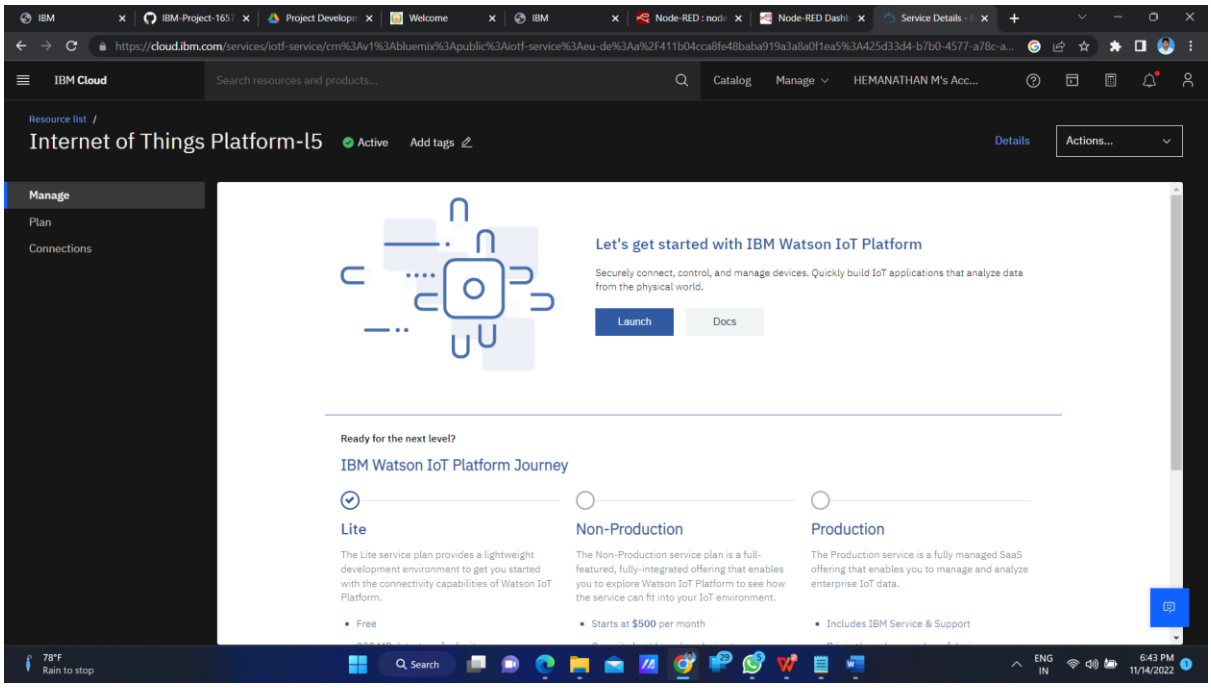
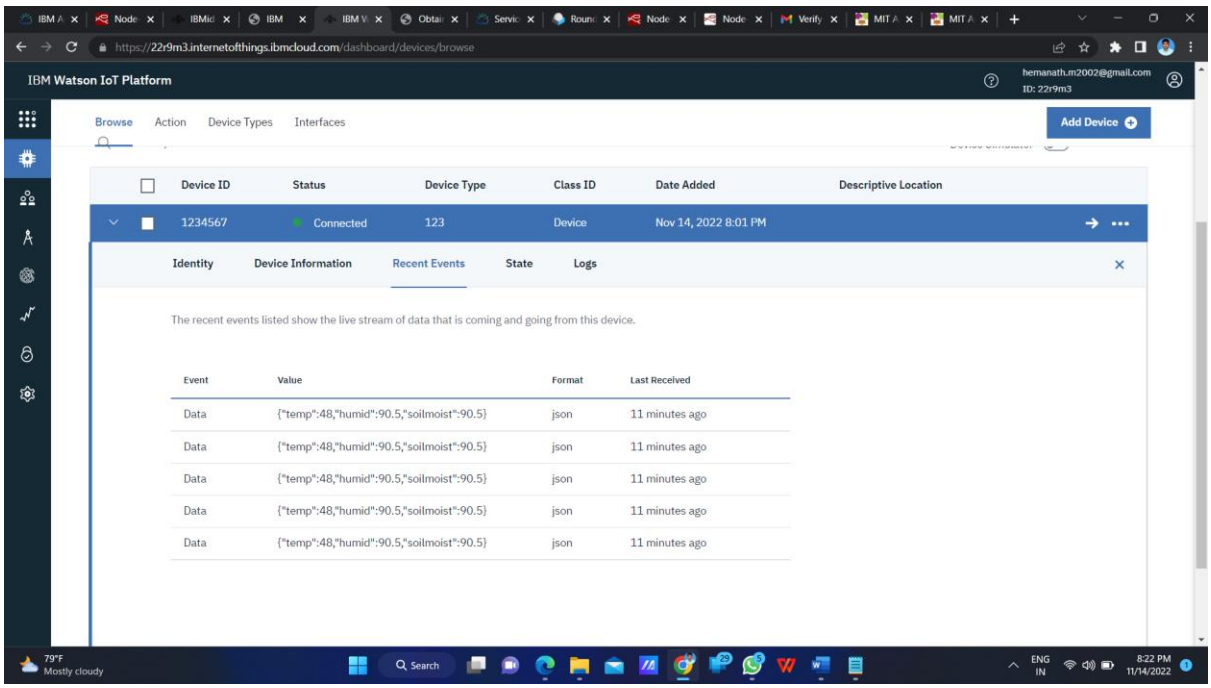
Sprint - 4
Connection between IOT Device & IBM Cloud

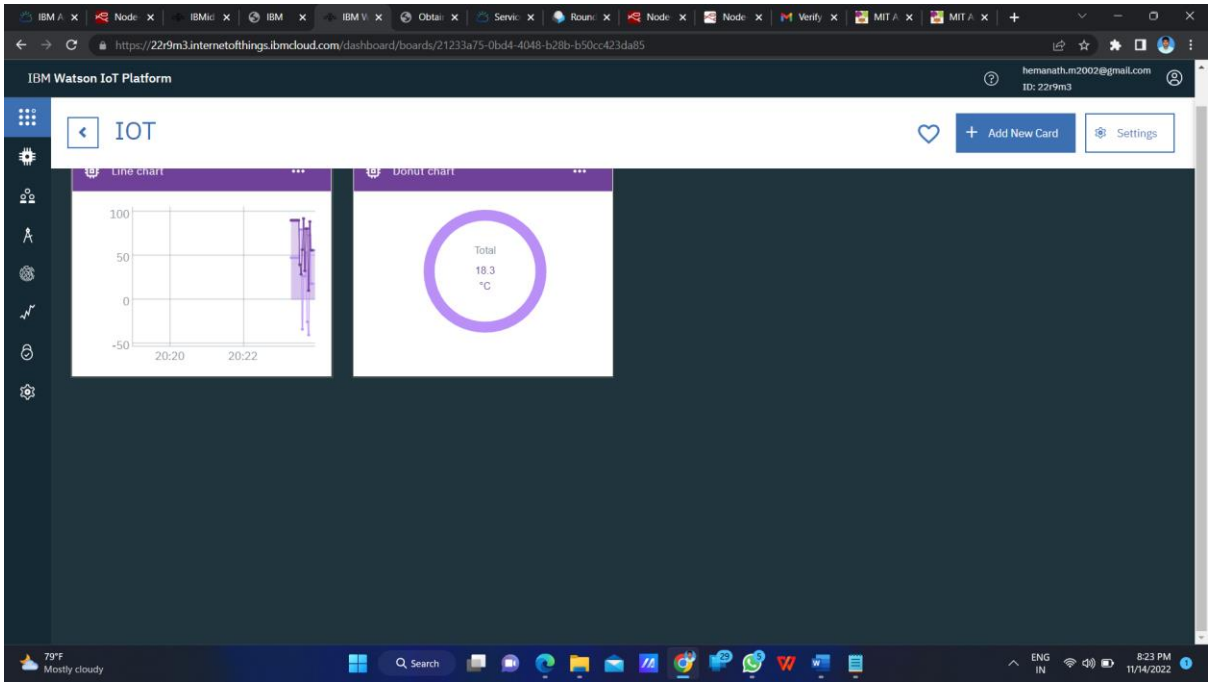
Team ID	PNT2022TMID37214
Project Name	Project - Smart Farmer – IOT Enabled Smart Farming Application

Cloud Configuration

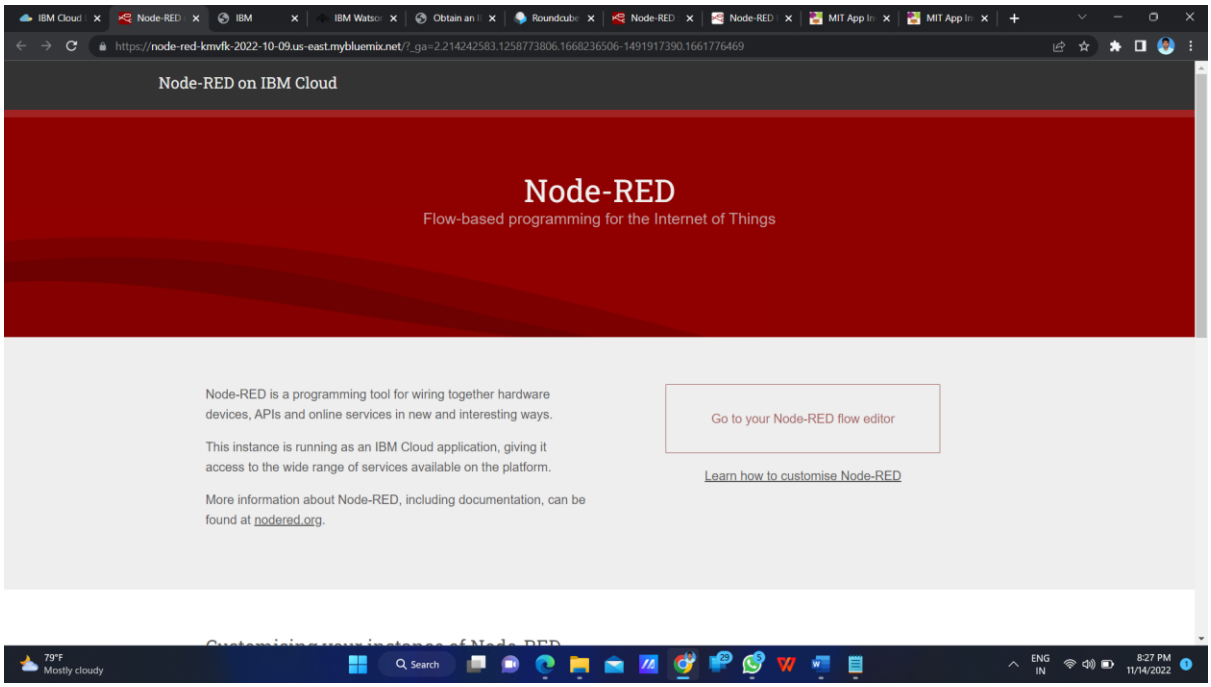
IOT WATSON PLATFORM

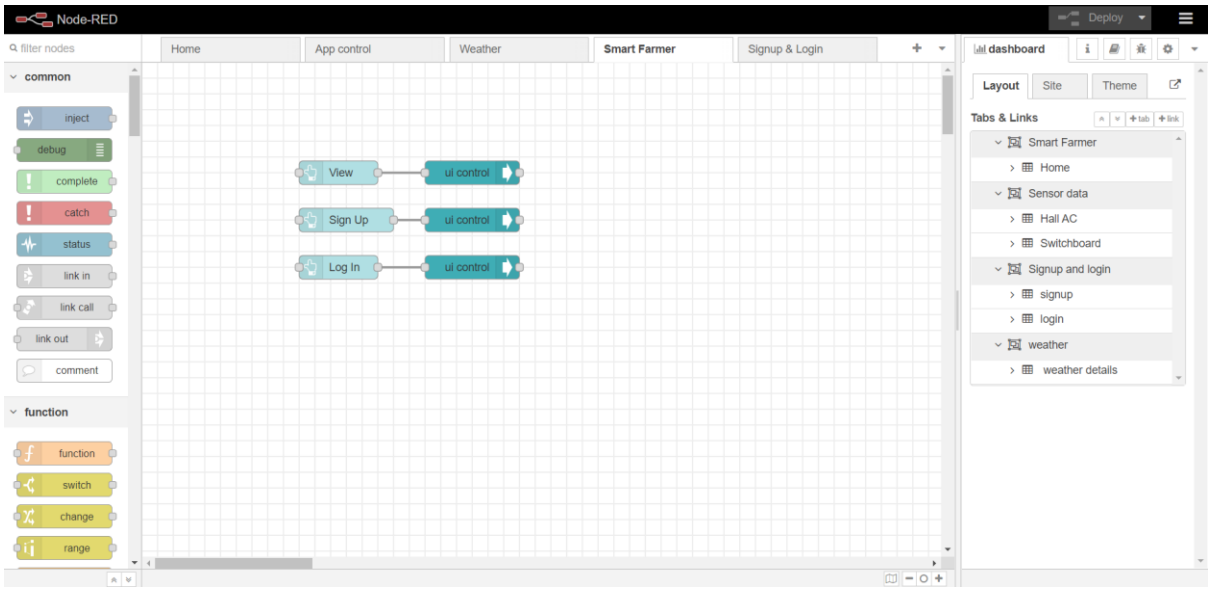


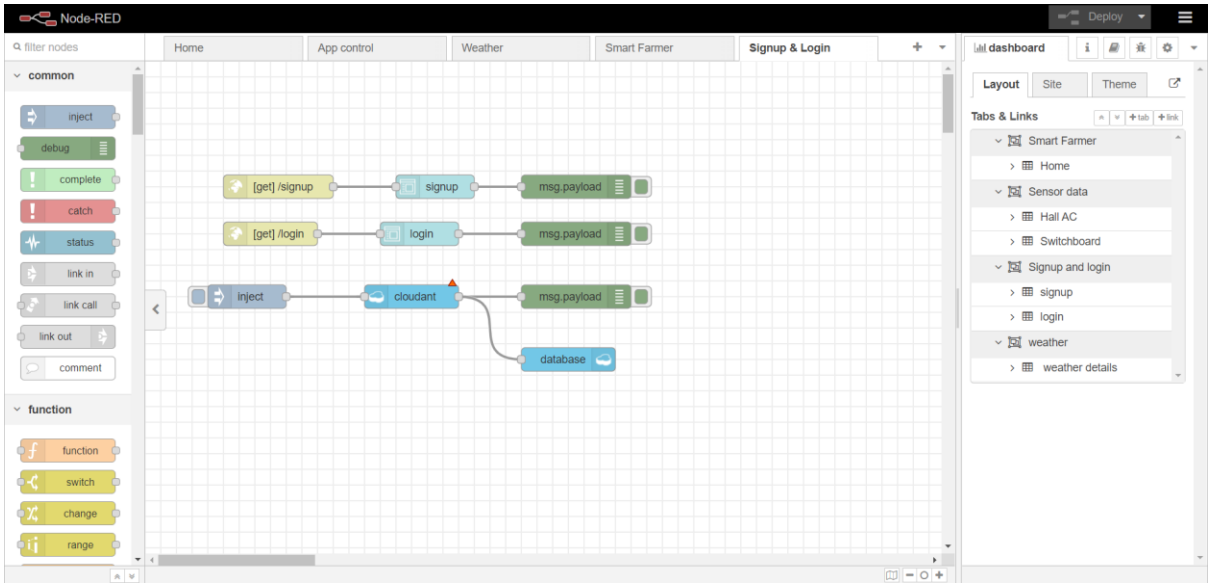


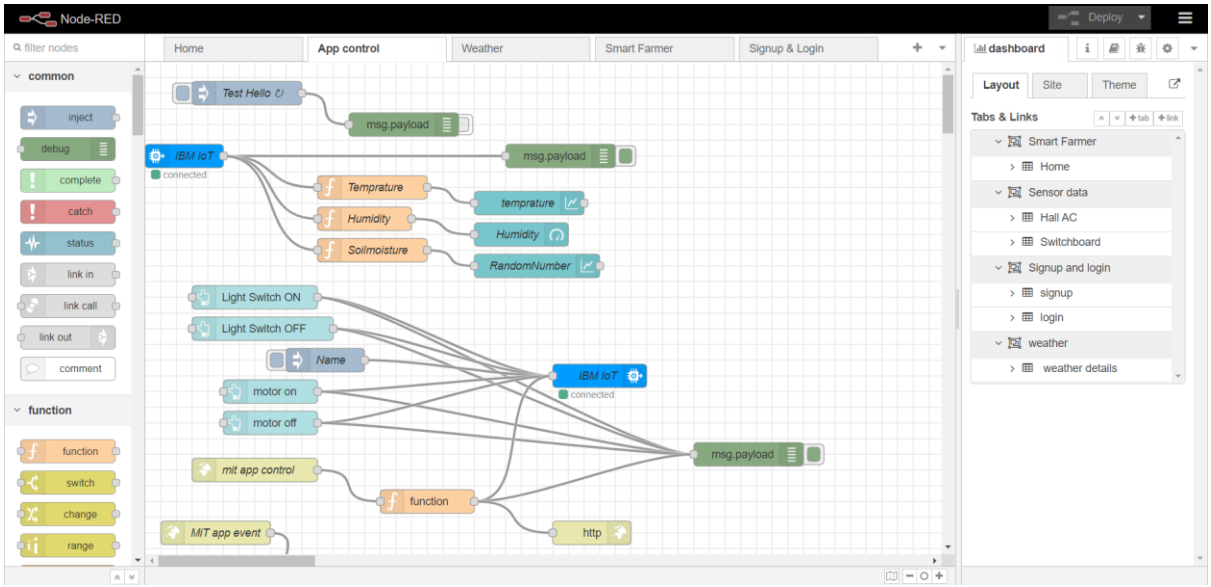

https://22r9m3.internetofthings.ibmcloud.com/dashboard/boards/21233a75-0bd4-4048-b28b-b50cc423da85
IBM Watson IoT Platform
hemanath.m2002@gmail.com
ID: 22r9m3
IOT
Add New Card
Settings
Line chart
Donut chart
100
50
0
-50
20:20
20:22
Total
18.3
°C
79°F
Mostly cloudy
Search
ENG
IN
8:23 PM
11/14/2022

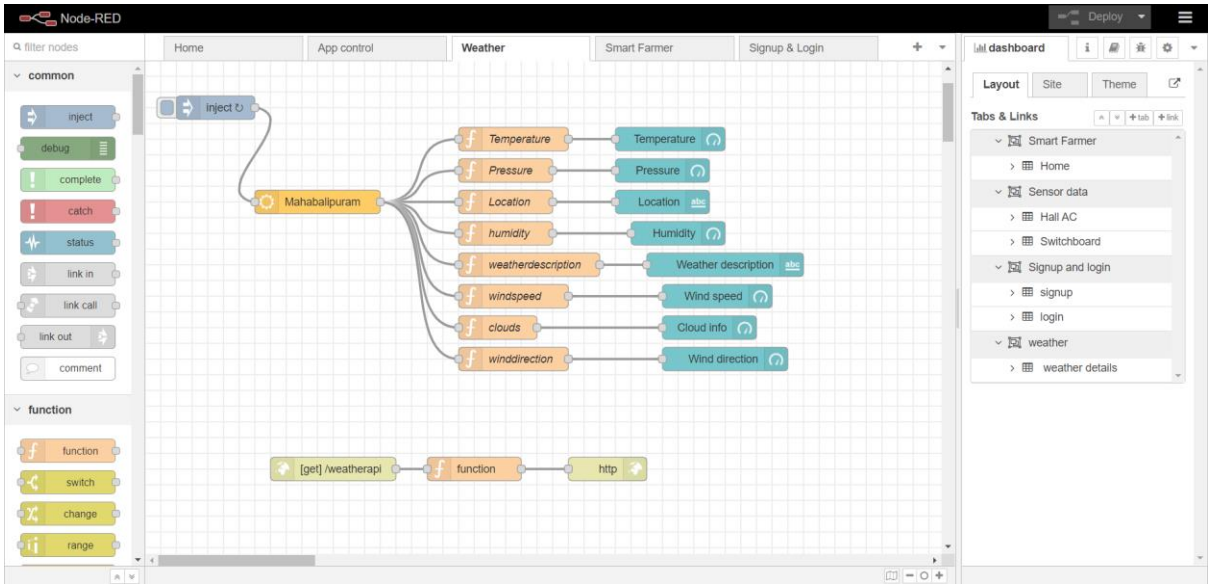
NodeRed

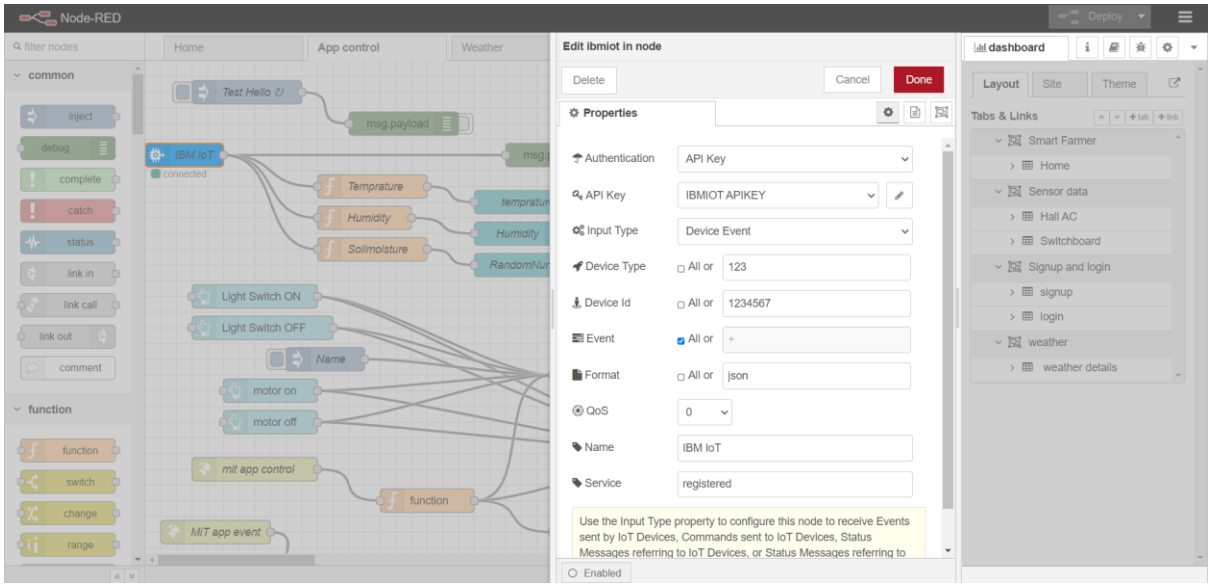

https://node-red-kmvfk-2022-10-09.us-east.mybluemix.net/?_ga=2.214242583.1258773806.1668236506-1491917390.1661776469
Node-RED on IBM Cloud
Node-RED
Flow-based programming for the Internet of Things
Node-RED is a programming tool for wiring together hardware devices, APIs and online services in new and 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
79°F
Mostly cloudy
Search
ENG
IN
8:27 PM
11/14/2022

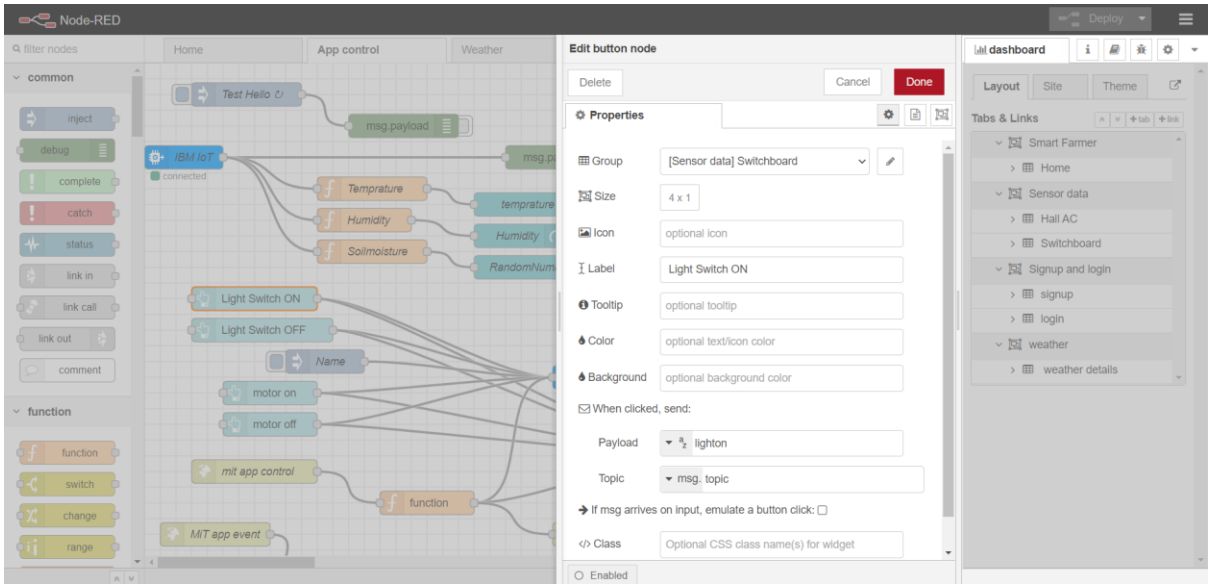

Node-RED
Deploy
filter nodes
Home
App control
Weather
Smart Farmer
Signup & Login
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
View
Sign Up
Log In
ui control
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details

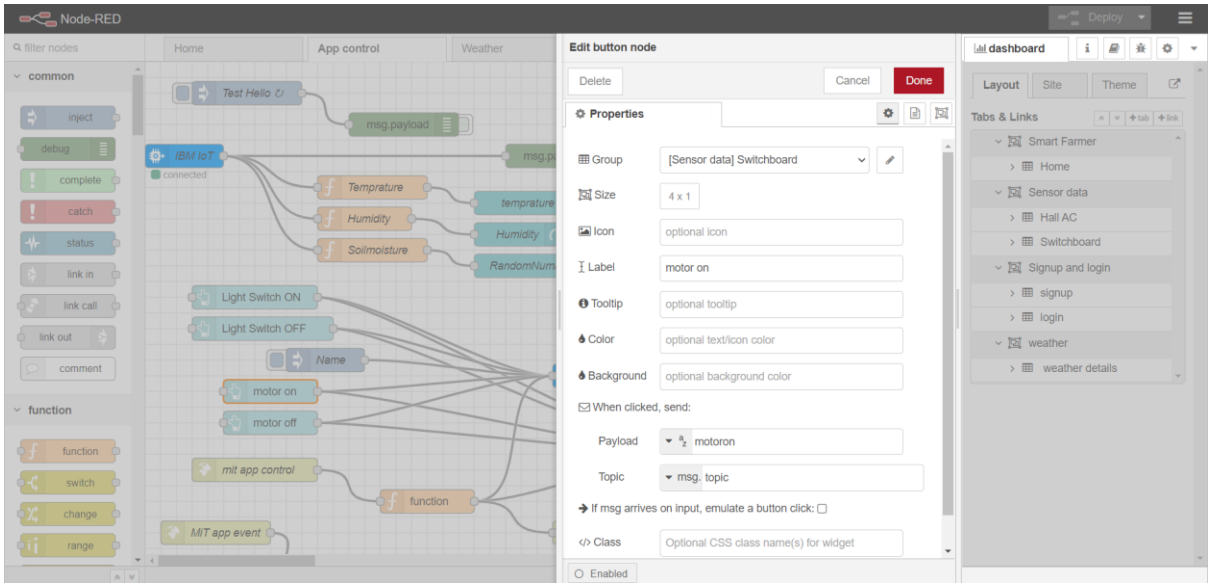

Node-RED
Deploy
filter nodes
Home
App control
Weather
Smart Farmer
Signup & Login
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
[get] /signup
signup
msg.payload
[get] /login
login
msg.payload
inject
cloudant
msg.payload
database
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details

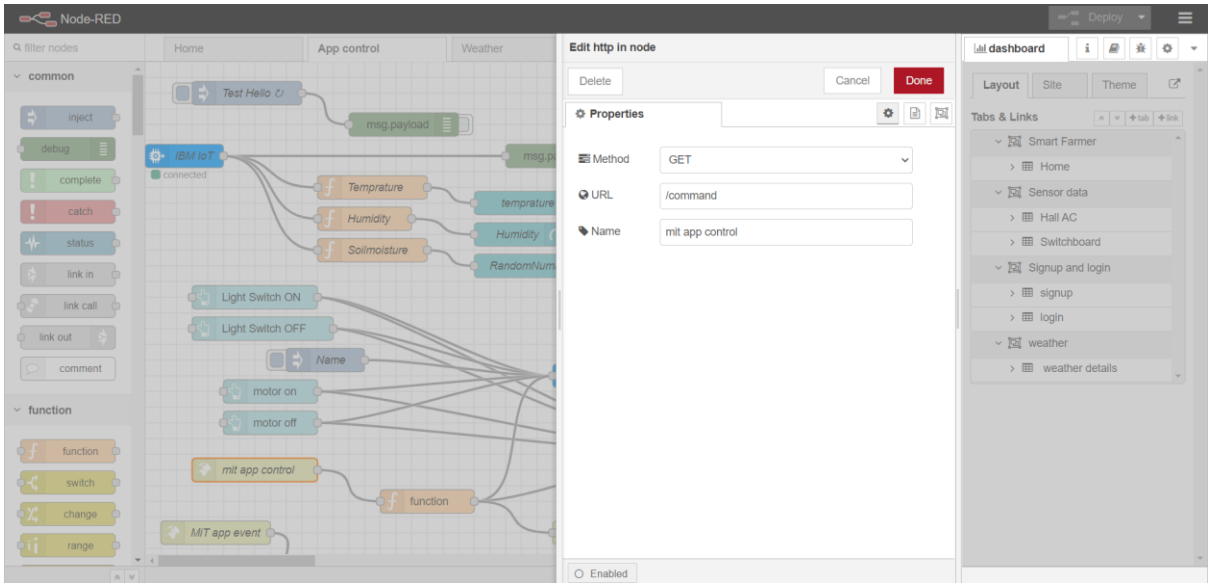

Node-RED
Deploy
filter nodes
Home
App control
Weather
Smart Farmer
Signup & Login
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
Test Hello
msg.payload
IBM IoT
connected
msg.payload
Temprature
Humidity
Soilmoisture
temprature
Humidity
RandomNumber
Light Switch ON
Light Switch OFF
Name
motor on
motor off
IBM IoT
connected
mit app control
function
msg.payload
MIT app event
http
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details

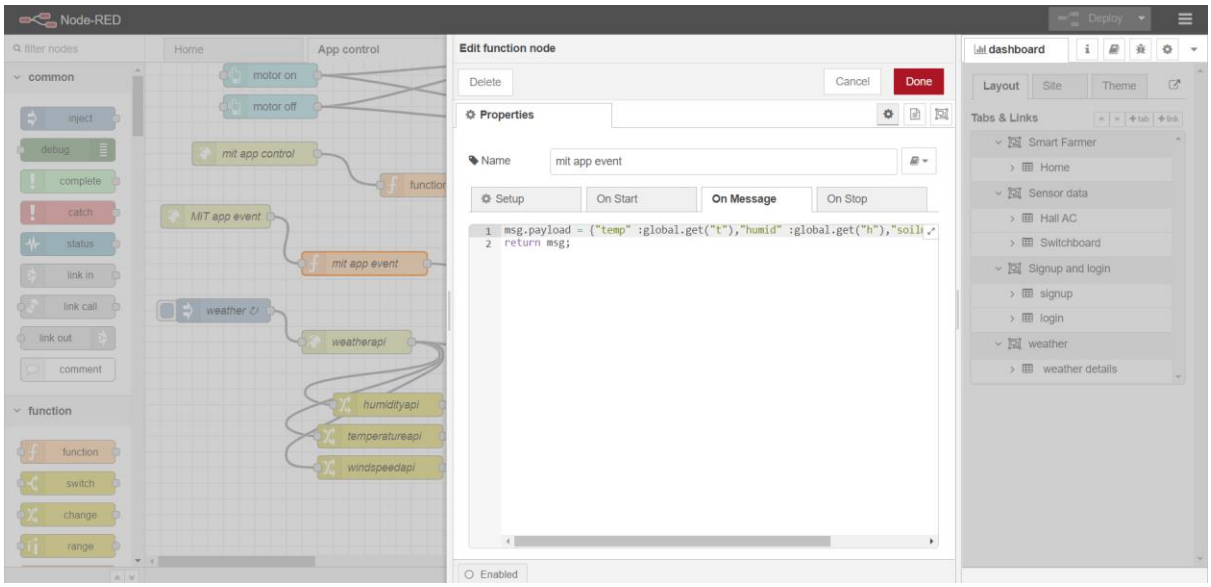

Node-RED
Deploy
filter nodes
Home
App control
Weather
Smart Farmer
Signup & Login
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
inject
Mahabalipuram
Temperature
Pressure
Location
humidity
weatherdescription
windspeed
clouds
winddirection
Temperature
Pressure
Location
Humidity
Weather description
Wind speed
Cloud info
Wind direction
[get] /weatherapi
function
http
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details

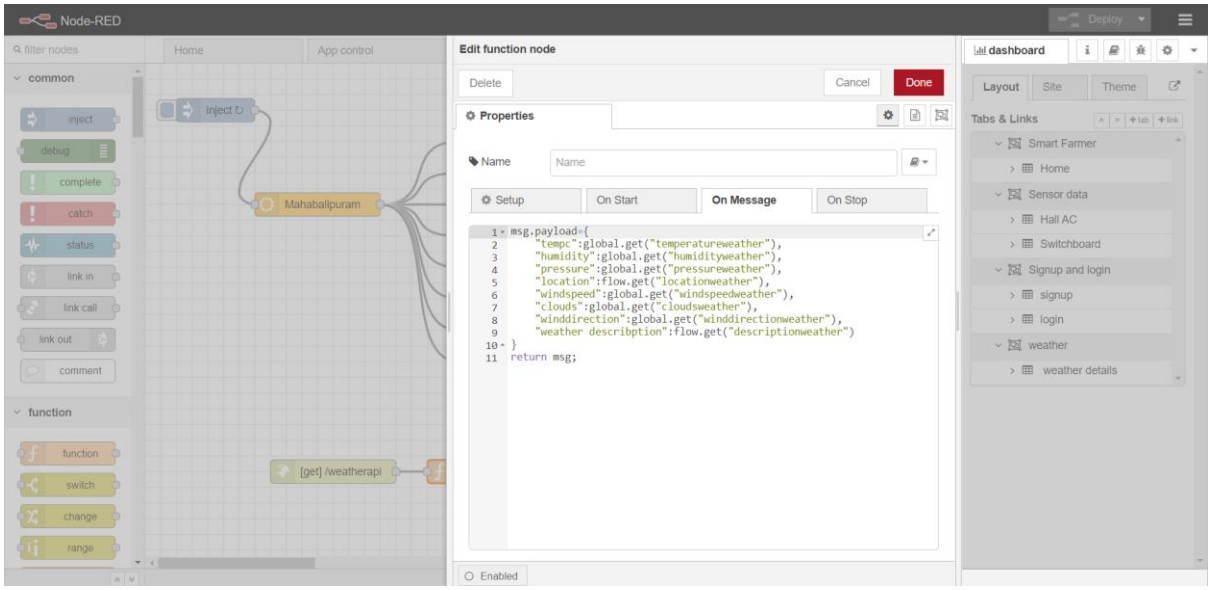

Node-RED
Deploy
Home
App control
Weather
Test Hello
msg.payload
IBM IoT
connected
Temprature
Humidity
Soilmoisture
Light Switch ON
Light Switch OFF
Name
motor on
motor off
mit app control
function
MIT app event
Edit ibmiot in node
Delete
Cancel
Done
Properties
Authentication
API Key
API Key
IBMIOT APIKEY
Input Type
Device Event
Device Type
All or
123
Device Id
All or
1234567
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
Enabled
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details

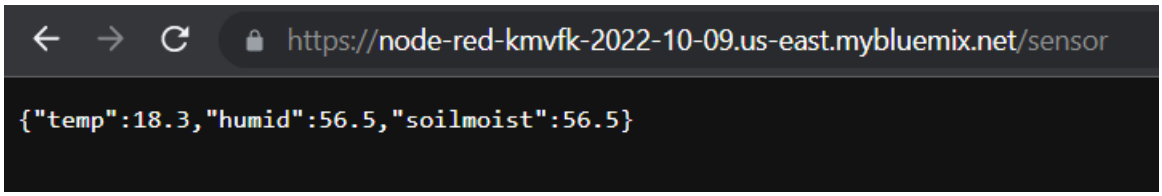

Node-RED
Deploy
Home
App control
Weather
Test Hello
msg.payload
IBM IoT
connected
Temprature
Humidity
Soilmoisture
Light Switch ON
Light Switch OFF
Name
motor on
motor off
mit app control
function
MIT app event
Edit button node
Delete
Cancel
Done
Properties
Group
[Sensor data] Switchboard
Size
4 x 1
Icon
optional icon
Label
Light Switch ON
Tooltip
optional tooltip
Color
optional text/icon color
Background
optional background color
When clicked, send:
Payload
lighton
Topic
msg.
topic
If msg arrives on input, emulate a button click:
Class
Optional CSS class name(s) for widget
Enabled
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details

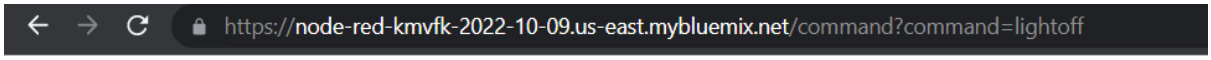

Node-RED
Deploy
filter nodes
Home
App control
Weather
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
Test Hello
msg.payload
IBM IoT
connected
Temprature
Humidity
Soilmoisture
Light Switch ON
Light Switch OFF
Name
motor on
motor off
mit app control
function
MIT app event
Edit button node
Delete
Cancel
Done
Properties
Group
[Sensor data] Switchboard
Size
4 x 1
Icon
optional icon
Label
motor on
Tooltip
optional tooltip
Color
optional text/icon color
Background
optional background color
When clicked, send:
Payload
motoron
Topic
msg. topic
If msg arrives on input, emulate a button click:
Class
Optional CSS class name(s) for widget
Enabled
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details


Node-RED
Deploy
filter nodes
Home
App control
Weather
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
Test Hello
msg.payload
IBM IoT
connected
Temprature
Humidity
Soilmoisture
Light Switch ON
Light Switch OFF
Name
motor on
motor off
mit app control
function
MIT app event
Edit http in node
Delete
Cancel
Done
Properties
Method
GET
URL
/command
Name
mit app control
Enabled
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details

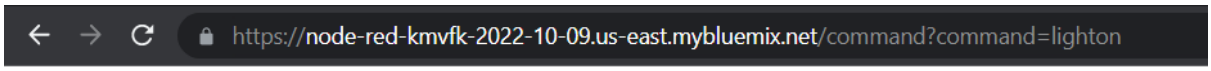

Node-RED
Deploy
filter nodes
Home
App control
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
motor on
motor off
mit app control
MIT app event
mit app event
weather
weatherapi
humidityapi
temperatureapi
windspeedapi
Edit function node
Delete
Cancel
Done
Properties
Name
mit app event
Setup
On Start
On Message
On Stop
1 msg.payload = {"temp" :global.get("t"),"humid" :global.get("h"),"soili
2 return msg;
Enabled
dashboard
Layout
Site
Theme
Tabs & Links
Smart Farmer
Home
Sensor data
Hall AC
Switchboard
Signup and login
signup
login
weather
weather details



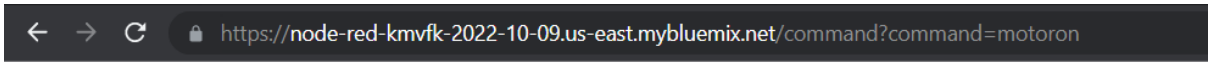




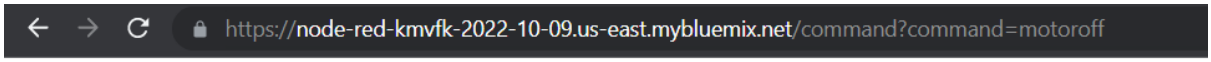
lightoff



lighton

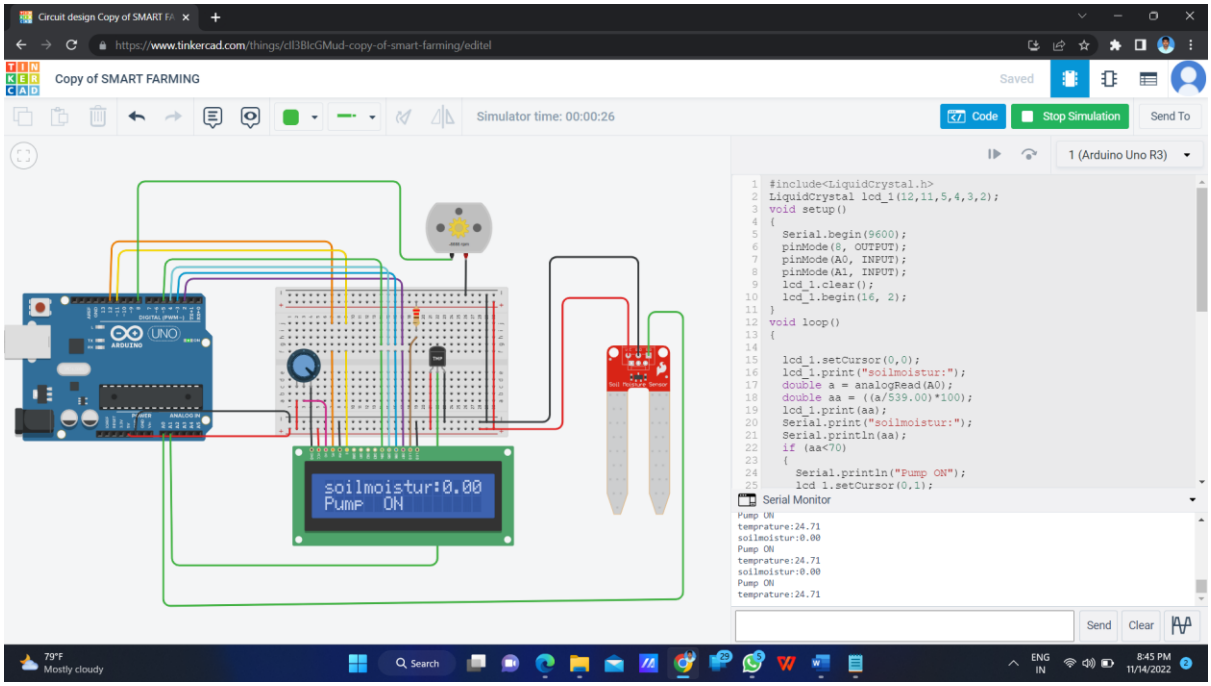


motoron



motoroff

IOT Device



Wokwi Simulator

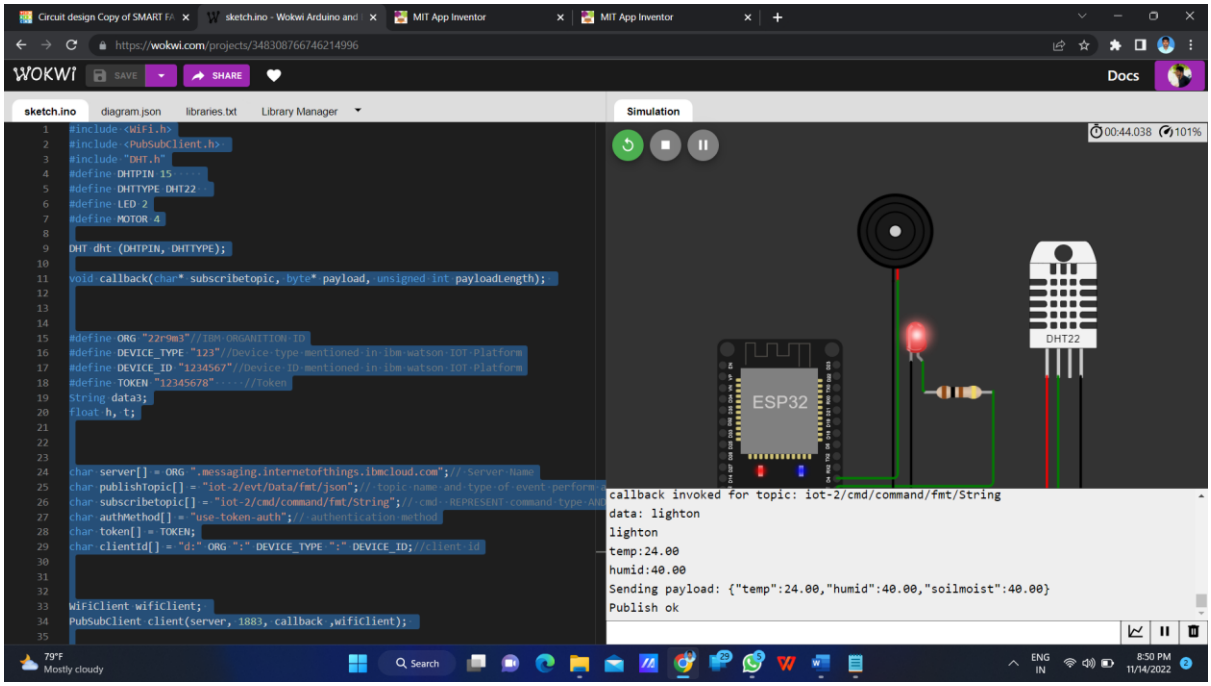
Code:

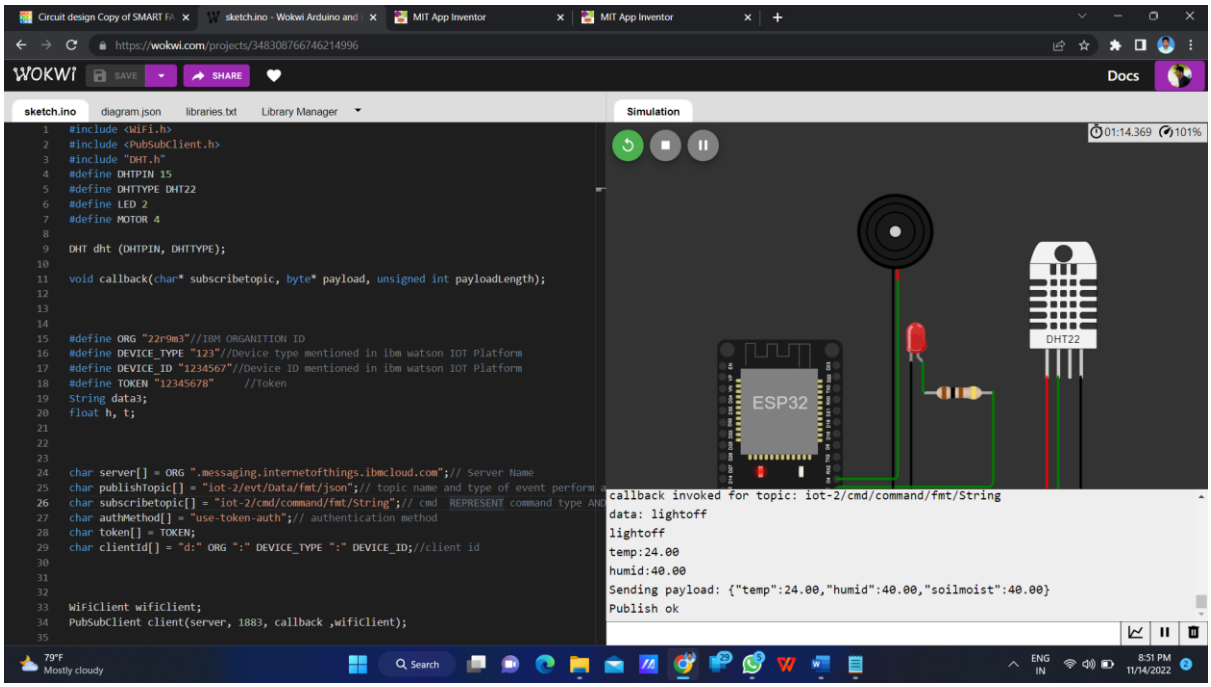
```
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTPIN 15
#define DHTTYPE DHT22
#define LED 2
#define MOTOR 4

DHT dht (DHTPIN, DHTTYPE);

void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);



#define ORG "22r9m3"//IBM ORGANITION ID
#define DEVICE_TYPE "123"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234567"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float h, t;
```

```
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
event perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT
command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id



WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);


void setup()
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED,OUTPUT);
  pinMode(MOTOR,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()
{

  h = dht.readHumidity();
  t = dht.readTemperature();
  Serial.print("temp:");
  Serial.println(t);
  Serial.print("humid:");
  Serial.println(h);

  PublishData(t, h);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}




void PublishData(float temp, float humid) {
  mqttconnect();
```

```
  String payload = "{\"temp\":";
  payload += temp;
  payload += "," "\"humid\":";
  payload += humid;
  payload += "," "\"soilmoist\":";
  payload += humid;
  payload += "}";


  Serial.print("Sending payload: ");
  Serial.println(payload);


  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");
  } else {
    Serial.println("Publish failed");
  }

}


void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }

     initManagedDevice();
     Serial.println();
  }
}
void wificonnect()
{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

```
void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{

  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  for (int i = 0; i < payloadLength; i++) {
    data3 += (char)payload[i];
  }
  Serial.println("data: "+ data3);
  if(data3=="lighton")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);
  }
  else if(data3=="motoron")
  {
Serial.println(data3);
digitalWrite(MOTOR,HIGH);
  }
  else if(data3=="motoroff")
  {
Serial.println(data3);
digitalWrite(MOTOR,LOW);
  }
  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);
  }
data3="";
}
```


Circuit design Copy of SMART FA
sketch.ino - Wokwi Arduino and
MIT App Inventor
MIT App Inventor
https://wokwi.com/projects/348308766746214996
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTPIN 15
#define DHTTYPE DHT22
#define LED 2
#define MOTOR 4
DHT dht (DHTPIN, DHTTYPE);
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "22r9m3"//IBM ORGANITION ID
#define DEVICE_TYPE "123"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234567"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678" //Token
String data3;
float h, t;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";//Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";//topic name and type of event perform
char subscribetopic[] = "iot-2/cmd/command/fmt/String";//cmd REPRESENT command type AN
char authMethod[] = "use-token-auth";//authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
Simulation
00:44.038
101%
ESP32
DHT22
callback invoked for topic: iot-2/cmd/command/fmt/String
data: lighton
lighton
temp:24.00
humid:40.00
Sending payload: {"temp":24.00,"humid":40.00,"soilmoist":40.00}
Publish ok
79°F
Mostly cloudy
Search
ENG
IN
8:50 PM
11/14/2022


Circuit design Copy of SMART FA
sketch.ino - Wokwi Arduino and
MIT App Inventor
MIT App Inventor
https://wokwi.com/projects/348308766746214996
WOKWI
SAVE
SHARE
Docs
sketch.ino
diagram.json
libraries.txt
Library Manager
#include <WiFi.h>
#include <PubSubClient.h>
#include "DHT.h"
#define DHTPIN 15
#define DHTTYPE DHT22
#define LED 2
#define MOTOR 4
DHT dht (DHTPIN, DHTTYPE);
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
#define ORG "22r9m3"//IBM ORGANITION ID
#define DEVICE_TYPE "123"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234567"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678" //Token
String data3;
float h, t;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command type AN
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
Simulation
01:14.369
101%
ESP32
DHT22
callback invoked for topic: iot-2/cmd/command/fmt/String
data: lightoff
lightoff
temp:24.00
humid:40.00
Sending payload: {"temp":24.00,"humid":40.00,"soilmoist":40.00}
Publish ok
79°F
Mostly cloudy
Search
ENG
IN
8:51 PM
11/14/2022

Python Script

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


#Provide your IBM Watson Device Credentials
organization = "22r9m3"
deviceType = "123"
deviceId = "1234567"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO


def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="lighton":
        print ("led is on")
    elif status == "lightoff":
        print("led is off")
    elif status == "motoron":
        print("motor is on")
    elif status == "motoroff":
        print("motor is off")
    else :
        print ("please send proper command")

    #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)
        soilmoist=random.randint(0,100)

        data = { 'temp' : temp, 'humid': humid, 'soilmoist': soilmoist  }
        #print data
        def myOnPublishCallback():
            print ("Published Temperature = %s C" % temp, "Humidity = %s %%" %
humid,"Soilmoisture = %s %%" % soilmoist, "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoTF")
        time.sleep(10)

        deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud
deviceCli.disconnect()
```