Sprint-1

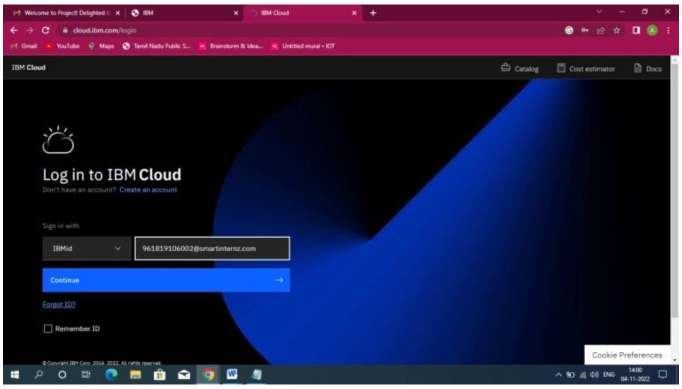
TEAM ID	*PNT2022TMID34551*
PROJECT NAME	*Smart Farmer-IOT Enabled Smart Farming Application*

RESOURCES PAGES

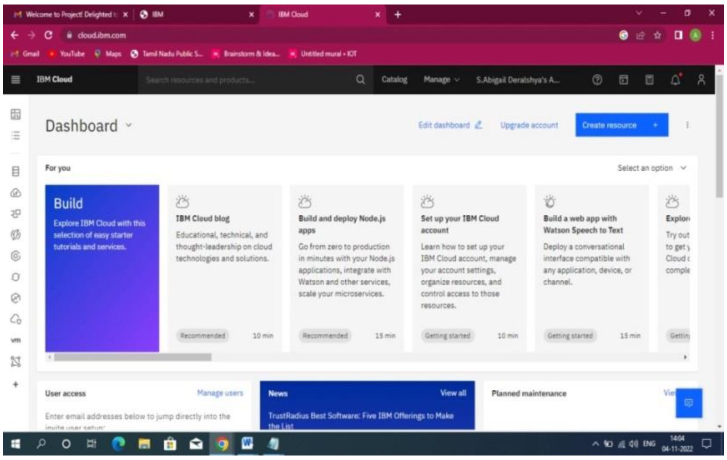
IBM CLOUD:

➢ LOGIN TO THE WEBSITE: IBM CLOUD LOGIN

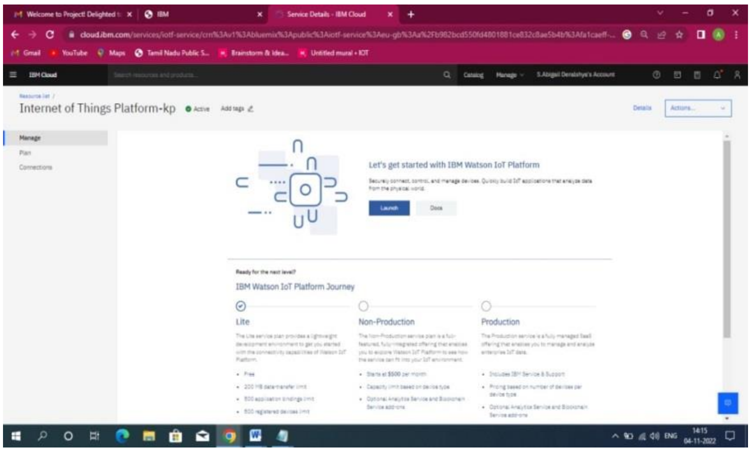
1. Enter The SI Mail Id And Create Your Password .



2. After Login Page Dashboard Will Be Displayed And Click The Catalog And Open Internet of Things:

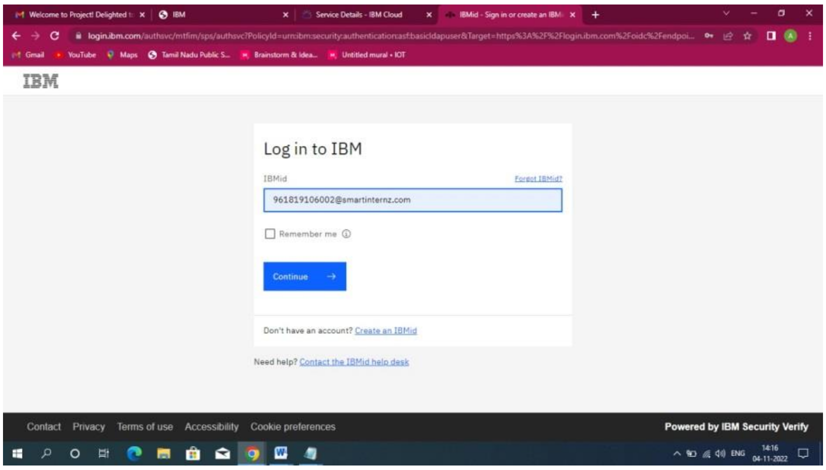

cloud.ibm.com
IBM Cloud
Search resources and products...
Catalog
Manage
S.Abigail Deralshya's A...
Dashboard
Edit dashboard
Upgrade account
Create resource +
For you
Select an option
Build
Explore IBM Cloud with this selection of easy starter tutorials and services.
IBM Cloud blog
Educational, technical, and thought-leadership on cloud technologies and solutions.
Recommended
10 min
Build and deploy Node.js apps
Go from zero to production in minutes with your Node.js applications, integrate with Watson and other services, scale your microservices.
Recommended
15 min
Set up your IBM Cloud account
Learn how to set up your IBM Cloud account, manage your account settings, organize resources, and control access to those resources.
Getting started
10 min
Build a web app with Watson Speech to Text
Deploy a conversational interface compatible with any application, device, or channel.
Getting started
15 min
User access
Manage users
Enter email addresses below to jump directly into the invite user setup
News
View all
TrustRadius Best Software: Five IBM Offerings to Make the List
Planned maintenance
14:04
04-11-2022

3. After That you will find this page and click the launch button:


Service Details - IBM Cloud
cloud.ibm.com/services/iotf-service/crn%3Av1%3Abluemix%3Apublic%3Aiotf-service%3Aeu-gb%3Aa%2Fb982bcd550fd4801881ce832c8ae5b4b%3Afa1caeff-...
IBM Cloud
Catalog
Manage
S.Abigail Deralshya's Account
Resource list /
Internet of Things Platform-kp
Active
Add tags
Details
Actions...
Manage
Plan
Connections
Let's get started with IBM Watson IoT Platform
Securely connect, control, and manage devices. Quickly build IoT applications that analyze data from the physical world.
Launch
Docs
Ready for the next level?
IBM Watson IoT Platform Journey
Lite
Non-Production
Production
14:15
04-11-2022

➢ TO LOGIN TO IBM WATSON

1 .After that login the Page and give your SI Mail id and password



Log in to IBM

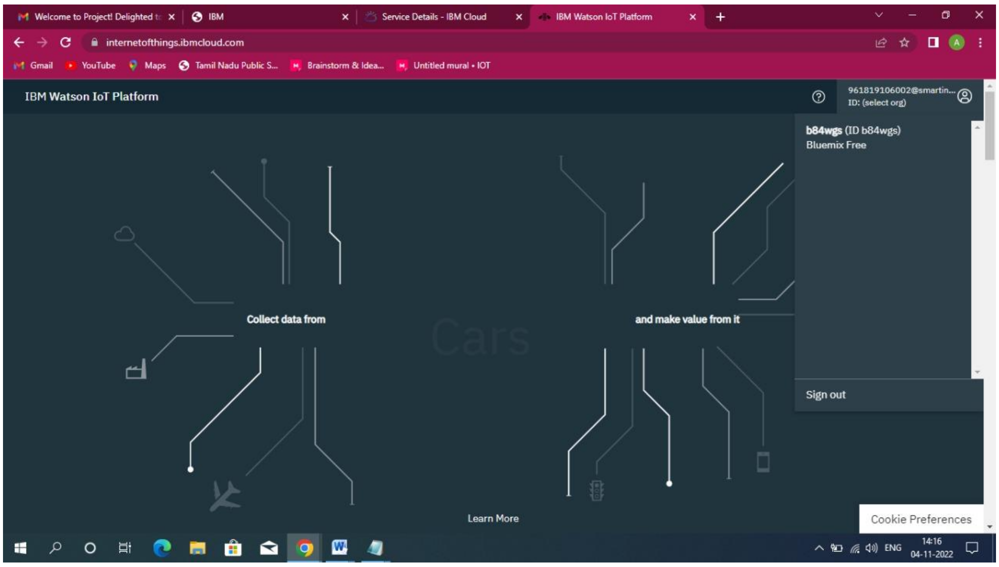
IBMid

961819106002@smartinternz.com

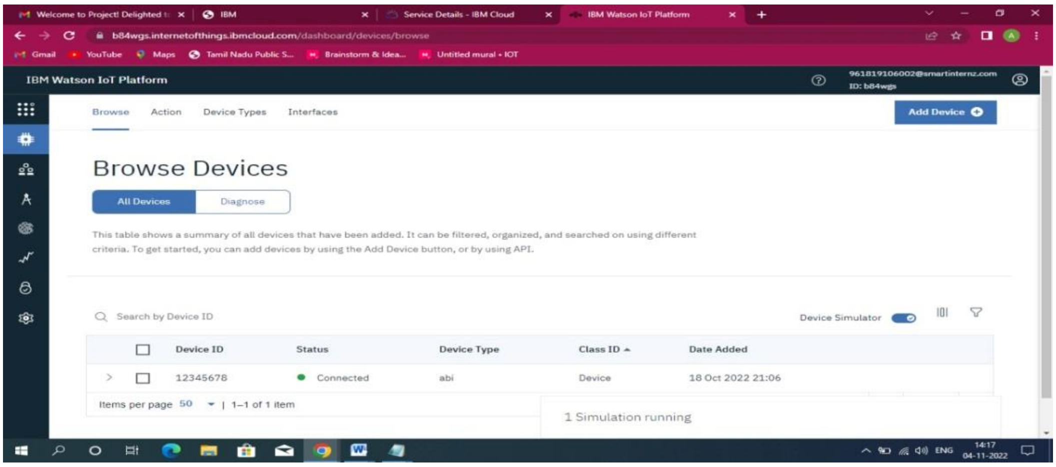
□ Remember me

login.ibm.com/authsvc/mtfim/sps/authsvc?PolicyId=urn:ibm:security:authentication:asf:basicldapuser&Target=https%3A%2F%2Flogin.ibm.com
IBM
Forgot IBMid?
Continue
Don't have an account? Create an IBMid
Need help? Contact the IBMid help desk
Contact
Privacy
Terms of use
Accessibility
Cookie preferences
Powered by IBM Security Verify

2. Then sign in the right side corner id given in the Ibm Watson.



3. After that you will find the Browse Device Page:



Browse Devices

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Device ID	Status	Device Type	Class ID ▲	Date Added
12345678	● Connected	abi	Device	18 Oct 2022 21:06

Items per page 50 | 1–1 of 1 item

1 Simulation running

4. **Create a Device by entering all the required details given below and click finish**:

Organization ID:

b84wgs

Device Type:

abi

Device ID:

12345678

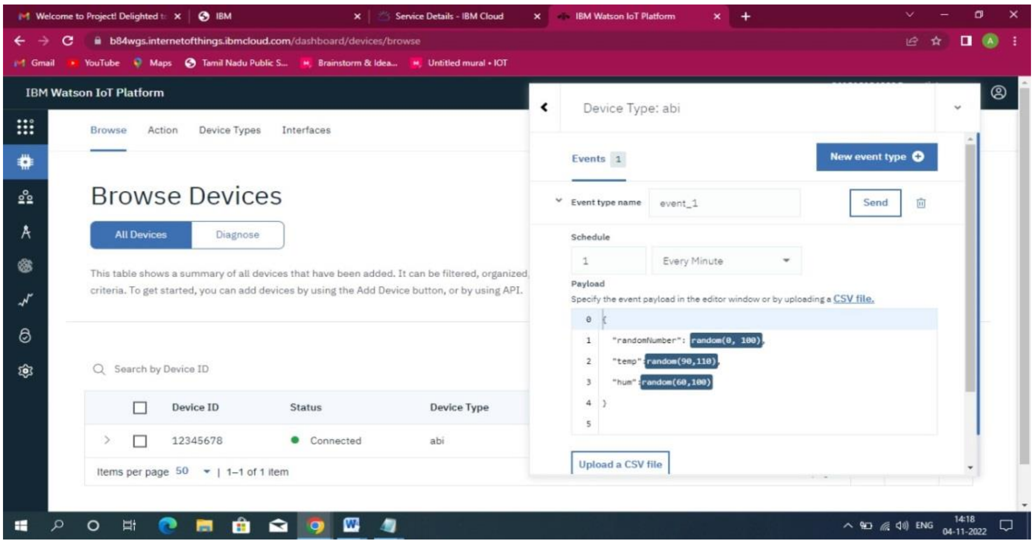
Authentication Method:

use-token-auth

Authentication Token:

87654321

5. And Then you will receive this page by stimulating the created device:



Browse Devices

This table shows a summary of all devices that have been added. It can be filtered, organized, criteria. To get started, you can add devices by using the Add Device button, or by using API.

	Device ID	Status	Device Type
□	12345678	● Connected	abi

Items per page 50 | 1–1 of 1 item

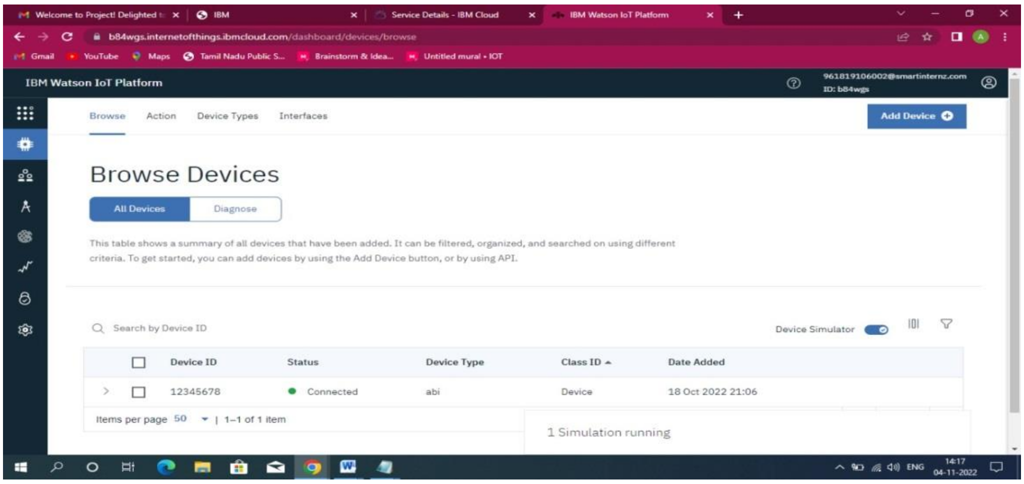
Event type name: event_1

Schedule: 1 Every Minute

Payload

Specify the event payload in the editor window or by uploading a CSV file.

```
{
  "randomNumber": random(0, 100),
  "temp":random(90,110),
  "hum":random(60,100)
}
```



Browse Devices

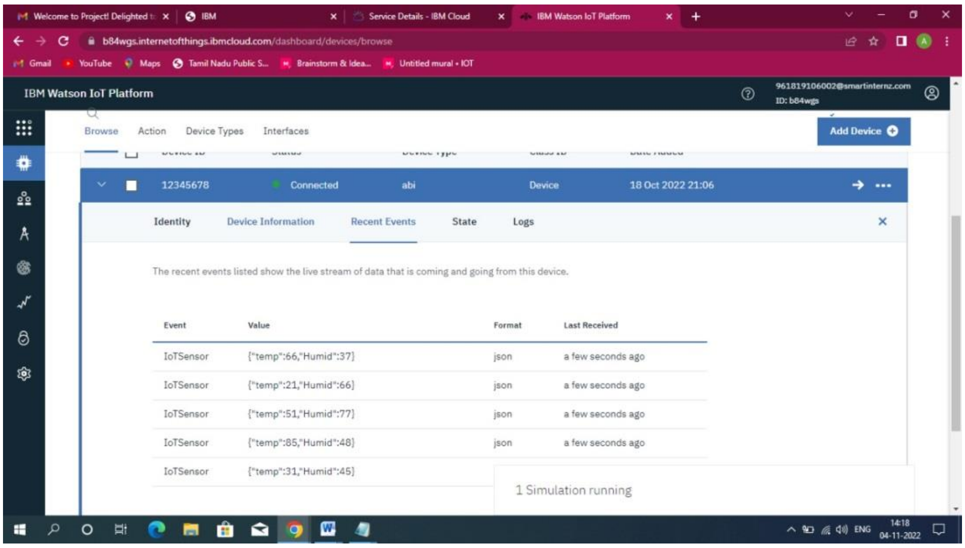
This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

	Device ID	Status	Device Type	Class ID ▲	Date Added
□	12345678	● Connected	abi	Device	18 Oct 2022 21:06

Items per page 50 | 1–1 of 1 item

1 Simulation running

STATUS PAGE (RESULT):



IBM Watson IoT Platform

961819106002@smartinternz.com
ID: b84wgs

12345678 Connected abi Device 18 Oct 2022 21:06

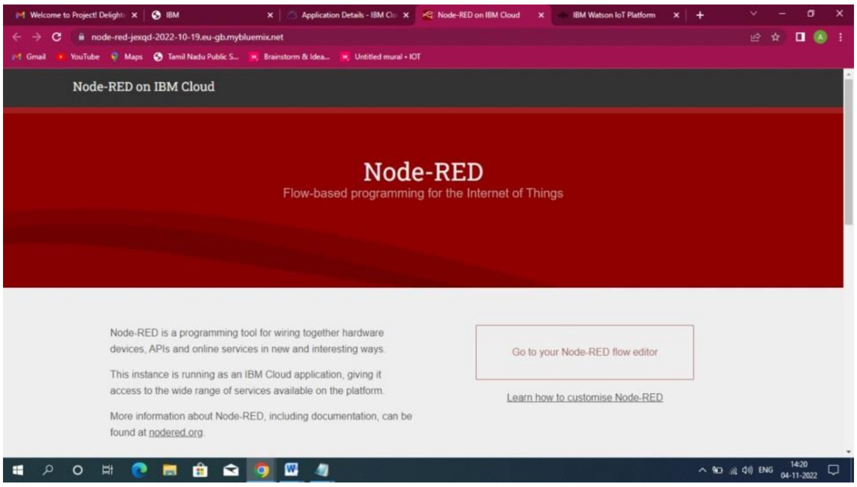
The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
IoTSensor	{"temp":66,"Humid":37}	json	a few seconds ago
IoTSensor	{"temp":21,"Humid":66}	json	a few seconds ago
IoTSensor	{"temp":51,"Humid":77}	json	a few seconds ago
IoTSensor	{"temp":85,"Humid":48}	json	a few seconds ago
IoTSensor	{"temp":31,"Humid":45}		

1 Simulation running

NODE RED:

➢ Login Page of the Node Red:



Node-RED on IBM Cloud

Node-RED
Flow-based programming for the Internet of Things

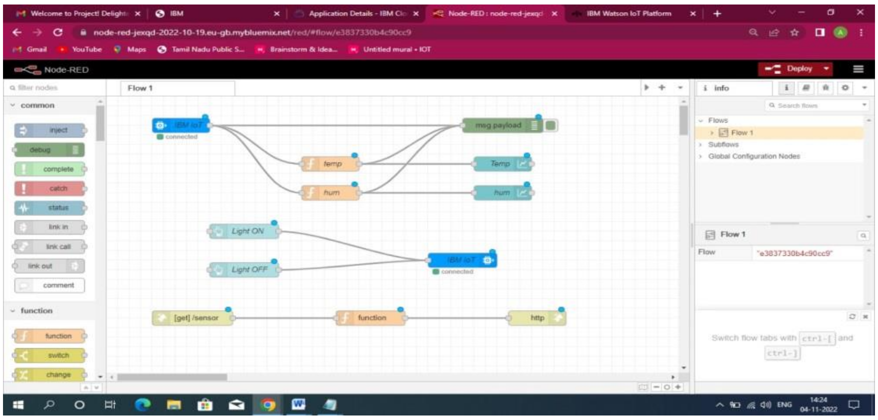
Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways.

This instance is running as an IBM Cloud application, giving it access to the wide range of services available on the platform.

More information about Node-RED, including documentation, can be found at nodered.org.

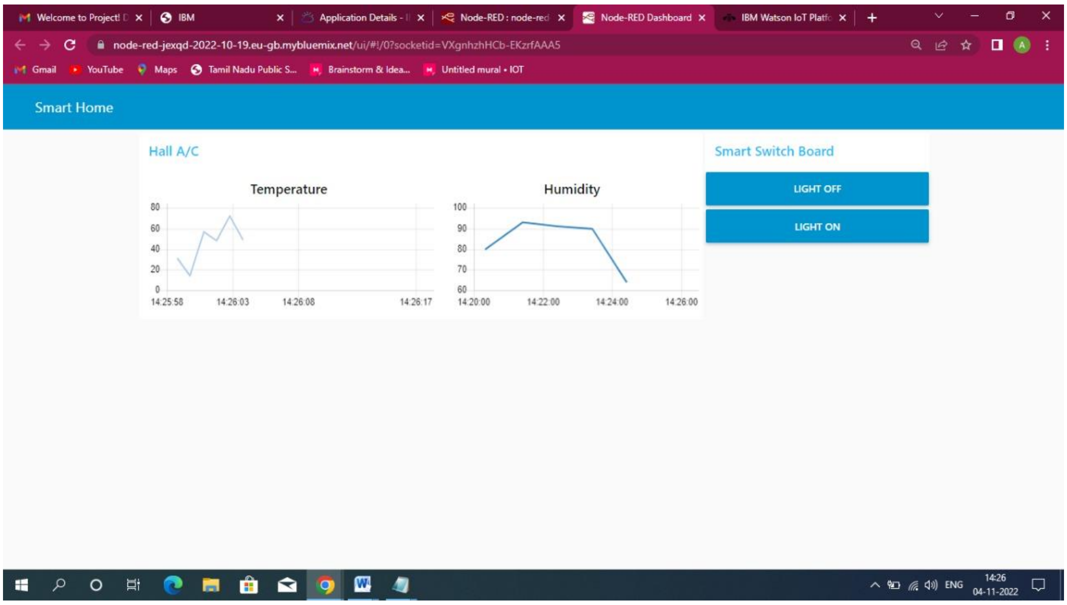
Go to your Node-RED flow editor
Learn how to customise Node-RED

➢ Developing Route based on the program:

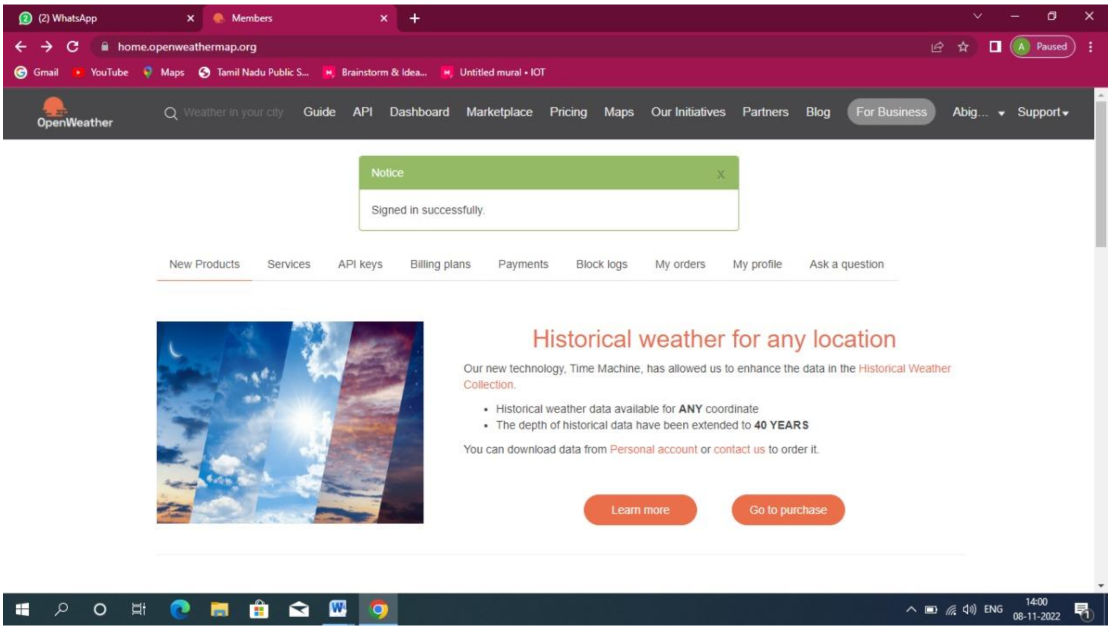


Output :

(The output will be displayed in graphical representation.)



OPEN WEATHER:



Signed in successfully.

Historical weather for any location

Our new technology, Time Machine, has allowed us to enhance the data in the Historical Weather Collection.

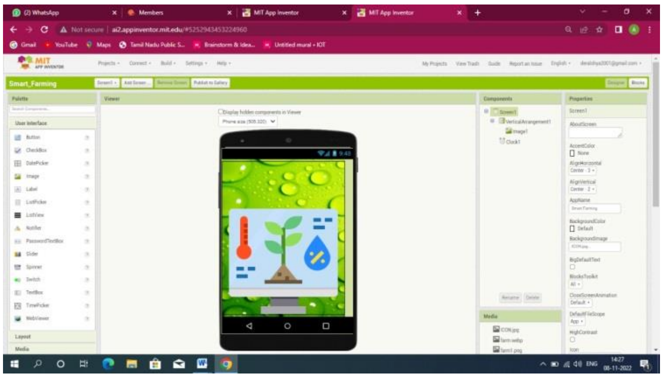
- Historical weather data available for **ANY** coordinate
- The depth of historical data have been extended to **40 YEARS**

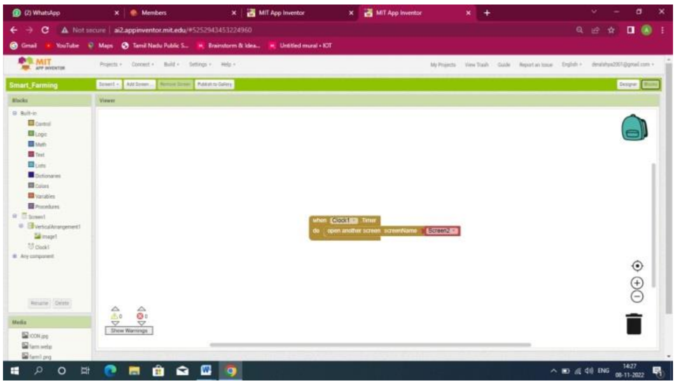
You can download data from Personal account or contact us to order it.

By using this website we can get the longitude and latitude values of the particular location.

MIT APP INVENTOR: ICON PAGE:

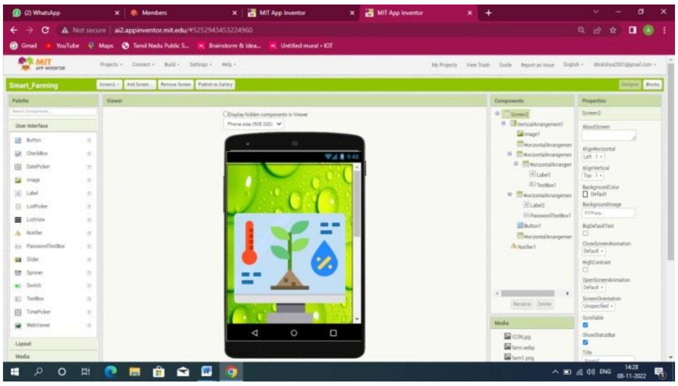
For Screen 1:

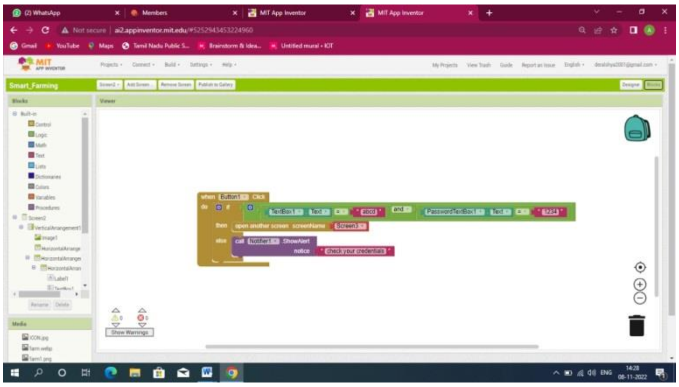




```
when Clock1 .Timer
do  open another screen  screenName  Screen2
```

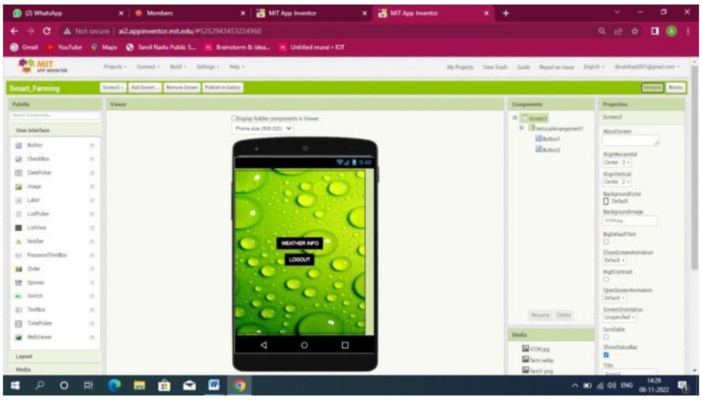
For Screen 2:

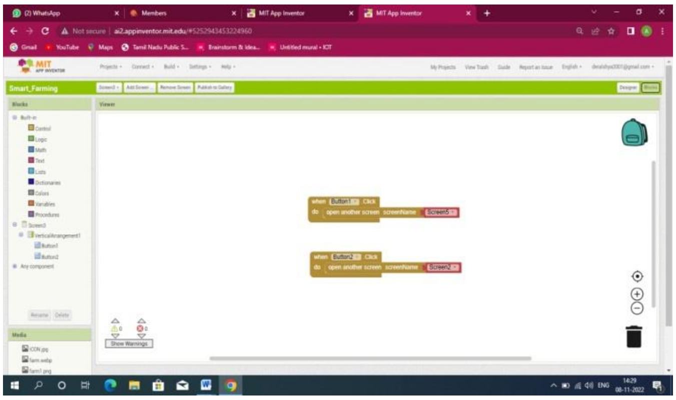




```
when Button1 .Click
do  if  TextBox1 . Text  =  " abcd "  and  PasswordTextBox1 . Text  =  " 1234 "
    then  open another screen  screenName  Screen3
    else  call Notifier1 .ShowAlert
                          notice  " check your credentials "
```

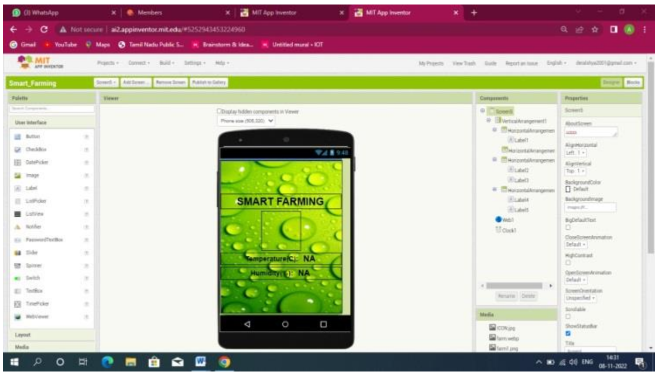
For Screen 3:

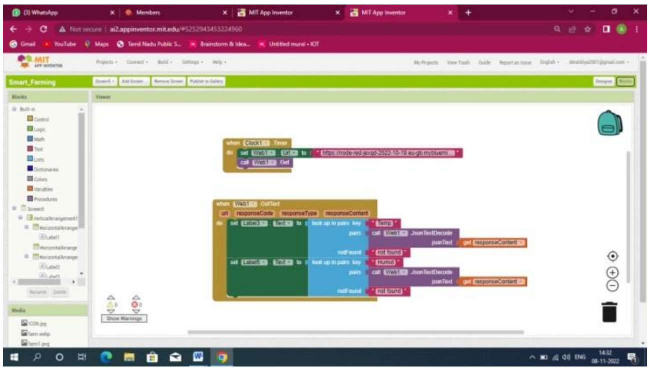




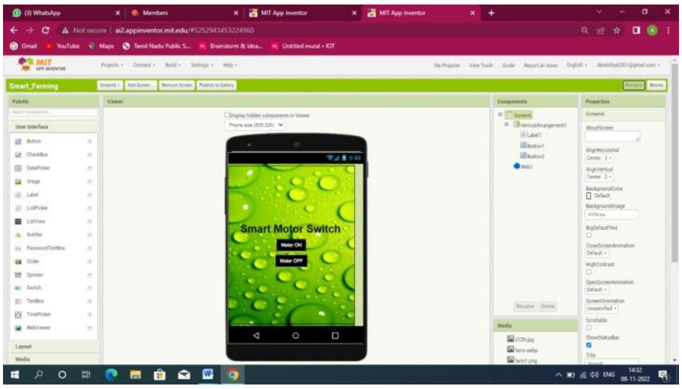
```
when Button1 .Click
do  open another screen  screenName  Screen5

when Button2 .Click
do  open another screen  screenName  Screen2
```

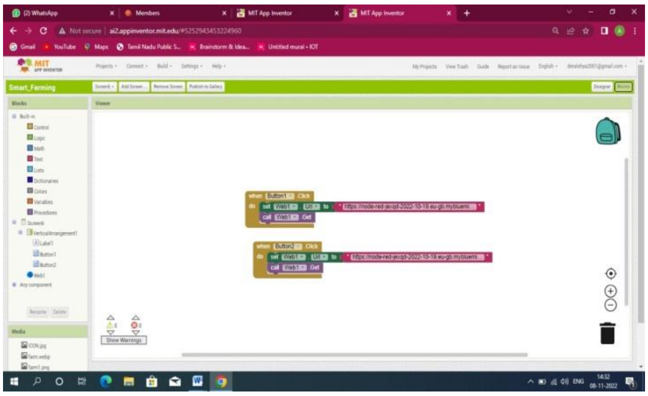
For Screen 4:





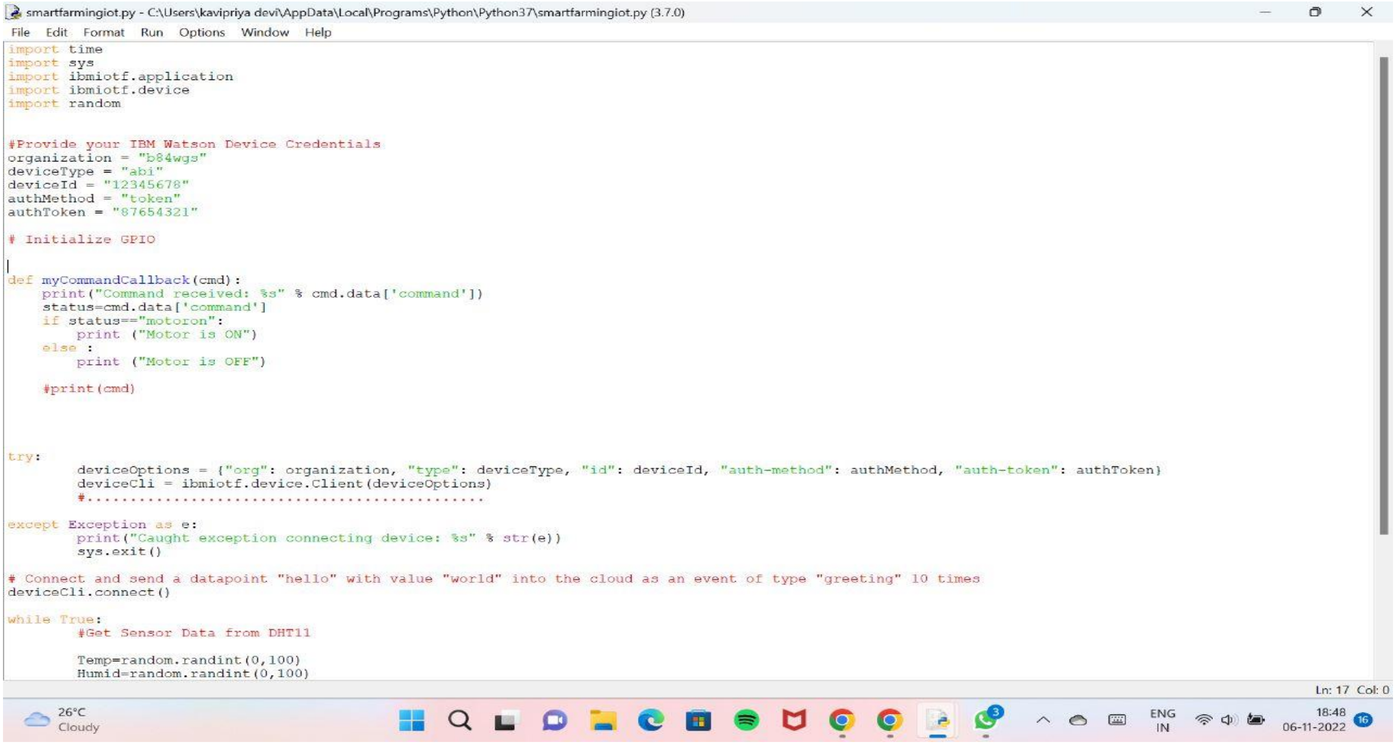
For Screen 5:



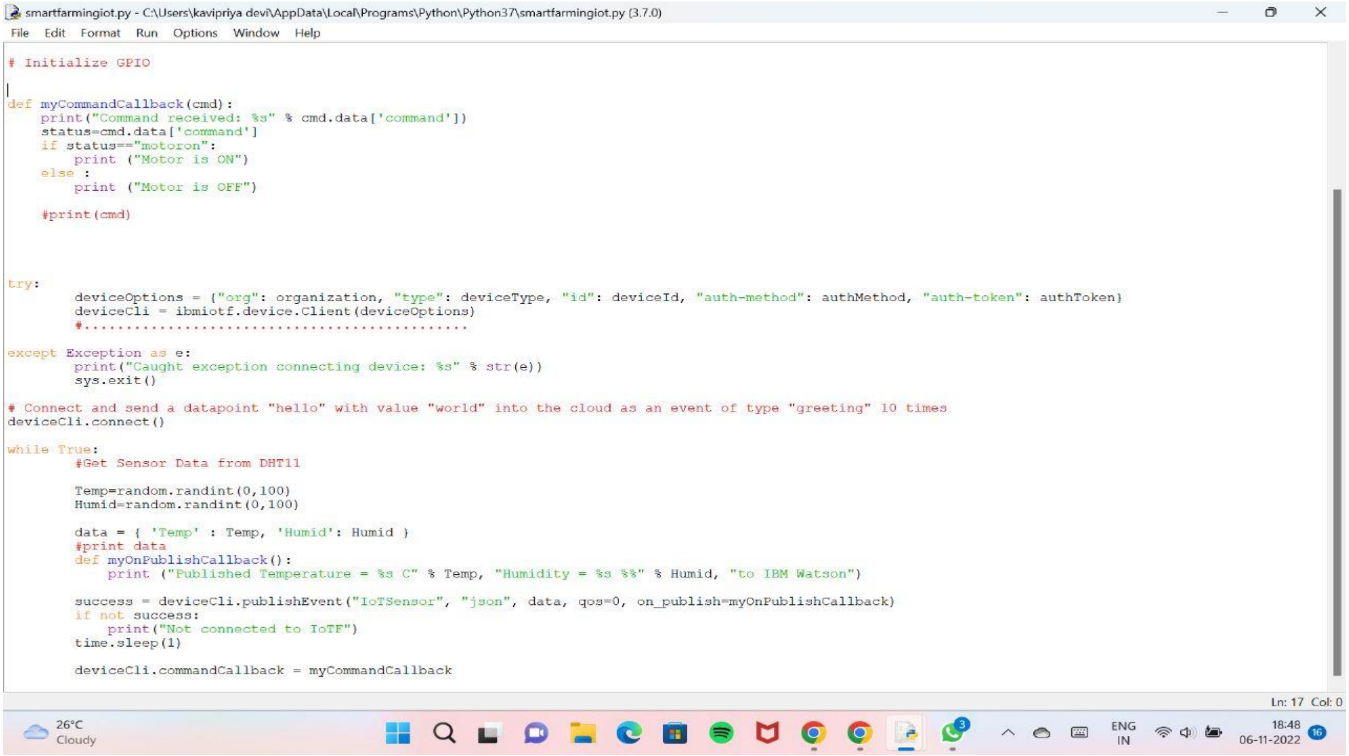


Code in Python IDLE:

PROGRAM:


smartfamingiot.py - C:\Users\kavipriya devi\AppData\Local\Programs\Python\Python37\smartfarmingiot.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 17 Col: 0
26°C
Cloudy
ENG
IN
18:48
06-11-2022

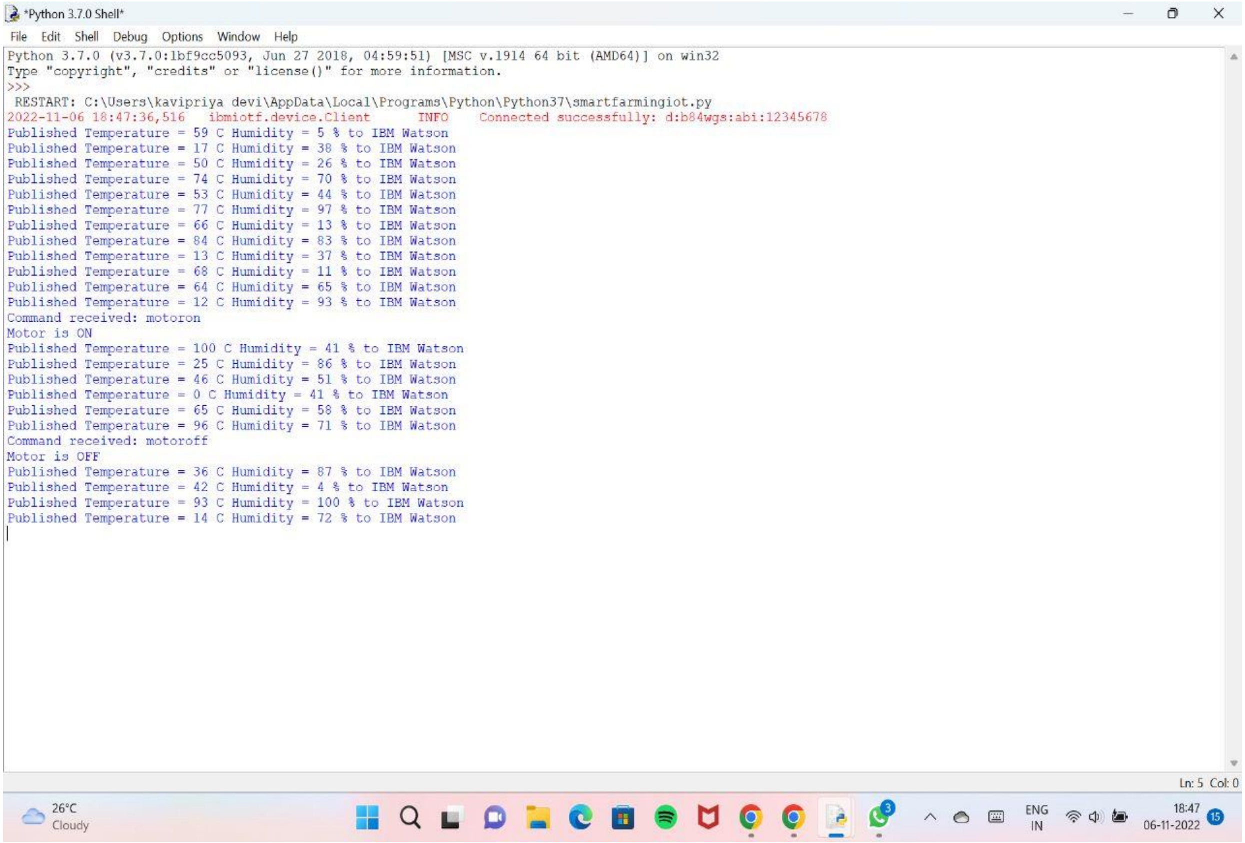
```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random


#Provide your IBM Watson Device Credentials
organization = "b84wgs"
deviceType = "abi"
deviceId = "12345678"
authMethod = "token"
authToken = "87654321"

# Initialize GPIO


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("Motor is ON")
    else :
        print ("Motor is OFF")

    #print(cmd)



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

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



smartfarmingiot.py - C:\Users\kavipriya devi\AppData\Local\Programs\Python\Python37\smartfarmingiot.py (3.7.0)

```
# Initialize GPIO

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="motoron":
        print ("Motor is ON")
    else :
        print ("Motor is OFF")

    #print(cmd)



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

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

        data = { 'Temp' : Temp, 'Humid': Humid }
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % Temp, "Humidity = %s %%" % Humid, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(1)

        deviceCli.commandCallback = myCommandCallback
```

OUTPUT:



Python 3.7.0 Shell

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\kavipriya devi\AppData\Local\Programs\Python\Python37\smartfarmingiot.py
2022-11-06 18:47:36,516   ibmiotf.device.Client      INFO    Connected successfully: d:b84wgs:abi:12345678
Published Temperature = 59 C Humidity = 5 % to IBM Watson
Published Temperature = 17 C Humidity = 38 % to IBM Watson
Published Temperature = 50 C Humidity = 26 % to IBM Watson
Published Temperature = 74 C Humidity = 70 % to IBM Watson
Published Temperature = 53 C Humidity = 44 % to IBM Watson
Published Temperature = 77 C Humidity = 97 % to IBM Watson
Published Temperature = 66 C Humidity = 13 % to IBM Watson
Published Temperature = 84 C Humidity = 83 % to IBM Watson
Published Temperature = 13 C Humidity = 37 % to IBM Watson
Published Temperature = 68 C Humidity = 11 % to IBM Watson
Published Temperature = 64 C Humidity = 65 % to IBM Watson
Published Temperature = 12 C Humidity = 93 % to IBM Watson
Command received: motoron
Motor is ON
Published Temperature = 100 C Humidity = 41 % to IBM Watson
Published Temperature = 25 C Humidity = 86 % to IBM Watson
Published Temperature = 46 C Humidity = 51 % to IBM Watson
Published Temperature = 0 C Humidity = 41 % to IBM Watson
Published Temperature = 65 C Humidity = 58 % to IBM Watson
Published Temperature = 96 C Humidity = 71 % to IBM Watson
Command received: motoroff
Motor is OFF
Published Temperature = 36 C Humidity = 87 % to IBM Watson
Published Temperature = 42 C Humidity = 4 % to IBM Watson
Published Temperature = 93 C Humidity = 100 % to IBM Watson
Published Temperature = 14 C Humidity = 72 % to IBM Watson
```