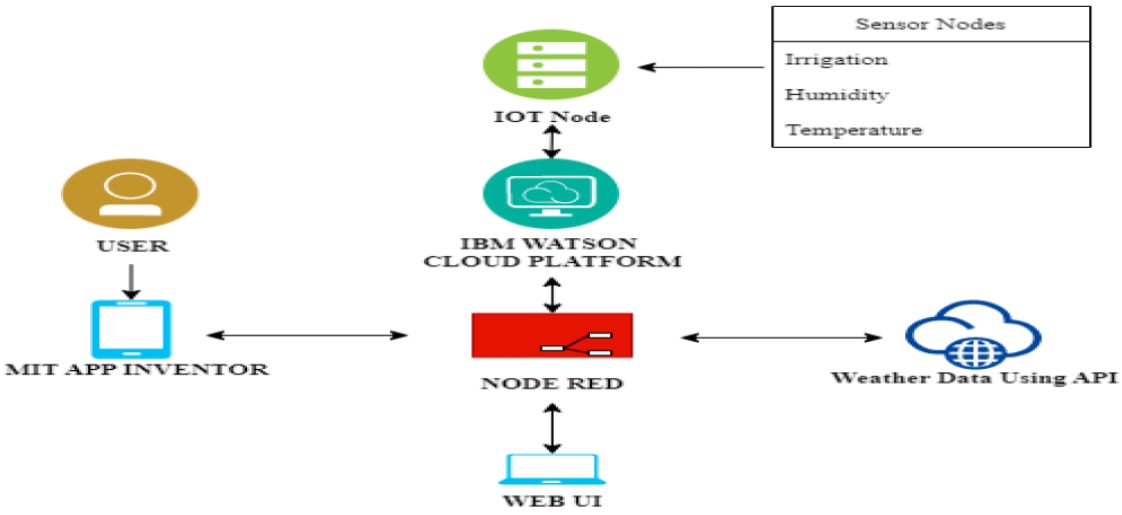
Project Development -Delivery of Sprint-4

Date	17 NOV 2022
Team ID	PNT2022TMID26547
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 cmments.py - C:\Users\B.SOMESHWARAN\Desktop\IBM\Project Development Phase\sprint -1\python code with cmments.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
```

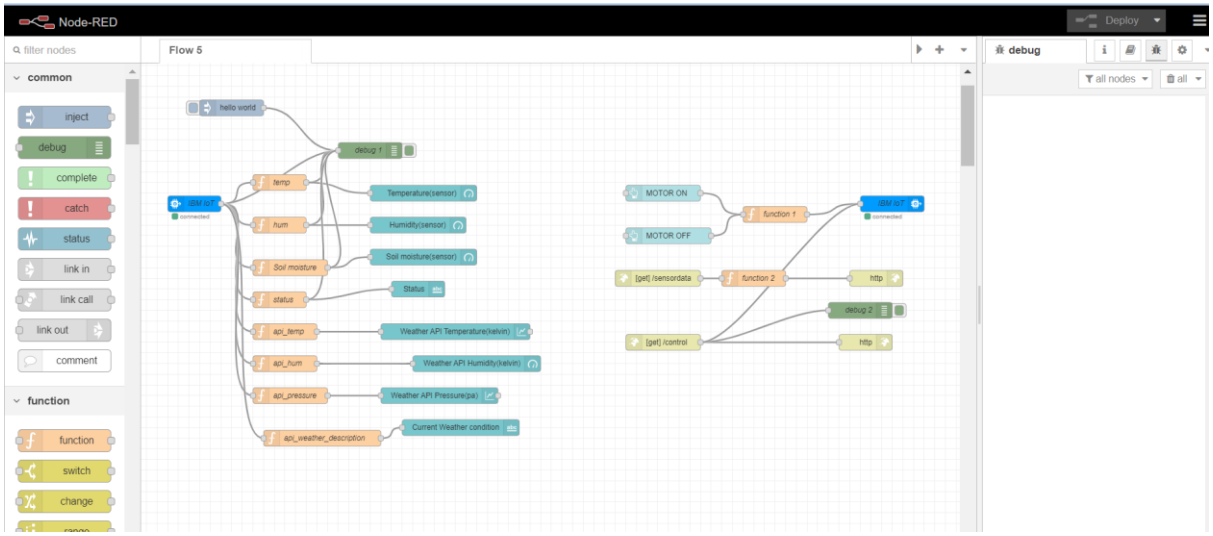
```
    temp=random.randint(-20,125)#geneating ranom values for temperature
    hum=random.randint(0,100)#geneating ranom values for humidity
    soilmoisture=random.randint(0,1023)#analog sensor
    sm_percentage=(soilmoisture/1023)*100
    sm_percentage=int(sm_percentage)#geneating ranom values for soilmoisture
    myData={'temperature':temp, 'humidity':hum,'soilmoisture':sm_percentage,'status':status,'api_temperature':api_temperature,'api_pressure':api_pressure,'ap
    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()
```

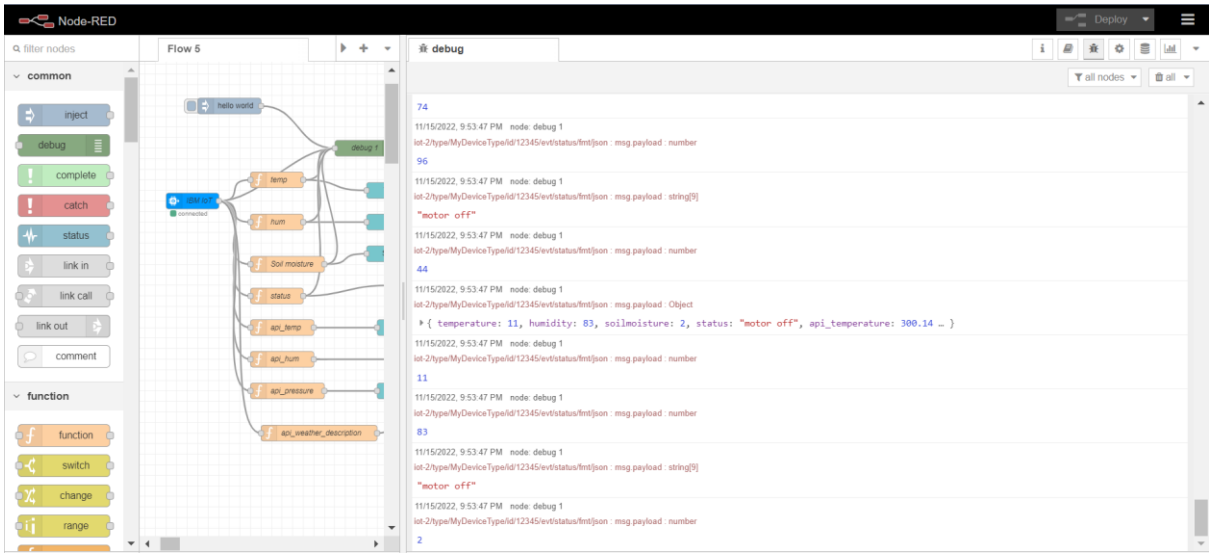
Running Module

NODE RED Flow Connections

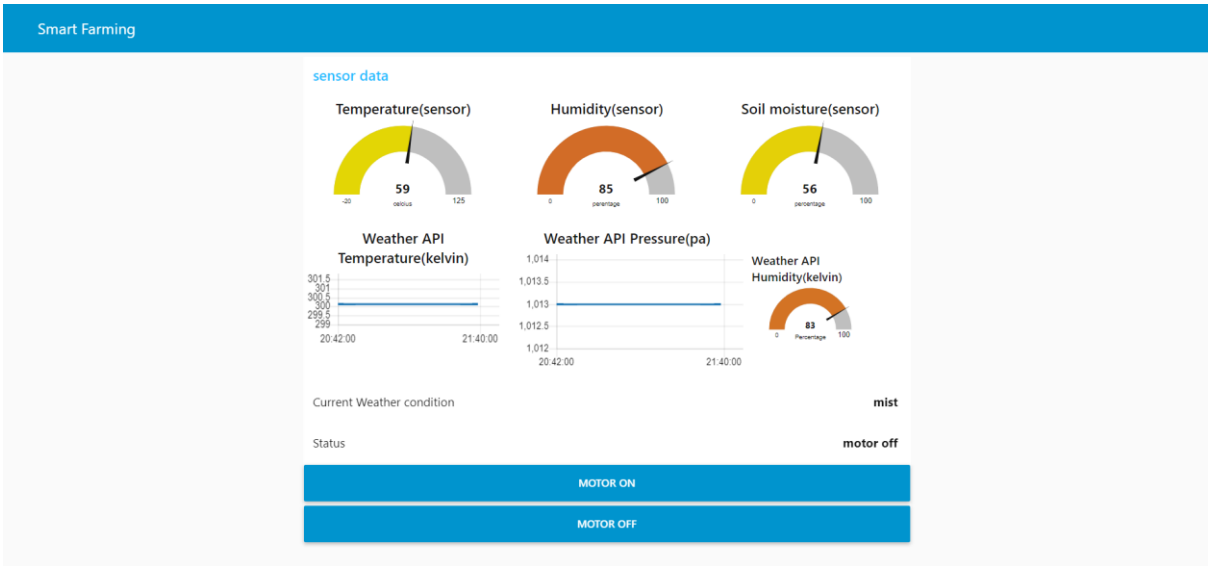
- **Interfacing IBM Cloud**
- **Intefacing & Getting Sensor Datas**
- **Connecting MIT App Inventor**
- **Weather Map Parameters**



Live Publish Data Output Of Node Red

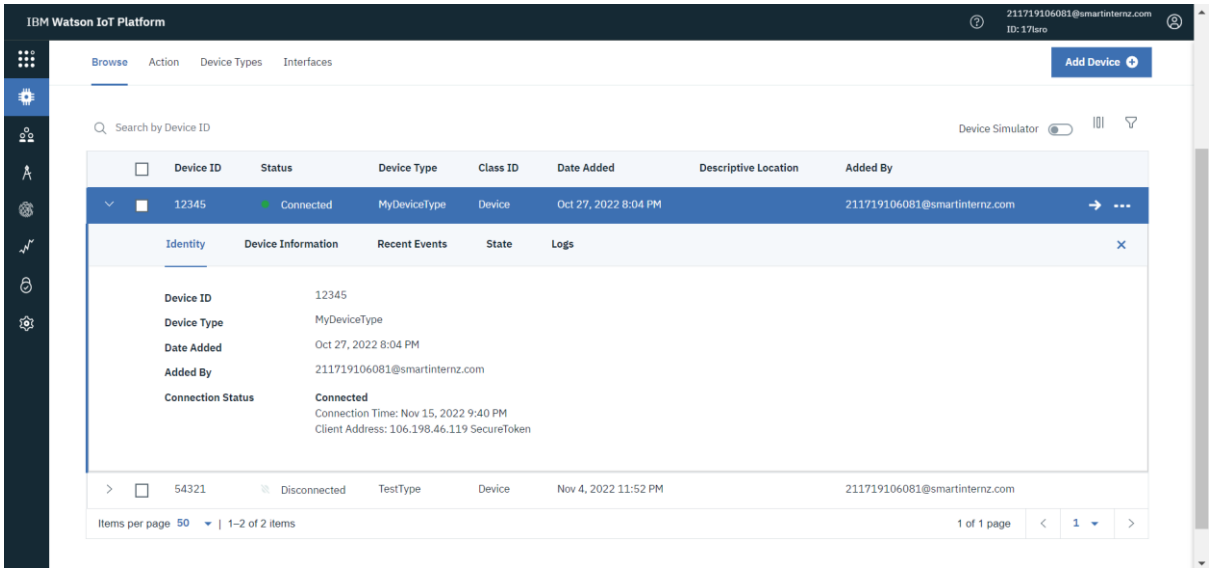


Web API Output



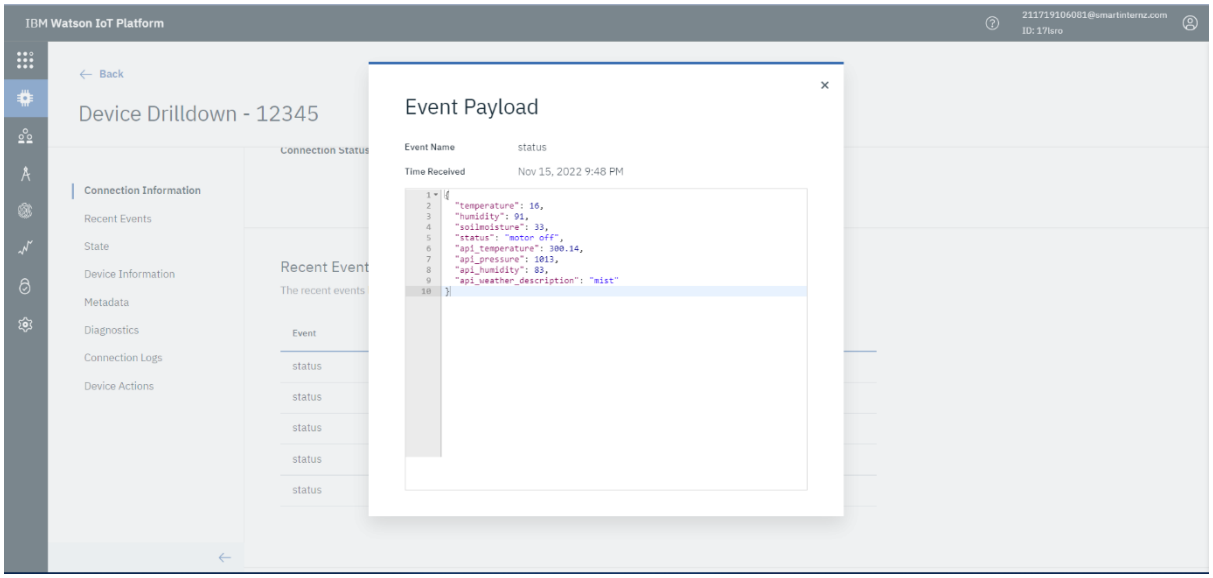
IBM Watson IoT Platform

- **Device Connected Details**



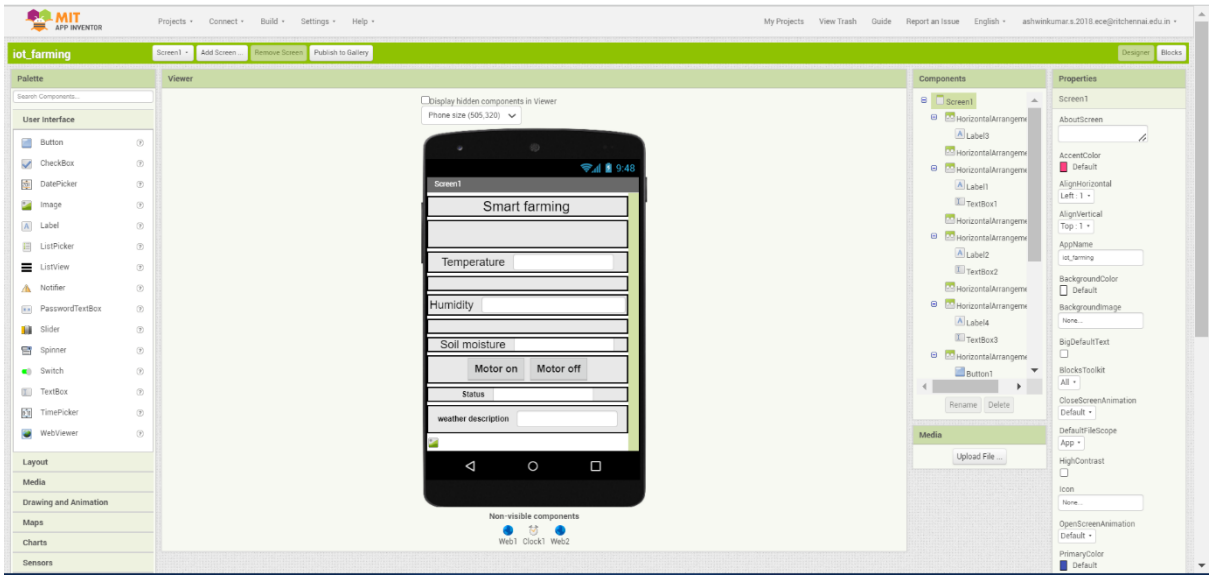
Live Date Output Of IBM Watson Iot Platform

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

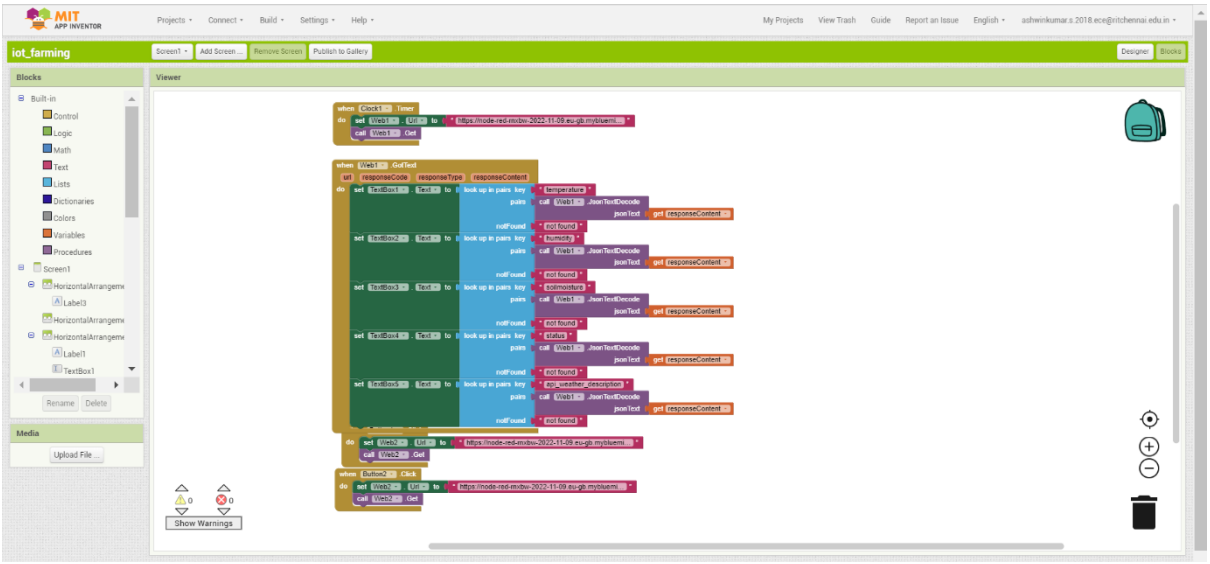


MIT APP INVENTOR

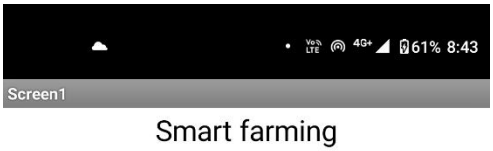
- **Design**



Back End Process(Block)



Mobile Application Ouput



Smart farming

Temperature 90

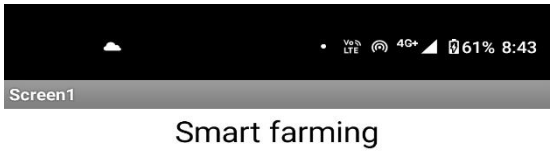
Humidity 50

Soil moisture 57

Motor on Motor off

Status motor off

weather description mist



Smart farming

Temperature 113

Humidity 48

Soil moisture 94

Motor on Motor off

Status motor on

weather description mist