ASSIGNMENT-4

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

#define echoPin 18 #define trigPin 5 void callback(char* subscribetopic, byte*
payload, unsigned int payloadLength);

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

#define ORG "vbzdj5"//IBM ORGANITION ID
#define DEVICE_TYPE "nodeMcu"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "123456"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678910"     //Token

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

void setup() // configureing the ESP32
{
  Serial.begin(115200);
  pinMode(ledPin, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  wificonnect();
  mqttconnect();
}
float readDistanceCM() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW); int
  duration = pulseIn(echoPin, HIGH);
  return duration * 0.034 / 2;
}

void loop() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
```

```
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW); int duration =
  pulseIn(echoPin, HIGH); float distance =
  duration * 0.034 / 2;

  Serial.print("Distance: ");
  Serial.println(distance);

  if(distance <100)
   Serial.println("Alert");
  else if(distance >100)
   Serial.println("Distance is maintained");
  PublishData(distance)
  ; delay(1000); if
  (!client.loop()) {
   mqttconnect();
  }
}


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

void PublishData(float distance) {
  mqttconnect();//function call for connecting to ibm
  /* creating the String in in form JSon to update the data to ibm
     cloud
  */
  String payload = "{\"Distance\":";
  payload += distance; payload +=
  "," "\"Status\":"; payload +=
  "\"Alert\""; 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 definition 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");
  }
}
```