ASSIGNMENT 4

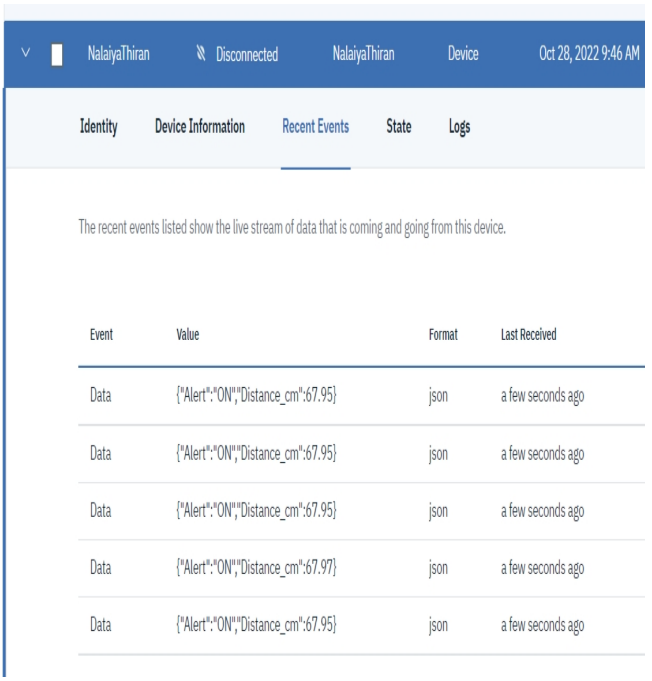
| | |
|---|---|
| Assignment Date | 29-10-2022 |
| Student Name | Vishali K |
| Team ID | PNT2022TMID48664 |
| Project Name | Personal Assistance Device Who are self reliant |
| Maximum marks | 2 Marks |

1.Write a code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100cms send “Alert” to IBM could and display in the device recent events. Upload document with wokwi share link and images of cloud.

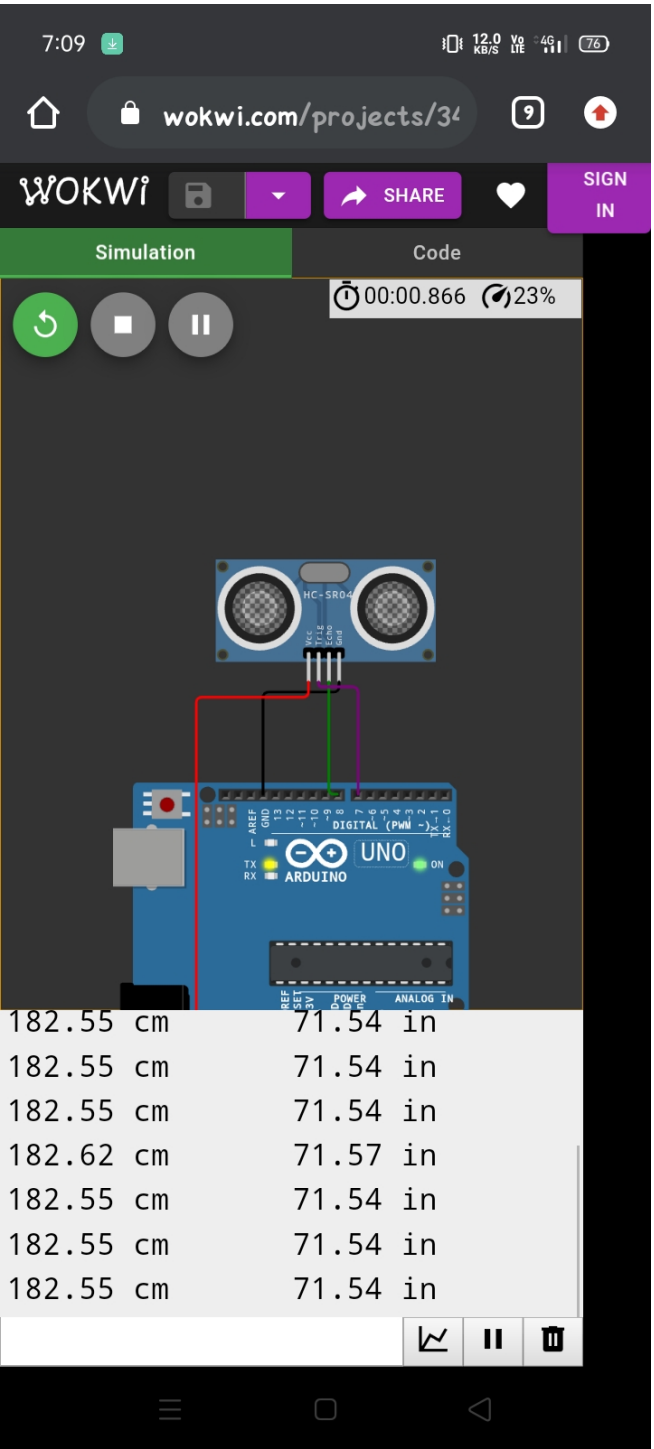
```
    #include <WiFi.h>
#include <PubSubClient.h>
WiFiClient wifiClient; String
data3;
    #define ORG "s8ov1q"
    #define DEVICE_TYPE "gayathri"
    #define DEVICE_ID "gayathri123"
    #define TOKEN "1234
    #define speed 0.034
    #define led 14
char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; char
publishTopic[] = "iot-2/evt/Gayathri/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String"; char
authMethod[] = "use-token-auth";
    char token[] = TOKEN;
    char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient); void
publishData();


const int trigpin=5; const
int echopin=18; String
command; String data="";

long duration; float
dist;
```

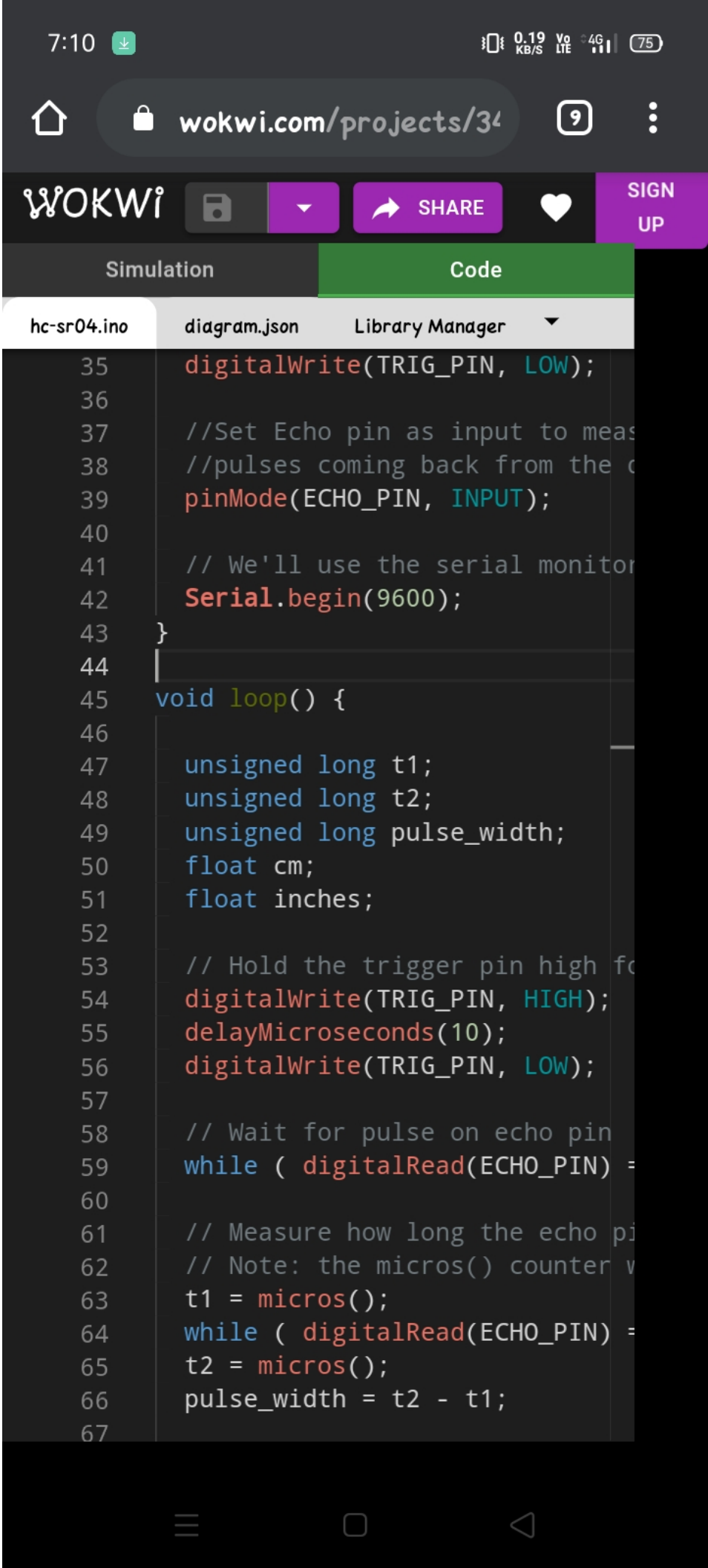
```
   void setup()
   {
     Serial.begin(115200);
     pinMode(led, OUTPUT);
     pinMode(trigpin, OUTPUT);
     pinMode(echopin, INPUT);
     wifiConnect(); mqttConnect();
   }

   void loop() {
     bool isNearby = dist < 100; digitalWrite(led,
     isNearby);

     publishData(); delay(500);if
     (!client.loop()) {
     mqttConnect();
  }
}

void wifiConnect() {
  Serial.print("Connecting to "); Serial.print("Wifi");
  WiFi.begin("Wokwi-GUEST", "", 6);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.print("WiFi connected, IP address: ");
Serial.println(WiFi.localIP());
}

void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);

    while (!client.connect(clientId, authMethod, token))
     {
      Serial.print("."); delay(500);
    }

    initManagedDevice();

    Serial.println();

  }
}

void initManagedDevice() {
  if (client.subscribe(topic)) {
    // Serial.println(client.subscribe(topic));
    Serial.println("IBM subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
```

```
}
void publishData()
{
  digitalWrite(trigpin,LOW);
  digitalWrite(trigpin,HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin,LOW);
  duration=pulseIn(echopin,HIGH);
  dist=duration*speed/2; if(dist<100){
    String payload = "{\"Alert Distance\":"; payload
    += dist;
    payload += "}";

    Serial.print("\n"); Serial.print("Sending
    payload: "); Serial.println(payload);
      if(client.publish(publishTopic, (char*) payload.c_str())) {
      Serial.println("Warning crosses 110cm -- it automaticaly of the loop");
      digitalWrite(led,HIGH);}

  }
    if(dist>101 && dist<111){
    String payload = "{\"Normal Distance\":"; payload
    += dist;
    payload += "}";

    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);

    }

  }



  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++){
    dist += (char)payload[i];
  }
  Serial.println("data:"+ data3);
  if(data3=="lighton"){
    Serial.println(data3);
    digitalWrite(led,HIGH);
```



```
}
data3="";
}
```


7:09
wokwi.com/projects/34
WOKWi
SHARE
SIGN IN
Simulation
Code
00:00.866
23%
HC-SR04
UNO
ARDUINO
182.55 cm 71.54 in
182.55 cm 71.54 in
182.55 cm 71.54 in
182.62 cm 71.57 in
182.55 cm 71.54 in
182.55 cm 71.54 in
182.55 cm 71.54 in


7:10
wokwi.com/projects/34
WOKWi
SHARE
SIGN UP
Simulation
Code
hc-sr04.ino
diagram.json
Library Manager
digitalWrite(TRIG_PIN, LOW);
//Set Echo pin as input to meas
//pulses coming back from the
pinMode(ECHO_PIN, INPUT);
// We'll use the serial monitor
Serial.begin(9600);
}
void loop() {
unsigned long t1;
unsigned long t2;
unsigned long pulse_width;
float cm;
float inches;
// Hold the trigger pin high fo
digitalWrite(TRIG_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
// Wait for pulse on echo pin
while ( digitalRead(ECHO_PIN) =
// Measure how long the echo pi
// Note: the micros() counter v
t1 = micros();
while ( digitalRead(ECHO_PIN) =
t2 = micros();
pulse_width = t2 - t1;