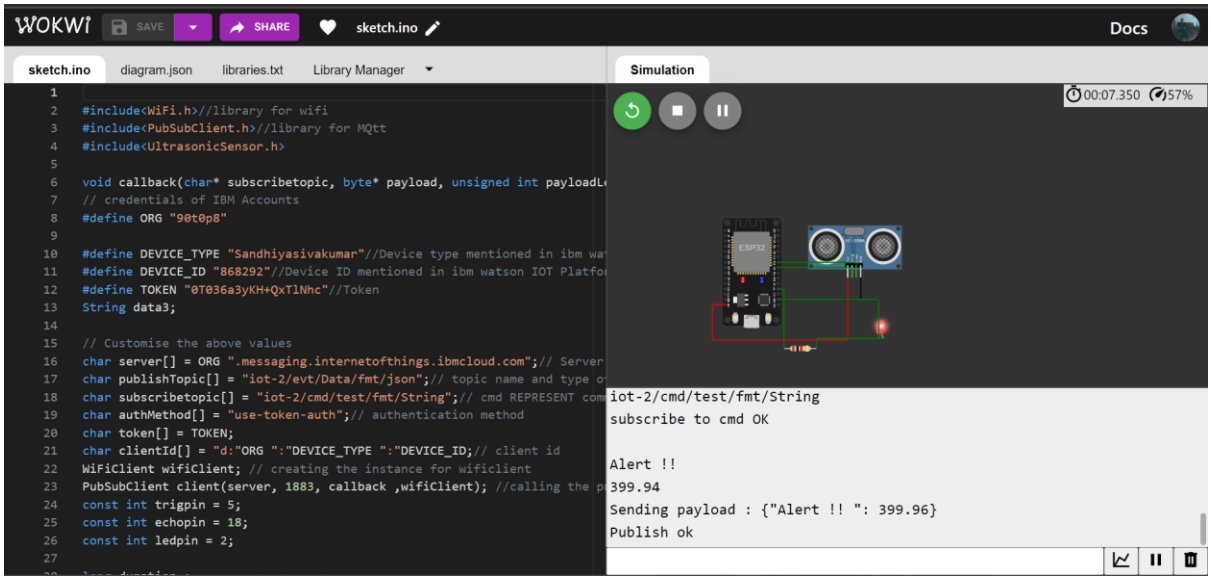
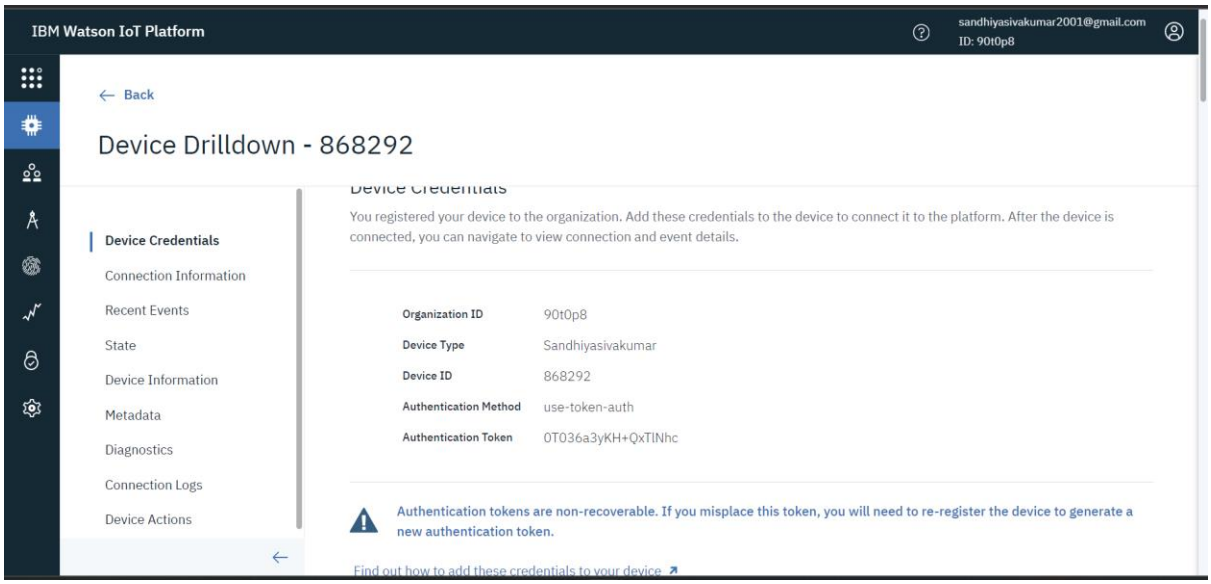
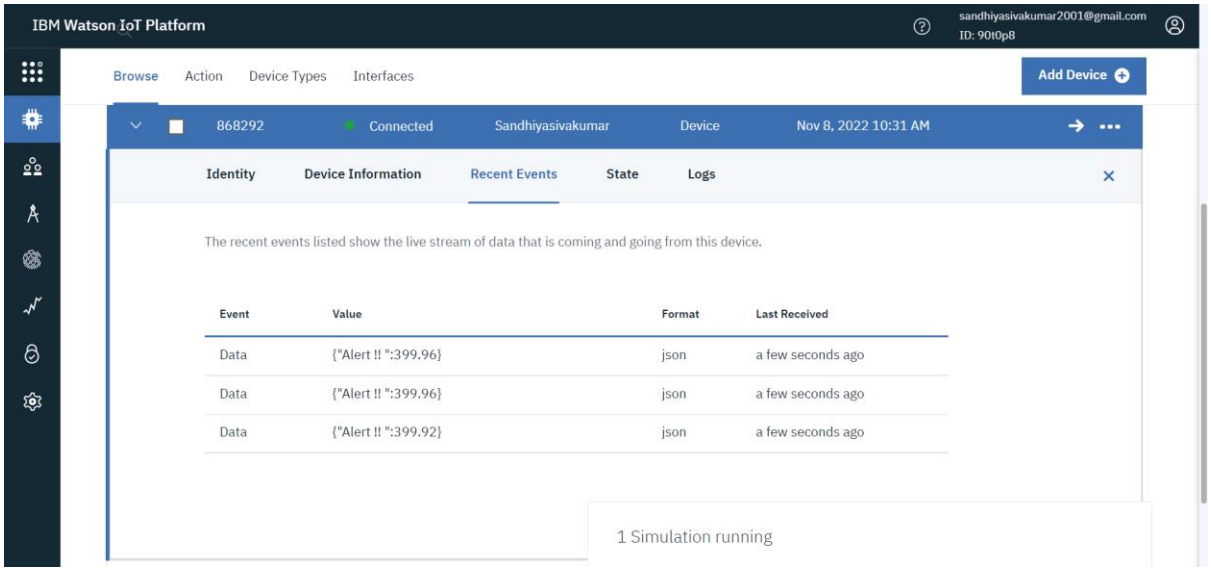
ASSIGNMENT 4







Code:

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

void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
// credentials of IBM Accounts
#define ORG "90t0p8"

#define DEVICE_TYPE "Sandhiyasivakumar"//Device type mentioned in ibm watson
IOT Platform
#define DEVICE_ID "868292"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "0T036a3yKH+QxTlNhc"//Token
String data3;

// 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
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command
type
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
const int trigpin = 5;
const int echopin = 18;
```

```
const int ledpin = 2;

long duration ;
float distance;
#define sound_speed 0.034
void setup()
{
 Serial.begin(115200);
 pinMode(trigpin, OUTPUT);
 pinMode(echopin, OUTPUT);
 pinMode(ledpin, OUTPUT);
 wificonnect();
 mqttconnect();
}
void loop()
{
 digitalWrite(trigpin, LOW);
 digitalWrite(trigpin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigpin, LOW);
 duration= pulseIn(echopin,HIGH);
 distance = duration * sound_speed /2;
 if(distance>=100)
 {
 PublishData(distance);
 delay(1000);
 if(!client.loop())
 {
 mqttconnect();
 }
 digitalWrite(ledpin, HIGH);
 Serial.println("Alert !!");
 Serial.println(distance);
 }
 else
 {
 digitalWrite(ledpin, LOW);
 }
 delay(10); // this speeds up the simulation
}
// 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 = "{\"Alert !! \": ";
 payload += distance;
 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
 }
 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
 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=="lighton")
 {
 Serial.println(data3);
 }
 else
 {
 Serial.println(data3);
 }
data3="";
}
```