

## Source Code :

<b>DATE</b>	<b>19 NOVEMBER 2022</b>
<b>TEAM ID</b>	<b>PNT2022TMID42644</b>
<b>PROJECT TITLE</b>	<b>INDUSTRY-SPECIFIC INTELLIGENT FIRE MANAGEMENT SYSTEM</b>

```
#include <time.h>
#include <WiFi.h>
#include <PubSubClient.h>
bool exhaust_fan_on =
false; bool sprinkler_on =
false; float temperature =
0; int gas_level = 0; int
flame = 0; String
flame_status = "";
String accident_status = "";
String sprinkler_status = ""; void
setup() {
Serial.begin(99900); }
void loop() { //setting a
random seed
srand(time(0)); //initial
variable temperature =
random(- 20,125);
gas_level = random(0,200);
int flamereading =
random(200,1024); flame =
map(flamereading,0,1024,0,
2);
//set a flame status
Serial.print("Temperature : ");
Serial.println(temperature);

Serial.print("Gas_level : ");
Serial.println(gas_level);

Serial.print("Flame : ");
Serial.println(flame);

switch (flame) { case
0:
flame_status = "No Fire";
Serial.println("Flame Status : "+flame_status);
break; case 1:
```

```

flame_status = "Fire is Detected";
Serial.println("Flame Status : "+flame_status); break;
}
//Gas Detection if(gas_level
> 100){
Serial.println("Gas Status : Gas leakage Detected");
} else{
exhaust_fan_on = false;
Serial.println("Gas Status : No Gas leakage Detected");
}
//send the sprinkler status
if(flame){
sprinkler_status
="Sprinkler ON";
Serial.println("Sprinkler Status : "+sprinkler_status);
} else{
sprinkler_status = "Sprinkler OFF";
Serial.println("Sprinkler Status : "+sprinkler_status);
}
//toggle the fan according to gas
if(gas_level > 100)
{
exhaust_fan_on = true;
Serial.println("Exhaust fan Status : Fan ON");
}
else{
exhaust_fan_on = false;
Serial.println("Exhaust fan Status : Fan OFF");
}
Serial.println("");
Serial.println("");
Serial.println(" -----#####-----");
");
Serial.println("");
Serial.println(""); delay(3000);

}

```

## OUTPUT :

The screenshot displays the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, showing a C++ program that simulates a fire detection system. The code includes libraries for time, WiFi, and PubSubClient. It defines variables for fan status, sprinkler status, temperature, gas level, and flame. The setup function initializes the serial port at 999000 baud. The loop function sets a random seed, initializes variables, and generates random values for temperature, gas level, and flame. It then maps the flame reading to a status and prints the current state.

```
1 #include <time.h>
2 #include <WiFi.h>
3 #include <PubSubClient.h>
4
5 bool exhaust_fan_on = false;
6 bool sprinkler_on = false;
7 float temperature = 0;
8 int gas_level = 0;
9 int flame = 0;
10 String flame_status = "";
11 String accident_status = "";
12 String sprinkler_status = "";
13 void setup() {
14   Serial.begin(999000);
15 }
16 void loop() {
17   //setting a random seed
18   srand(time(0));
19   //initial variable
20   temperature = random(-
21     20,125);
22   gas_level = random(0,200);
23   int flamereading =
24     random(200,1024);
25   flame =
26     map(flamereading,0,1024,0,
27     2);
28   //set a flame status
29   Serial.print("Temperature : ");
```

On the right, the 'Simulation' tab is active, showing the output of the program. The output is divided into two sections by a dashed line with a hash symbol. The first section shows the initial state where the gas level is 15 and the exhaust fan is off. The second section shows a state where the gas level is 116 and the exhaust fan is on.

Simulation

Temperature : 116.00  
Gas\_level : 15  
Flame : 1  
Flame Status : Fire is Detected  
Gas Status : No Gas leakage Detected  
Sprinkler Status : Sprinkler ON  
Exhaust fan Status : Fan OFF

-----#####-----

Temperature : 72.00  
Gas\_level : 116  
Flame : 1  
Flame Status : Fire is Detected  
Gas Status : Gas leakage Detected  
Sprinkler Status : Sprinkler ON  
Exhaust fan Status : Fan ON