

## QUESTION:

Write code and connections in Wokwi for the ultrasonic sensor. Whenever the distance is less than 100cms 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.

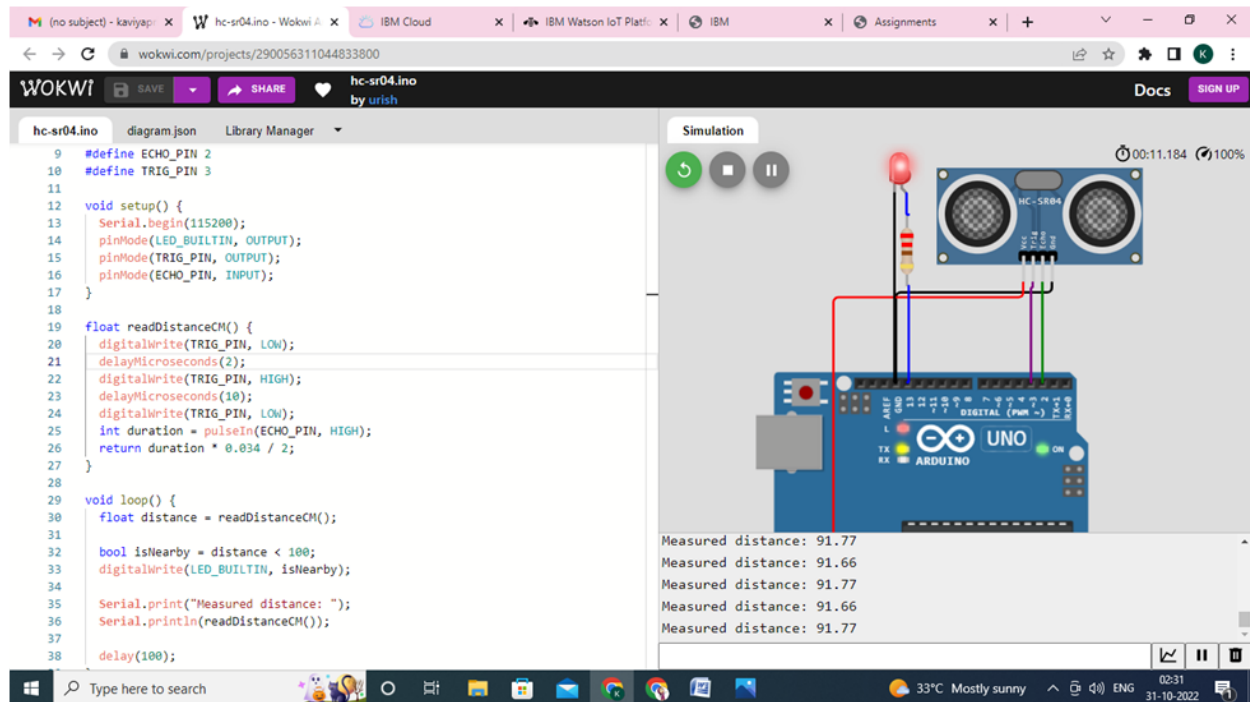
## PROGRAM:

```
/*  
  HC-SR04 Ultrasonic Sensor Example.  
  
  Turn the LED on when an object is within 100cm range.  
  
  Copyright (C) 2021, Uri Shaked  
*/  
  
#define ECHO_PIN 2  
#define TRIG_PIN 3  
  
void setup()  
{  
  Serial.begin(115200);  
  pinMode(LED_BUILTIN, OUTPUT);  
  pinMode(TRIG_PIN, OUTPUT);  
  pinMode(ECHO_PIN, INPUT);  
}  
  
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() {  
  float distance = readDistanceCM();  
  
  bool isNearby = distance < 100;  
  digitalWrite(LED_BUILTIN, isNearby);  
  
  Serial.print("Measured distance: ");  
  Serial.println(readDistanceCM());  
  
  delay(100);  
}
```

## CONNECTION:



The screenshot displays the Wokwi IDE interface. On the left, the code for the Arduino sketch is shown, including pin definitions, setup, and the main loop. The right side features a simulation window showing an Arduino Uno board connected to an HC-SR04 ultrasonic sensor. The sensor's VCC is connected to the 5V pin, GND to GND, Trig to digital pin 2, and Echo to digital pin 3. A red LED is connected to the Arduino's GND and a 10k resistor to the Trig pin. The simulation output shows the measured distance as 91.66-91.77 cm.

```
9 #define ECHO_PIN 2  
10 #define TRIG_PIN 3  
11  
12 void setup() {  
13   Serial.begin(115200);  
14   pinMode(LED_BUILTIN, OUTPUT);  
15   pinMode(TRIG_PIN, OUTPUT);  
16   pinMode(ECHO_PIN, INPUT);  
17 }  
18  
19 float readDistanceCM() {  
20   digitalWrite(TRIG_PIN, LOW);  
21   delayMicroseconds(2);  
22   digitalWrite(TRIG_PIN, HIGH);  
23   delayMicroseconds(10);  
24   digitalWrite(TRIG_PIN, LOW);  
25   int duration = pulseIn(ECHO_PIN, HIGH);  
26   return duration * 0.034 / 2;  
27 }  
28  
29 void loop() {  
30   float distance = readDistanceCM();  
31  
32   bool isNearby = distance < 100;  
33   digitalWrite(LED_BUILTIN, isNearby);  
34  
35   Serial.print("Measured distance: ");  
36   Serial.println(readDistanceCM());  
37  
38   delay(100);  
39 }
```

Measured distance: 91.77  
Measured distance: 91.66  
Measured distance: 91.77  
Measured distance: 91.66  
Measured distance: 91.77

## OUTPUT:

The screenshot displays the IBM Watson IoT Platform web interface. The top navigation bar includes tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various platform functions. The main content area shows details for a device with ID '115200', which is 'Disconnected' and of type 'ESP32'. The 'Recent Events' tab is selected, showing a table of events. The table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The events listed are all 'event\_1' with values representing an 'Alert distance' in JSON format, such as '21', '41', '43', '23', and '0'. A status message at the bottom of the table indicates '4 Simulations running'.

Event	Value	Format	Last Received
event_1	("Alert distance":21)	json	a few seconds ago
event_1	("Alert distance":41)	json	a few seconds ago
event_1	("Alert distance":43)	json	a few seconds ago
event_1	("Alert distance":23)	json	a few seconds ago
event_1	("Alert distance":0)		

4 Simulations running

## WOKWI LINK:

<https://wokwi.com/projects/347026184010203732>