

SPRINT-2

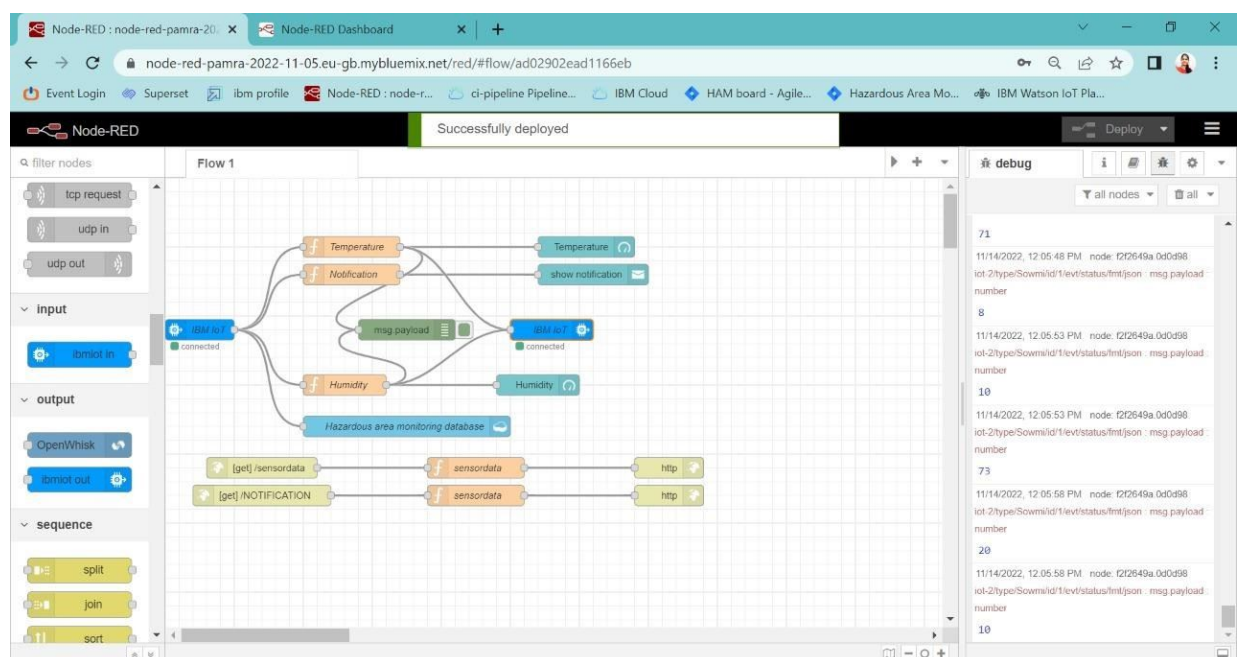
SOFTWARE

Date	19 November 2022
Team ID	PNT2022TMID16791
Project Name	Hazardous Area Monitoring for Industrial Power Plant powered by IOT

PROCEDURE:

- Step1: Develop node red application.
 Step2: Install the required nodes from manage palette option.
 Step3: Connect the node flow. Step4: Deploy the flow.

NODE RED:



FUNCTION NODE COMMAND TO INDICATE THE TEMPERATURE:

`msg.payload=msg.payload.temperature; global.set('temperature',msg.payload);`

`return msg;` FUNCTION NODE COMMAND TO INDICATE THE HUMIDITY:

```
msg.payload=msg.payload.humidity; global.set('humidity',msg.payload);  
return msg;
```

FUNCTION NODE COMMAND TO SHOW THE STATUS:

```
temperature=msg.payload.temperature; humidity=msg.payload.humidity;  
if(temperature>50  
& humidity>50){ msg.payload="Temperature and  
Humidity Alert!";  
}  
else if(temperature>50){ msg.payload="Temperature  
Alert!";  
}  
else if(humidity>50){ msg.payload="Humidity  
Alert!";  
}  
else {msg.payload="safe";} return msg;
```

FUNCTION NODE COMMAND TO GET THE HTTP REQUEST FOR SENSOR DATA:

```
msg.payload={  
  "temperature":global.get('temperature'),  
  "humidity":global.get('humidity')  
}  
return msg;
```

FUNCTION NODE COMMAND TO GET THE HTTP REQUEST FOR STATUS:

```
msg.payload={  
  "temperature":global.get('temperature'),
```

```

    "humidity":global.get('humidity')
}
temperature=msg.payload.temperature; humidity=msg.payload.humidity;
if(temperature>50
& humidity>50){ msg.payload="Temperature and
Humidity Alert!";
}
else if(temperature>50){ msg.payload="Temperature
Alert!";
}
else if(humidity>50){ msg.payload="Humidity
Alert!";
}
else {msg.payload="safe";} return msg;

```

NOTIFICATION NODE:

The screenshot shows the Node-RED web interface in a browser. The main workspace displays a flow with an 'IBM IoT' node connected to several function nodes. The 'Edit notification node' dialog is open, showing the following configuration:

- Layout:** Top Right
- Timeout (S):** 5
- Border:** (optional) border highlight colour
- Send to all browser sessions:** ☒
- Accept raw HTML/JavaScript input in msg.payload to format popup:** ☐
- Class:** [msg.className]
- Topic:** STATUS
- Name:** Name
- Enabled:** ☐ Enabled

The debug console on the right shows a series of messages from the 'iot-2type:Somnild/1/evt/status/rmt/json' topic, each containing a JSON payload with temperature and humidity values:

```

{ temperature: 21, humidity: 42 }
{ temperature: 75, humidity: 52 }
{ temperature: 16, humidity: 80 }
{ temperature: 10, humidity: 19 }
{ temperature: 20, humidity: 8 }

```

GAUGE-TEMPERATURE NODE:

The screenshot shows the Node-RED web interface in a browser. The main workspace displays a flow with an 'IBM IoT' node connected to several output nodes, including a 'Gauge' node. The 'Edit gauge node' dialog is open, showing the following properties:

- Size:** auto
- Type:** Gauge
- Label:** Temperature
- Value format:** {{value}}
- Units:** Centigrade
- Range:** min -20, max 125
- Colour gradient:** A gradient bar with green, yellow, and red segments.
- Sectors:** -20, optional, optional, 125
- Enabled:** ☐ Enabled

The debug console on the right shows a series of messages from the 'iot-2/type/SowmiId/1/evt/status/fmt/json' topic, each containing a JSON payload with temperature and humidity values:

```
{ "temperature": 66, "humidity": 91 }
```

```
{ "temperature": -13, "humidity": 2 }
```

```
{ "temperature": 21, "humidity": 42 }
```

```
{ "temperature": 75, "humidity": 52 }
```

```
{ "temperature": 16, "humidity": 80 }
```

GAUGE-HUMIDITY NODE:

The screenshot shows the Node-RED web interface with the 'Edit gauge node' dialog open for a 'Humidity' gauge. The properties are as follows:

- Group:** [Dashboard] Sensor data
- Size:** auto
- Type:** Gauge
- Label:** Humidity
- Value format:** {{value}}
- Units:** Percentage
- Range:** min 0, max 100
- Colour gradient:** A gradient bar with green, yellow, and red segments.
- Enabled:** ☐ Enabled

The debug console on the right shows a series of messages from the 'iot-2/type/SowmiId/1/evt/status/fmt/json' topic, each containing a JSON payload with temperature and humidity values:

```
{ "temperature": 16, "humidity": 80 }
```

```
{ "temperature": 10, "humidity": 19 }
```

```
{ "temperature": 20, "humidity": 8 }
```

```
{ "temperature": 15, "humidity": 45 }
```

```
{ "temperature": 67, "humidity": 3 }
```