

## ASSIGNMENT 4

### ESP 32 AND ULTRASONIC SENSOR DATA SENDS TO IBM CLOUD

<b>PROJECT</b>	IoT ENABLED - REAL TIME WATER QUALITY MONITORING AND CONTROL SYSTEM
<b>NAME</b>	JAYASHREE S
<b>PROJECT ID</b>	PNT2022TMID47600
<b>DATE</b>	22 OCT 2022

#### QUESTION:

Write a code and connections in wowki for the ultrasonic sensor.

Whenever the distance is less than 100cms send an “alert” to IBM cloud and display in the device receive recent events.

Upload documents with wowki share link and images of IBM cloud

#### PROGRAM:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "5vyvsk"//IBM ORGANITION ID
#define DEVICE_TYPE "REAL_TIME_WATER_QUALITY"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "PNT2022TMID47600"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "8vsl8edAhw2kYeYulZ" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
const int trigPin = 2;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
void setup() {
Serial.begin(115200);
pinMode(trigPin, OUTPUT);
```

```

pinMode(echoPin, INPUT);
wificonnect();
mqttconnect();
}
void loop()
{
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = duration * SOUND_SPEED/2;
Serial.print("Distance (cm): ");
Serial.println(distance);
if(distance<100)
{
Serial.println("ALERT!!");
delay(1000);
PublishData(distance);
delay(1000);
if (!client.loop()) {
mqttconnect();
}
}
delay(1000);
}
void PublishData(float dist) {
mqttconnect();
String payload = "{\"Distance\": ";
payload += dist;
payload += ", \"ALERT!!\": \"\" \"Distance less than 100cms\"";
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();

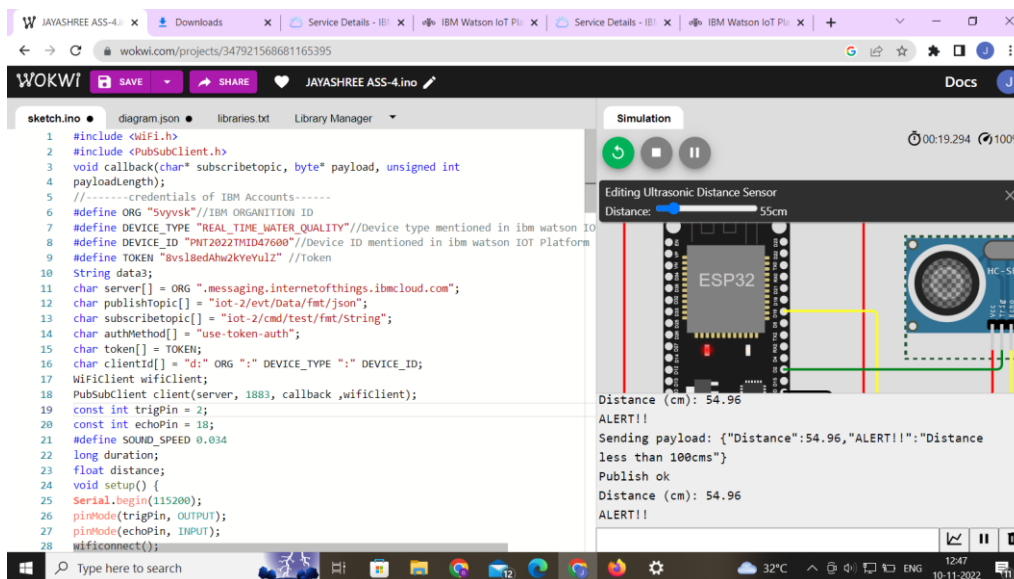
```

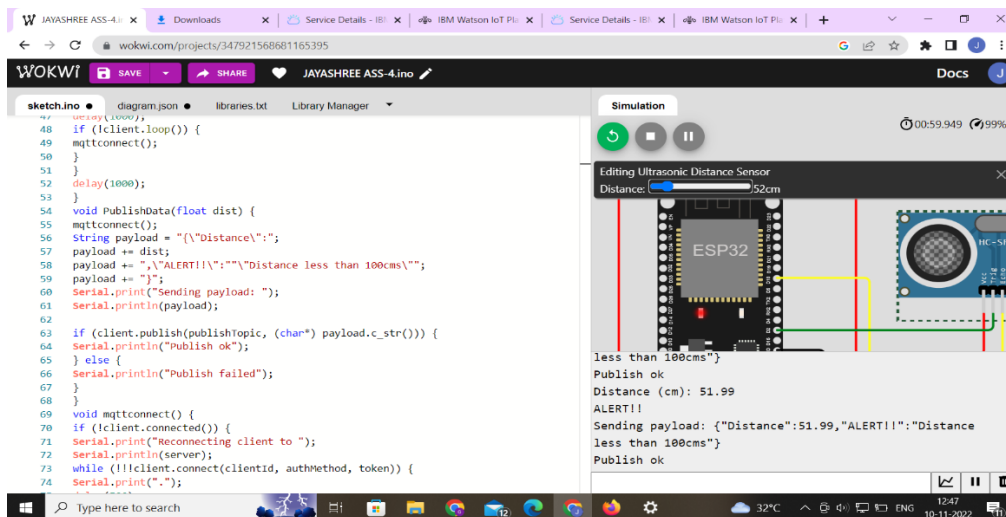
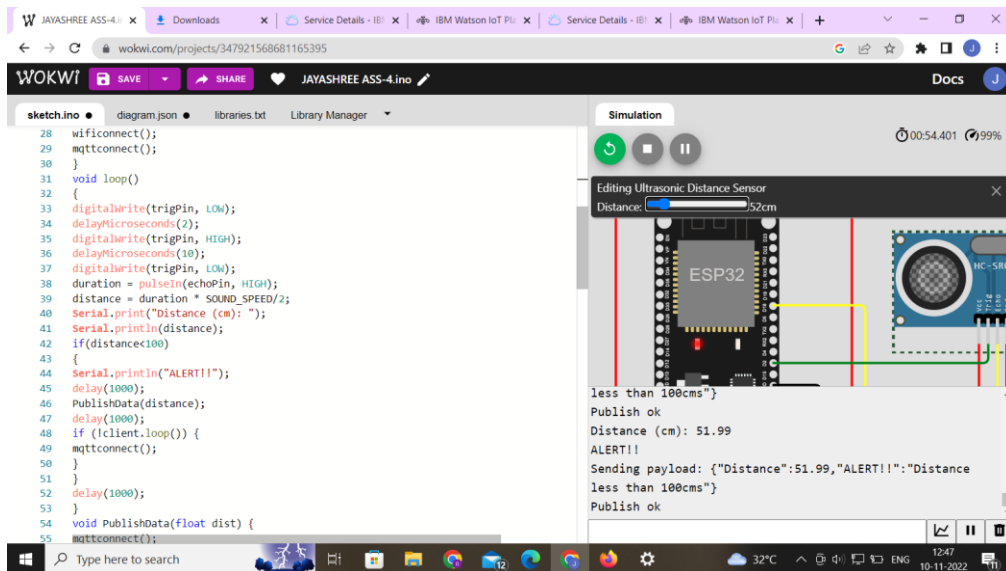
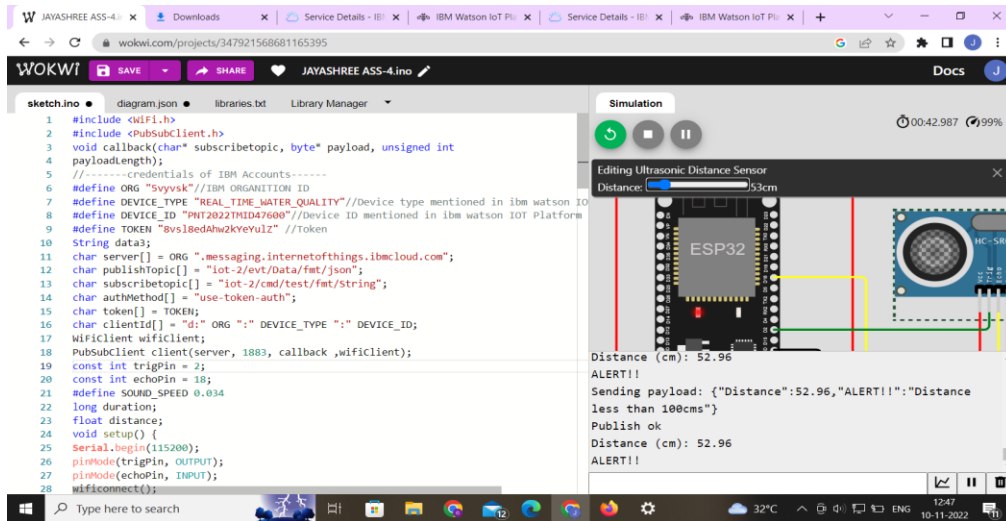
```

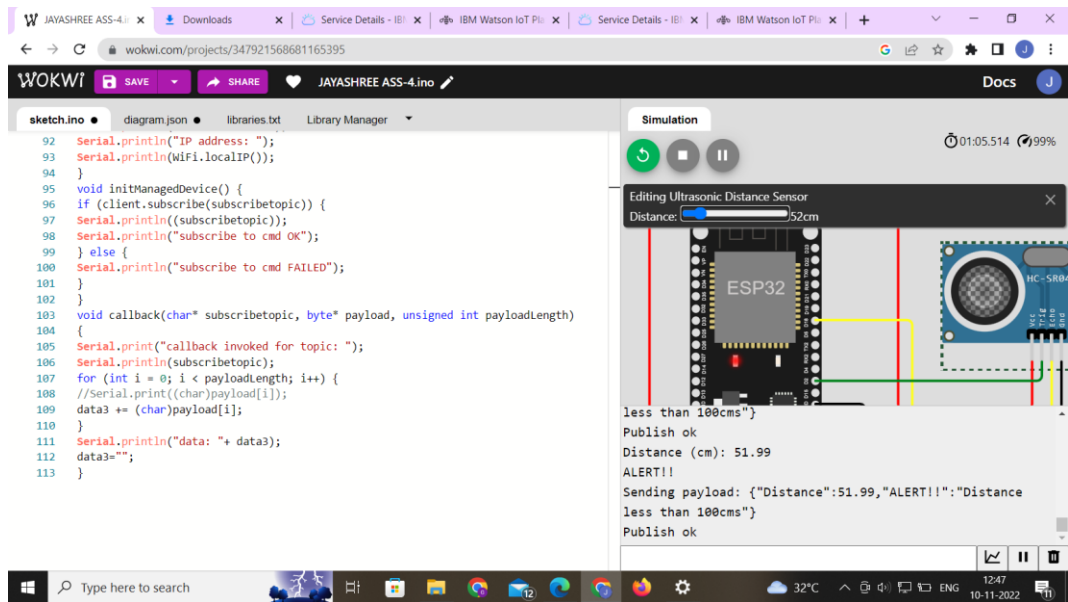
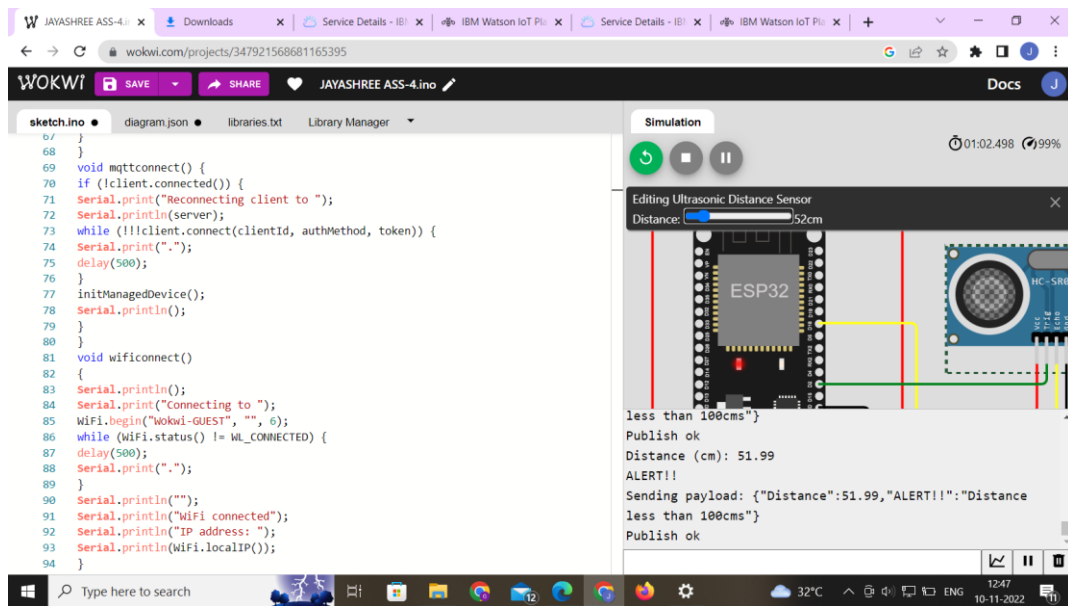
Serial.print("Connecting to ");
WiFi.begin("Wokwi-GUEST", "", 6);
while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}
void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    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++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }
  Serial.println("data: "+ data3);
  data3="";
}

```

## OUTPUT:







## IBM Watson IoT platform:

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area displays a table of devices. The selected device is 'PNT2022TMID47600', which is 'Disconnected' and has a 'REAL\_TIME\_WATER\_QUALITY' device type. Below the device list, a modal window titled 'Recent Events' is open, showing a table of recent events. The events table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The events are all 'Data' type, with a value of '{"Distance":51.99,"ALERT!!":"Distance less than ..."' and a format of 'json'. The last received time for all events is 'a few seconds ago'.

Event	Value	Format	Last Received
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago

This screenshot is identical to the one above, showing the IBM Watson IoT Platform dashboard with the 'Recent Events' modal window open. It displays the same device 'PNT2022TMID47600' and the same table of recent events, all of which are 'Data' type with a value of '{"Distance":51.99,"ALERT!!":"Distance less than ..."' and a format of 'json', received 'a few seconds ago'.

Event	Value	Format	Last Received
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago
Data	{"Distance":51.99,"ALERT!!":"Distance less than ..."	json	a few seconds ago

## Wokwi link:

<https://wokwi.com/projects/347921568681165395>

