

## Project development phase

### Sprint - II

Date	04 November 2022
Team ID	PNT2022TMID13566
Project Name	Project - Industry-specific intelligent fire management system
Maximum Marks	20 marks

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

**OUTPUT:**

IBM-Project-2847-1658484090/Sprint-

sketch.ino - Wokwi Arduino and E

New Issue - IBM-EPBL/IBM-Project-2847

wokwi.com/projects/348062828084593236

WOKWI

SAVE

SHARE

sketch.ino

Docs

M

sketch.ino

diagram.json

libraries.txt

Library Manager

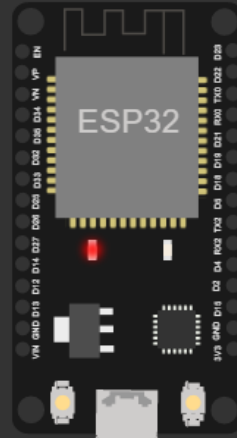
```
1  #include <time.h>
2  #include <WiFi.h>
3  #include <PubSubClient.h>
4
5  #define ORG "pq685h"
6  #define DEVICE_TYPE "NodeMCU"
7  #define DEVICE_ID "12345"
8  #define TOKEN "12345678"
9
10 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
11 char publishTopic[] = "iot-2/evt/data/fmt/json";
12 char authMethod[] = "use-token-auth";
13 char token[] = TOKEN;
14 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
15
16 WiFiClient wificlient;
17 PubSubClient client(server, 1883, wificlient);
18
19 float temperature = 0;
20 int gas = 0;
21 int flame = 0;
22
23 String flame_status = "";
24 String Gas_status = "";
25 String exhaust_fan_status = "";
26 String sprinkler_status = "";
27
28
29 void setup() {
30   Serial.begin(99900);
31   wifiConnect();
```

Simulation

00:07.280

101%

ESP32



Connecting to Wifi...WiFi connected, IP address: 10.10.0.2  
Reconnecting MQTT client to pq685h.messaging.internetofthings.ibmcloud.com  
  
Publish OK  
Publish OK  
Publish OK  
Publish OK  
Publish OK

Type here to search

File...

fi...

IB...

sk...

IB...

io...

id...

W...

S...

ENG

20:55

13-11-2022

## **CODE:**

```
#include <time.h>
#include <WiFi.h>
#include <PubSubClient.h>
#define ORG "wt19pm"
#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.printl

```

```
n("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();
} }
```