SPRINT – 3 DEVELOPMENT OF PYTHON SCRIPT

Date	15 November 2022
Team ID	PNT2022TMID35844
Project Name	IoT Based Smart Crop Protection System for Agriculture

DESCRIPTION :

The random sensor data's are generated and automation has been implemented through the python code to implement IoT based crop protection system. And the code gives the response to the IoT Device in IBM Watson Platform.

PYTHON CODE :

```
#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

float floatMap(float x, float in_min, float in_max, float out_min, float out_max) {

 return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;

}


DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht
connected


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


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


#define ORG "wckx64"//IBM ORGANITION ID

#define DEVICE_TYPE "ibmiot"//Device type mentioned in ibm watson IOT Platform

#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "12345678"     //Token
```

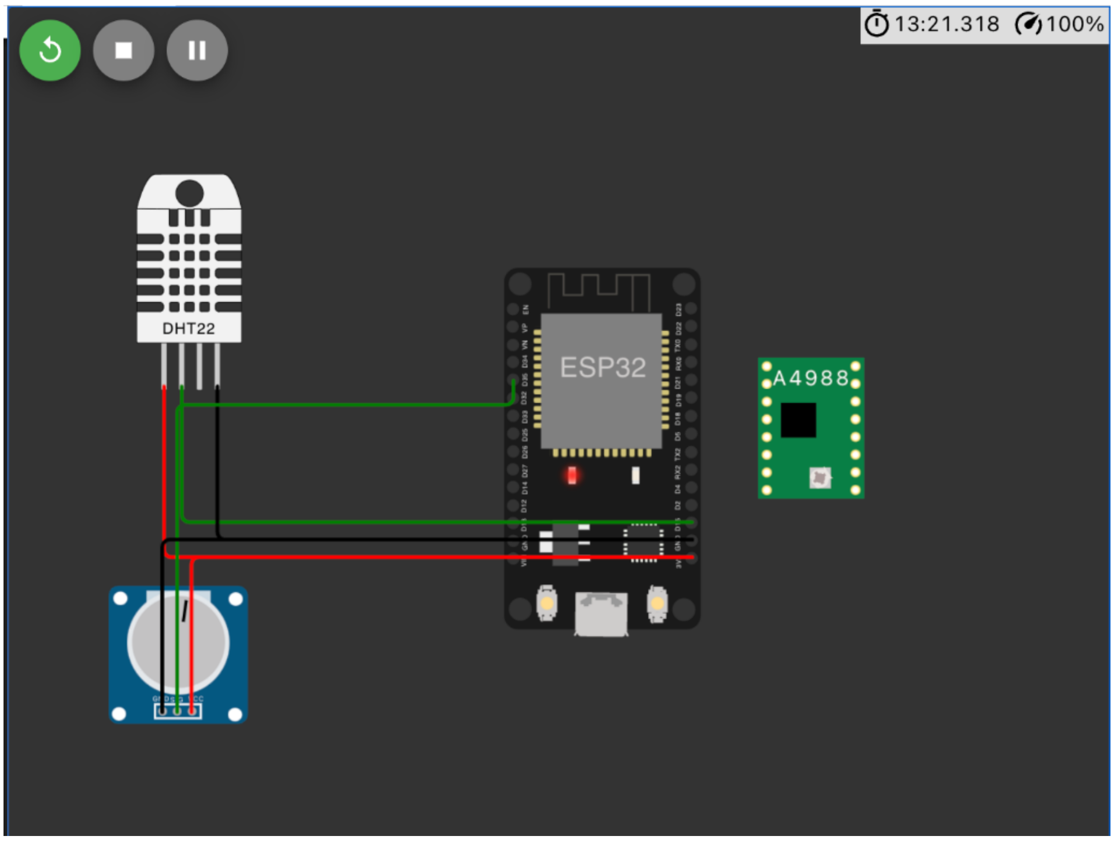
```
String data3;
float h, t;


//-------- 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 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; // creating the instance for wificlient
PubSubClient 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();

 int analogValue = analogRead(35);

 float voltage = floatMap(analogValue, 0, 4095, 0, 14);


 delay(1000);

 Serial.print("temp:");

 Serial.println(t);

 Serial.print("Humid:");

 Serial.println(h);

 Serial.print(" ph value ");

 Serial.println(voltage);


 PublishData(t);

 PublishData1(h);

 PublishData2(voltage);

  delay(1000);

 if (!client.loop()) {

   mqttconnect();

 }

}




/*....................................retrieving to
Cloud..............................*/
```

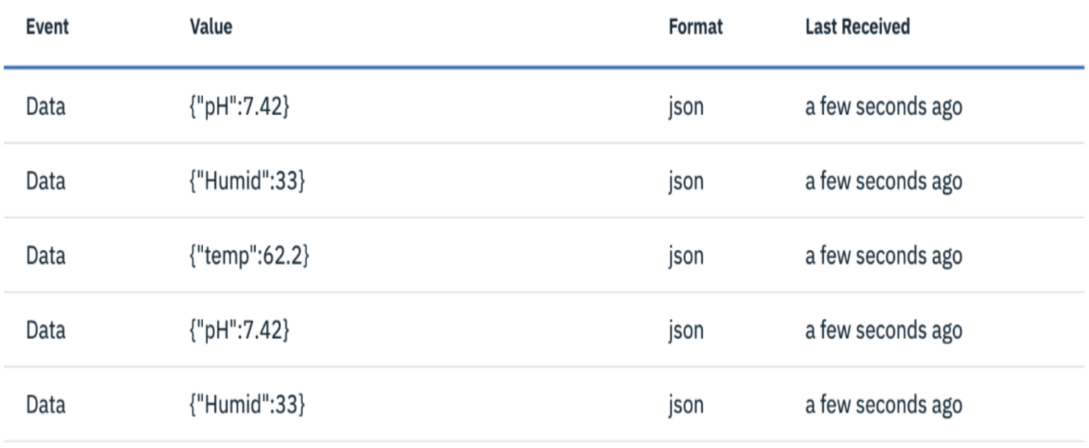
```
void PublishData(float temp)
{
 mqttconnect();//function call for connecting to ibm
 /*
    creating the String in in form JSon to update the data to ibm cloud
 */
 String payload = "{\"temp\":";
 payload += temp;
payload += "}";




  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 print publish failed
 } else {
   Serial.println("Publish failed");
 }
 }
void PublishData1( 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 = "{\"Humid\":";
 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 print publish failed

 } else {

   Serial.println("Publish failed");

 }

 }

void PublishData2(float voltage) {

 mqttconnect();//function call for connecting to ibm

 /*

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

 */

 String payload = "{\"pH\":";

payload += voltage;

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 print publish 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 to establish 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="";
}
```

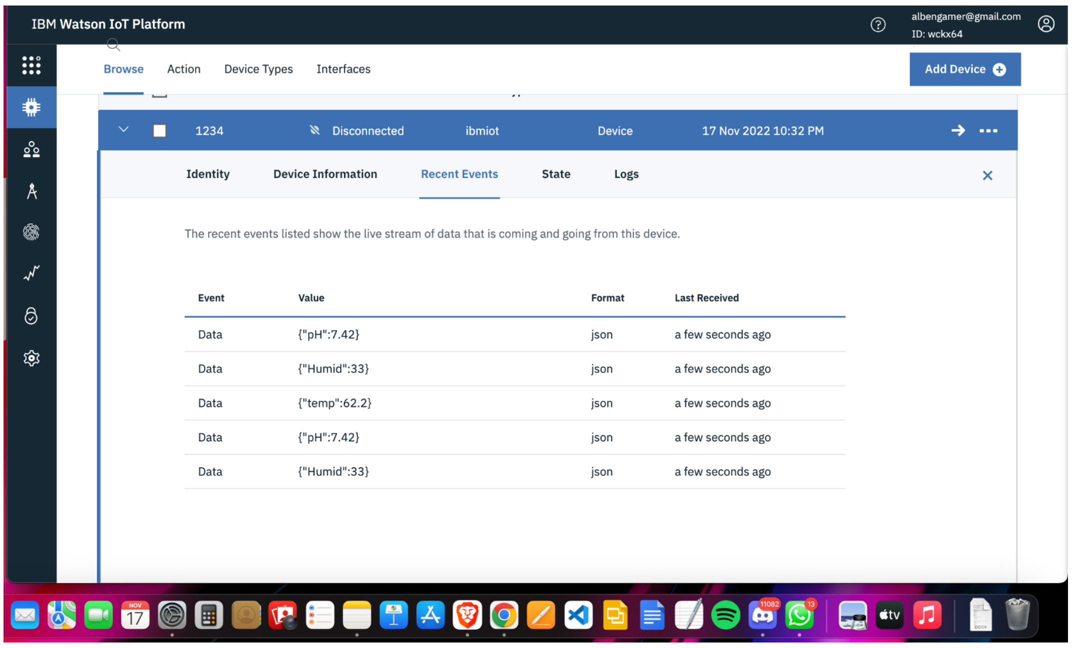
CONNECTION DIAGRAM :



OUTPUT IN IBM WATSON IoT PLATFORM



Event	Value	Format	Last Received
Data	{"pH":7.42}	json	a few seconds ago
Data	{"Humid":33}	json	a few seconds ago
Data	{"temp":62.2}	json	a few seconds ago
Data	{"pH":7.42}	json	a few seconds ago
Data	{"Humid":33}	json	a few seconds ago



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

Event	Value	Format	Last Received
Data	{"pH":7.42}	json	a few seconds ago
Data	{"Humid":33}	json	a few seconds ago
Data	{"temp":62.2}	json	a few seconds ago
Data	{"pH":7.42}	json	a few seconds ago
Data	{"Humid":33}	json	a few seconds ago