IBM ASSIGNMENT 4

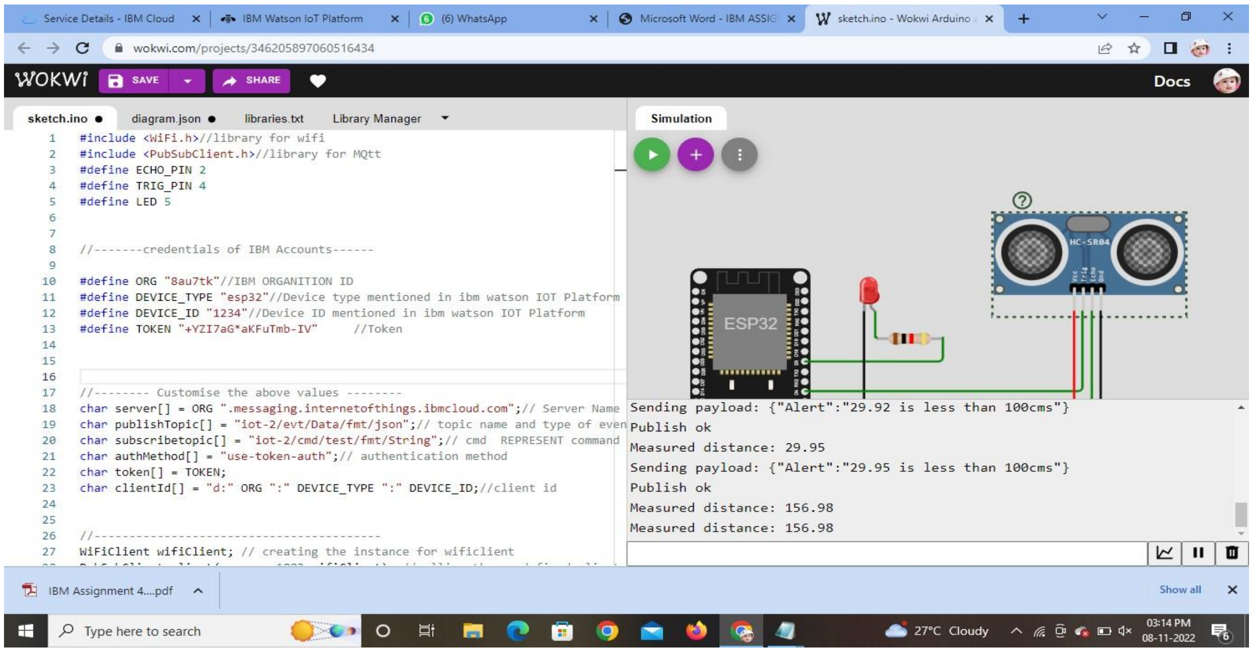
- J R GHAJENDHIRAN PNT2022TMID22142

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#define ECHO_PIN 2
#define TRIG_PIN 4
#define LED 5


//-------credentials of IBM Accounts------
#define ORG "8au7tk"//IBM ORGANITION ID
#define DEVICE_TYPE "esp32"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "+YZI7aG*aKFuTmb-IV"     //Token



//-------- 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/test/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,wifiClient); //calling the predefined client id by passing parameter like
server id,portand wificredential void setup()// configureing the ESP32
{
  Serial.begin(115200);
pinMode(TRIG_PIN,         OUTPUT);
pinMode(ECHO_PIN,           INPUT);
pinMode(LED,OUTPUT);   delay(10);
Serial.println();   wificonnect();
mqttconnect();
}
```

```
float readDistanceCM() {    digitalWrite(TRIG_PIN,
LOW);                                         delayMicroseconds(2);
digitalWrite(TRIG_PIN,                                        HIGH);
delayMicroseconds(10);      digitalWrite(TRIG_PIN,
LOW);   int duration = pulseIn(ECHO_PIN, HIGH);
return duration * 0.034 / 2;
}

void loop()// Recursive Function
{    float  distance  =  readDistanceCM();    bool
isNearby = distance < 100;   digitalWrite(LED,
isNearby);
  Serial.print("Measured distance: ");   Serial.println(distance);
delay(100);
  if (isNearby == 1){
PublishData(distance);
  }   delay(1000);   if (!client.loop())
{
mqttconnect();
  }
}



/*....................................retrieving to
Cloud...............................*/
 void  PublishData(float  distance)  {      mqttconnect();//function  call  for
connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud   */
  String payload = "{\"Alert\":""\"";   payload +=
distance;    payload += " is less than 100cms\"";
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");
  }
}
```

Picture:-

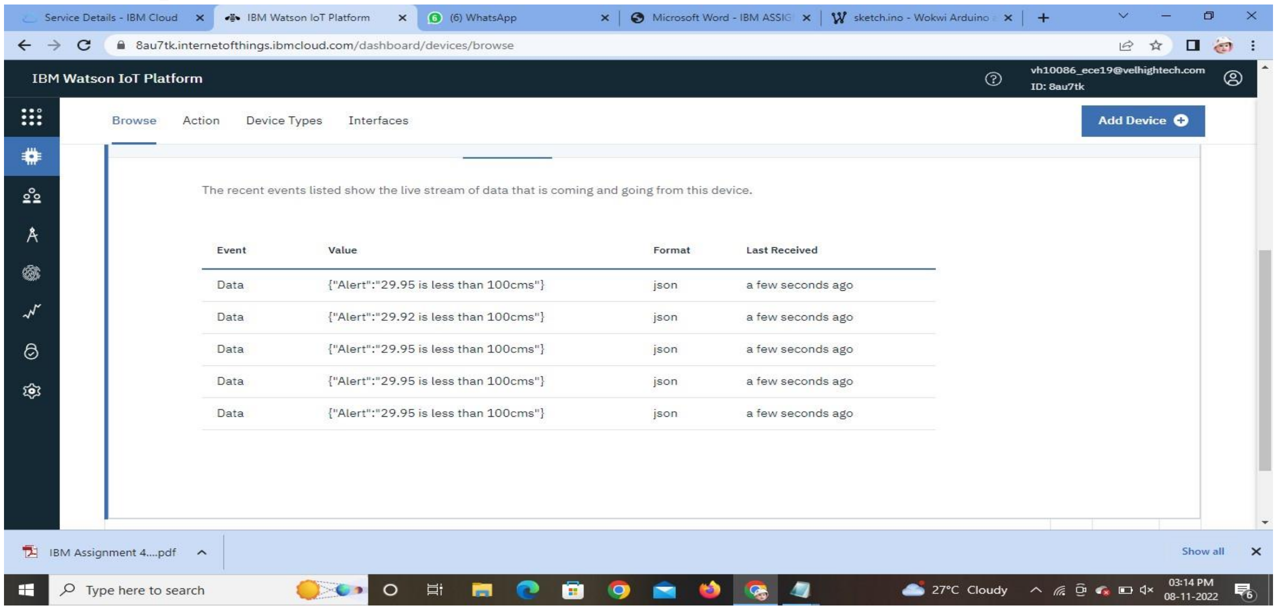


```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#define ECHO_PIN 2
#define TRIG_PIN 4
#define LED 5


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

#define ORG "8au7tk"//IBM ORGANITION ID
#define DEVICE_TYPE "esp32"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "+YZI7aG*aKFuTmb-IV"     //Token



//-------- 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/test/fmt/String";// cmd  REPRESENT command
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
```

```
Sending payload: {"Alert":"29.92 is less than 100cms"}
Publish ok
Measured distance: 29.95
Sending payload: {"Alert":"29.95 is less than 100cms"}
Publish ok
Measured distance: 156.98
Measured distance: 156.98
```

Link:- https://wokwi.com/projects/348100798012457556

Cloud output:-



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

Event	Value	Format	Last Received
Data	{"Alert":"29.95 is less than 100cms"}	json	a few seconds ago
Data	{"Alert":"29.92 is less than 100cms"}	json	a few seconds ago
Data	{"Alert":"29.95 is less than 100cms"}	json	a few seconds ago
Data	{"Alert":"29.95 is less than 100cms"}	json	a few seconds ago
Data	{"Alert":"29.95 is less than 100cms"}	json	a few seconds ago