub (dat
    client.p
    print("P
while True:
    s=random
    h=random
    t=random
    data={"s
    pub(data
    client.c
    time.sle
client.disco
```

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: C:\Users\Akash\OneDrive\Desktop\IBM\project\akash.py ========
Published data Successfully: %s2022-11-19 16:12:55,171   ibmiotf.device.Client
    INFO    Connected successfully: d:nq4lh2:abcd:123
 {'sm': 96, 'hum': 96, 'temp': 5}
Published data Successfully: %s {'sm': 67, 'hum': 14, 'temp': 19}
Published data Successfully: %s {'sm': 53, 'hum': 68, 'temp': 99}
Published data Successfully: %s {'sm': 77, 'hum': 81, 'temp': 94}
```

2. Python code connection with IBM IOT platform and displaying the readings.

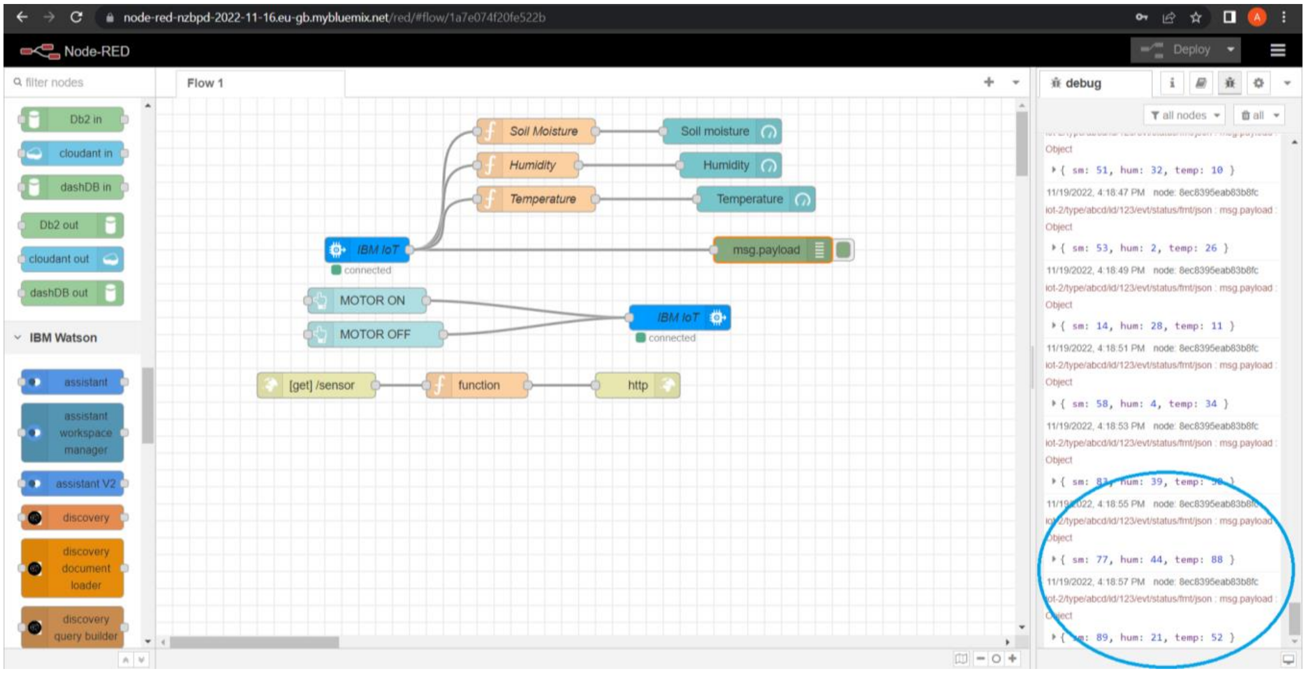


Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
123	● Connected	abcd	Device	Nov 16, 2022 3:22 PM	

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

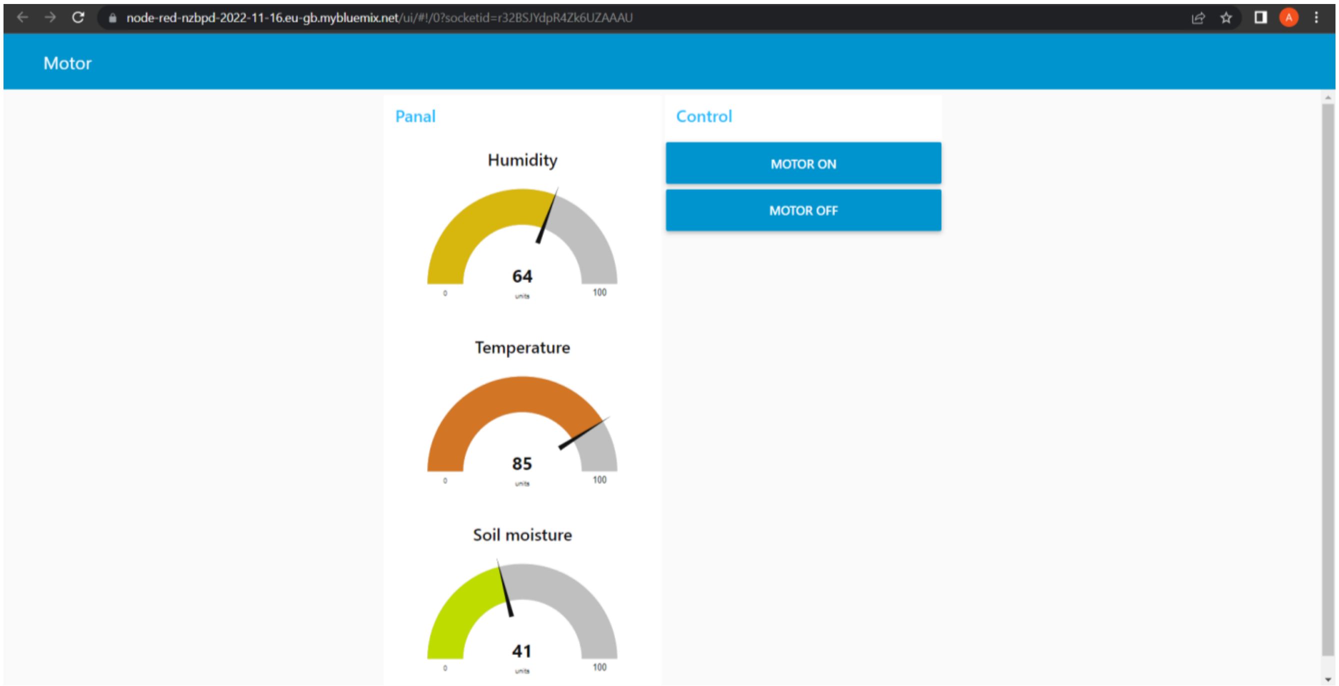
Event	Value	Format	Last Received
status	{"sm":31,"hum":1,"temp":84}	json	a few seconds ago
status	{"sm":43,"hum":10,"temp":79}	json	a few seconds ago
status	{"sm":44,"hum":66,"temp":56}	json	a few seconds ago

3. IBM IoT connection with Node-Red platform

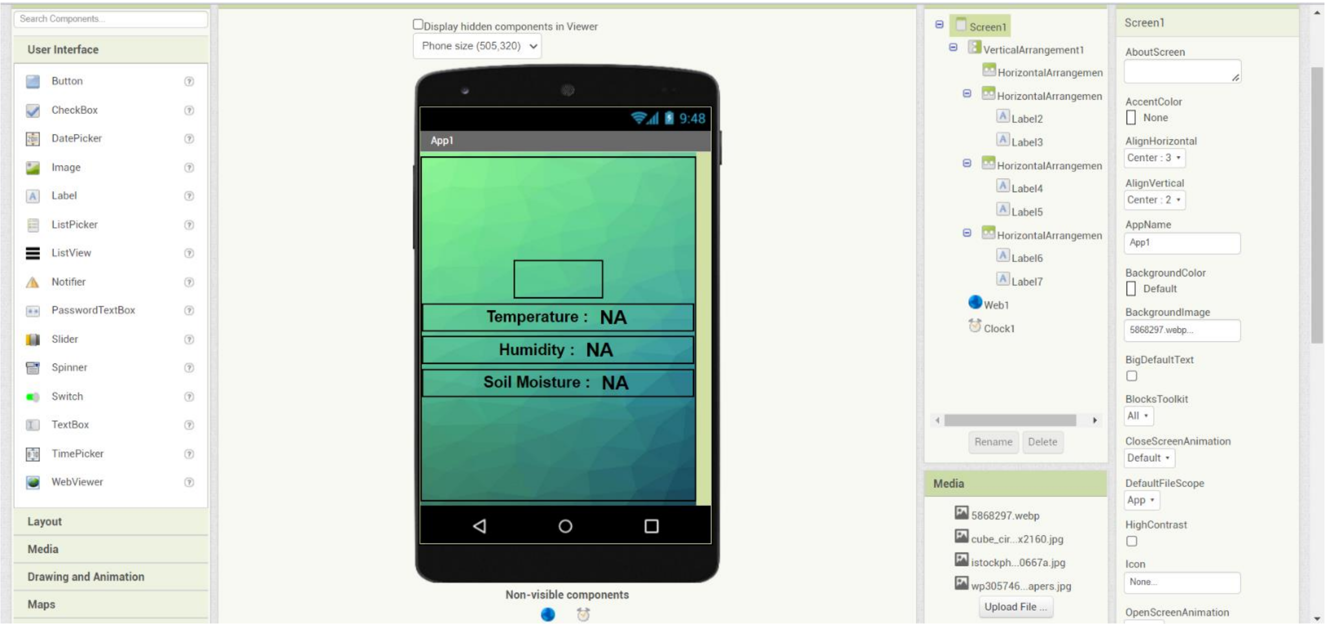


```
Object
▸{ sm: 51, hum: 32, temp: 10 }
11/19/2022, 4:18:47 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
▸{ sm: 53, hum: 2, temp: 26 }
11/19/2022, 4:18:49 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
▸{ sm: 14, hum: 28, temp: 11 }
11/19/2022, 4:18:51 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
▸{ sm: 58, hum: 4, temp: 34 }
11/19/2022, 4:18:53 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
▸{ sm: 83, hum: 39, temp: 30 }
11/19/2022, 4:18:55 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
▸{ sm: 77, hum: 44, temp: 88 }
11/19/2022, 4:18:57 PM  node: 8ec8395eab83b8fc
iot-2/type/abcd/id/123/evt/status/fmt/json : msg.payload :
Object
▸{ sm: 89, hum: 21, temp: 52 }
```

4. Readings visualization in WEB UI



5. After downloading the developed mobile application by scanning the QR code in the user’s application, the user(farmer) can see himself/herself in mobile phone and can be operated in from remote area without wasting time in going to field.



AboutScreen

AccentColor: None

AlignHorizontal: Center : 3

AlignVertical: Center : 2

AppName: App1

BackgroundColor: Default

BackgroundImage: 5868297.webp...

BigDefaultText: □

BlocksToolkit: All

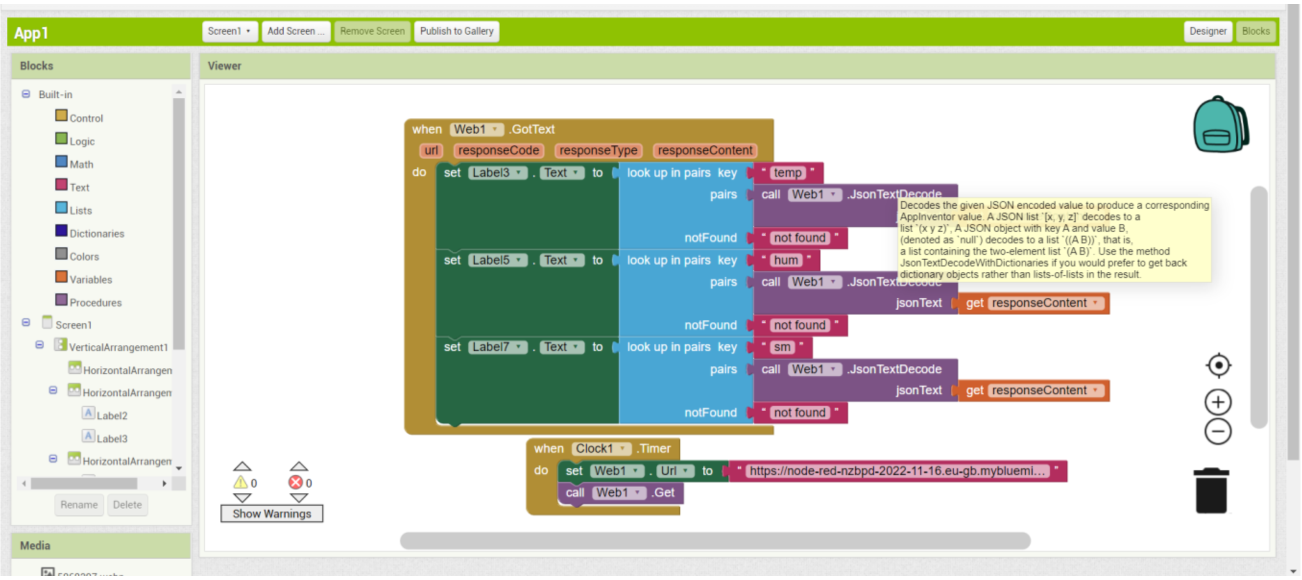
CloseScreenAnimation: Default

DefaultFileScope: App

HighContrast: □

Icon: None...

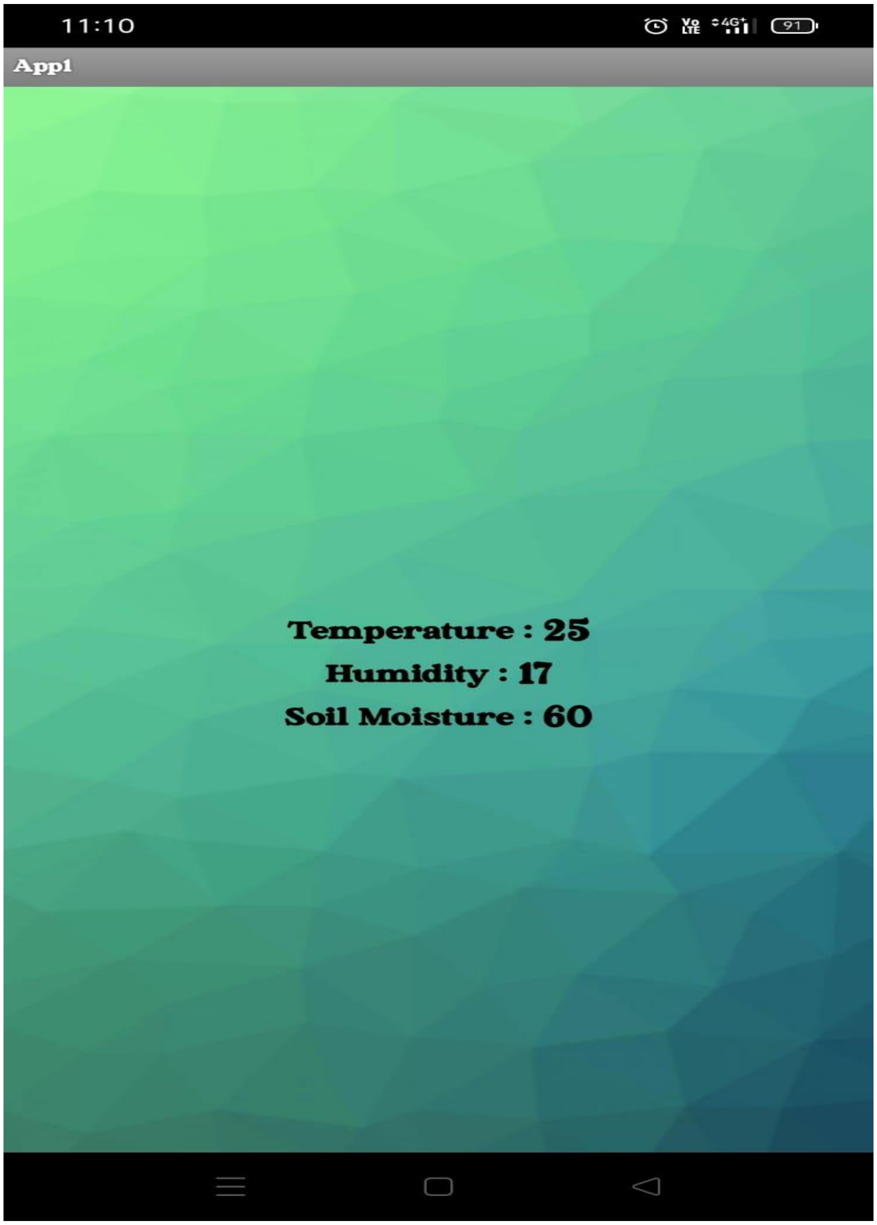
OpenScreenAnimation



```
when Web1 .GotText
  url  responseCode  responseType  responseContent
do  set Label3 . Text to  look up in pairs  key  " temp "
                                            pairs  call Web1 .JsonTextDecode
                                         notFound  " not found "
    set Label5 . Text to  look up in pairs  key  " hum "
                                            pairs  call Web1 .JsonTextDecode
                                                                   jsonText  get responseContent
                                         notFound  " not found "
    set Label7 . Text to  look up in pairs  key  " sm "
                                            pairs  call Web1 .JsonTextDecode
                                                                   jsonText  get responseContent
                                         notFound  " not found "

when Clock1 .Timer
do  set Web1 . Url to  " https://node-red-nzbpd-2022-11-16.eu-gb.mybluemi... "
    call Web1 .Get
```

Decodes the given JSON encoded value to produce a corresponding AppInventor value. A JSON list `[x, y, z]` decodes to a list `(x y z)`, A JSON object with key A and value B, (denoted as `null`) decodes to a list `((A B))`, that is, a list containing the two-element list `(A B)`. Use the method JsonTextDecodeWithDictionaries if you would prefer to get back dictionary objects rather than lists-of-lists in the result.



Temperature : 25

Humidity : 17

Soil Moisture : 60

CONCLUSION:

The objectives are achieved and the data flow is constructed as per the project flow mentioned in the Smartintenz Guided project.