ASSIGNMENT – 3

QUESTION:

Build wowki product, use ultrasonic sensor and detect the distance from the object. Whenever distance is less than 100cms upload the value to the ibm cloud.in recent device events upload the data from wokwi.

- Example: distance is 20 cms. Upload the 20 value to the ibm cloud in recent event in the ibm iot platform device

- Submit the Assignment in PDF format in the Git repo.
- PDF should have wokwi share link, connections image, code, IBM cloud recent events image(Screenshot)

- Everyone in the team should submit the assignment as it is an individual task.

LINK:

https://wokwi.com/projects/364311694669799425

Code:

```
 #include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt

#define LED 5
#define LED2 4 #define
LED3 2 int LDR = 32; int
LDRReading = 0;   int
threshold_val = 800;
int lEDBrightness = 0;
int flag=0;
 void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
```

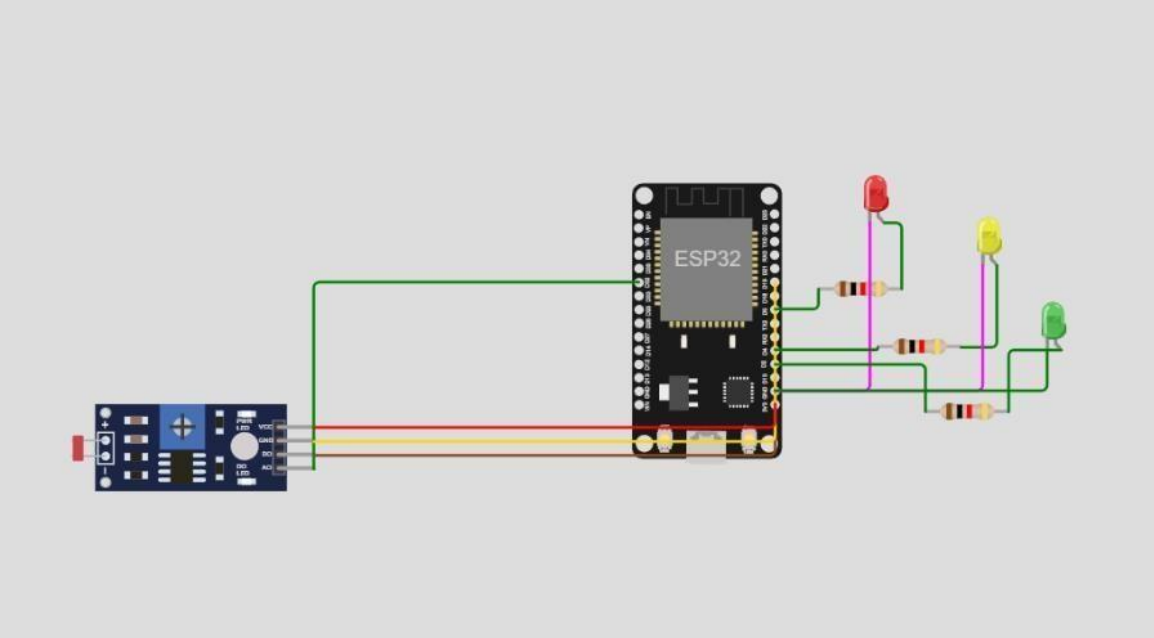
```
//-------credentials of IBM Accounts------

#define ORG "stuloy"//IBM ORGANITION ID
#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3; float h, t;

//-------- 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, callback ,wifiClient); //calling the predefined
client id by passing parameter like server id,portand wificredential void
setup()// configureing the ESP32
{
  Serial.begin(115200);

pinMode(LED,OUTPUT);
pinMode(LED2,OUTPUT);
pinMode(LED3,OUTPUT);
delay(10);
Serial.println();
wificonnect();
mqttconnect();
}  void loop()// Recursive
Function
{


  //PublishData(t, h);
  //delay(1000);

  /* LDRReading = analogRead(LDR);
  Serial.print("LDR READING:");
Serial.println(LDRReading);
if (LDRReading >threshold_val){
lEDBrightness = map(LDRReading, 0, 1023, 0, 255);
```

```
  Serial.print("LED BRIGHTNESS:");
Serial.println(lEDBrightness);
analogWrite(LED, lEDBrightness);
analogWrite(LED2, lEDBrightness);
analogWrite(LED3, lEDBrightness);
  }   else{
analogWrite(LED, 0);
analogWrite(LED2, 0);
analogWrite(LED3, 0);
  }
delay(300);*/
  if (!client.loop()) {
mqttconnect();
  }
}



/*.....................................retrieving to
Cloud.............................*/  /*void
PublishData(float temp, float humid) {
mqttconnect();//function call for connecting to
ibm*/
  /*      creating the String in in form JSon to update the data to ibm cloud
*/
  /*String payload = "{\"temperature\":";
payload += temp;   payload += ","
"\"humidity\":";   payload += humid;
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
```

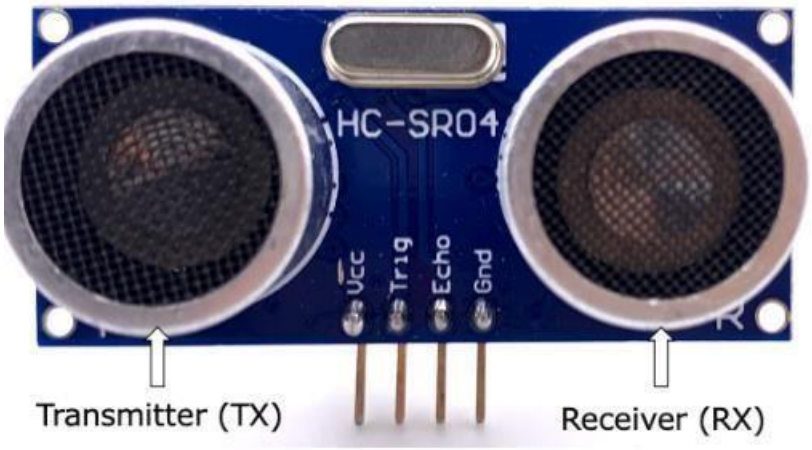
```
(!!!client.connect(clientId,    authMethod,    token))    {
Serial.print(".");       delay(500);
    }
initManagedDevice();
     Serial.println();
  } }  void wificonnect() //function defination for
wificonnect
{
  Serial.println();
  Serial.print("Connecting to ");

  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");
  }
}  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);
if(data3=="lighton1")
  {
Serial.println(data3); digitalWrite(LED,HIGH);

  }    else
if(data3=="lightoff1")
  {
```

```
Serial.println(data3);
digitalWrite(LED,LOW);

  }   else
if(data3=="lighton2")
  {
Serial.println(data3);
digitalWrite(LED2,HIGH);   }    else
if(data3=="lightoff2")
  {
Serial.println(data3); digitalWrite(LED2,LOW);
   }   else
if(data3=="lighton3")
  {
Serial.println(data3);
digitalWrite(LED3,HIGH);

  }    else
if(data3=="lightoff3")
  {
Serial.println(data3); digitalWrite(LED3,LOW);
   }
data3="";


}
```

CIRCUIT DIAGRAM:



IBM CLOUD RECENT EVENT IMAGE:



. .