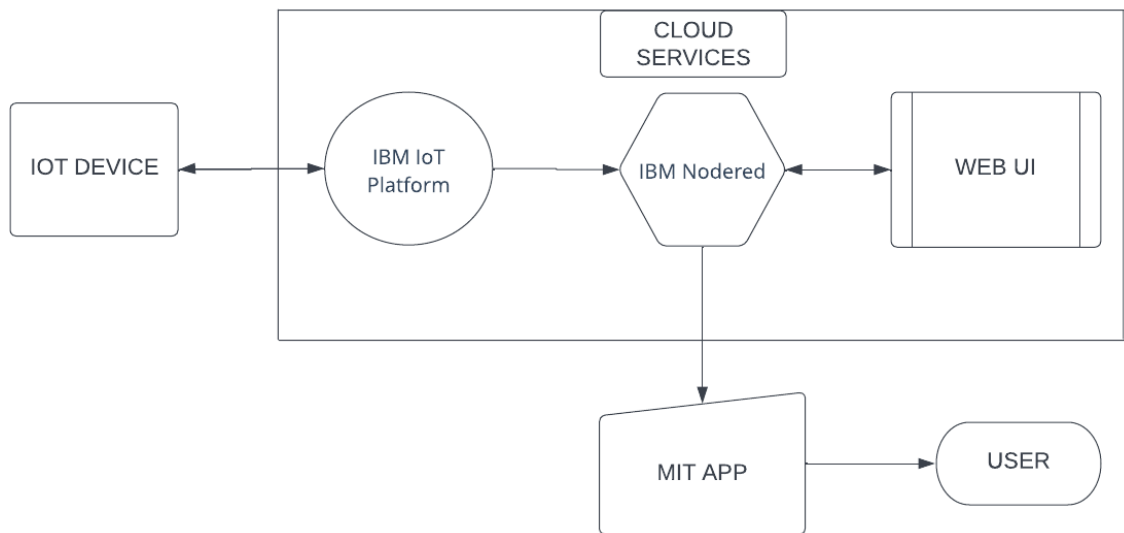


SPRINT 4

Date	16 th November - 2022
Team ID	PNT2022TMID42737
Project Name	Project – Smart Farmer-IoT Enabled smart Farming Application

Over All Flow chart:



Python code :

```
import wiotp.sdk.device
import time
import os
import datetime
import random
myConfig = {
    "identity":{
        "orgId": "ik6mgw",
```

```

        "typeId": "smartfarmer",
        "deviceId": "147852369"
    },
    "auth": {
        "token": "9790375943"
    }
}

client = wiotp.sdk.device.DeviceClient (config=myConfig, logHandlers=None)
client.connect()

```

```

def myCommandCallback(cmd) :
    print("Message received from IBM IoT Platform: %s" %
cmd.data['command'])
    m=cmd.data['command']
    if(m== "motoron"):
        print("Motor is switched on")
    elif (m== "motoroff"):
        print ("Motor is switched OFF")
    print(" ")

```

```

while True:
    soil=random.randint(0,100)
    temp=random.randint(0,100)
    hum=random.randint(0,100)
    myData={'soil_moisture': soil,
'temperature':temp,
'humidity':hum}

```

```

client.publishEvent(eventId="status",
msgFormat="json",
data=myData,
qos=0,
onPublish=None)

print("Published data Successfully: %s", myData)

time.sleep(10)

client.commandCallback = myCommandCallback

client.disconnect()

```

Python Code Simulation:

The screenshot shows a Jupyter Notebook environment with two panes. The left pane displays the output of the code, showing multiple 'Published data Successfully' messages with random sensor data. The right pane shows the Python code that generates this output.

Output (Left Pane):

```

Published data Successfully: %s ('soil_moisture': 77, 'temperature': 43, 'humidity': 22)
Published data Successfully: %s ('soil_moisture': 60, 'temperature': 36, 'humidity': 45)
Published data Successfully: %s ('soil_moisture': 96, 'temperature': 12, 'humidity': 42)
Published data Successfully: %s ('soil_moisture': 53, 'temperature': 82, 'humidity': 96)
Published data Successfully: %s ('soil_moisture': 23, 'temperature': 2, 'humidity': 90)
Published data Successfully: %s ('soil_moisture': 7, 'temperature': 38, 'humidity': 39)
Published data Successfully: %s ('soil_moisture': 100, 'temperature': 85, 'humidity': 86)
Published data Successfully: %s ('soil_moisture': 26, 'temperature': 19, 'humidity': 68)
Published data Successfully: %s ('soil_moisture': 2, 'temperature': 26, 'humidity': 49)
Published data Successfully: %s ('soil_moisture': 57, 'temperature': 15, 'humidity': 56)
Published data Successfully: %s ('soil_moisture': 8, 'temperature': 51, 'humidity': 65)
Published data Successfully: %s ('soil_moisture': 68, 'temperature': 91, 'humidity': 90)
Published data Successfully: %s ('soil_moisture': 65, 'temperature': 21, 'humidity': 50)
Published data Successfully: %s ('soil_moisture': 41, 'temperature': 60, 'humidity': 95)
Published data Successfully: %s ('soil_moisture': 67, 'temperature': 20, 'humidity': 4)
Published data Successfully: %s ('soil_moisture': 19, 'temperature': 5, 'humidity': 78)
Published data Successfully: %s ('soil_moisture': 64, 'temperature': 80, 'humidity': 53)
Published data Successfully: %s ('soil_moisture': 13, 'temperature': 90, 'humidity': 52)
Published data Successfully: %s ('soil_moisture': 61, 'temperature': 20, 'humidity': 88)

```

Code (Right Pane):

```

import wiotsdk.device
import time
import os
import datetime
import random

myConfig = {
    "identity": {
        "orgId": "ikdmgw",
        "typeId": "smartfarmer",
        "deviceId": "147852369"
    },
    "auth": {
        "token": "9790375943"
    }
}

client = wiotsdk.device.DeviceClient (config=myConfig, logHandlers=None)
client.connect()

def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if (m=="motoron"):
        print("Motor is switched on")
    elif (m=="motoroff"):
        print("Motor is switched OFF")
    print(" ")

while True:
    soil=random.randint(0,100)
    temp=random.randint(0,100)
    hum=random.randint(0,100)
    myData={'soil_moisture': soil, 'temperature':temp, 'humidity':hum}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
    time.sleep(10)
    client.commandCallback = myCommandCallback
    client.disconnect()

```

IBM IoT Watson Platform:

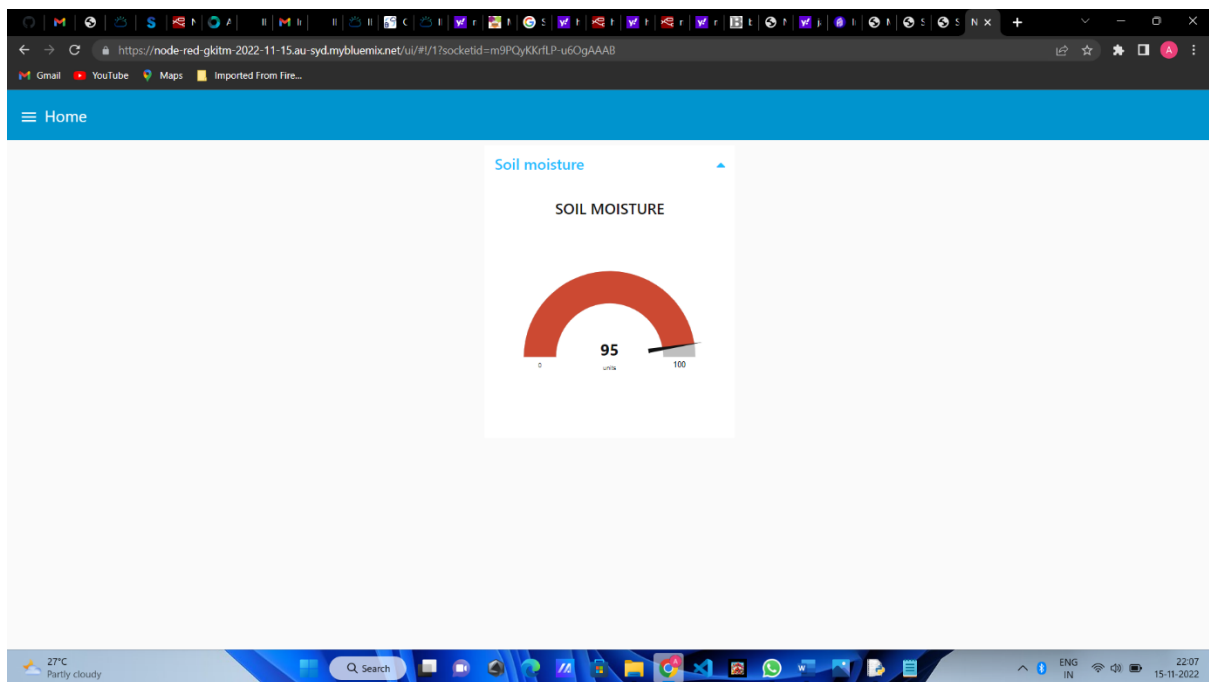
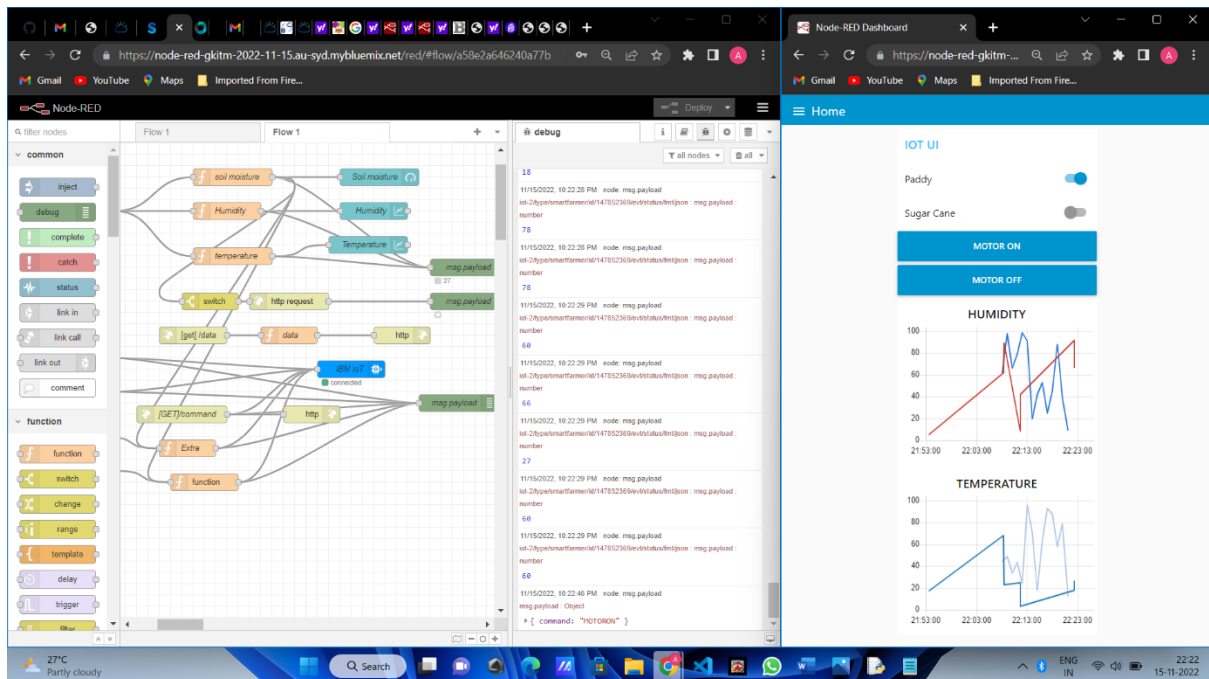
The screenshot displays the IBM Watson IoT Platform dashboard. The main interface shows a list of devices with columns for Device ID, Device Type, Date Added, Added By, and Connection Status. A device with ID 147852369 and type 'smartfarmer' is highlighted. A 'Simulations' window is open, showing a simulation of an event for the selected device. The simulation details include the event name 'event_1', the device ID '147852369', and the event payload: `{"soil_moisture":25,"temperature":33,"humidity":39}`. The simulation status indicates '1 event sent' and '57 bytes sent'.

Device ID	Device Type	Date Added	Added By	Connection Status
12345	Disconnected	rasberry	Device	Oct 2
147852369	Connected	smartfarmer	Device	Nov 5, 2022 10:45 AM
928451	Disconnected	assign4	Device	Oct 2

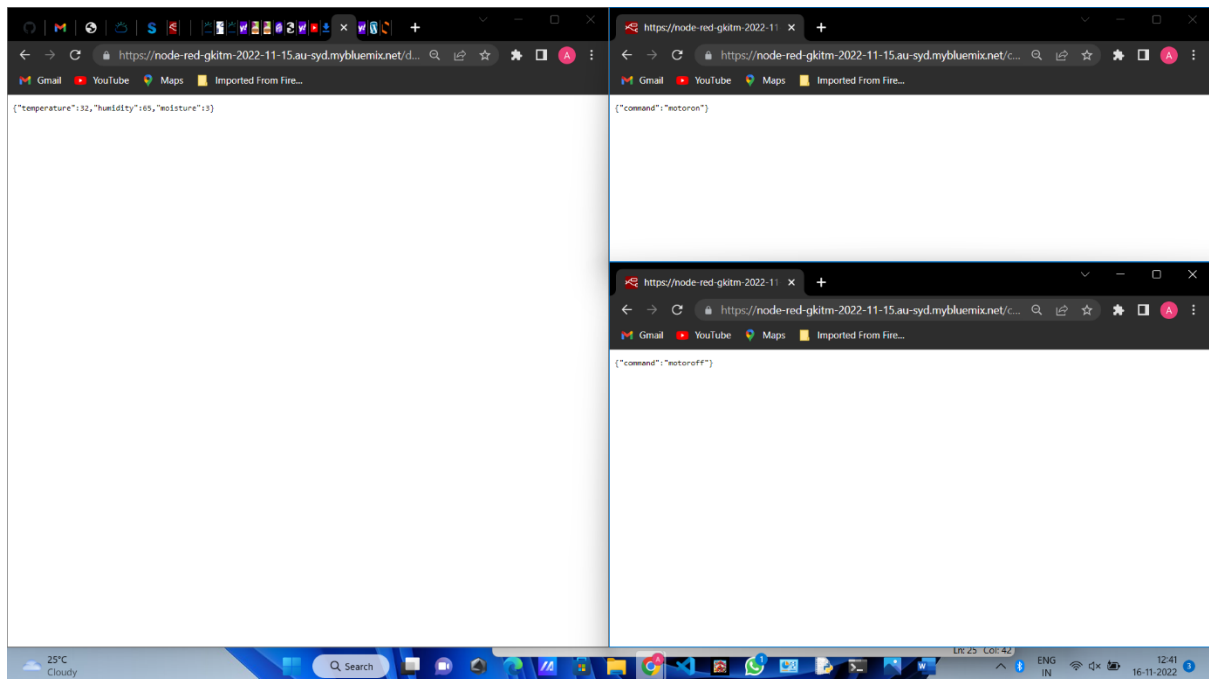
Node Red to Collect Data From IBM Cloud:

The screenshot shows the Node-RED interface with a flow titled 'Flow 1'. The flow starts with an 'IBM IoT' node, which connects to several function nodes: 'soil moisture', 'Humidity', and 'temperature'. These function nodes then connect to corresponding output nodes: 'Soil moisture', 'Humidity', and 'Temperature'. The flow also includes a 'switch' node, an 'http request' node, and a 'msg.payload' node. The flow is configured to collect data from the IBM IoT platform and send it to the output nodes. The interface also shows a list of flows on the right side, including 'Flow 1' and 'Subflows'.

Node Red Web UI:

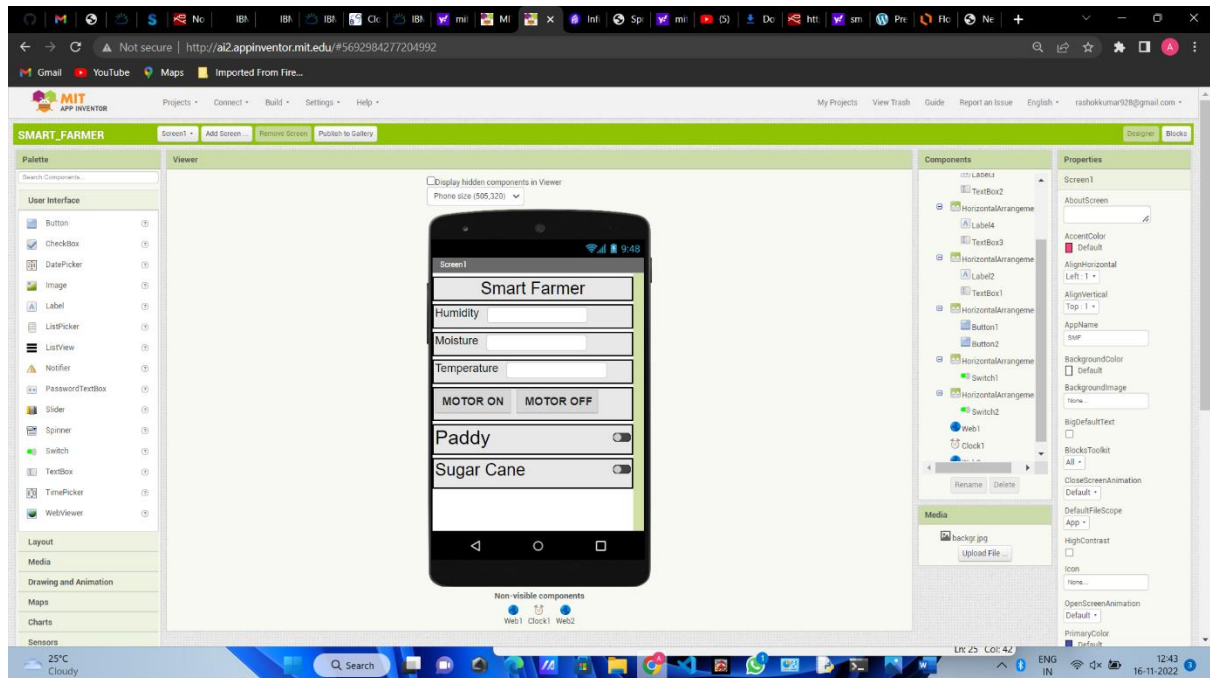


Node Red Web UI Command Lines:

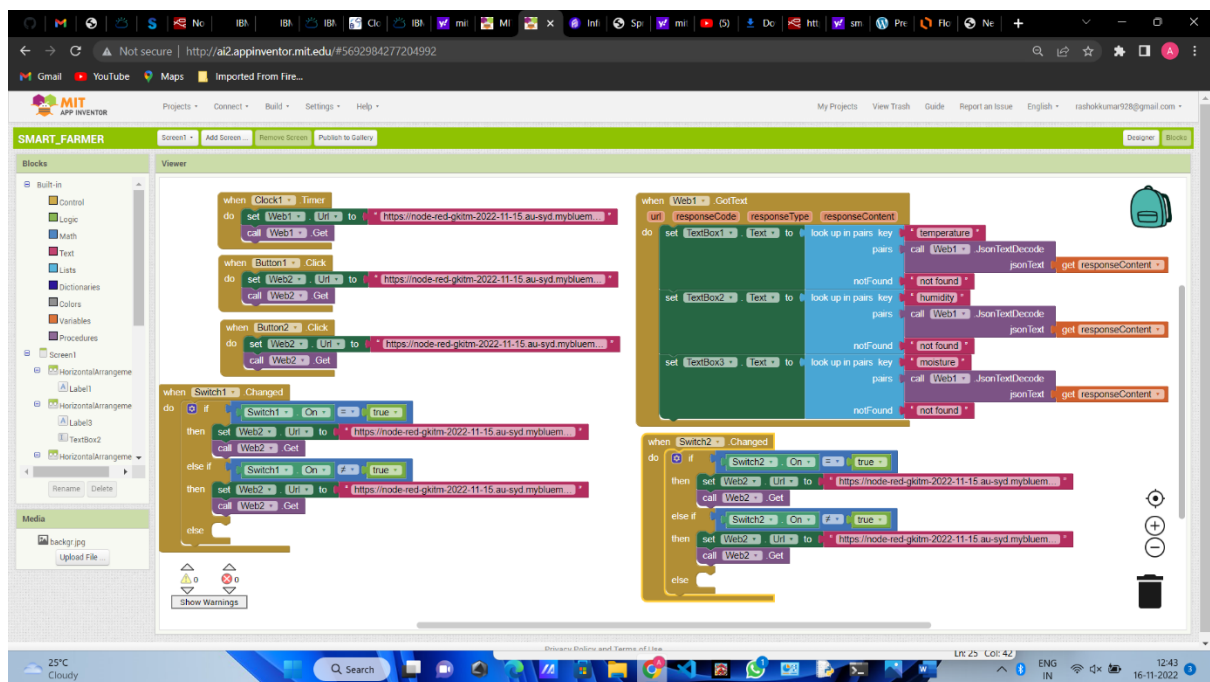


MIT APP INVENTOR :

User Interface:



MIT APP – Blocks Coding:



Working Of MIT APPLICATION :





Result :

