Project Development – Delivery of Sprint-1

|  |  |
| --- | --- |
| Team ID | PNT2022TMID47580 |
| Project Name | Smart Solutions for Railways |

**Sprint-1 Code:**

/\*\*

   ESP32 + Ultrasonic Sensor in Wokwi

Here We use Ultrasonic Sensor to detect whether the Train crossing over the area, also the sensor detects every motion through its region we use 4 sensors.

If the motion is detected below 100 in all the 4 ultrasonic sensors, then the alert message is sent to the IOT Watson

\*/

#include <WiFi.h>//library for wifi

#include <PubSubClient.h>//library for MQtt

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

#define ORG "7kb26g"//IBM ORGANITION ID

#define DEVICE\_TYPE "krishna"//Device type mentioned in ibm watson IOT Platform

#define DEVICE\_ID "24052002"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "tDWoWy\_nHPVS!HVaTd"     //Token

const int TRIG\_PIN\_1 = 5;

const int TRIG\_PIN\_2 = 19;

const int TRIG\_PIN\_3 = 21;

const int TRIG\_PIN\_4 = 22;

const int ECHO\_PIN\_1 = 4;

const int ECHO\_PIN\_2 = 2;

const int ECHO\_PIN\_3 = 15;

const int ECHO\_PIN\_4 = 18;

const int RED\_LIGHT = 25;

const int GREEN\_LIGHT = 33;

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

void setup() {

**Serial**.begin(115200);

  pinMode(TRIG\_PIN\_1, OUTPUT);

  pinMode(TRIG\_PIN\_2, OUTPUT);

  pinMode(TRIG\_PIN\_3, OUTPUT);

  pinMode(TRIG\_PIN\_4, OUTPUT);

  pinMode(ECHO\_PIN\_1, INPUT);

  pinMode(ECHO\_PIN\_2, INPUT);

  pinMode(ECHO\_PIN\_3, INPUT);

  pinMode(ECHO\_PIN\_4, INPUT);

  pinMode(RED\_LIGHT, OUTPUT);

  pinMode(GREEN\_LIGHT, OUTPUT);

  wificonnect();

  mqttconnect();

}

float readDistance1() {

  digitalWrite(TRIG\_PIN\_1, LOW);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN\_1, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN\_1, LOW);

  int duration = pulseIn(ECHO\_PIN\_1, HIGH);

  return duration \* 0.034 / 2;

}

float readDistance2() {

  digitalWrite(TRIG\_PIN\_2, LOW);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN\_2, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN\_2, LOW);

  int duration = pulseIn(ECHO\_PIN\_2, HIGH);

  return duration \* 0.034 / 2;

}

float readDistance3() {

  digitalWrite(TRIG\_PIN\_3, LOW);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN\_3, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN\_3, LOW);

  int duration = pulseIn(ECHO\_PIN\_3, HIGH);

  return duration \* 0.034 / 2;

}

float readDistance4() {

  digitalWrite(TRIG\_PIN\_4, LOW);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN\_4, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN\_4, LOW);

  int duration = pulseIn(ECHO\_PIN\_4, HIGH);

  return duration \* 0.034 / 2;

}

void loop() {

  float distance1 = readDistance1();

  float distance2 = readDistance2();

  float distance3 = readDistance3();

  float distance4 = readDistance4();

**Serial**.println(distance1);

**Serial**.println(distance2);

**Serial**.println(distance3);

**Serial**.println(distance4);

  if(distance1<=100 && distance2<=100 && distance3<=100 && distance4<=100){

**Serial**.println("TARAIN IS ARRIVING");

      PublishData();

      digitalWrite(RED\_LIGHT, HIGH);

      delay(700);

      digitalWrite(RED\_LIGHT, LOW);

  }

  else{

**Serial**.println("TRAIN IS NOT ARRIVING");

    digitalWrite(GREEN\_LIGHT, HIGH);

    delay(700);

    digitalWrite(GREEN\_LIGHT, LOW);

  }

  delay(1000);

  if (!client.loop()) {

    mqttconnect();

  }

}

/\*.....................................retrieving to Cloud...............................\*/

void PublishData() {

  mqttconnect();//function call for connecting to ibm

  /\*

     creating the String in in form JSon to update the data to ibm cloud

  \*/

  bool status=true;

  String payload = "{\"ALERT\_MESSAGE\": \"TRAIN IS ARRIVING\"";

  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");

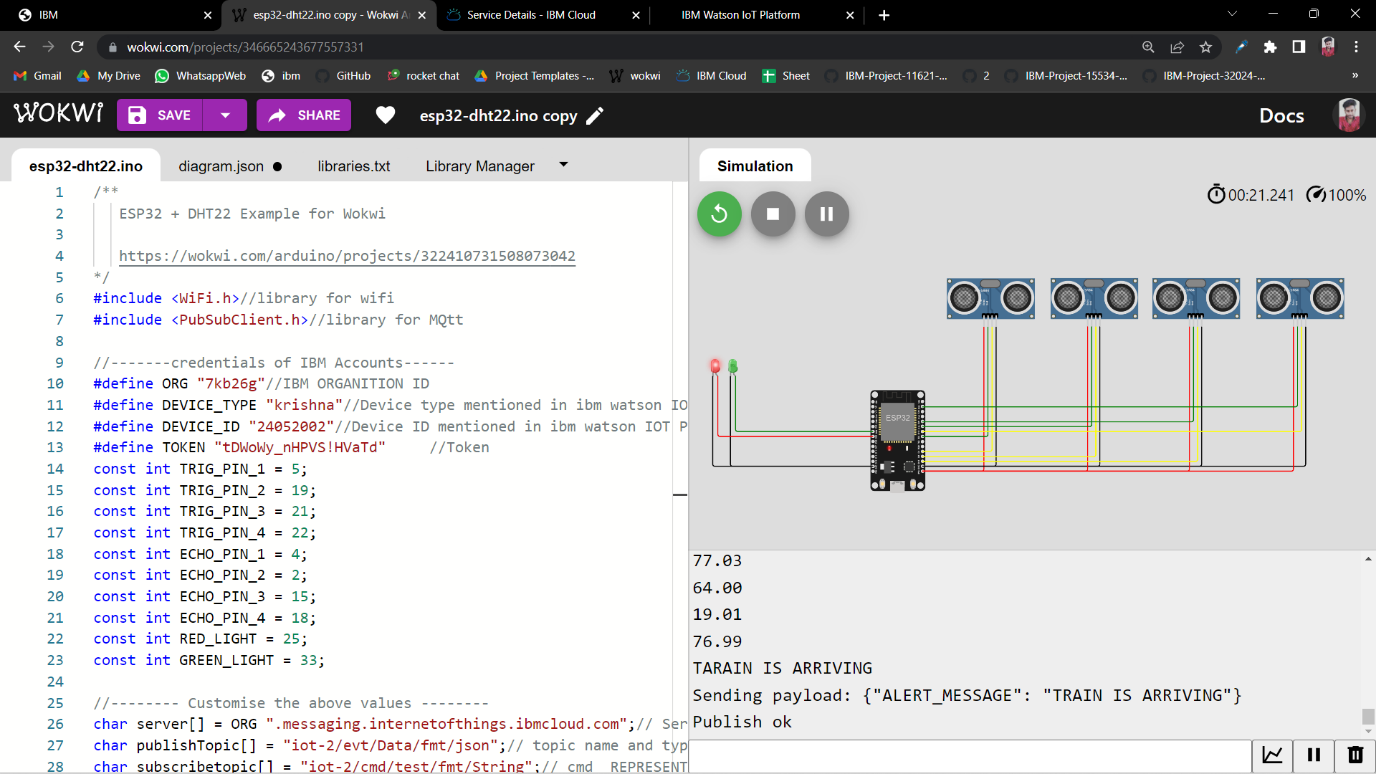
  } else {

**Serial**.println("subscribe to cmd FAILED");

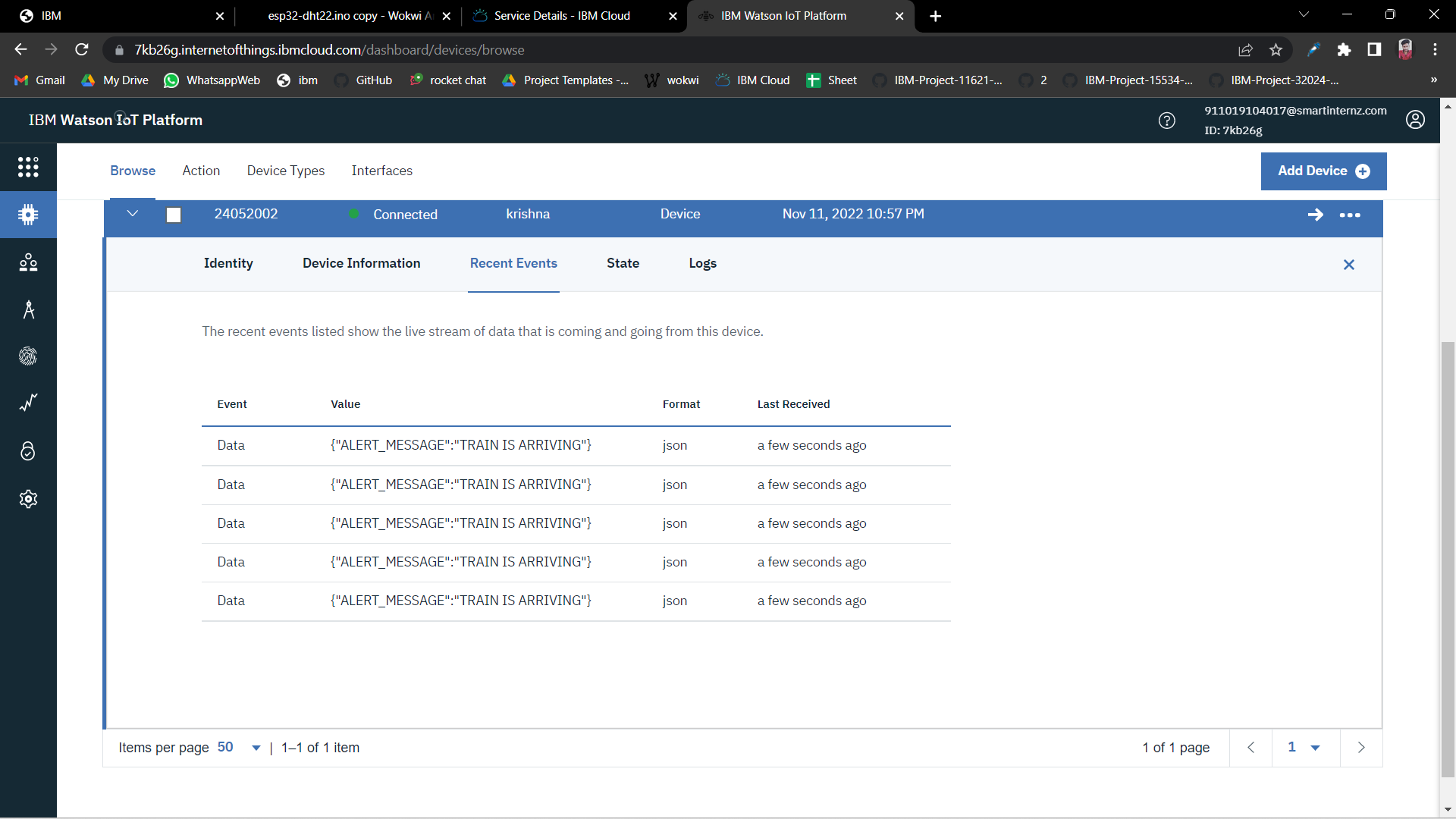
  }

}

**Output:**



**IBM Cloud Image:**



**Web Application:**

* For this Project We use Flutter Framework for developing the Web Application.
* In Sprint 1, Our Team Developed a Login and Registration UI using Flutter Framework and Dart Language.
* Here the Maria Database is used for Storing the Login Information.

**Login Page:**

**Registration Page:**

****