

Project development phase

Sprint - III

Date	11 November 2022
Team ID	PNT2022TMID13566
Project Name	Project - Industry-specific intelligent fire management system

LINK: <https://wokwi.com/projects/348062828084593236>

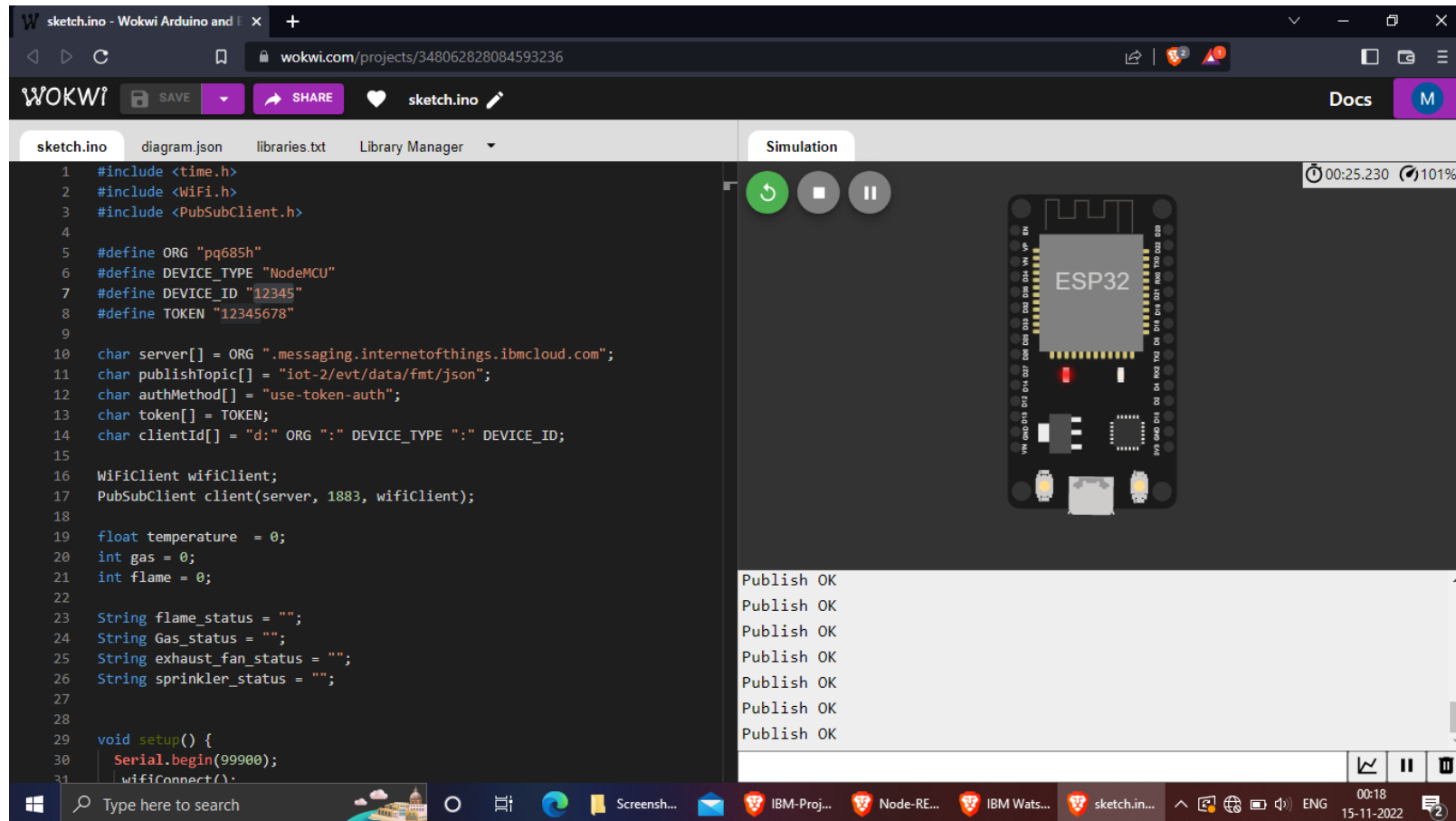
NODE-RED DASHBOARD UILINK:

<https://node-red-iwivz-2022-11-13.eu-gb.mybluemix.net/ui/#!/0?socketid=RNNTsORzKbrlp-UqAAAu>

WEB UI LINK : <https://node-red-dashboard059.eu-gb.mybluemix.net/fire>

OUTPUT:

WOKWI SIMULATOR



IBM WATSON OUTPUT

The screenshot displays the IBM Watson IoT Platform interface in a web browser. The URL is `pq685h.internetofthings.ibmcloud.com/dashboard/devices/browse`. The user is logged in as `mathevanvj15@gmail.com` with ID `pq685h`. The dashboard shows a list of devices, with one device (ID 12345) selected. The device is connected and is a NodeMCU. The 'Recent Events' tab is active, showing a live stream of data events. The events are listed in a table with columns: Event, Value, Format, and Last Received. The events are JSON objects containing gas, temperature, and flame data. A status box at the bottom right indicates '0 Simulations running'.

IBM Watson IoT Platform

mathevanvj15@gmail.com
ID: pq685h

Add Device

Device ID Status Device Type Class ID Date Added Descriptive Location

12345 Connected NodeMCU Device Nov 13, 2022 4:18 PM

Identity Device Information Recent Events State Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
data	{"gas":201,"temperature":106,"flame":807,"fire_...	json	a few seconds ago
data	{"gas":535,"temperature":-5,"flame":618,"fire_st...	json	a few seconds ago
data	{"gas":909,"temperature":113,"flame":706,"fire_...	json	a few seconds ago
data	{"gas":424,"temperature":5,"flame":231,"fire_sta...	json	a few seconds ago
data	{"gas":608,"temperature":103,"flame":777,"fire_...	json	a few seconds ago

0 Simulations running

TRANSFERRING DATA FROM IBM WATSON INTO NODE-RED

The screenshot displays the Node-RED web interface in a browser. The main workspace shows a flow named 'Flow 1' with the following components:

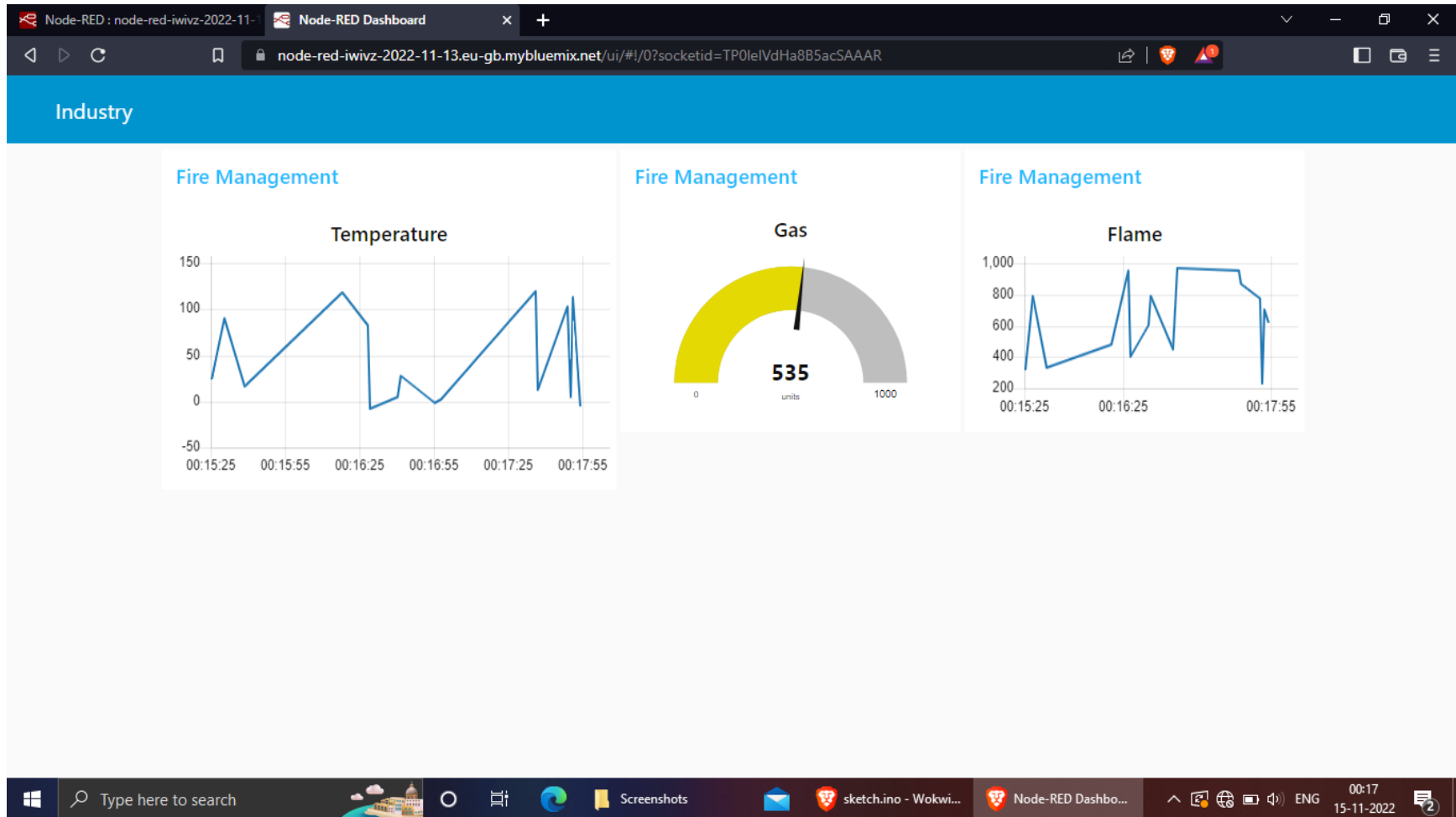
- IBM IoT Node:** A blue node labeled 'connected' that receives data from the IBM Watson IoT platform.
- Function Nodes:** Seven orange nodes with function icons, each connected to the IBM IoT node:
 - temp
 - Gas
 - Flame
 - Fire Status
 - Sprinkler Status
 - Gas Status
 - Exhaust Fan Status
- msg.payload Node:** A green node that receives data from the function nodes.
- Output Nodes:** Three nodes connected to the msg.payload node:
 - Temperature (chart node)
 - Gas (gauge node)
 - Flame (chart node)

The right-hand sidebar shows the 'debug' console with a list of messages. The messages are as follows:

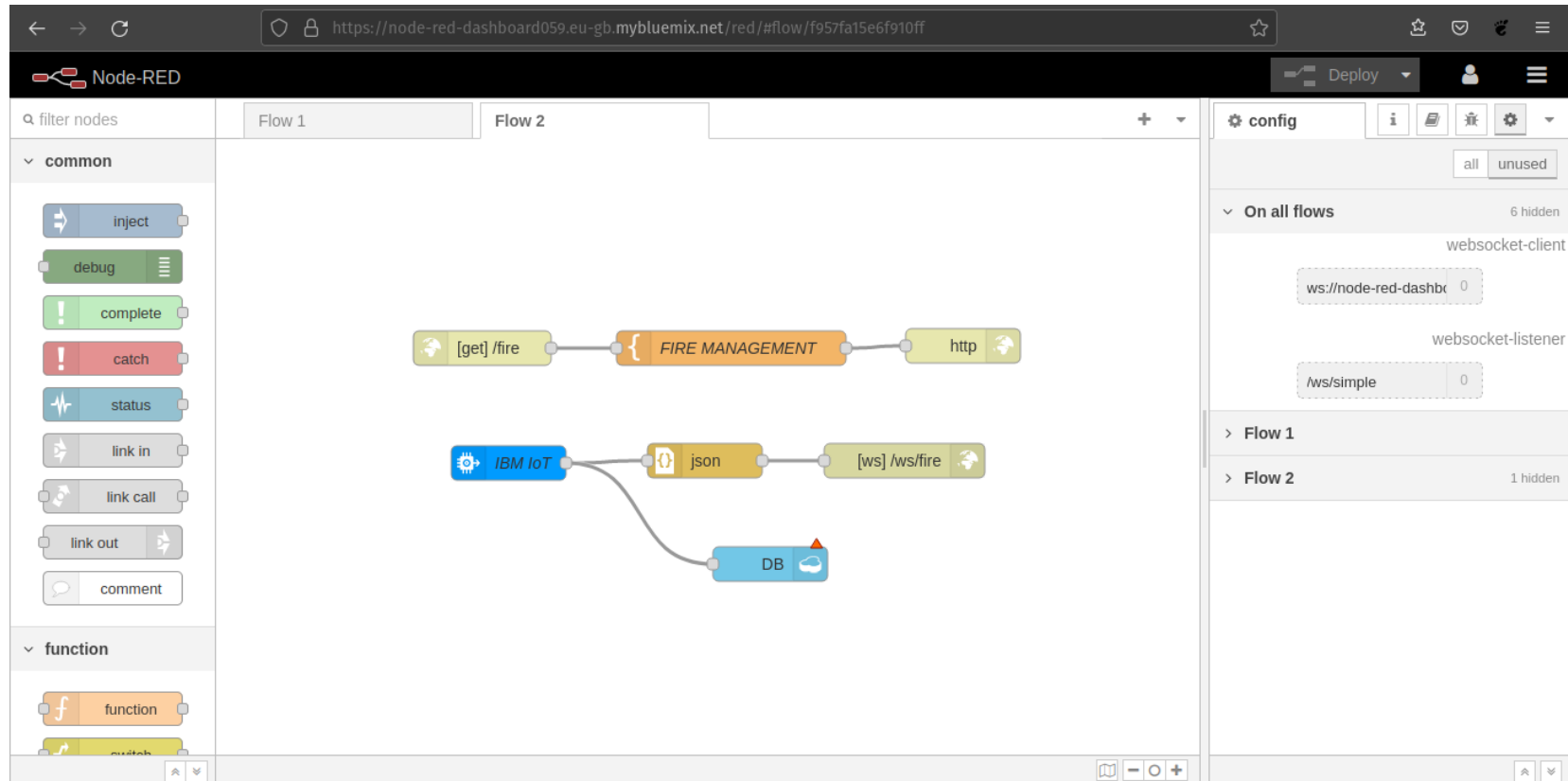
```
..
msg.payload : number
670
11/15/2022, 12:15:39 AM node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : number
332
11/15/2022, 12:15:40 AM node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[7]
"No Fire"
11/15/2022, 12:15:41 AM node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[11]
"Not Working"
11/15/2022, 12:15:42 AM node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[23]
"Gas Leakage is Detected"
11/15/2022, 12:15:43 AM node: f2f2649a.0d0d98
iot-2/type/NodeMCU/id/12345/evt/data/fmt/json :
msg.payload : string[7]
"Working"
```

The bottom of the image shows the Windows taskbar with the search bar and several open applications: sketch.ino - Wokwi..., Node-RED : node-r..., and a system clock showing 00:17 on 15-11-2022.

NODE DASHBOARD



TRANSFERRING DATA FROM NODE-RED INTO WEB UI

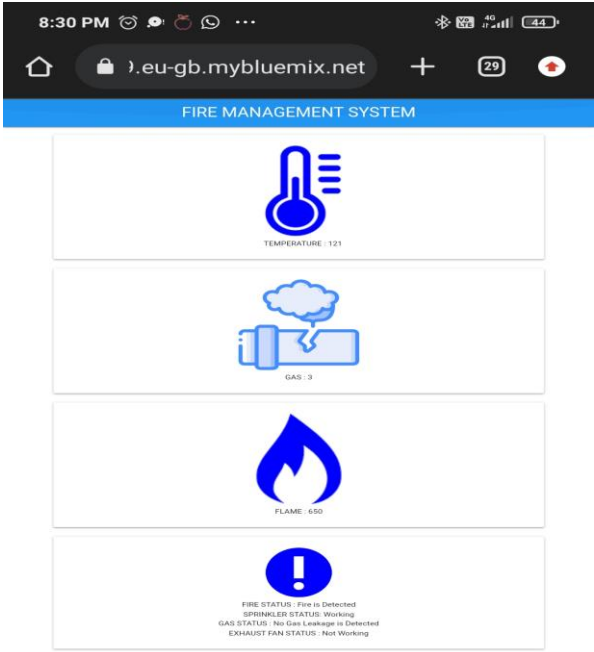


WEB UI

DESKTOP VIEW



MOBILE VIEW



[←](#) [→](#) [↺](#) https://5dd9166f-e1b8-473e-bd7c-b2917517f267-bluemix.cloudant.com/dashboard.html#/database/db/_all_docs ☆ ⚙️ 📄 🔔

↕

< db :

Document ID ▾

Options ⚙️ { } JSON 📖 🔔

All Documents +

Query

Permissions

Changes

Design Documents +

☐

Table Metadata { } JSON 🗒

Create Document

	id	key	value
☐ 📄	657846f21e0cb8ead462fd89321d28...	657846f21e0cb8ead462fd89321d28...	{ "rev": "1-1c9683229f242d4133b7f..." }
☐ 📄	657846f21e0cb8ead462fd89321dd3...	657846f21e0cb8ead462fd89321dd3...	{ "rev": "1-8aeed9d453a632f539ee9c..." }
☐ 📄	657846f21e0cb8ead462fd8932201e...	657846f21e0cb8ead462fd8932201e...	{ "rev": "1-7b6df30912cf9fde43ca8b..." }
☐ 📄	657846f21e0cb8ead462fd8932203d...	657846f21e0cb8ead462fd8932203d...	{ "rev": "1-a9bec25d7f94ccc71ce692..." }
☐ 📄	70ea2e4bb2a9c635be3ce2603a25a...	70ea2e4bb2a9c635be3ce2603a25a...	{ "rev": "1-b567b4cce122c31e1666fc..." }
☐ 📄	70ea2e4bb2a9c635be3ce2603a268...	70ea2e4bb2a9c635be3ce2603a268...	{ "rev": "1-217497b95c16c3d228800..." }
☐ 📄	70ea2e4bb2a9c635be3ce2603a272...	70ea2e4bb2a9c635be3ce2603a272...	{ "rev": "1-a01738b27517a2bb4b93b..." }
☐ 📄	70ea2e4bb2a9c635be3ce2603a273...	70ea2e4bb2a9c635be3ce2603a273...	{ "rev": "1-13230a9f364a021a02422..." }
☐ 📄	7170def319e06e12e85b74c728897...	7170def319e06e12e85b74c728897...	{ "rev": "1-4bdfcbf4dbbf888784fc24d..." }
☐ 📄	7170def319e06e12e85b74c7288b7...	7170def319e06e12e85b74c7288b7...	{ "rev": "1-5b1a46d23a6c259bd5b97..." }
☐ 📄	7170def319e06e12e85b74c7288c2...	7170def319e06e12e85b74c7288c2...	{ "rev": "1-782ab5b4e98aed22641a1..." }

Showing document 1 - 20. Documents per page: 20 ▾ < >

CODE:

```
#include <time.h>
#include <WiFi.h>
#include <PubSubClient.h>

#define ORG "pq685h"
#define DEVICE_TYPE "NodeMCU"
#define DEVICE_ID "12345"
#define TOKEN "12345678"

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/data/fmt/json";
char authMethod[] = "use-token-auth";          char
token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

WiFiClient wifiClient;
PubSubClient client(server, 1883, wifiClient);

float temperature = 0;
int gas = 0; int flame
= 0;
String flame_status = "";
```

```
String Gas_status = "";  
String exhaust_fan_status = "";  
String sprinkler_status = "";
```

```
void setup() {  
  Serial.begin(99900);  
  wifiConnect();  mqttConnect();  
}
```

```
void loop() {
```

```
  srand(time(0));
```

```
  //initial variables and random generated data
```

```
  temperature = random(-20,125);  gas =  
  random(0,1000);  int flamereading =  
  random(200,1024);  flame =  
  map(flamereading,200,1024,0,2);
```

```
  //set a flame status
```

```
  switch (flame) {  case 0:  
    flame_status = "No Fire";
```

```
        break;    case 1:
flame_status = "Fire is Detected";
        break;
    }
```

```
//send the sprinkler status
```

```
    if(flame==1){
        sprinkler_status = "Working";
    }
    else{
        sprinkler_status = "Not Working";

    }
```

```
//toggle the fan according to gas reading
```

```
    if(gas > 100){
        Gas_status = "Gas Leakage is Detected";
        exhaust_fan_status = "Working";
    }
    else{
        Gas_status = "No Gas Leakage is Detected";
        exhaust_fan_status = "Not Working";
    }
```

```
}
```

```
//json format for IBM Watson
```

```
String payload = "{";    payload+="\"gas\":";
payload+=gas;    payload+=",";
payload+="\"temperature\":";
payload+=(int)temperature;    payload+=",";
payload+="\"flame\":";    payload+=flamereading;
payload+=",";
payload+="\"fire_status\":"+"\""+flame_status+"\"",";
payload+="\"sprinkler_status\":"+"\""+sprinkler_status+"\"",";
payload+="\"Gas_status\":"+"\""+Gas_status+"\"",";
    payload+="\"exhaust_fan_status\":"+"\""+exhaust_fan_status+"\""}";
```

```
if(client.publish(publishTopic, (char*) payload.c_str()))
{
    Serial.println("Publish OK");
}
else{
    Serial.println("Publish failed");
}
delay(1000);
```

```
    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);  
    }  
  
    Serial.println();  
  }  
}
```