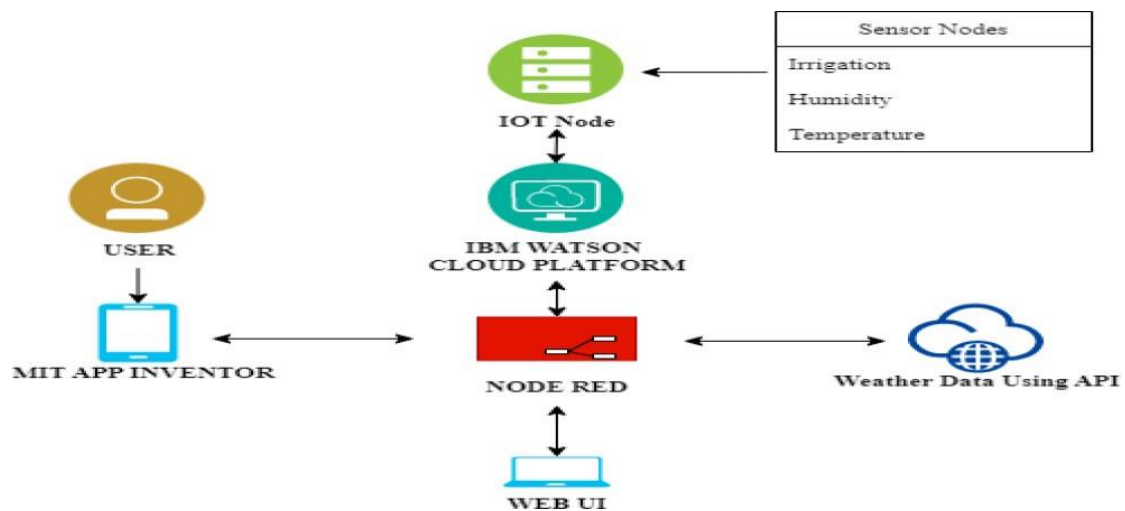


Project Development -Delivery of Sprint-4

Date	17 NOV 2022
Team ID	PNT2022TMID17431
Project Name	Project -Smart farmer-IOT enabled smart Farming Application

Flow Diagram



Python Code:

- For Connecting IBM Cloud
- For NODE RED
- Weather Map Information
- MIT App Inventor

```
#IBM Watson IOT Platform
```

```
#pip install wiotp-sdk
```

```
import wiotp.sdk.device
```

```
import time
```

```
import random

import requests, json

ms=0

# Enter your API key here
api_key = "a0db30a689a774b93ffcb58ef2eddfda"

# base_url variable to store url
base_url = "http://api.openweathermap.org/data/2.5/weather?"

# Give city name
city_name = 'Chennai, IN'

# complete_url variable to store
# complete url address
complete_url = base_url + "appid=" + api_key + "&q=" + city_name


status='motor off'

myConfig = {
    "identity": {
        "orgId": "17lsro",
        "typeId": "MyDeviceType",
        "deviceId": "12345"
    },
    "auth": {
        "token": "GkatKdiUS?UVHKvnAD"
    }
}
```

```

def myCommandCallback(cmd):

    print("Message received from IBM IoT Platform: %s" %
cmd.data['command'])

    m=cmd.data['command']

    if(m=="MOTOR ON"):#if motor is on

        print("MOTOR IS ON")

        global status

        status='motor on'

        myData={'temperature':temp,
'humidity':hum,'soilmoisture':sm_percentage,'status':status,'api_temperature':
api_temperature,'api_pressure':api_pressure,'api_humidity':api_humidity,'api
_weather_description':api_weather_description}

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

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


    time.sleep(2)

    elif(m=="MOTOR OFF"):#if motor is off

        print("MOTOR IS OFF")


        status='motor off'

        myData={'temperature':temp,
'humidity':hum,'soilmoisture':sm_percentage,'status':status,'api_temperature':
api_temperature,'api_pressure':api_pressure,'api_humidity':api_humidity,'api
_weather_description':api_weather_description}

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

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

```

```
time.sleep(2)
```

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

```
while True:
```

```
    # get method of requests module
```

```
    # return response object
```

```
    response = requests.get(complete_url)
```

```
    # json method of response object
```

```
    # convert json format data into
```

```
    # python format data
```

```
    x = response.json()
```

```
    # Now x contains list of nested dictionaries
```

```
    # Check the value of "cod" key is equal to
```

```
    # "404", means city is found otherwise,
```

```
    # city is not found
```

```
    if x["cod"] != "404":
```

```
        y = x["main"]
```

```
api_temperature = y["temp"]#getting api temperature data
```

```
api_pressure = y["pressure"]#getting api pressure data
```

```
api_humidity = y["humidity"] #getting api humidity data
```

```
z = x["weather"]
```

```
api_weather_description = z[0]["description"]#getting api weather  
condition data
```

```
temp=random.randint(-20,125)#geneating ranom values for temperature
```

```
hum=random.randint(0,100)#geneating ranom values for humidity
```

```
soilmoisture=random.randint(0,1023)#analog sensor
```

```
sm_percentage=(soilmoisture/1023)*100
```

```
sm_percentage=int(sm_percentage)#geneating ranom values for  
soilmoisture
```

```
myData={'temperature':temp,  
'humidity':hum,'soilmoisture':sm_percentage,'status':status,'api_temperature':  
api_temperature,'api_pressure':api_pressure,'api_humidity':api_humidity,'api  
_weather_description':api_weather_description}
```

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

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

```
client.commandCallback = myCommandCallback
```

```
time.sleep(2)
```

```
time.sleep(2)
```

```
client.disconnect()
```

```
python code with cmnts.py - C:\Users\B.SOMESHWARAN\Desktop\IBM\Project Development Phase\sprint -1\python code with cmnts.py (3.8.10)
File Edit Format Run Options Window Help
#IBM Watson IOT Platform
#pip install wiotp-sdk
import wiotp.sdk.device
import time
import random
import requests, json

ms=0
# Enter your API key here
api_key = "a0db30a689a774b93ffcb58ef2eddfda"
# base_url variable to store url
base_url = "http://api.openweathermap.org/data/2.5/weather?"
# Give city name
city_name = 'Chennai, IN'
# complete_url variable to store
# complete url address
complete_url = base_url + "appid=" + api_key + "&q=" + city_name

status='motor off'
myConfig = {
    "identity": {
        "orgId": "171sro",
        "typeId": "MyDeviceType",
        "deviceId": "12345"
    },
    "auth": {
        "token": "GkatKdiUS?UVHKvnAD"
    }
}

def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="MOTOR ON"):#if motor is on
        print("MOTOR IS ON")
        global status
        status='motor on'
        myData={'temperature':temp, 'humidity':hum, 'soilmoisture':sm_percentage, 'status':status, 'api_temperature':api_temperature, 'api_pressure':api_pressure}
        client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
        print("Published data Successfully: %s", myData)

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

while True:
    # get method of requests module
    # return response object
    response = requests.get(complete_url)
    # json method of response object
    # convert json format data into
    # python format data
    x = response.json()
    # Now x contains list of nested dictionaries
    # Check the value of "cod" key is equal to
    # "404", means city is found otherwise,
    # city is not found
    if x["cod"] != "404":

        y = x["main"]

        api_temperature = y["temp"]#getting api temperature data

        api_pressure = y["pressure"]#getting api pressure data

        api_humidity = y["humidity"] #getting api humidity data

        z = x["weather"]

        api_weather_description = z[0]["description"]#getting api weather condition data
```

```

temp=random.randint(-20,125)#geneating ranom values for temperature
hum=random.randint(0,100)#geneating ranom values for humidity
soilmoisture=random.randint(0,1023)#analog sensor
sm_percentage=(soilmoisture/1023)*100
sm_percentage=int(sm_percentage)#geneating ranom values for soilmoisture
myData={'temperature':temp, 'humidity':hum,'soilmoisture':sm_percentage,'status':status,'api_temperature':api_temperature,'api_pressure':api_pressure,'api_humidity':api_humidity}
client.publish(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
print("Published data Successfully: %s" % myData)
client.commandCallback = myCommandCallback
time.sleep(2)

time.sleep(2)
client.disconnect()

```

Running Module

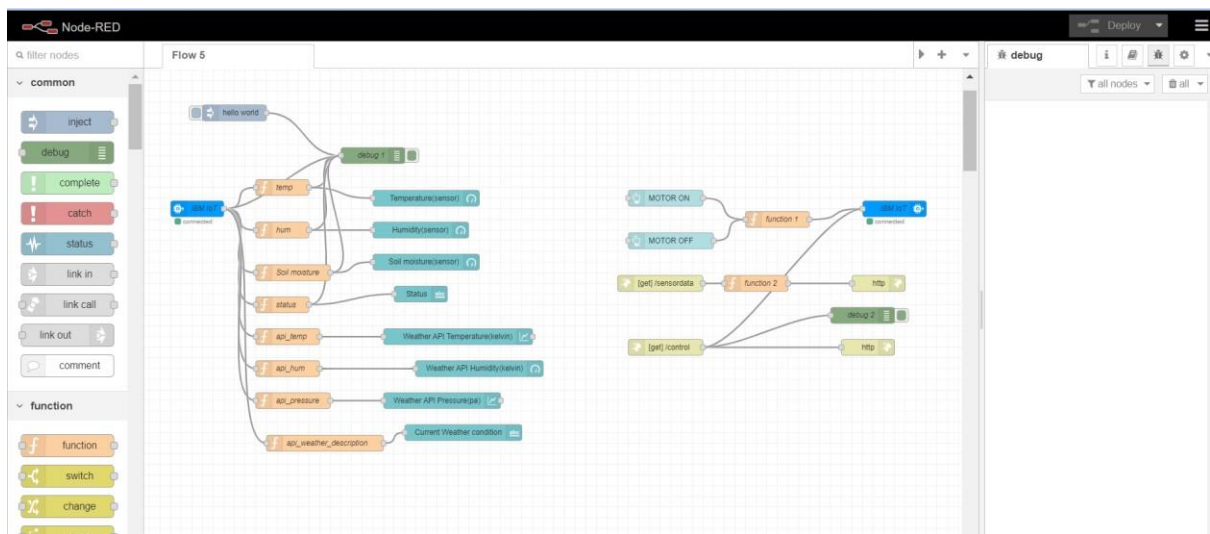
```

IDLE Shell 3.8.10
File Edit Shell Debug Options Window Help
- RESTART: C:\Users\B.SOMESHWARAN\Desktop\IEM\Project Development Phase\sprint -l\python code with cmnts.py
2022-11-15 21:26:16,286 wiotp.sdk.device.client.DeviceClient INFO Connected successfully: d:171src:MyDeviceType:12345
Published data Successfully: %s {'temperature': 60, 'humidity': 34, 'soilmoisture': 57, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 96, 'humidity': 85, 'soilmoisture': 70, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 93, 'humidity': 3, 'soilmoisture': 8, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 1013
, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 50, 'humidity': 23, 'soilmoisture': 60, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 76, 'humidity': 16, 'soilmoisture': 94, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 86, 'humidity': 51, 'soilmoisture': 56, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': -6, 'humidity': 27, 'soilmoisture': 22, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 59, 'humidity': 62, 'soilmoisture': 13, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 12, 'humidity': 4, 'soilmoisture': 81, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 101
3, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 74, 'humidity': 89, 'soilmoisture': 50, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': -1, 'humidity': 14, 'soilmoisture': 77, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 18, 'humidity': 66, 'soilmoisture': 81, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 98, 'humidity': 15, 'soilmoisture': 100, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 1
013, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': -11, 'humidity': 17, 'soilmoisture': 96, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 1
013, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 102, 'humidity': 87, 'soilmoisture': 47, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 1
013, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 49, 'humidity': 57, 'soilmoisture': 47, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 15, 'humidity': 3, 'soilmoisture': 84, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 101
3, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 48, 'humidity': 96, 'soilmoisture': 49, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}
Published data Successfully: %s {'temperature': 103, 'humidity': 3, 'soilmoisture': 84, 'status': 'motor off', 'api_temperature': 300.14, 'api_pressure': 10
13, 'api_humidity': 83, 'api_weather_description': 'mist'}

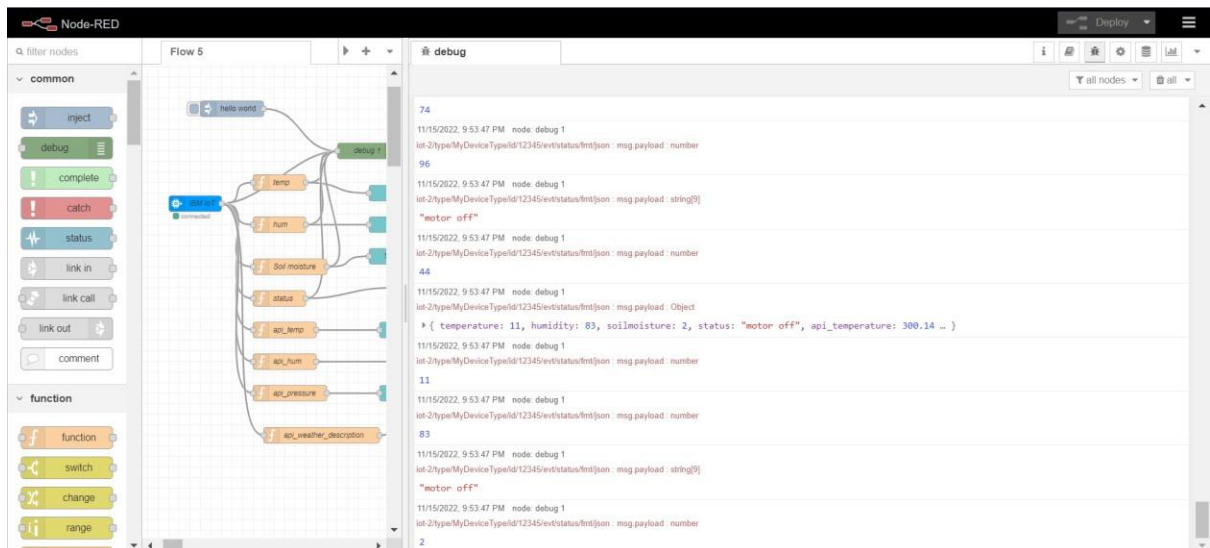
```

NODE RED Flow Connections

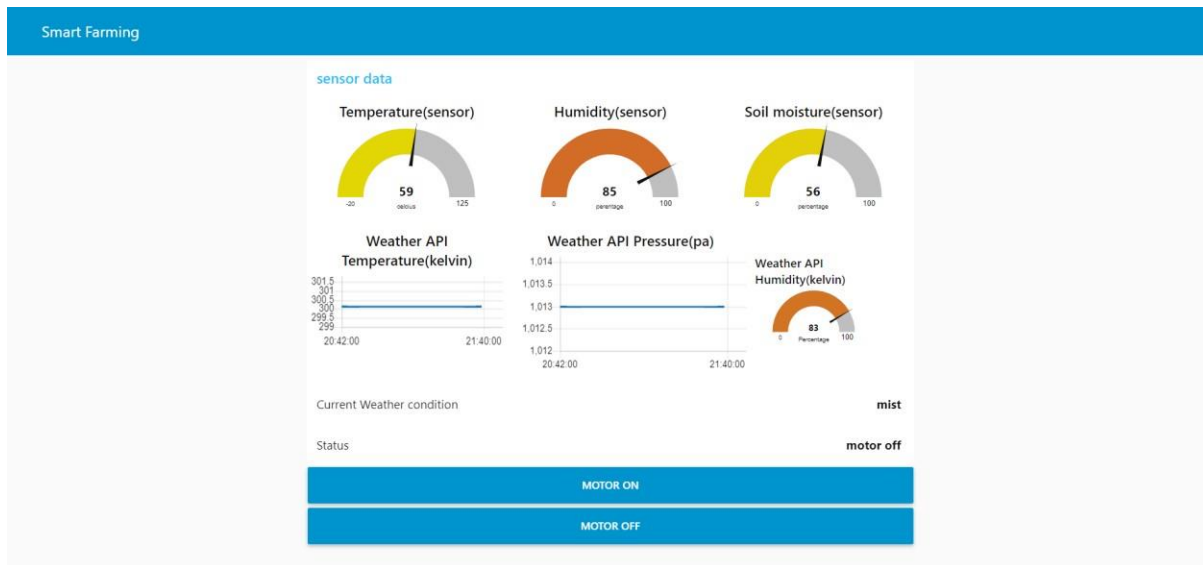
- Interfacing IBM Cloud
- Intefacing & Getting Sensor Datas
- Connecting MIT App Inventor
- Weather Map Parameters



Live Publish Data Output Of Node Red



Web API Output



IBM Watson IoT Platform

- Device Connected Details

The screenshot shows the IBM Watson IoT Platform interface. The top navigation bar includes "Browse", "Action", "Device Types", and "Interfaces". The main content area displays a table of devices. The first device, ID 12345, is connected and has a status of "MyDeviceType". The second device, ID 54321, is disconnected and has a status of "TestType". The table also shows the date added, descriptive location, and added by for each device. A "Device Simulator" toggle is visible on the right.

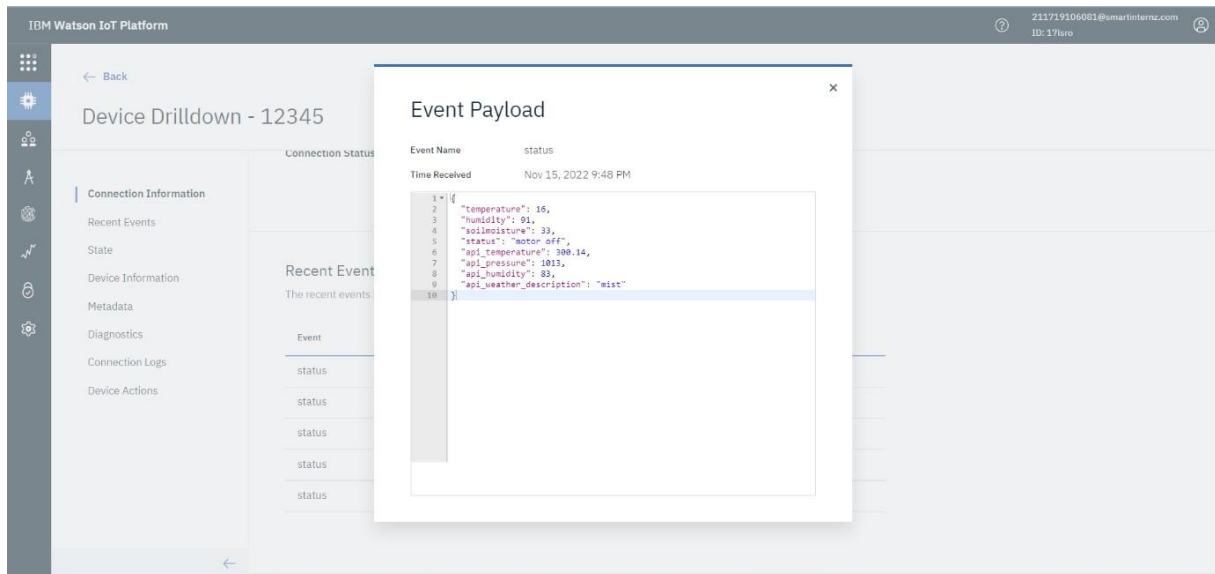
Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location	Added By
12345	Connected	MyDeviceType	Device	Oct 27, 2022 8:04 PM		211719106081@smartinternz.com
54321	Disconnected	TestType	Device	Nov 4, 2022 11:52 PM		211719106081@smartinternz.com

Items per page: 50 | 1-2 of 2 items

1 of 1 page

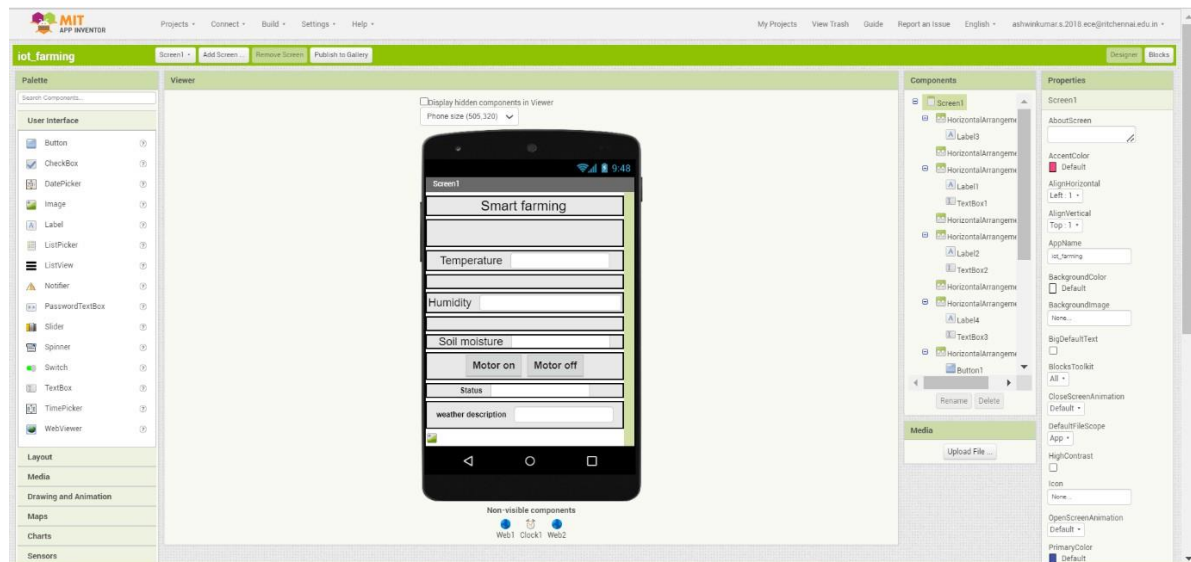
Live Date Output Of IBM Watson IoT Platform

- Sensor Output Data
- Weather Condition
- Weather Map Parameters In Current Location

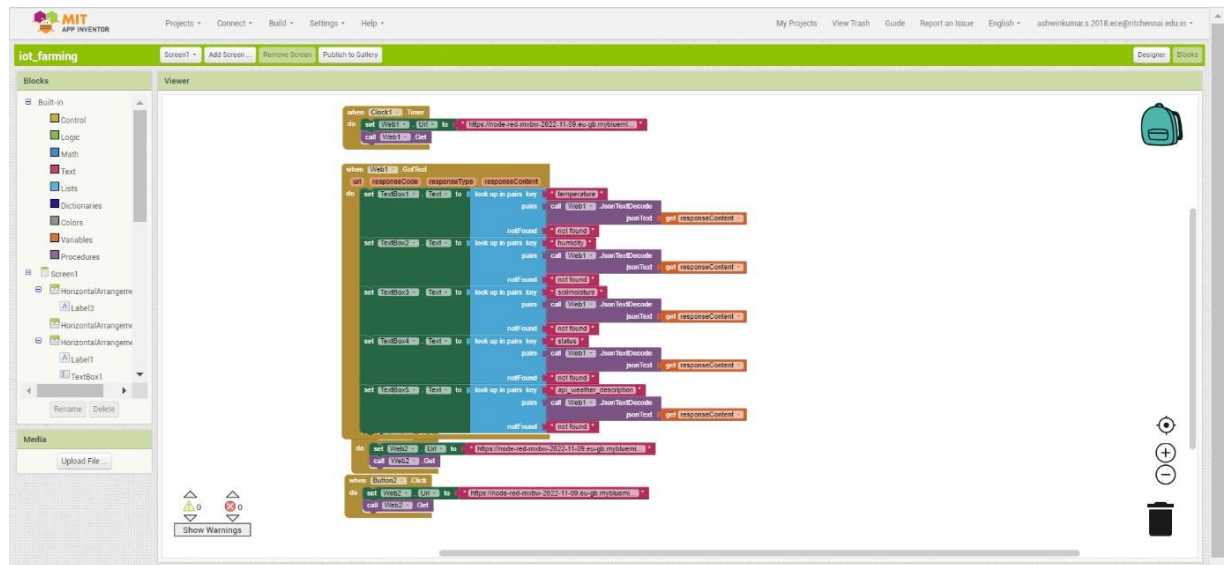


MIT APP INVENTOR

- Design



Back End Process(Block)



Mobile Application Ouput



Smart farming



Smart farming

Temperature

Humidity

Soil moisture

Status

weather description

Temperature

Humidity

Soil moisture

Status

weather description

