Smart Farmer-IOT Enabled Smart FarmingApplication

Project Development Phase

Sprint Delivery -1

Introduction:

The main aim of this project is to help farmers automate their farms by providing them with a Web App through which they can monitor the parameters of the field like Temperature,soil moisture, humidity etc and control the equipment like water motor and other devices remotely via the internet without their actual presence in the field.

Problem Statement:

Farmers need to deal with many problems like coping with climate change, soil erosion and Biodiversity loss. Farmers are to be present at farm for its maintenance irrespective of the weather conditions.They have to ensure that the crops are well watered and the farm status is monitored by them physically.Farmers have to stay most of the time in field in order to get a good yield.In difficult times like in the presence of pandemic also they have to work hard in their fields risking their lives to provide food for the country.

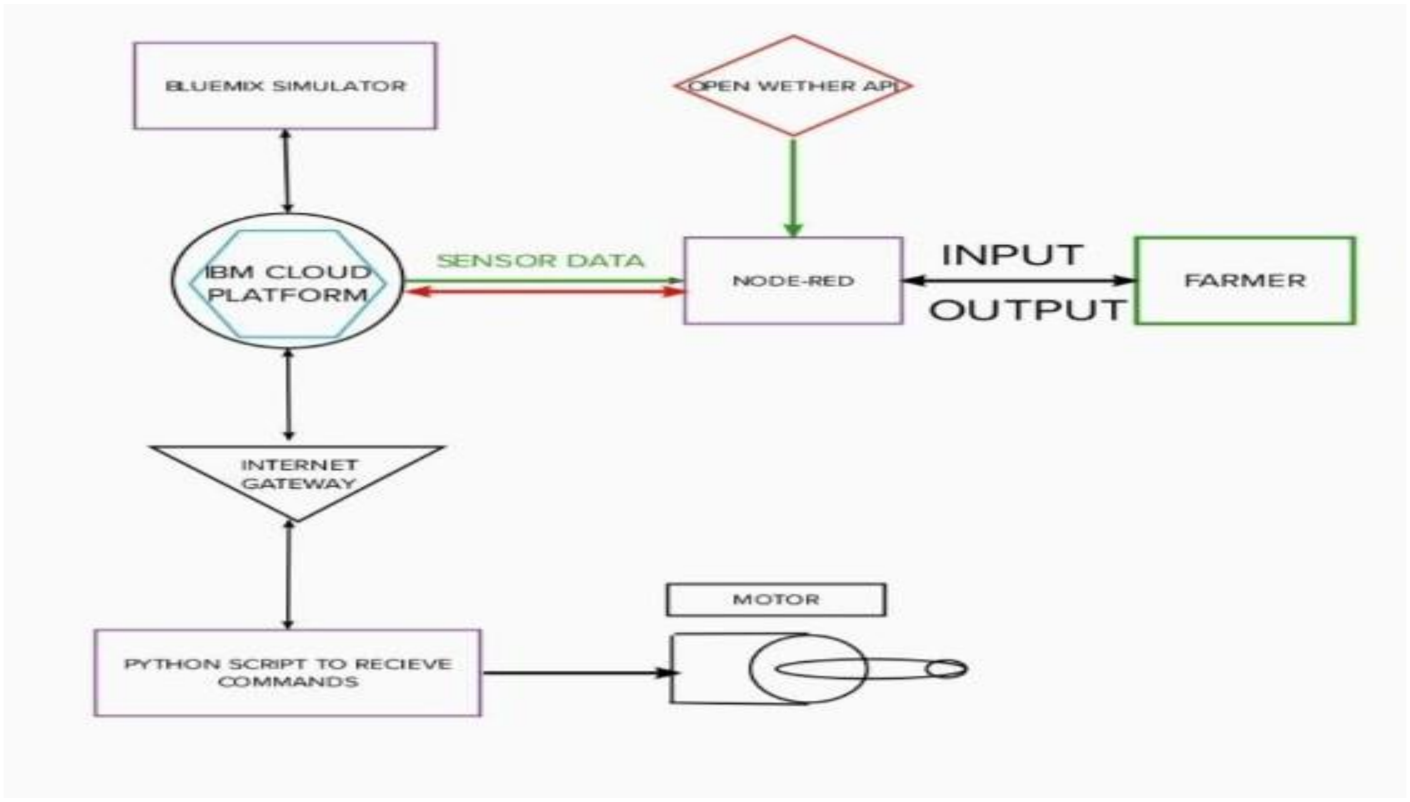
Proposed Solution:

To provide an efficient decision support system using wireless sensor networks which handle different activities of the farm and give useful information related to soil moisture, Temperature and Humidity content. Due to the weather condition, water level increases, Farmers get a lot of distractions which is not good for Agriculture.

Theoretical Analysis:

Block Diagram:

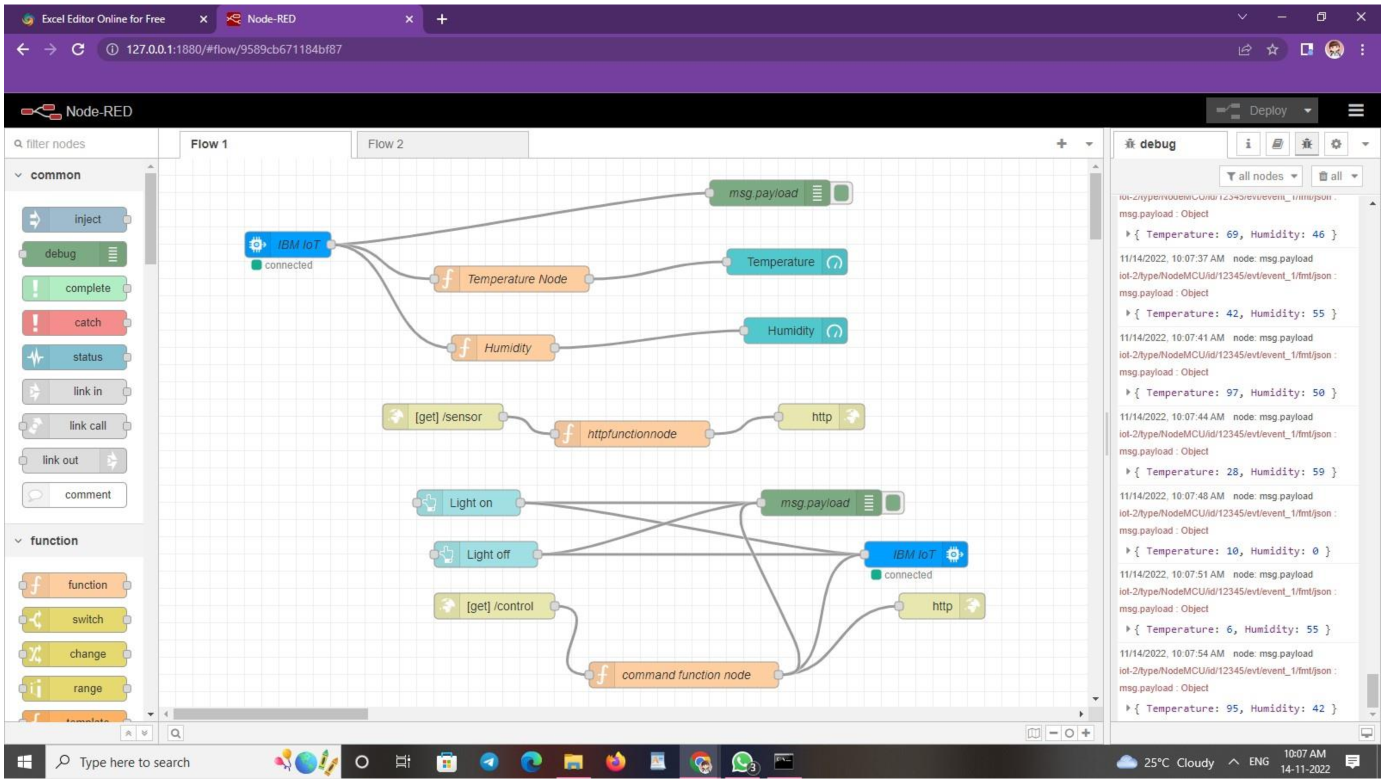
In order to implement the solution, the following approach asshown in the block diagram is used



Required Software Installation:

Node-Red:

Node-RED is a flow-based development tool for visualprogramming developed originally by IBM for wiring togetherhardware devices, APIs and online services as part of the Internetof Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.



```
msg.payload : Object
{ Temperature: 69, Humidity: 46 }
11/14/2022, 10:07:37 AM   node: msg.payload
iot-2/type/NodeMCU/id/12345/evt/event_1/fmt/json :
msg.payload : Object
{ Temperature: 42, Humidity: 55 }
11/14/2022, 10:07:41 AM   node: msg.payload
iot-2/type/NodeMCU/id/12345/evt/event_1/fmt/json :
msg.payload : Object
{ Temperature: 97, Humidity: 50 }
11/14/2022, 10:07:44 AM   node: msg.payload
iot-2/type/NodeMCU/id/12345/evt/event_1/fmt/json :
msg.payload : Object
{ Temperature: 28, Humidity: 59 }
11/14/2022, 10:07:48 AM   node: msg.payload
iot-2/type/NodeMCU/id/12345/evt/event_1/fmt/json :
msg.payload : Object
{ Temperature: 10, Humidity: 0 }
11/14/2022, 10:07:51 AM   node: msg.payload
iot-2/type/NodeMCU/id/12345/evt/event_1/fmt/json :
msg.payload : Object
{ Temperature: 6, Humidity: 55 }
11/14/2022, 10:07:54 AM   node: msg.payload
iot-2/type/NodeMCU/id/12345/evt/event_1/fmt/json :
msg.payload : Object
{ Temperature: 95, Humidity: 42 }
```

Installation :

- First install npm/node.js.
- Open cmd prompt.
- Type => npm install node-red.

To run the application :

- Open cmd prompt.
- Type=>Node-RED.
- Then open http://127.0.0.1:1880/ on your browser.

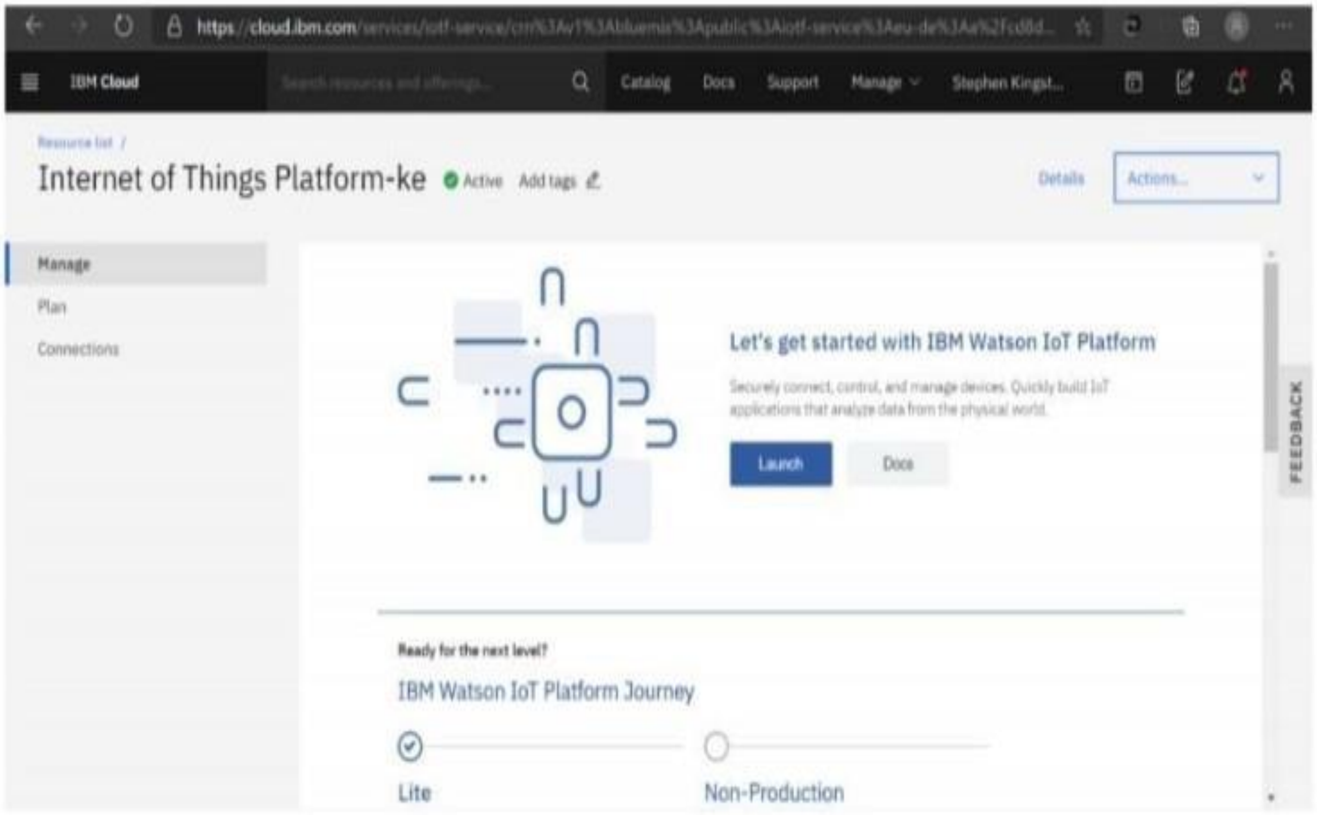
Installation of IBM IoT and Dashboard nodes for NodeRED:

In order to connect to IBM Watson IoT platform and create theWEB UI these nodes are required.

- IBM IoT node.
- Dashboard node.

IBM Watson IoT Platform:

A fully managed, cloud-hosted service with capabilities for deviceregistration,connectivity,control, rapid visualization and data storage.IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices



https://cloud.ibm.com/services/iotf-service/crn%3Av1%3Abluemix%3Apublic%3Aiotf-service%3Aeu-de%3Aa%2Fcd8d...
IBM Cloud
Search resources and offerings...
Catalog
Docs
Support
Manage
Stephen Kingst...
Resource list /
Internet of Things Platform-ke
Active
Add tags
Details
Actions...
Manage
Plan
Connections
Launch
Docs
FEEDBACK

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.

Ready for the next level?

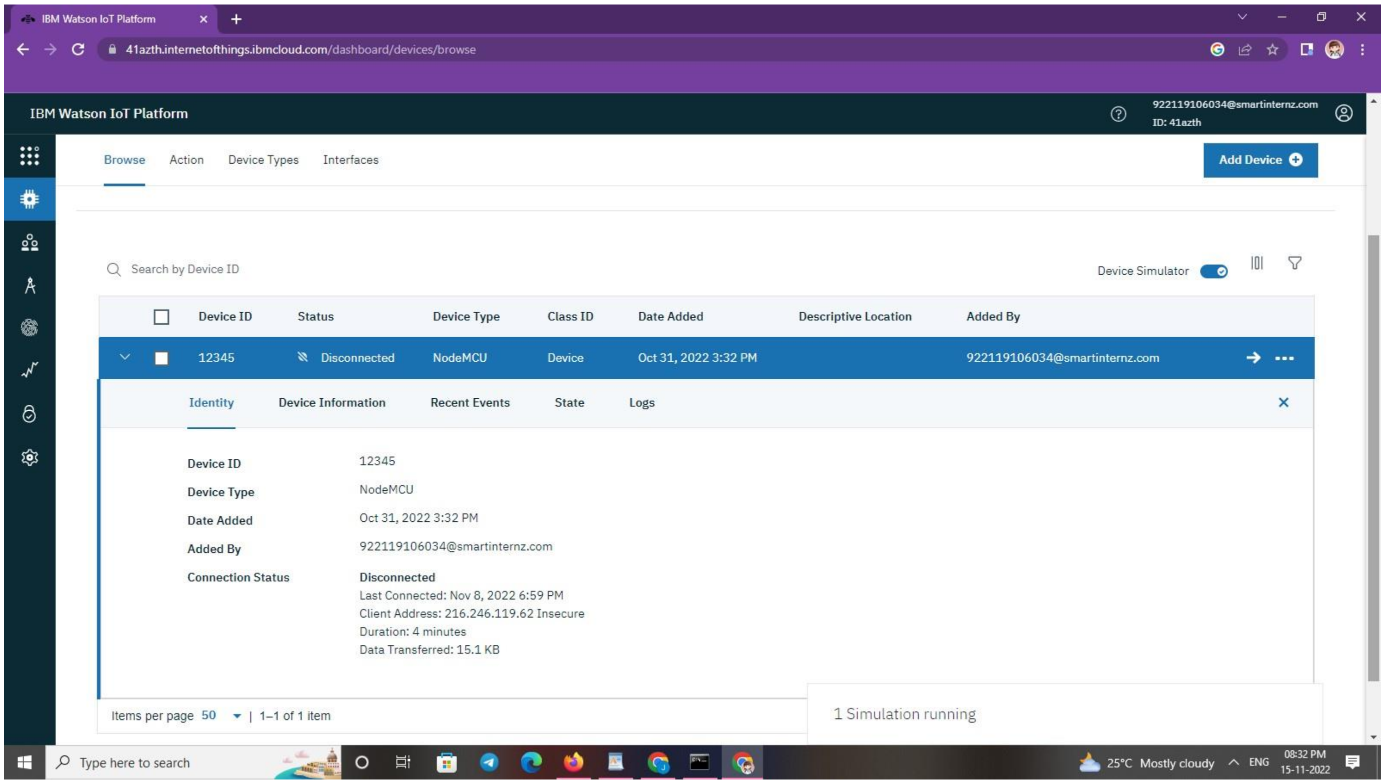
IBM Watson IoT Platform Journey

Lite

Non-Production

Steps to configure:

- Create an account in IBM cloud using your email ID
- Create IBM Watson Platform in services in your IBM cloudAccount.
- Launch the IBM Watson IoT Platform.
- Create a new device
- Give credentials like device type, device ID, Auth. Token
- Create API key and store API key and token elsewhere.



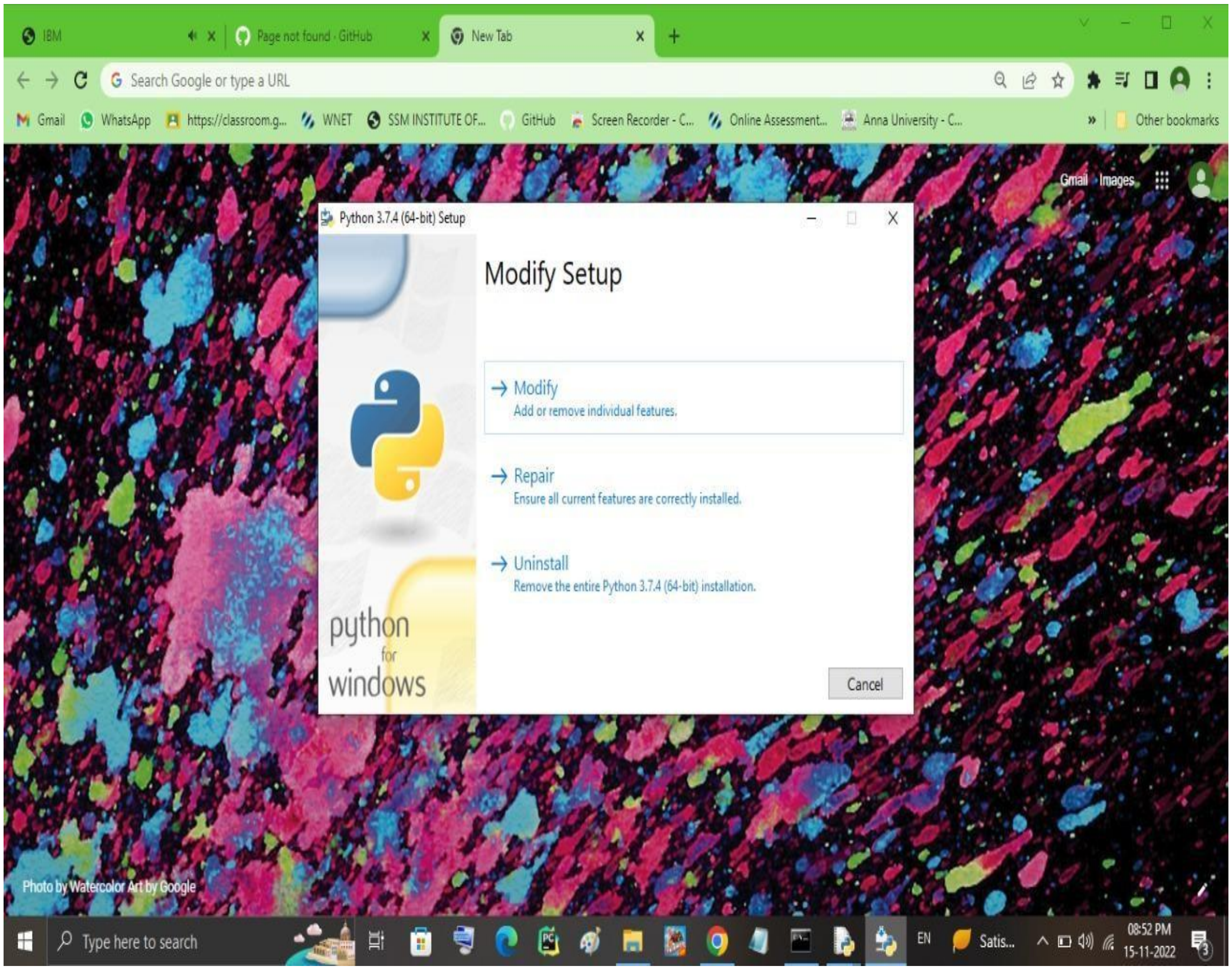
Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location	Added By
12345	Disconnected	NodeMCU	Device	Oct 31, 2022 3:32 PM		922119106034@smartinternz.com

Device ID	12345
Device Type	NodeMCU
Date Added	Oct 31, 2022 3:32 PM
Added By	922119106034@smartinternz.com
Connection Status	Disconnected Last Connected: Nov 8, 2022 6:59 PM Client Address: 216.246.119.62 Insecure Duration: 4 minutes Data Transferred: 15.1 KB

Python IDE:

Install Python3 compiler

Install any python IDE to execute python scripts, in my case I usedCommand Prompt to execute.


Python 3.7.4 (64-bit) Setup
Modify Setup
Modify
Add or remove individual features.
Repair
Ensure all current features are correctly installed.
Uninstall
Remove the entire Python 3.7.4 (64-bit) installation.
Cancel
python for windows

Code:

```
import wiotp.sdk.deviceimport time
import os import
datetimeimport random
myConfig = { "identity":
{
"orgId": "u9qhfi", "typeId":
"Devicetypel","deviceId":
"DeviceID1"
},
"auth": {
"token": ")hSb7_ZD+evl2fRhXi"
} }
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 (-20, 125)
            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 (2)
client.commandCallback = myCommandCallbackclient.disconnect ()
```

Arduino code for C :

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h>//library for MQtt#include "DHT.h"// Library for
dht11
#define DHTPIN 15                    // what pin we're connected to #define DHTTYPE DHT22
                                     // define type of sensor DHT 11#define LED 2
DHT dht (DHTPIN, DHTTYPE);// creating the instance by passingand type of dht connected

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

//-------credentials of IBM Accounts------

#define ORG "u9qhfi"//IBM ORGANIZATION ID
#define DEVICE_TYPE "Devicetype1"//Device type mentioned inibm watson IOT Platform
#define DEVICE_ID "DeviceID1"//Device ID mentioned in ibm
```

watson IOT Platform

```
#define TOKEN ")hSb7_ZD+ev12fRhXi"                              //Token
String data3;
float h, t;

//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and typeof 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 TESTOF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication methodchar token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id

//______________________________________________

WiFiClient wifiClient; // creating the instance for wificlientPubSubClient client(server, 1883, callback
,wifiClient);
//calling the predefined client id by passing parameter like
```

```
server id,portand wificredential
void setup()// configureing the ESP32
{

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




void loop()// Recursive Function
{

   h = dht.readHumidity();
   t = dht.readTemperature();
   Serial.print("temperature:");Serial.println(t);
   Serial.print("Humidity:");
```

```
  Serial.println(h);

  PublishData(t, h);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}
/*....................................retrieving to Cloud ............................................................................... */

void PublishData(float temp, float humid) { mqttconnect();//function call for connecting to ibm
  /*

     creating the String in in form JSon to update the data to ibm cloud
  */

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

```
Serial.print("Sending payload: ");
  Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str())) { Serial.println("Publish  ok");//  if  it  sucessfully
      upload  data  on  the cloud then it will print publish ok in Serial monitor or else it will printpublish failed
  } 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() //function defination for wificonnect
{

      Serial.println(); Serial.print("Connecting to ");

       WiFi.begin("Wokwi-GUEST",  "",  6);//passing  the  wifi  credentials  toestablish the connection
    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++) {
    //Serial.print((char)payload[i]);data3 +=
    (char)payload[i];
  }
```

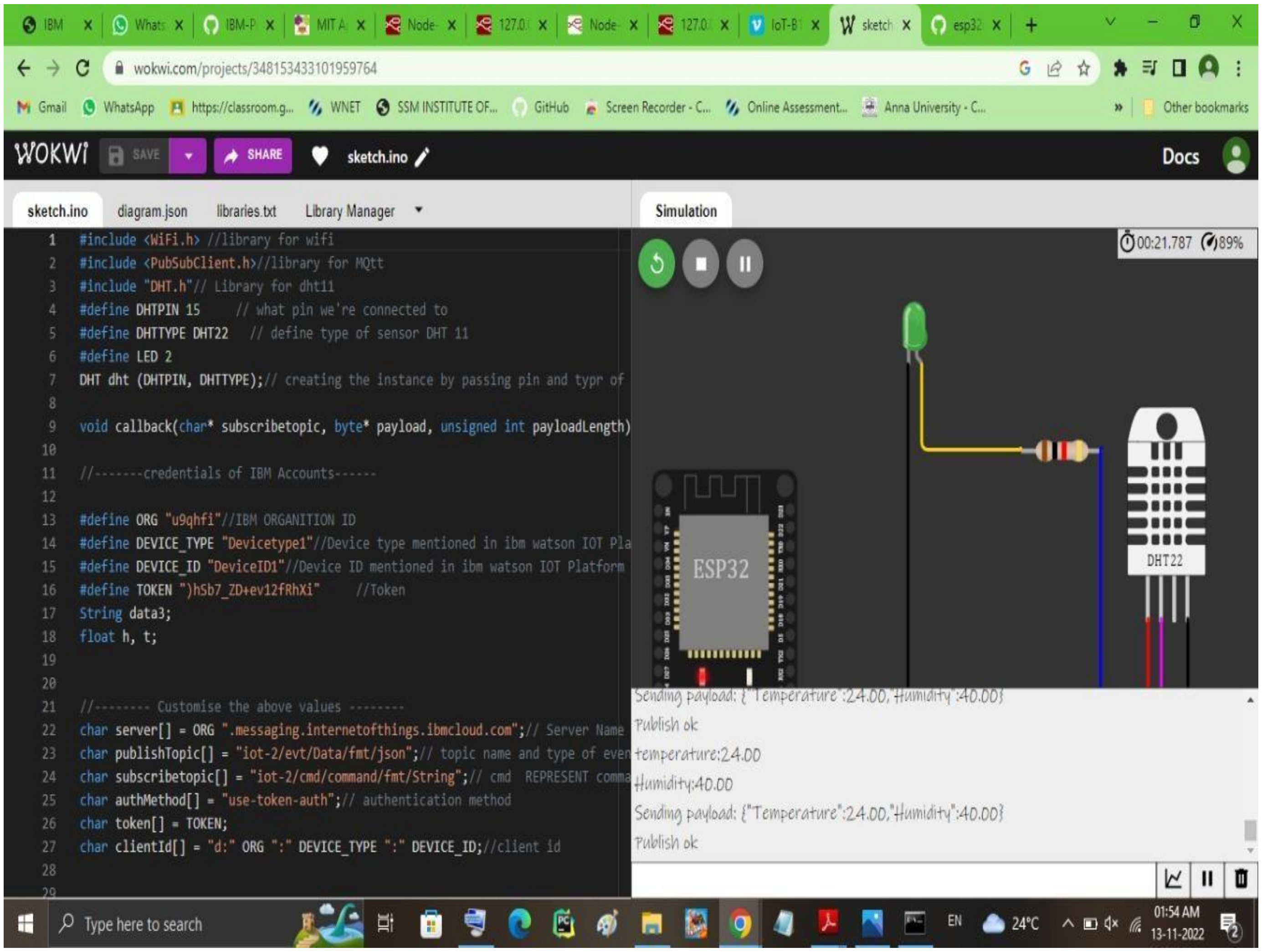
```
   Serial.println("data: "+ data3);
   if(data3=="lighton")
   {
Serial.println(data3);
digitalWrite(LED,HIGH);

   }


   else
   {
Serial.println(data3);
digitalWrite(LED,LOW);

   }
data3="";


}
```



```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11
#define LED 2
DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of

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

//-------credentials of IBM Accounts------

#define ORG "u9qhfi"//IBM ORGANITION ID
#define DEVICE_TYPE "Devicetype1"//Device type mentioned in ibm watson IOT Pla
#define DEVICE_ID "DeviceID1"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN ")hSb7_ZD+ev12fRhXi"     //Token
String data3;
float h, t;


//-------- Customise the above values --------
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of even
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT comma
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
```

```
Sending payload: {"Temperature":24.00,"Humidity":40.00}
Publish ok
temperature:24.00
Humidity:40.00
Sending payload: {"Temperature":24.00,"Humidity":40.00}
Publish ok
```

IoT Simulator:

In our project in the place of sensors we are going to use

IoT sensor simulator which give random readings to the connectedCloud.

The link to simulator:

https://41azth.internetofthings.ibmcloud.com/dashboard/devices/browse

We need to give the credentials of the created device in IBMWatson IoT Platform to connect cloud to simulator.