

Sprint - 2

Date	04 November 2022
Team ID	PNT2022TMID4760
Project Name	Project - Industry-specific intelligent fire management system
Maximum Marks	20 marks

Sprint-2	US-1	Configure the connection security and create API keys that are used in the Node-RED service for accessing the IBM IoT Platform.	10	High	Indhumathi K,Hariharan S
Sprint-2	US-2	Create a Node-RED service.	10	High	Athira V R, ArunRaj G

US-1 Configure the connection security and create API keys that are used in the Node-RED service for accessing the IBM IoT Platform.


US-2 Create a Node-RED service.

US-1 Configure the connection security and create API keys that are used in the Node-RED service for accessing the IBM IoT Platform.

The API key has been added.

Authentication tokens are non-recoverable. If you misplace this token, you will need to re-register the API key to generate a new authentication token.

Generated Details

API Key a-4aqwut-gahbbnkql5 

Authentication Token dtAhr+HB3E-xIpbAgZ 



Make a note of the generated authentication token. Lost authentication tokens cannot be recovered. If you lose the token, you must reregister the API to generate a new token.

API Key Information

Description -

Role Standard Application

Expires Never

US-2 Create a Node-RED service

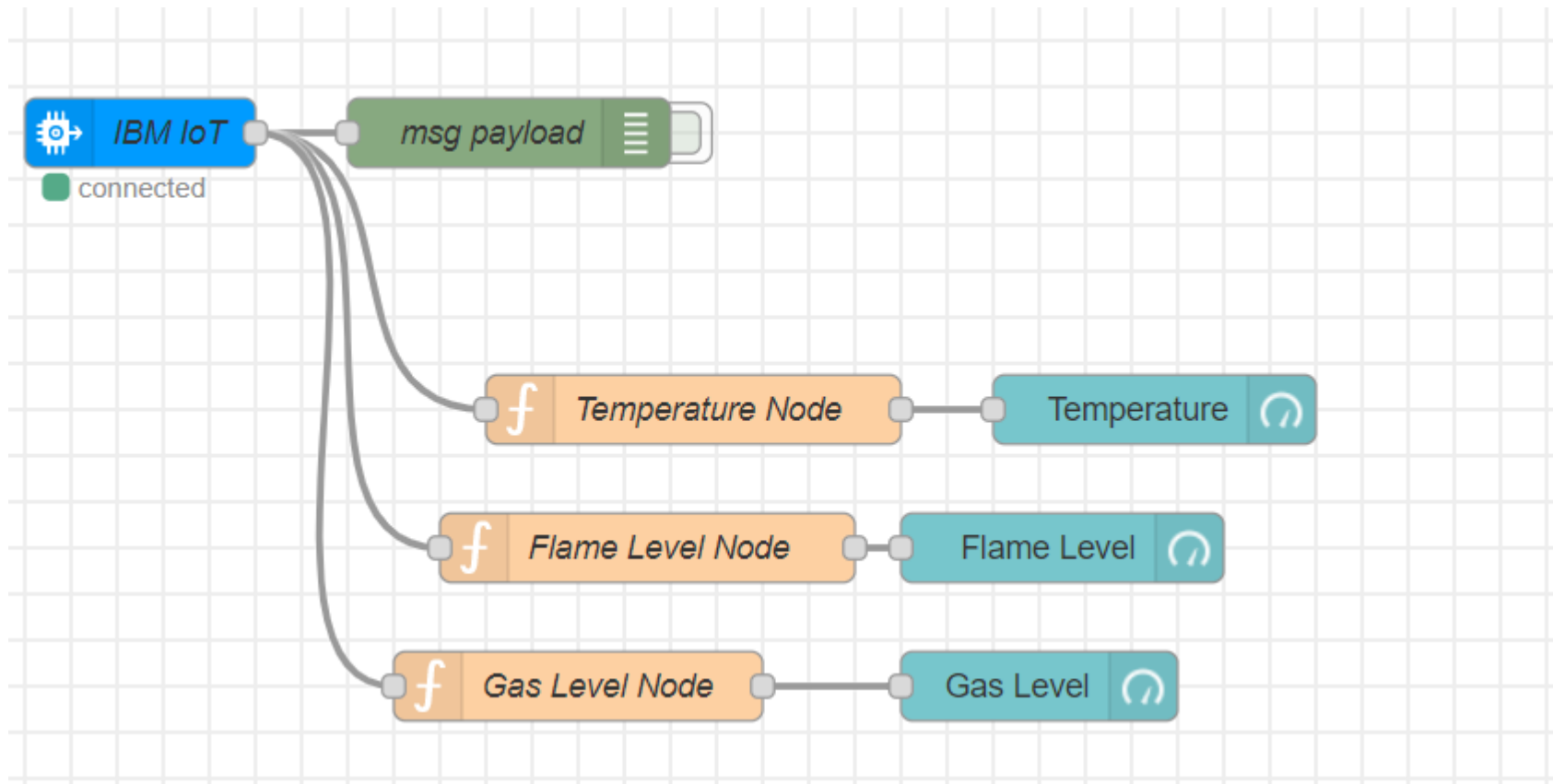


fig1 - Monitoring the sensor values - Temperature, Flame Level, Gas Level. These values are randomly generated by IBM WATSON IOT PLATFORM.

```
11/3/2022, 9:04:47 AM  node: msg payload
iot-2/type/B11M3EDeviceType/id/B11M3EDeviceID/evt/event_1/fmt/json : msg.payload : Object
  ▶ { Temperature: 1, Flame_Level: 62, Gas_Level: 38 }
```

```
11/3/2022, 9:04:50 AM  node: msg payload
iot-2/type/B11M3EDeviceType/id/B11M3EDeviceID/evt/event_1/fmt/json : msg.payload : Object
  ▶ { Temperature: 1, Flame_Level: 78, Gas_Level: 11 }
```

```
11/3/2022, 9:04:53 AM  node: msg payload
iot-2/type/B11M3EDeviceType/id/B11M3EDeviceID/evt/event_1/fmt/json : msg.payload : Object
  ▶ { Temperature: 99, Flame_Level: 36, Gas_Level: 55 }
```

```
11/3/2022, 9:04:56 AM  node: msg payload
iot-2/type/B11M3EDeviceType/id/B11M3EDeviceID/evt/event_1/fmt/json : msg.payload : Object
  ▶ { Temperature: 71, Flame_Level: 24, Gas_Level: 46 }
```

```
11/3/2022, 9:05:00 AM  node: msg payload
iot-2/type/B11M3EDeviceType/id/B11M3EDeviceID/evt/event_1/fmt/json : msg.payload : Object
  ▶ { Temperature: 38, Flame_Level: 92, Gas_Level: 63 }
```

```
11/3/2022, 9:05:03 AM  node: msg payload
iot-2/type/B11M3EDeviceType/id/B11M3EDeviceID/evt/event_1/fmt/json : msg.payload : Object
  ▶ { Temperature: 74, Flame_Level: 98, Gas_Level: 84 }
```

```
11/3/2022, 9:05:06 AM  node: msg payload
iot-2/type/B11M3EDeviceType/id/B11M3EDeviceID/evt/event_1/fmt/json : msg.payload : Object
  ▶ { Temperature: 87, Flame_Level: 81, Gas_Level: 44 }
```

Fig 2 - Temperature, Flame_Level, Gas_Level values displayed in deploy tab in node-red

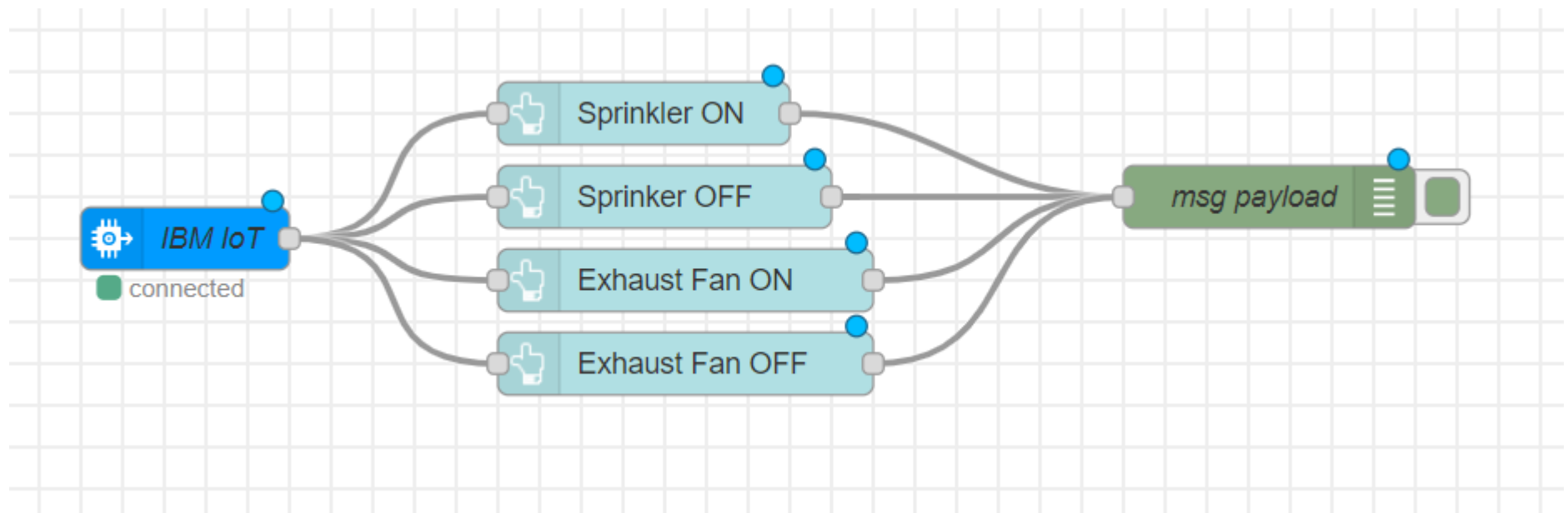


fig 3 - Control buttons (Sprinkler ON, Sprinkler OFF, Exhaust Fan ON, Exhaust Fan OFF)

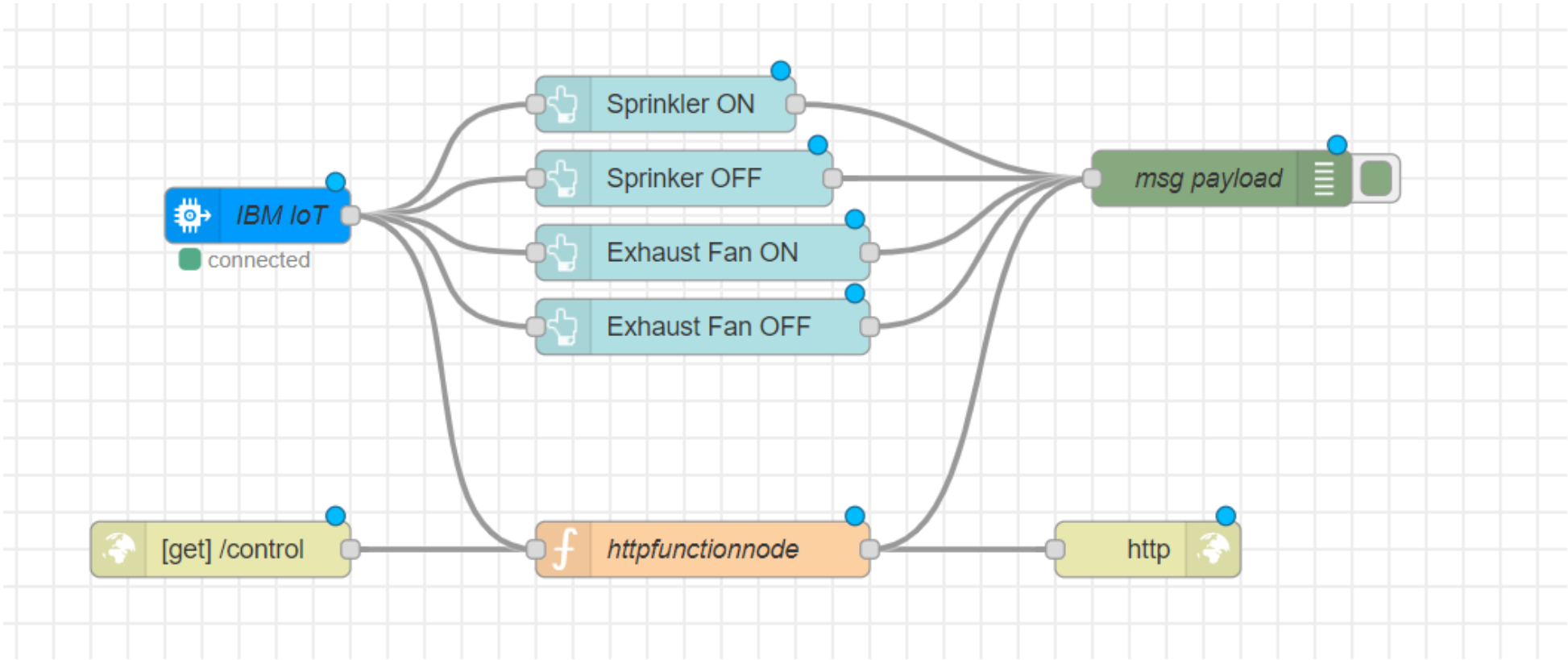


Fig 4 - Using HTTP in and HTTP response in network option, <http://127.0.0.1:1880/#flow/f74f1b96473dc208/control> will display the control options

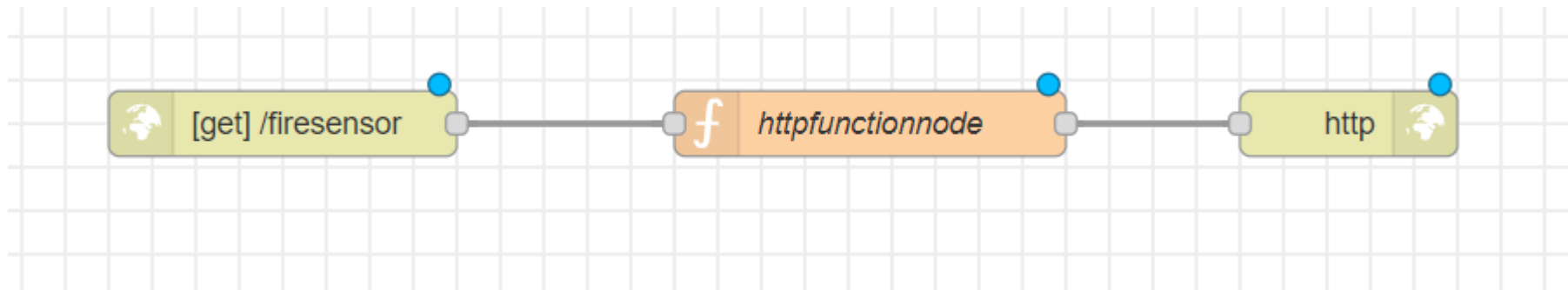


Fig 5 - Using HTTP in and HTTP response in network option, <http://127.0.0.1:1880/#flow/f74f1b96473dc208/firesensor> will display the sensor values like Temperature, Gas_Level and Flame_Level from the IBM WATSON IOT PLATFORM.

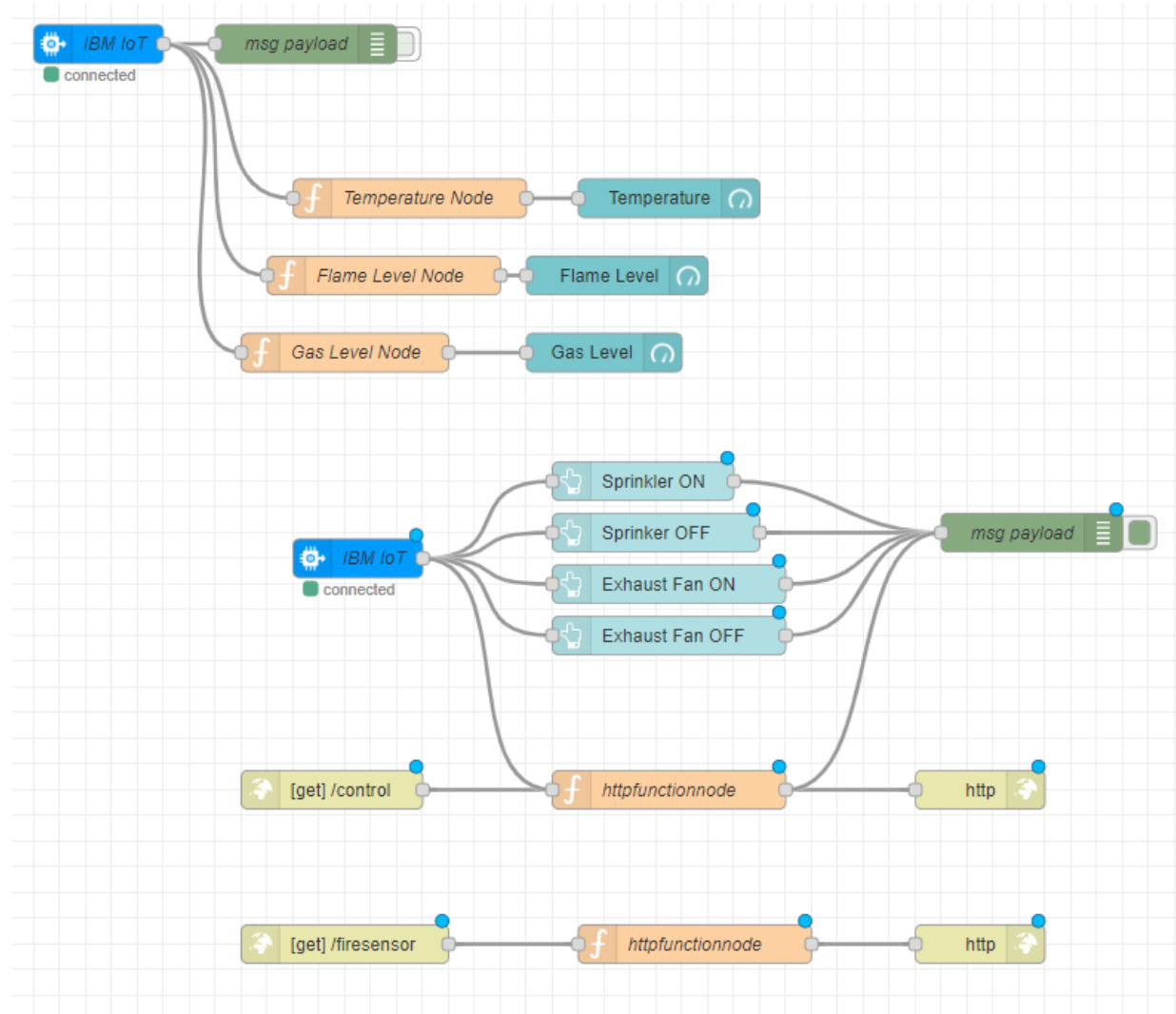


Fig 6 - Entire Node-Red connection for our project

Edit ibmiot in node

Delete

Cancel

Done

⚙ Properties

⚙

📄

🖼

☂ Authentication

API Key

▼

🔍 API Key

a6cb71b59d73b36b

▼

✎

⚙ Input Type

Device Event

▼

🔗 Device Type

☐ All or

B11M3EDeviceType

👤 Device Id

☐ All or

B11M3EDeviceID

📋 Event

☒ All or

+

📄 Format

☐ All or

json

⚙ QoS

0

▼

🔖 Name

IBM IoT

🔖 Service

registered

Fig 7 - Properties of IBM IOT are shown. The API key, Device Type, Device ID are taken from IBM IOT WATSON PLATFORM.

Edit function node

Delete Cancel Done

Properties

Name Temperature Node

Setup On Start **On Message** On Stop

```
1 msg.payload = msg.payload.Temperature
2 global.set('t',msg.payload)
3 return msg;
```

Edit function node

Delete

Cancel

Done

⚙️ Properties

⚙️ 📄 🖨️

🔖 Name

Flame Level Node

📄 ▼

⚙️ Setup

On Start

On Message

On Stop

1 msg.payload = msg.payload.Flame_Level

2 global.set("f",msg.payload)

3 return msg;

Edit function node

Delete

Cancel

Done

⚙️ Properties

⚙️ 📄 🖨️

🔖 Name

Gas Level Node

📄 ▼

⚙️ Setup

On Start

On Message

On Stop

1 msg.payload = msg.payload.Gas_Level

2 global.set("g",msg.payload)

3 return msg;

Fig 8 - Properties of Function Node -Temperature Node, Flame_Level Node, Gas_Level Node.

Edit gauge node

Delete

Cancel

Done

Properties

Group

[Control] Industry specific intelligent fire ▾

Size

auto

Type

Gauge ▾

Label

Temperature

Value format

{{value}}

Units

C

Range

min

max

Colour gradient

Sectors

0 10

Name

Fig 9 - Properties of Temperature Gauge.

Edit gauge node

Delete

Cancel

Done

⚙ Properties

⚙

📄

🖼

📊 Group

[Control] Industry specific intelligent fire ▾

✎

📏 Size

auto

☰ Type

Gauge ▾

🏷 Label

Flame Level

🏷 Value format

{{value}}

🏷 Units

units

Range

min 0

max 10

Colour gradient

Sectors

0

...

optional

...

optional

...

10

🏷 Name

Fig 9 - Properties of Flame_Level Gauge.

Edit gauge node

Delete

Cancel

Done

⚙ Properties

⚙

📄

🔗

📁 Group

[Control] Industry specific intelligent fire ▾

✎

📏 Size

auto

☰ Type

Gauge ▾

🏷 Label

Gas Level

🏷 Value format

{{value}}

🏷 Units

units

Range

min 0

max 10

Colour gradient

Sectors

0

...

optional

...

optional

...

10

🏷 Name

Fig 9 - Properties of Gas_Level Gauge.

Edit ibmiot in node

Delete

Cancel

Done

⚙ Properties

⚙

📄

🖨

🔑 Authentication

API Key

▼

🔑 API Key

a6cb71b59d73b36b

▼

✎

⚙ Input Type

Device Command

▼

🔑 Device Type

☐ All or

B11M3EDeviceType

👤 Device Id

☐ All or

B11M3EDeviceID

📋 Command

☐ All or

onoff

📄 Format

☐ All or

String

⚙ QoS

0

▼

🔑 Name

IBM IoT

🔑 Service

registered

Fig 9 - Properties of IBM IOT Node.

Edit button node

Delete

Cancel

Done

⚙️ Properties

⚙️

📄

🖼️

📊 Group

[Control] Industry specific intelligent fi

✎

📏 Size

auto

🖼️ Icon

optional icon

🏷️ Label

Sprinkler ON

📘 Tooltip

optional tooltip

🔥 Color

optional text/icon color

🔥 Background

optional background color

✉️ When clicked, send:

Payload

▼ {} {"command":"SprinklerON"} ...

Topic

▼ msg. topic

➡️ If msg arrives on input, emulate a button click:

☐

Fig 10 - Properties of Sprinkler ON button node.

Edit http in node

Delete

Cancel

Done

Properties

Method

GET

▼

URL

/control

Name

Name

Fig 10 - Properties of HTTP Node with method GET and URL /control,

Edit function node

Delete

Cancel

Done

Properties

Name

httpfunctionnode

Setup

On Start

On Message

On Stop

```
1 msg.payload = msg.payload.command
2 return msg;
```

Fig 11 - Properties of Control HTTP Function Node.

Edit function node

Delete

Cancel

Done

Properties

Name

httpfunctionnode

Setup

On Start

On Message

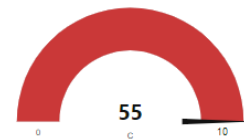
On Stop

```
1 msg.payload={ "Temperature":global.get('t'),
2               "Flame_Level":global.get('f'),
3               "Gas_Level":global.get('g')}
4 return msg;
```

Control

Industry specific intelligent fire management system

Temperature



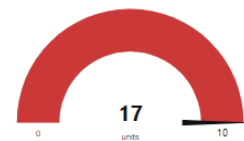
SPRINKLER ON

EXHAUST FAN ON

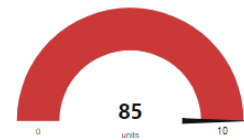
SPRINKLER OFF

EXHAUST FAN OFF

Flame Level



Gas Level



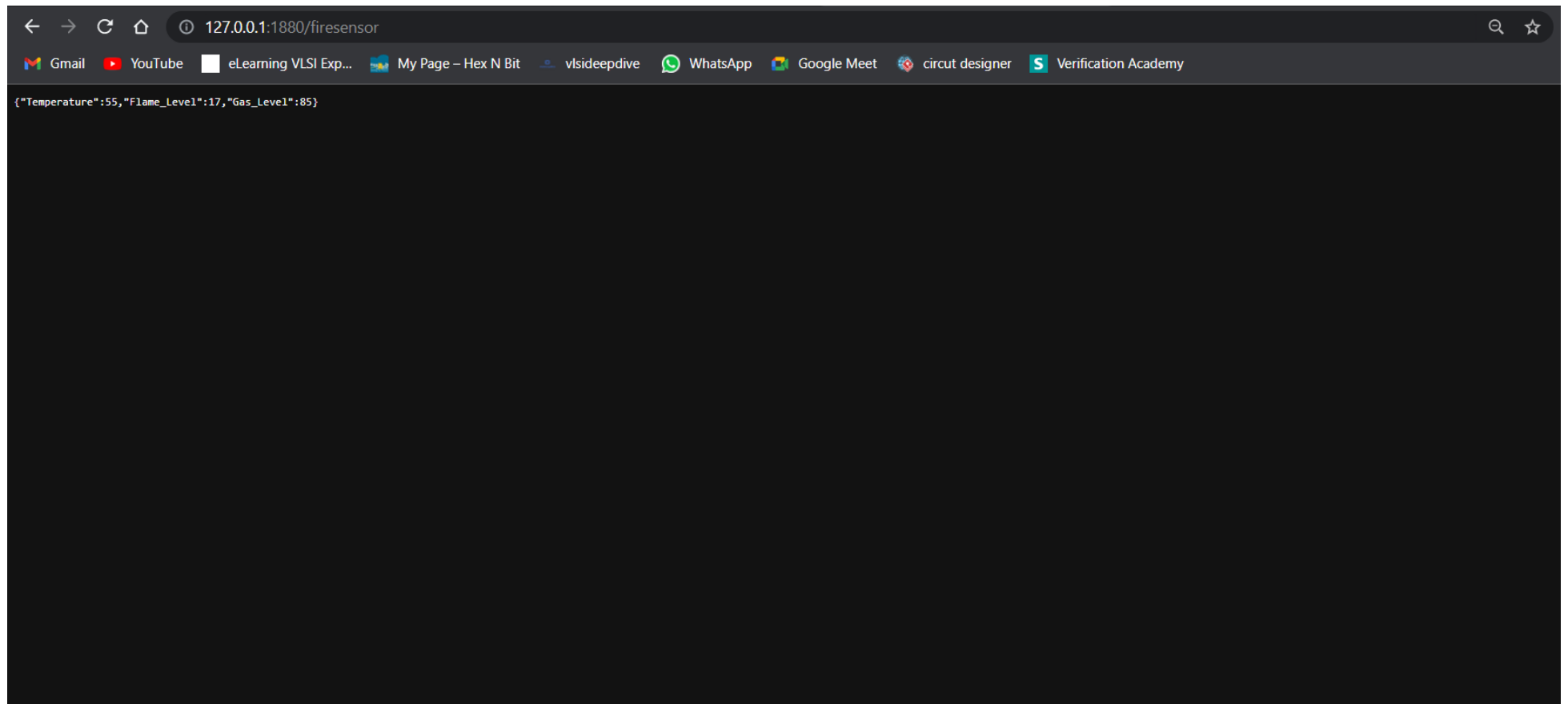


Fig 12 - Properties of Monitor HTTP Function Node

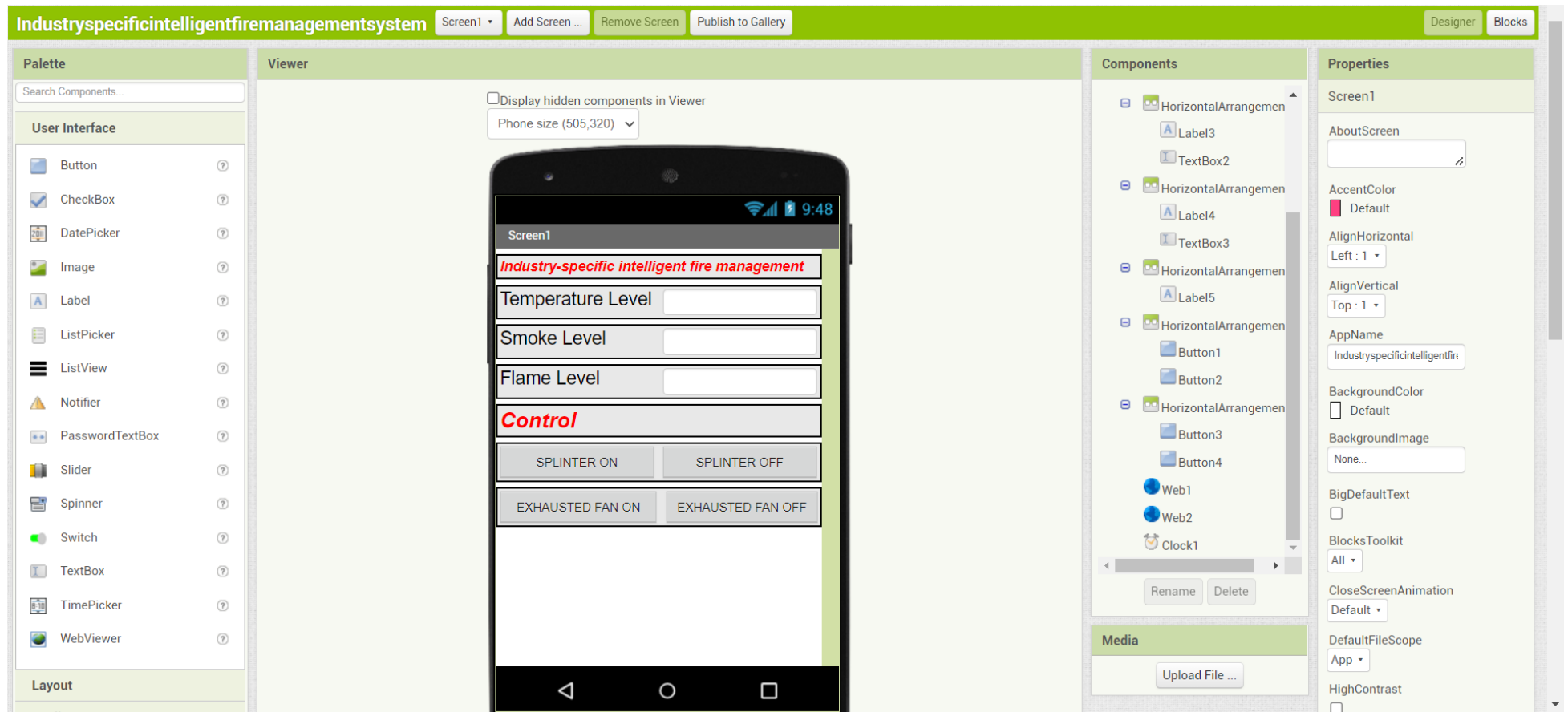


Fig 13 - Front-end APP for our project, to display the Temperature Level, Smoke Level and Flame Level with control buttons like Sprinkler ON and OFF and Exhaust Fan ON and OFF