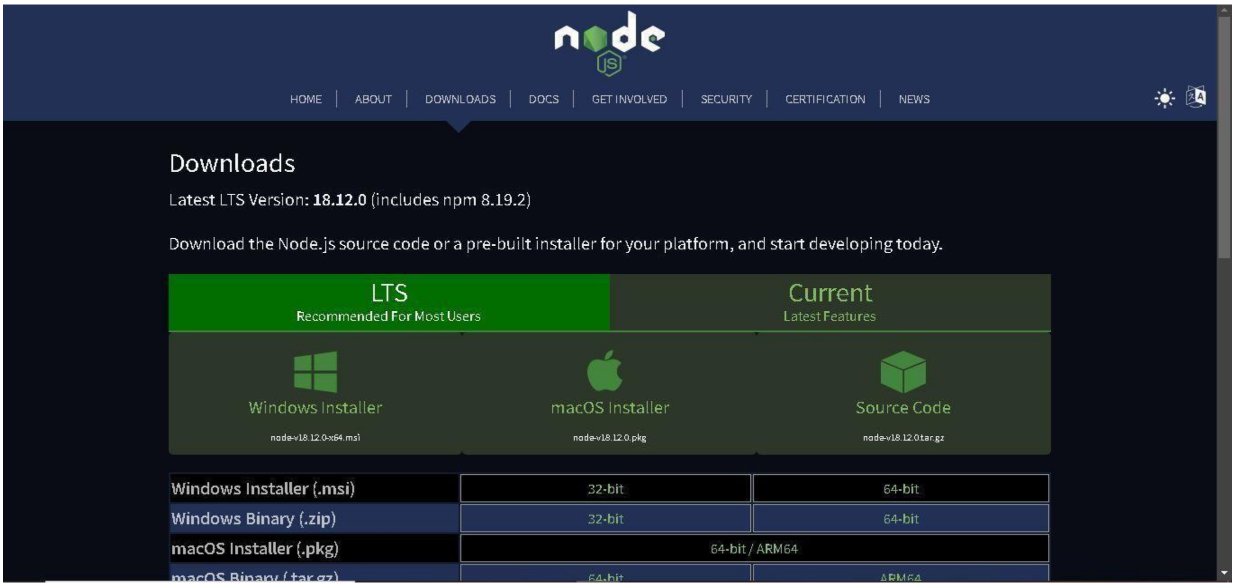
Team ID	PNT2022TMID07710
Project Name	Real Time River Water Quality Monitoring AndControl System

STEP 1: Download and Install node.js.



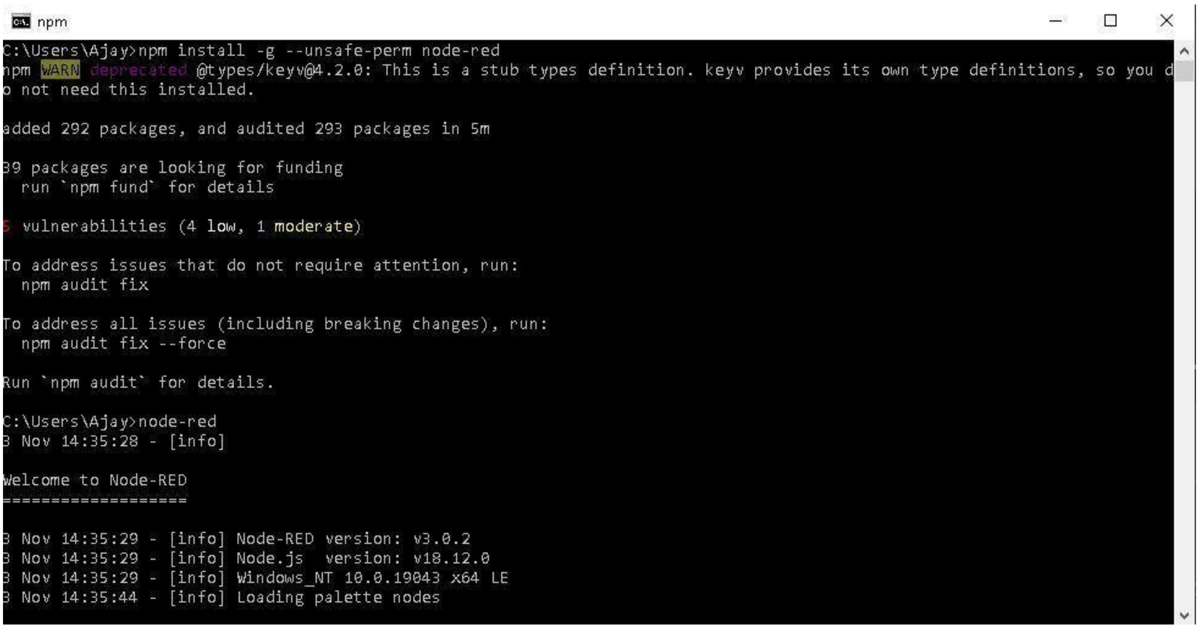
Downloads

Latest LTS Version: **18.12.0** (includes npm 8.19.2)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

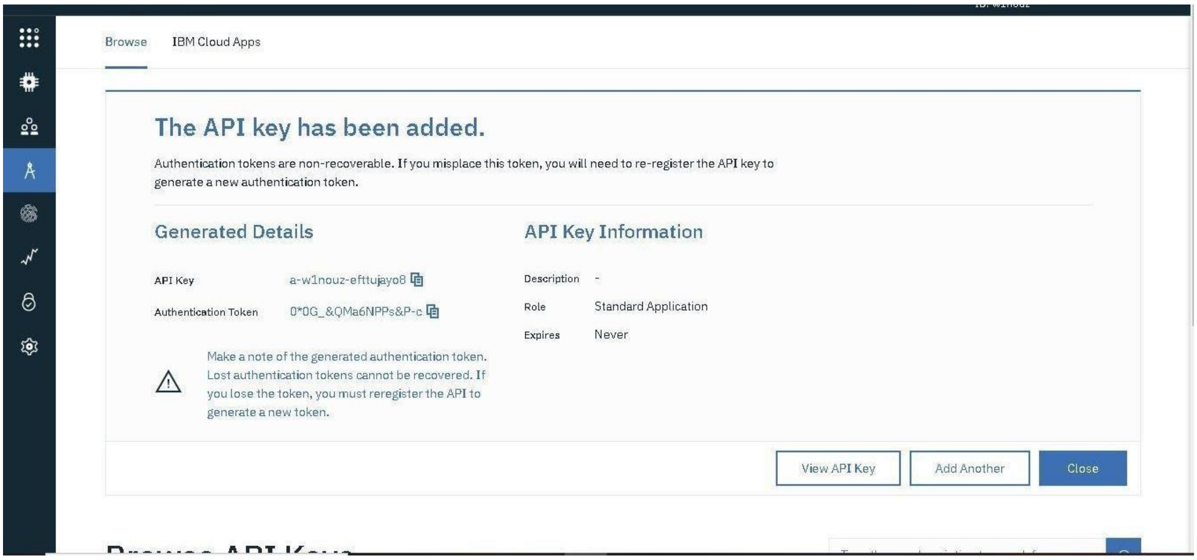
Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit / ARM64	
macOS Binary (.tar.gz)	64-bit	ARM64

STEP 2: Setup node.js and configure command prompt for error check. Open node-red from the generated link.



```
C:\Users\Ajay>npm install -g --unsafe-perm node-red
npm WARN deprecated @types/keyv@4.2.0: This is a stub types definition. keyv provides its own type definitions, so you d
o not need this installed.

added 292 packages, and audited 293 packages in 5m

39 packages are looking for funding
  run `npm fund` for details

5 vulnerabilities (4 low, 1 moderate)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

C:\Users\Ajay>node-red
3 Nov 14:35:28 - [info]

Welcome to Node-RED
===================

3 Nov 14:35:29 - [info] Node-RED version: v3.0.2
3 Nov 14:35:29 - [info] Node.js  version: v18.12.0
3 Nov 14:35:29 - [info] Windows_NT 10.0.19043 x64 LE
3 Nov 14:35:44 - [info] Loading palette nodes
```

STEP 3: Generating API key and Authentication token.



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-w1nouz-efttujayo8

Authentication Token 0*0G_&QMa6NPPs&P-c

⚠ 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.

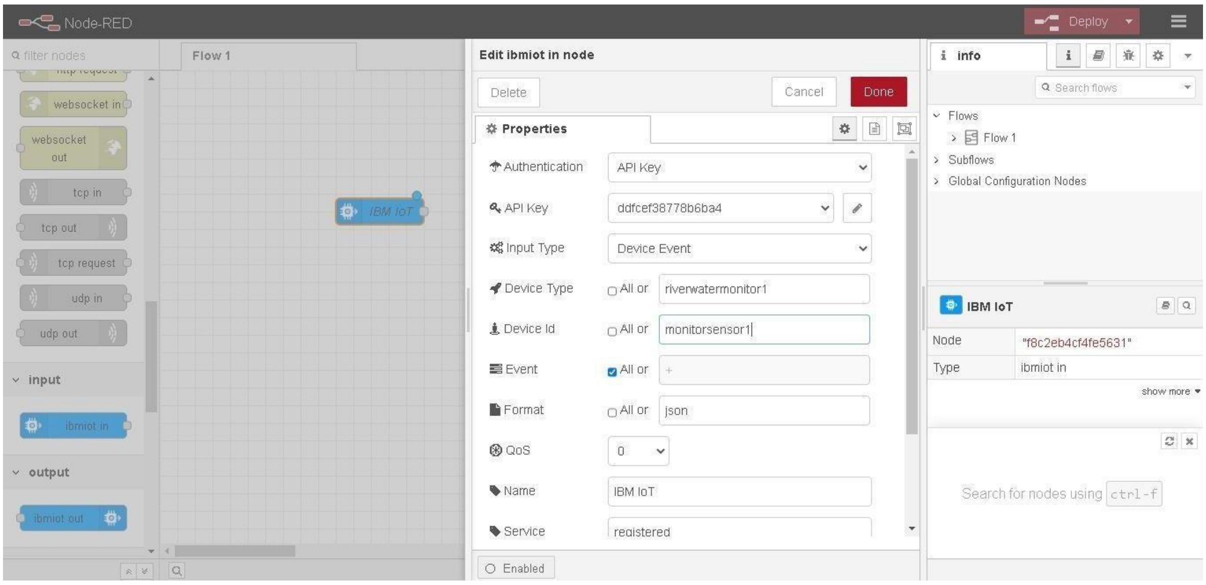
API Key Information

Description -

Role Standard Application

Expires Never

STEP 4: Edit Ibmiot in node.



Edit ibmiot in node

Authentication: API Key

API Key: ddfcef38778b6ba4

Input Type: Device Event

Device Type: □ All or riverwatermonitor1

Device Id: □ All or monitorsensor1

Event: ☑ All or +

Format: □ All or json

QoS: 0

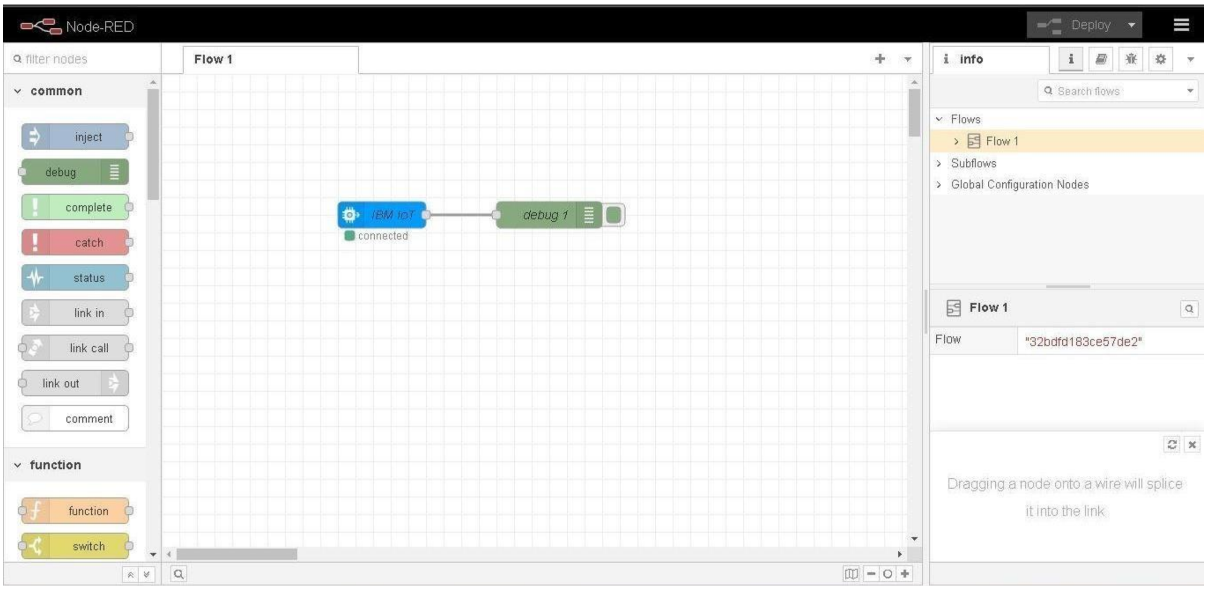
Name: IBM IoT

Service: registered

IBM IoT

Node	"f8c2eb4cf4fe5631"
Type	ibmiot in

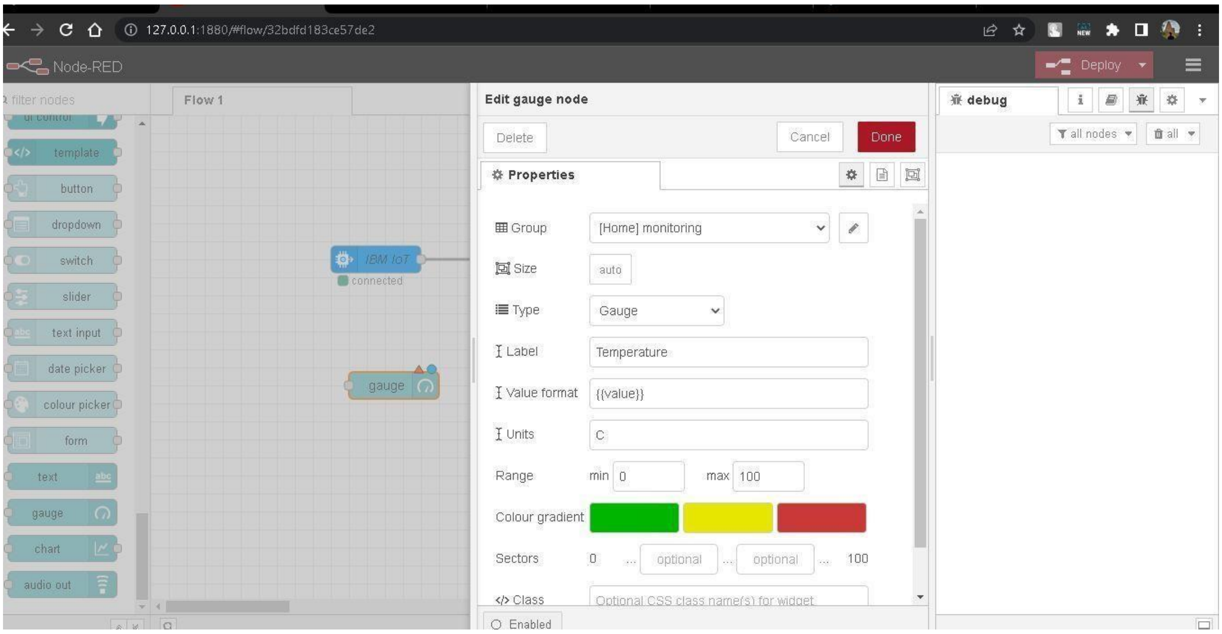
STEP 5: Connect Ibmiot in and debug 1 and deploy.



Flow "32bdfd183ce57de2"

Dragging a node onto a wire will splice it into the link

STEP 6: Edit gauge node (here the gauge nodes are named as Temperature, pH and Turbidity).



127.0.0.1:1880/#flow/32bdfd183ce57de2

Group [Home] monitoring

Size auto

Type Gauge

Label Temperature

Value format {{value}}

Units C

Range min 0 max 100

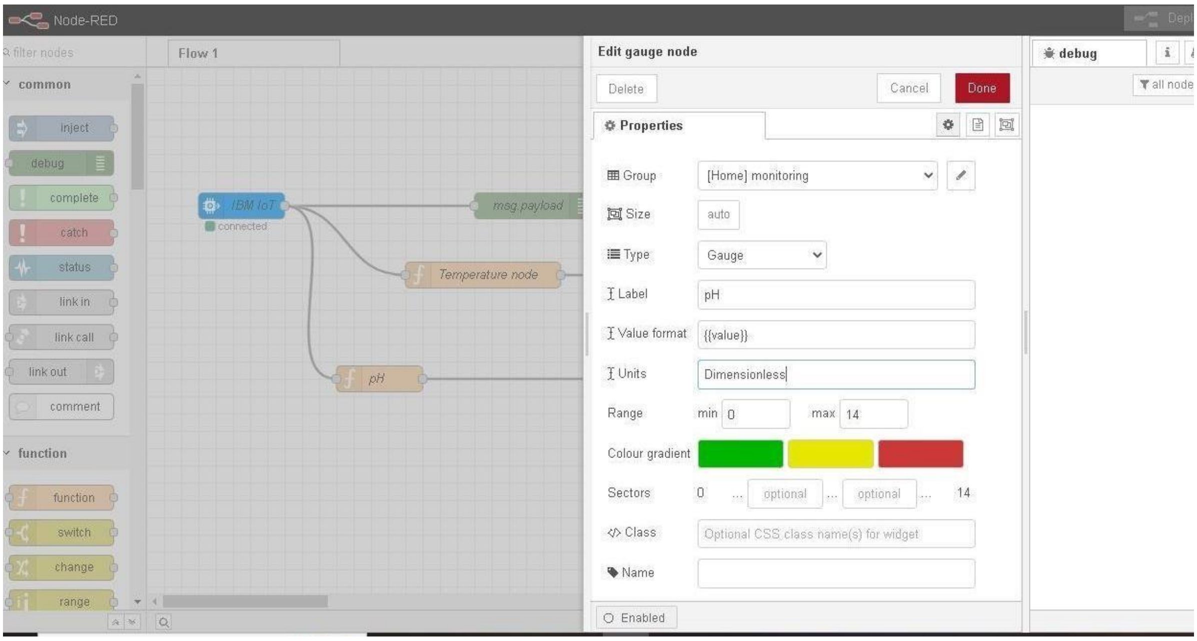
Colour gradient 🟩 🟨 🟥

Sectors 0 ... optional ... optional ... 100

Class Optional CSS class name(s) for widget

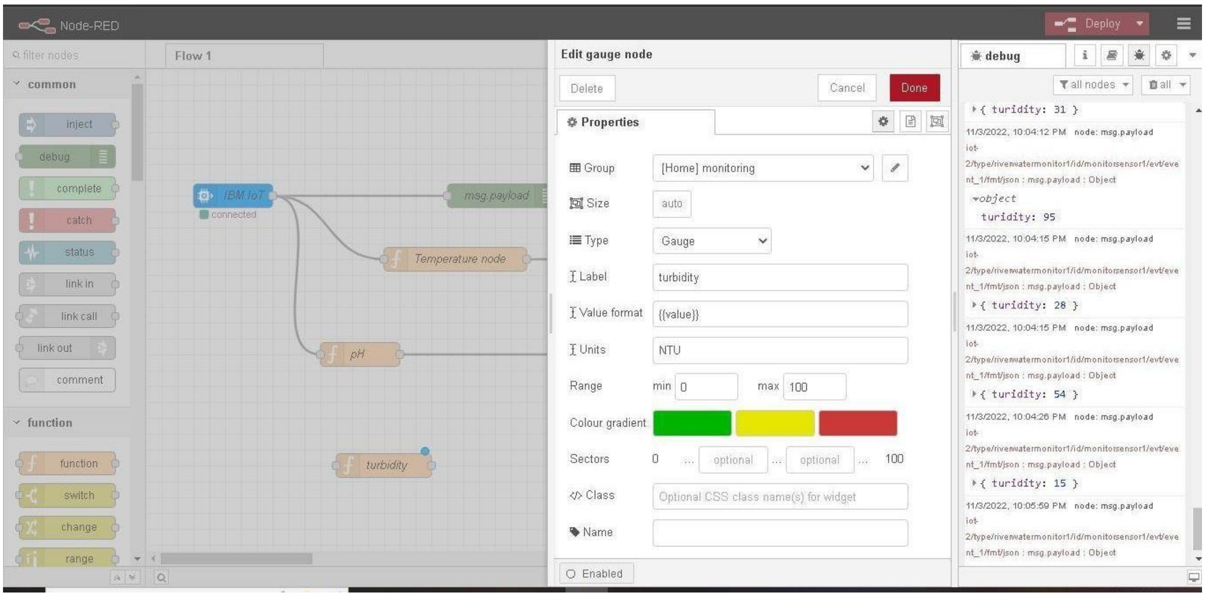
○ Enabled

Fig 1



Group: [Home] monitoring
Size: auto
Type: Gauge
Label: pH
Value format: {{value}}
Units: Dimensionless
Range: min 0 max 14
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 14
Class: Optional CSS class name(s) for widget
Name:

Fig 2

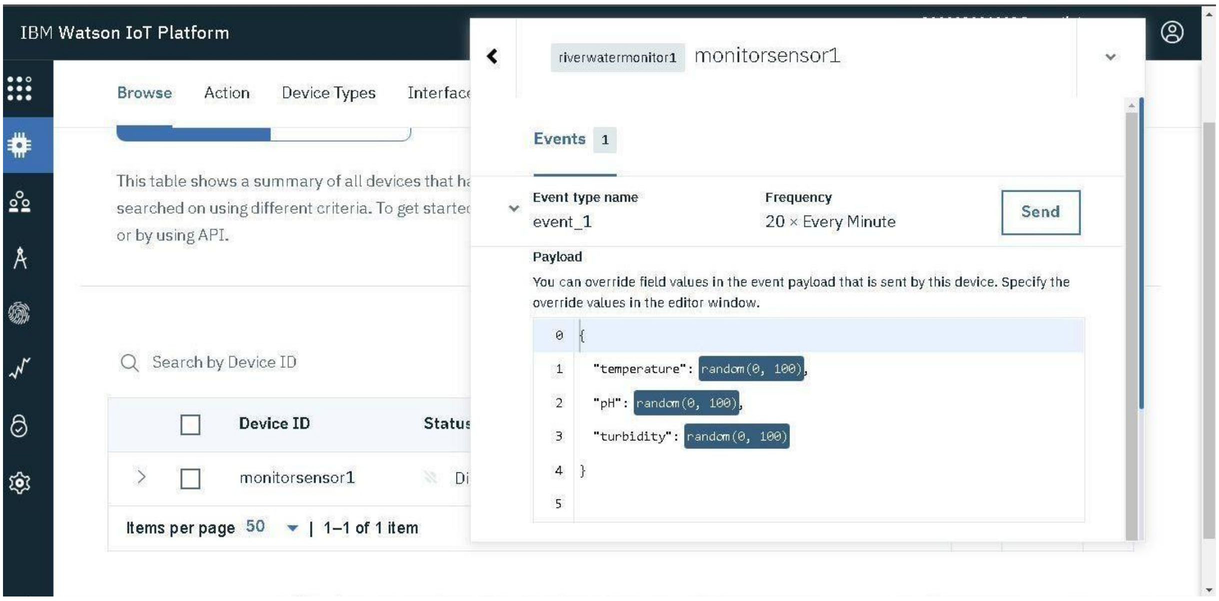


Group: [Home] monitoring
Size: auto
Type: Gauge
Label: turbidity
Value format: {{value}}
Units: NTU
Range: min 0 max 100
Colour gradient: 🟩 🟨 🟥
Sectors: 0 ... optional ... optional ... 100
Class: Optional CSS class name(s) for widget
Name:

```
▸{ turidity: 31 }
11/3/2022, 10:04:12 PM  node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▾object
  turidity: 95
11/3/2022, 10:04:15 PM  node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▸{ turidity: 28 }
11/3/2022, 10:04:15 PM  node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▸{ turidity: 54 }
11/3/2022, 10:04:26 PM  node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▸{ turidity: 15 }
11/3/2022, 10:05:59 PM  node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
```

Fig 3

STEP 7: Simulated program to get the random values.



This table shows a summary of all devices that ha searched on using different criteria. To get started or by using API.

	Device ID	Status
□	monitorsensor1	Di

Event type name: event_1

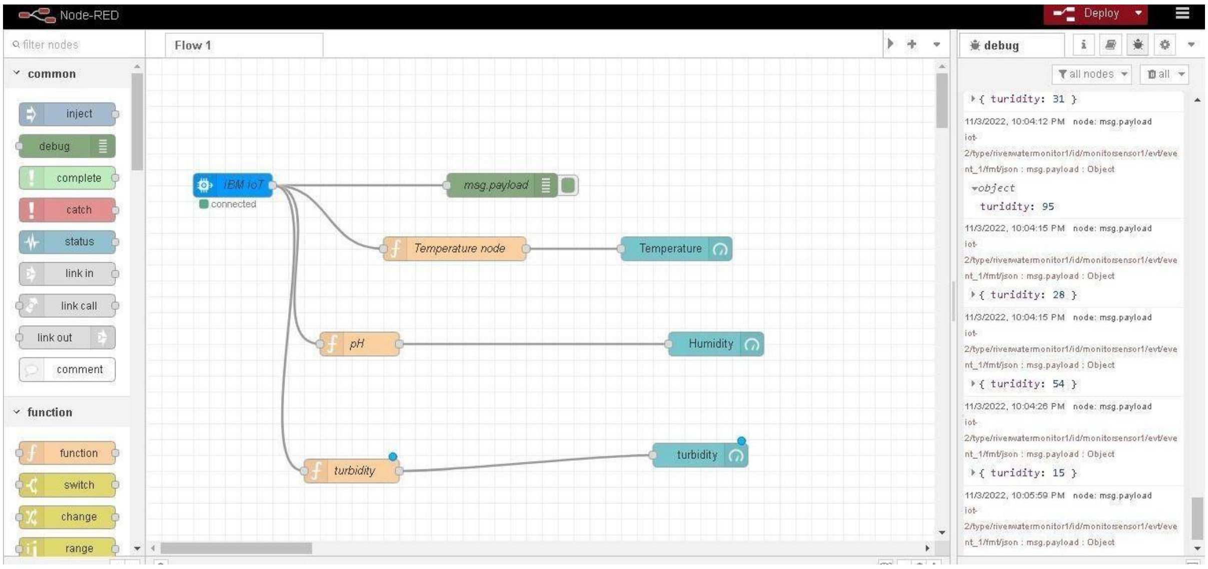
Frequency: 20 × Every Minute

Payload

You can override field values in the event payload that is sent by this device. Specify the override values in the editor window.

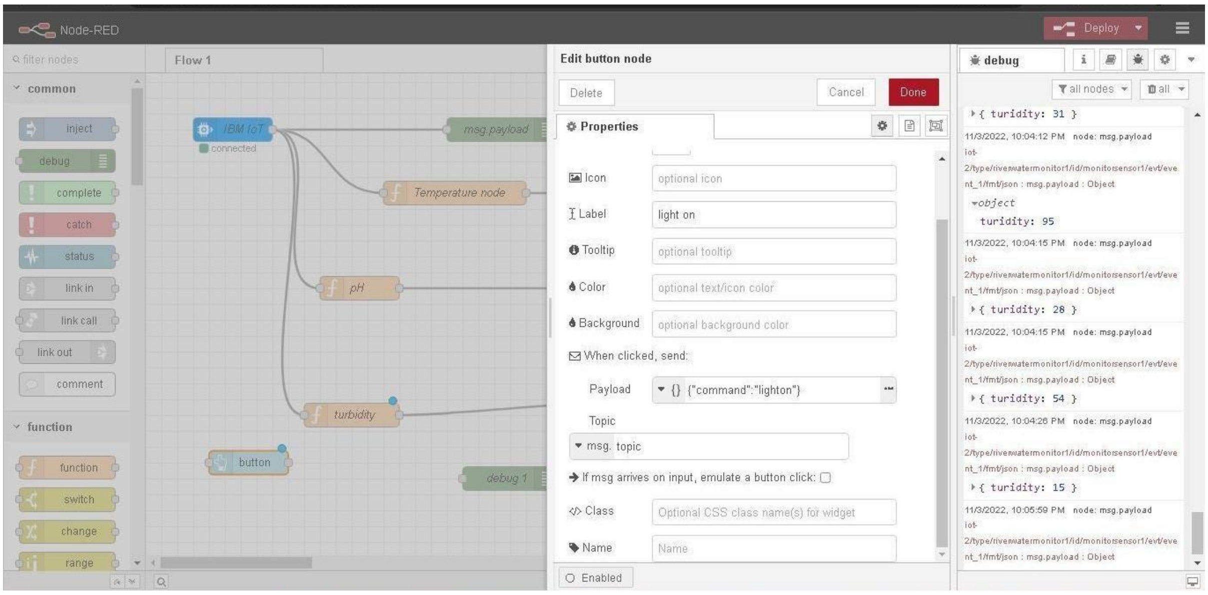
```
{
  "temperature": random(0, 100),
  "pH": random(0, 100),
  "turbidity": random(0, 100)
}

```

STEP 8: Generate debug message from IBM Watson IoT Platform and connect the nodes.



```
▸{ turidity: 31 }
11/3/2022, 10:04:12 PM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 ▾object
   turidity: 95
11/3/2022, 10:04:15 PM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 ▸{ turidity: 28 }
11/3/2022, 10:04:15 PM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 ▸{ turidity: 54 }
11/3/2022, 10:04:26 PM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 ▸{ turidity: 15 }
11/3/2022, 10:05:59 PM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
```

STEP 9: Edit button mode [light ON and light OFF].



Edit button node

Delete Cancel Done

Properties

Icon: optional icon

Label: light on

Tooltip: optional tooltip

Color: optional text/icon color

Background: optional background color

When clicked, send:

Payload: {} {"command":"lighton"}

Topic: msg. topic

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

Class: Optional CSS class name(s) for widget

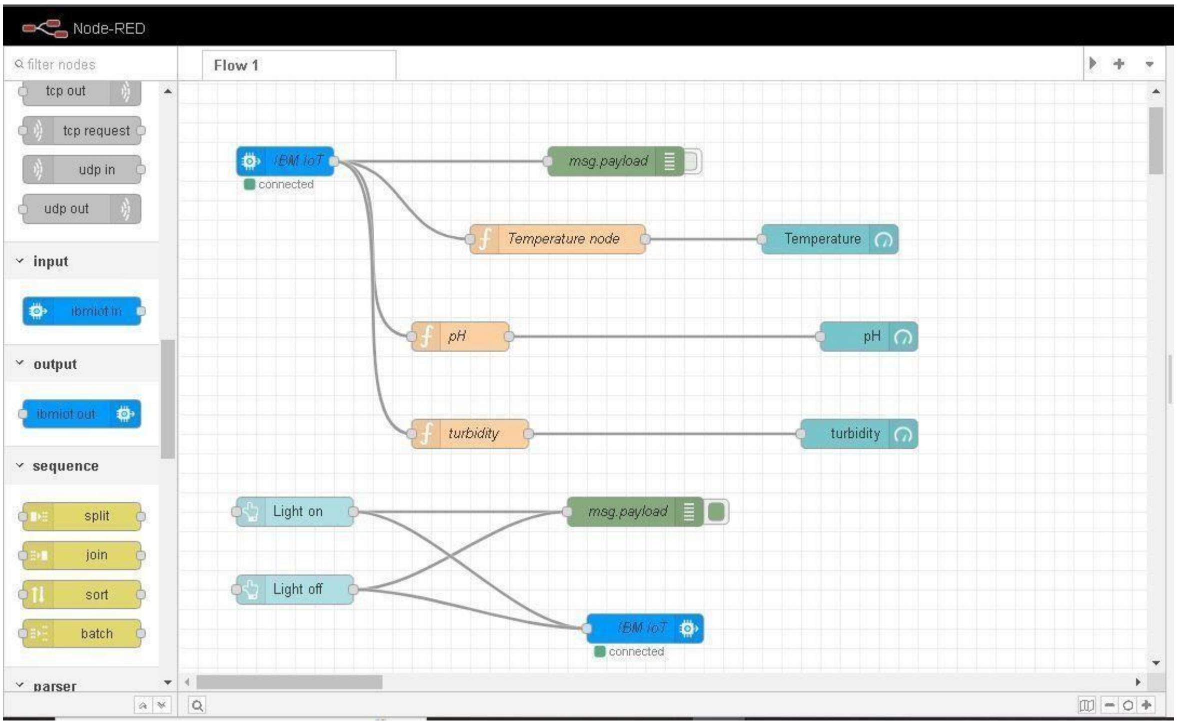
Name: Name

○ Enabled

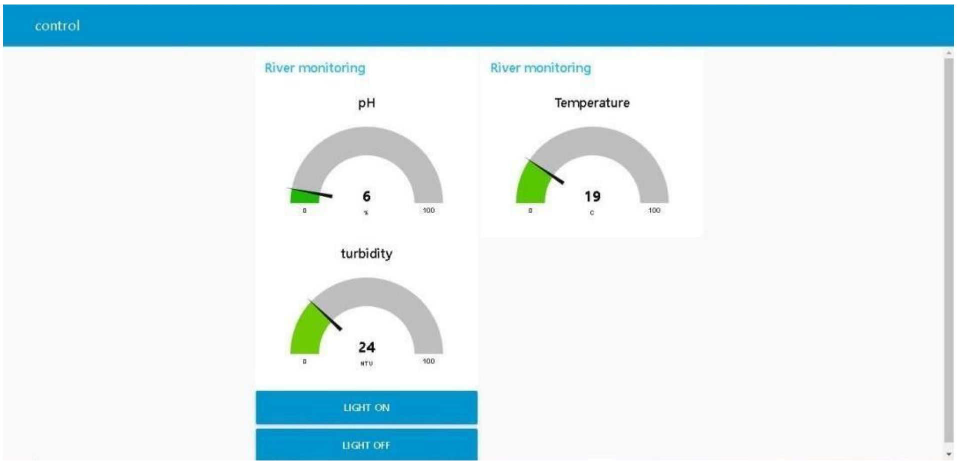
debug — all nodes — all

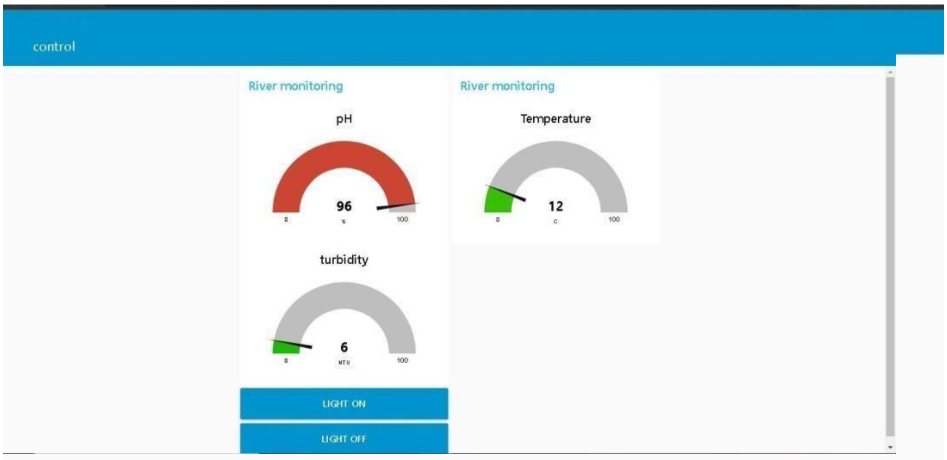
```
▸{ turidity: 31 }
11/3/2022, 10:04:12 PM   node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▾object
  turidity: 95
11/3/2022, 10:04:15 PM   node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▸{ turidity: 28 }
11/3/2022, 10:04:15 PM   node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▸{ turidity: 54 }
11/3/2022, 10:04:26 PM   node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
▸{ turidity: 15 }
11/3/2022, 10:05:59 PM   node: msg.payload
iot-2/type/riverwatermonitor1/id/monitorsensor1/evt/event_1/fmt/json : msg.payload : Object
```

STEP 10: Entire flow diagram in Node-RED.

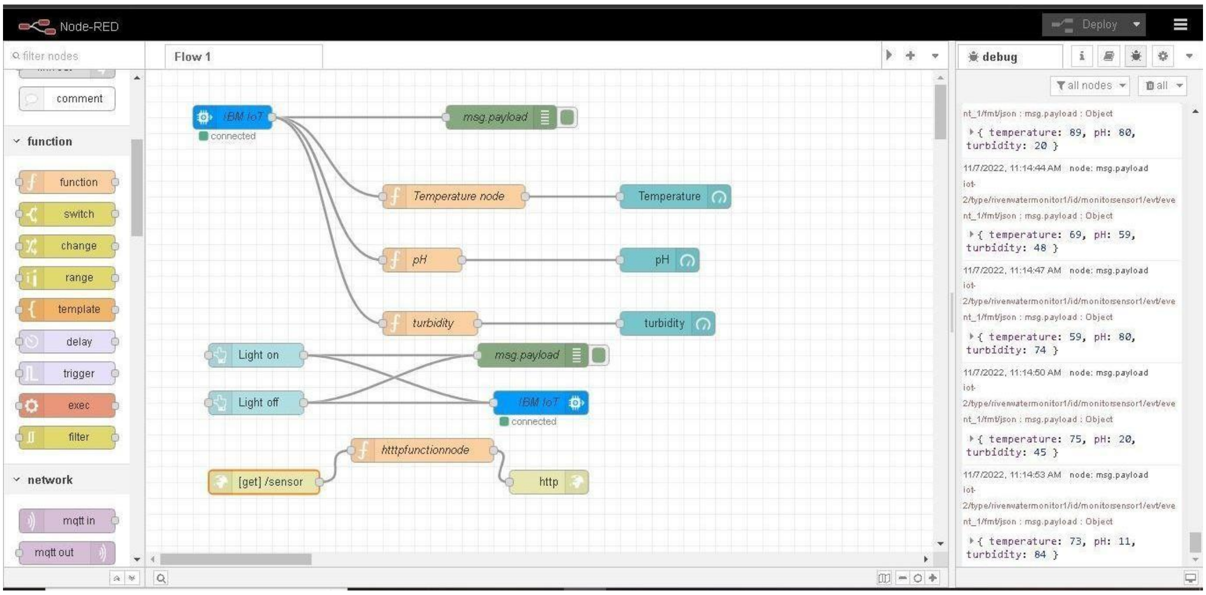


STEP 11: Generate the output from recent events.

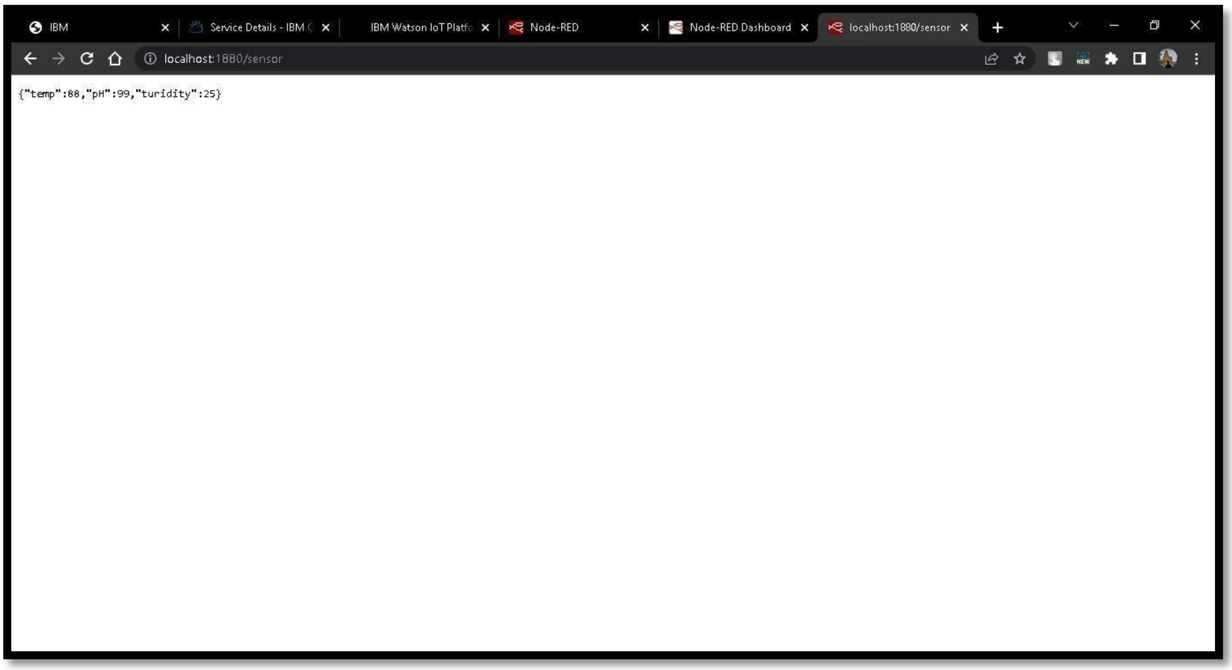




STEP 12: Implementing url in the function node to generate output.



```
nt_1/fmt/json : msg.payload : Object
 { temperature: 89, pH: 80,
turbidity: 20 }
11/7/2022, 11:14:44 AM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 { temperature: 69, pH: 59,
turbidity: 48 }
11/7/2022, 11:14:47 AM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 { temperature: 59, pH: 80,
turbidity: 74 }
11/7/2022, 11:14:50 AM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 { temperature: 75, pH: 20,
turbidity: 45 }
11/7/2022, 11:14:53 AM   node: msg.payload
iot-
2/type/riverwatermonitor1/id/monitorsensor1/evt/eve
nt_1/fmt/json : msg.payload : Object
 { temperature: 73, pH: 11,
turbidity: 84 }
```



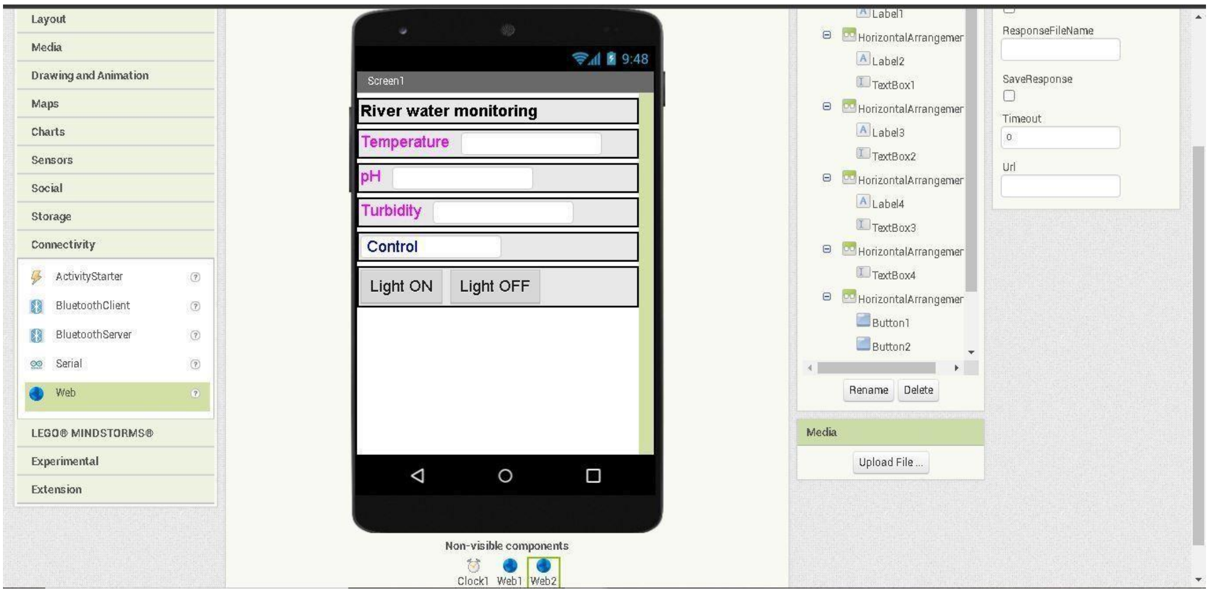
```
{"temp":88,"pH":99,"turidity":25}
```

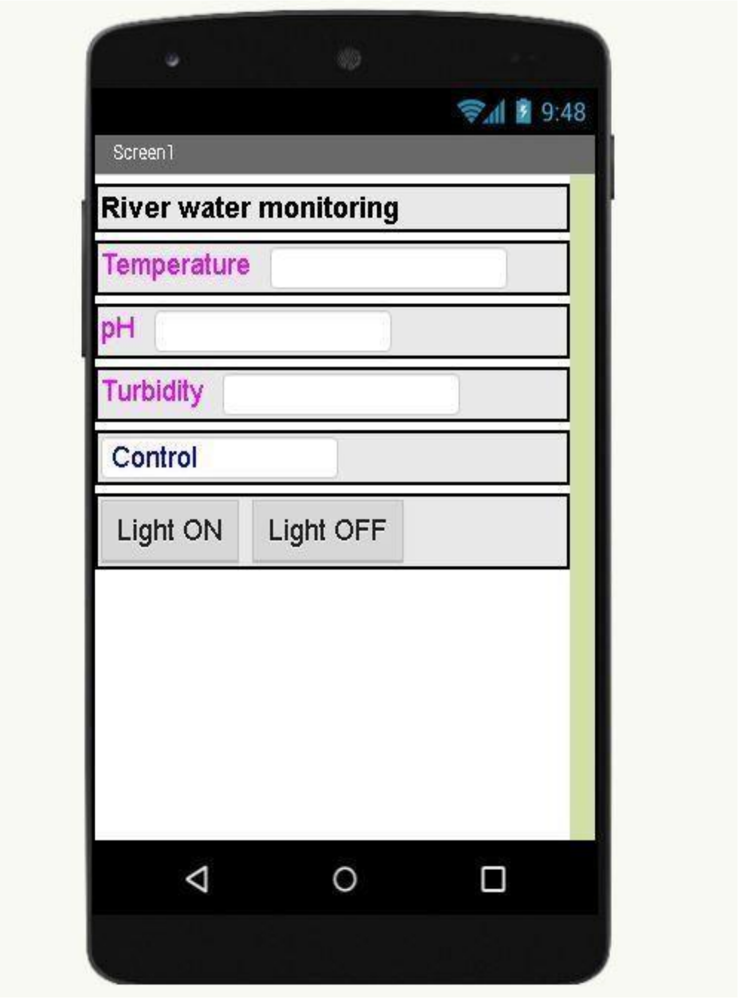
URL are:

localhost:1880/ui

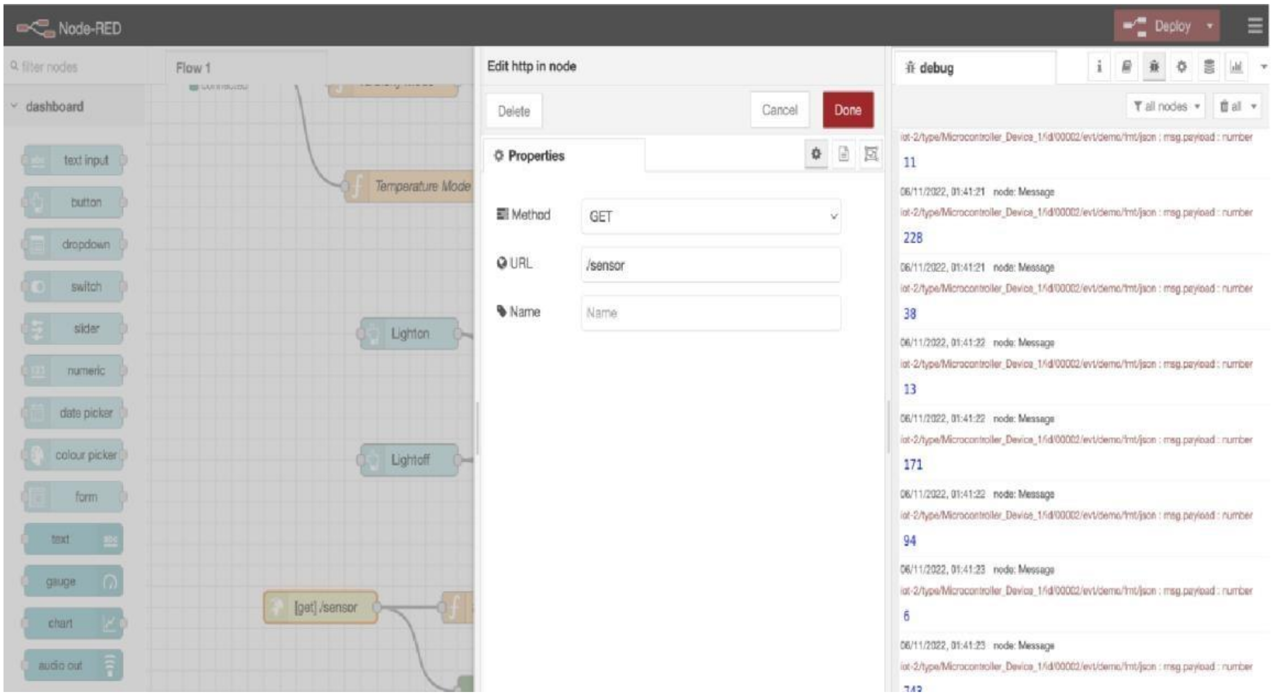
localhost:1880/sensor

STEP 13: MIT app inverter to design the app.





STEP-14: Http Request to communicate with app



Method: GET

URL: /sensor

Name: Name

```
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
11
06/11/2022, 01:41:21   node: Message
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
228
06/11/2022, 01:41:21   node: Message
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
38
06/11/2022, 01:41:22   node: Message
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
13
06/11/2022, 01:41:22   node: Message
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
171
06/11/2022, 01:41:22   node: Message
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
94
06/11/2022, 01:41:23   node: Message
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
6
06/11/2022, 01:41:23   node: Message
iot-2/type/Microcontroller_Device_1/id/00002/evt/demo/fmt/json : msg.payload : number
743
```