Assignment - 4

Assignment Date	19 October 2022
Student Name	Mr. Sanjeev V E
Student Roll Number	737819ECR160
Maximum Marks	2 Marks

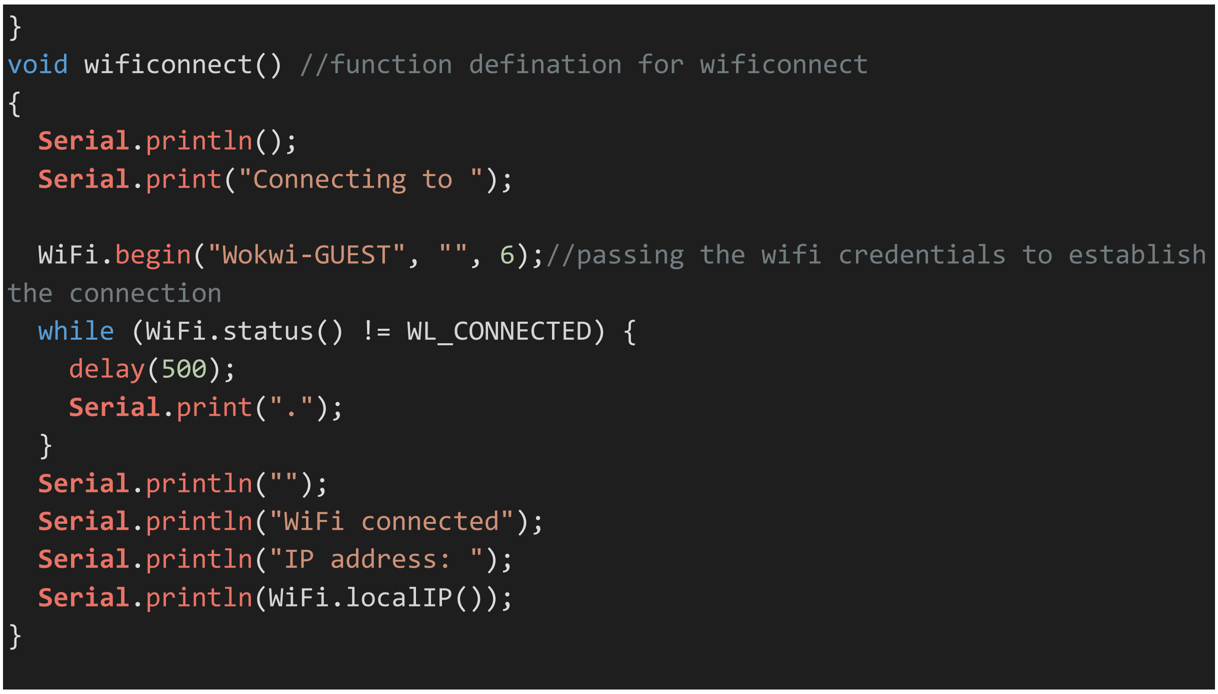
Question-1:

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device recent events. Upload document with wokwi share link and images of IBM cloud.

Solution:

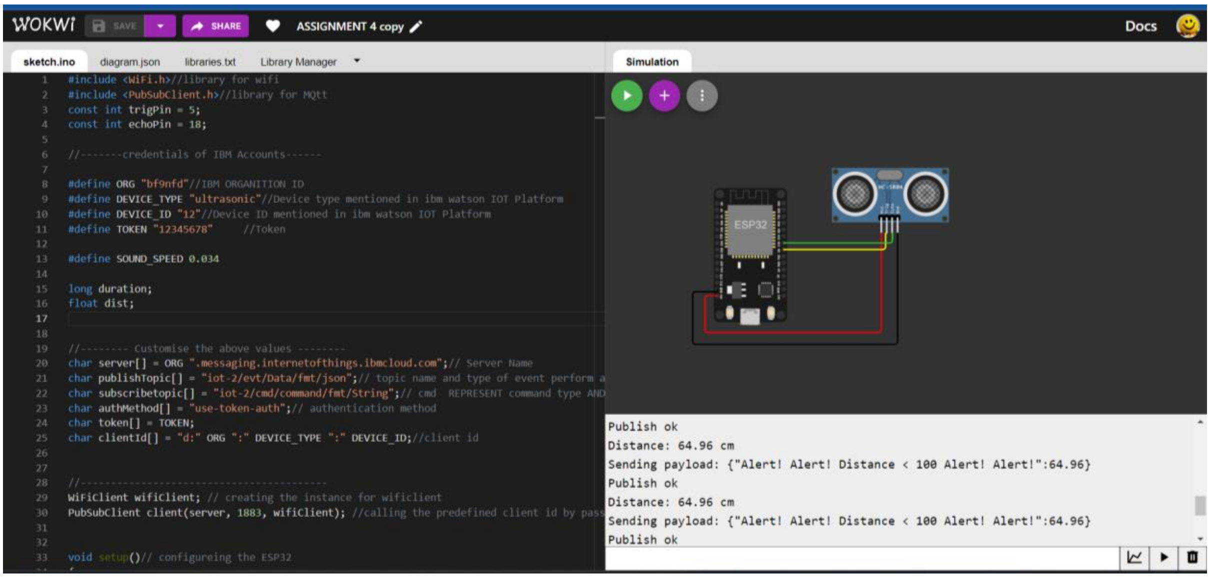
```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
const int trigPin = 5;
const int echoPin = 18;

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

#define ORG "bf9nfd"//IBM ORGANITION ID
#define DEVICE_TYPE "ultrasonic"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "12"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token

#define SOUND_SPEED 0.034

long duration;
float dist;


//-------- 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, wifiClient); //calling the predefined client
id by passing parameter like server id,portand wificredential


void setup()// configureing the ESP32
{
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{

  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  dist = duration * SOUND_SPEED/2;

  // Prints the distance in the Serial Monitor
  Serial.print("Distance: ");
  Serial.print(dist);
  Serial.println(" cm");
  delay(1000);

  PublishData(dist);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}



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

void PublishData(float dist) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in form JSon to update the data to ibm cloud
  */
  if(dist<100)
  {
  String payload = "{\"Alert! Alert! Distance < 100 Alert! Alert!\":";
  payload += dist;
  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");
  }
   }
   else{
     String payload = "{\"Distance of Object\":";
     payload += dist;
     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();
  }
```



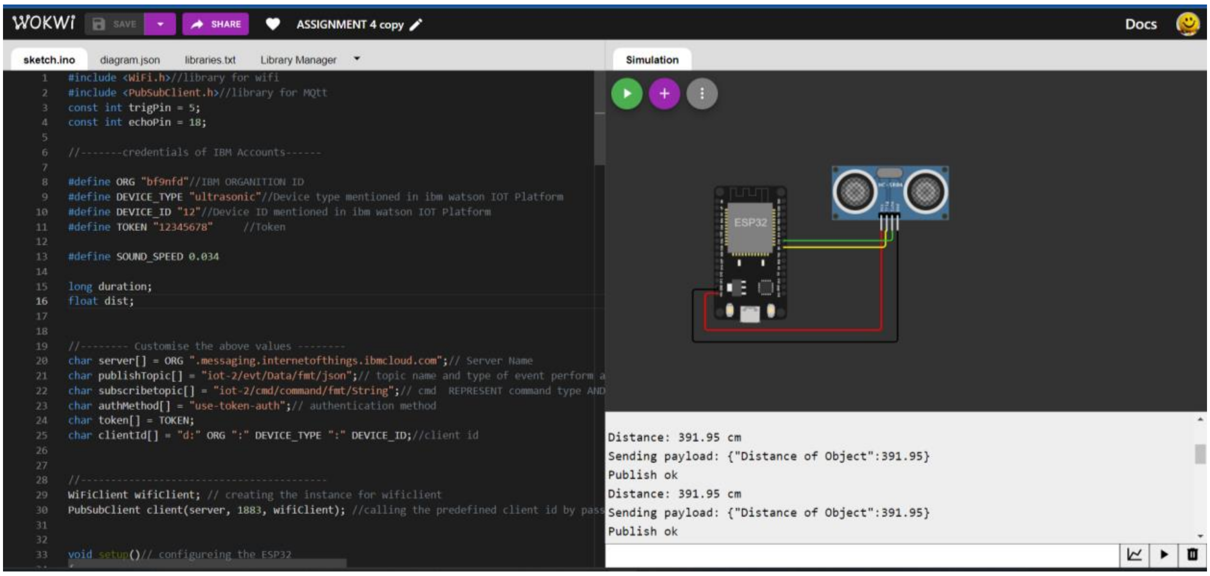
```
}
void wificonnect() //function defination for wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  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());
}
```

Output:



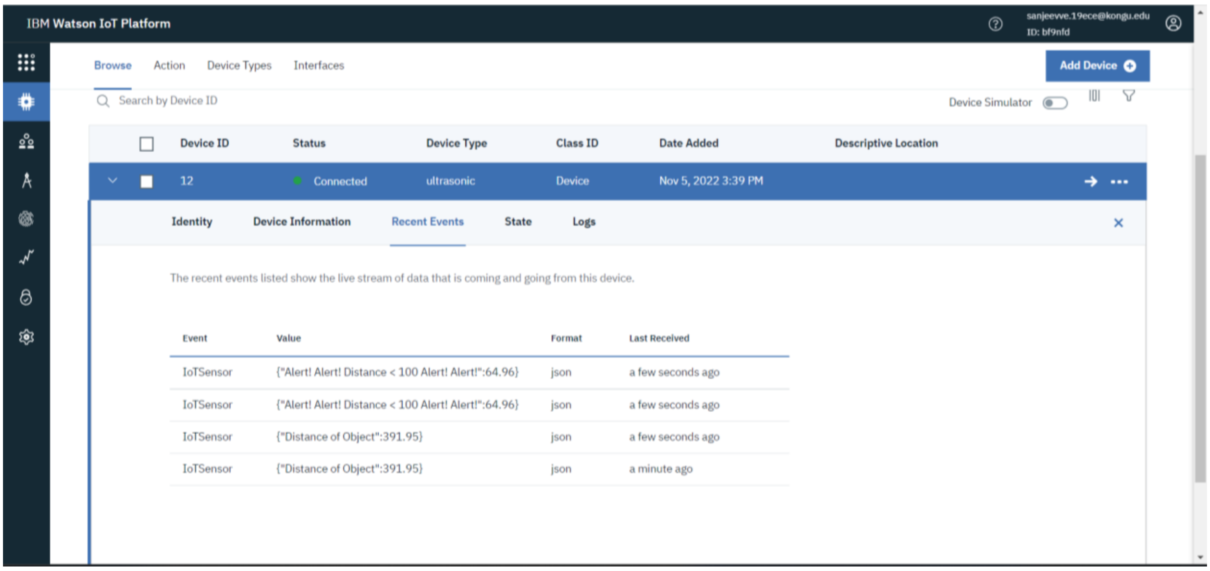
```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
const int trigPin = 5;
const int echoPin = 18;

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

#define ORG "bf9nfd"//IBM ORGANITION ID
#define DEVICE_TYPE "ultrasonic"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token

#define SOUND_SPEED 0.034

long duration;
float dist;


//-------- 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 a
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND
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 pass


void setup()// configureing the ESP32
```

```
Publish ok
Distance: 64.96 cm
Sending payload: {"Alert! Alert! Distance < 100 Alert! Alert!":64.96}
Publish ok
Distance: 64.96 cm
Sending payload: {"Alert! Alert! Distance < 100 Alert! Alert!":64.96}
Publish ok
```



```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
const int trigPin = 5;
const int echoPin = 18;

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

#define ORG "bf9nfd"//IBM ORGANITION ID
#define DEVICE_TYPE "ultrasonic"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "12"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token

#define SOUND_SPEED 0.034

long duration;
float dist;


//-------- 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 a
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd  REPRESENT command type AND
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 pass


void setup()// configureing the ESP32
```

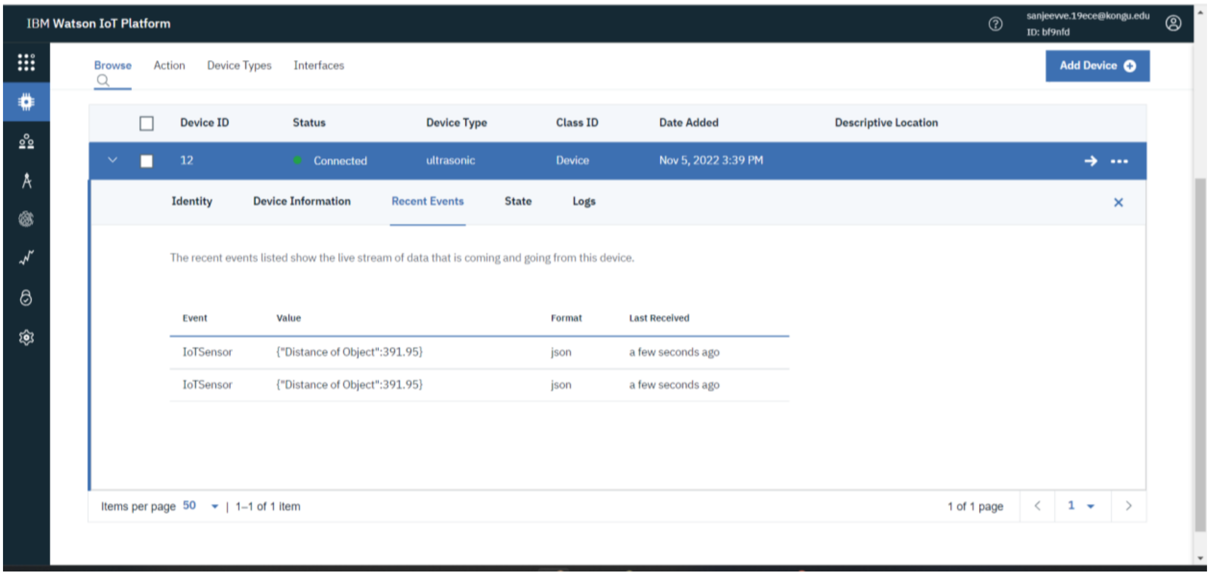
```
Distance: 391.95 cm
Sending payload: {"Distance of Object":391.95}
Publish ok
Distance: 391.95 cm
Sending payload: {"Distance of Object":391.95}
Publish ok
```



Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
12	Connected	ultrasonic	Device	Nov 5, 2022 3:39 PM	

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

Event	Value	Format	Last Received
IoTSensor	{"Alert! Alert! Distance < 100 Alert! Alert!":64.96}	json	a few seconds ago
IoTSensor	{"Alert! Alert! Distance < 100 Alert! Alert!":64.96}	json	a few seconds ago
IoTSensor	{"Distance of Object":391.95}	json	a few seconds ago
IoTSensor	{"Distance of Object":391.95}	json	a minute ago



Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
12	Connected	ultrasonic	Device	Nov 5, 2022 3:39 PM	

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

Event	Value	Format	Last Received
IoTSensor	{"Distance of Object":391.95}	json	a few seconds ago
IoTSensor	{"Distance of Object":391.95}	json	a few seconds ago

Items per page 50 | 1–1 of 1 item

1 of 1 page 1