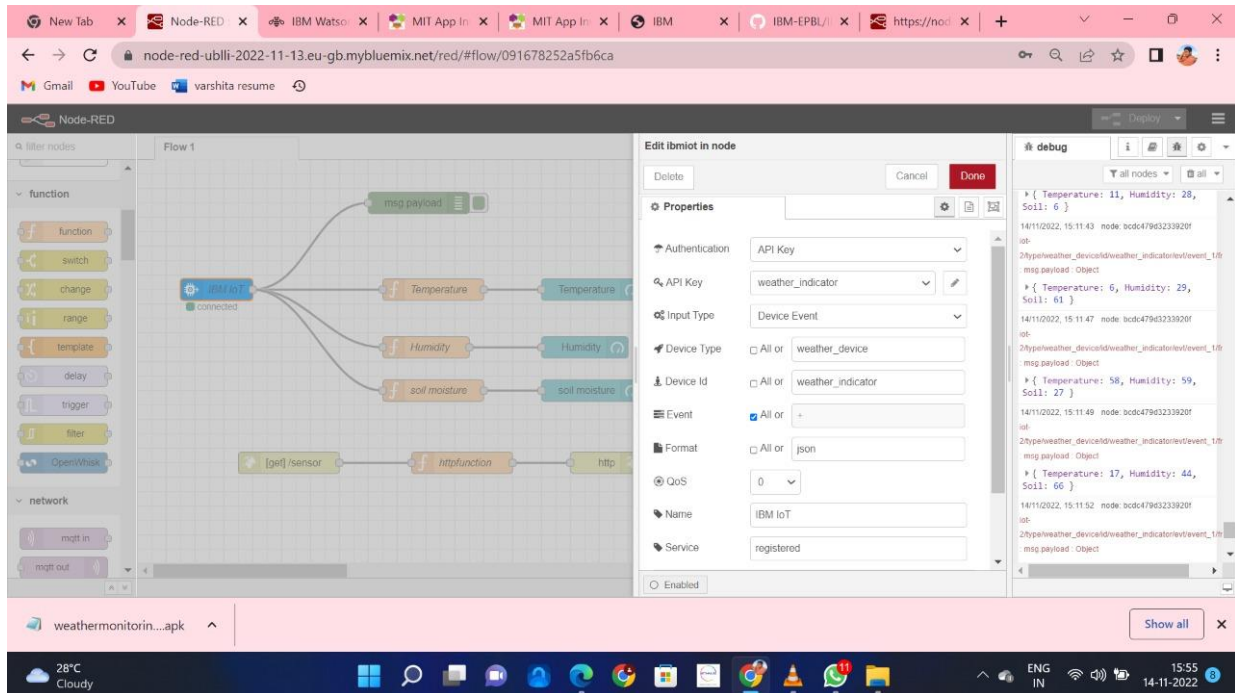


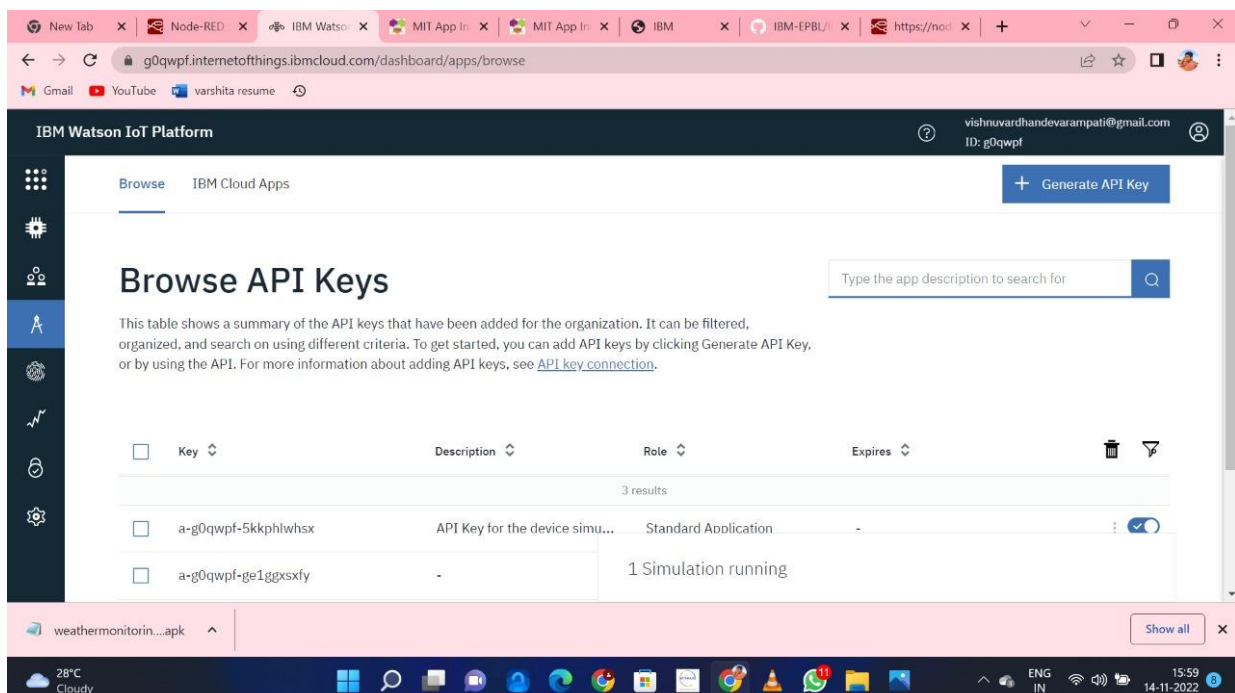
Build A Web Application Using Node-Red

Team ID	PNT2022TMID08457
Project Name	Smart Farmer-IOT Enabled Smart Farming Application

First open Node RED workspace and drag IBM iot input into the workspace. It will ask API key, device id, device type etc.



For Generating API keys open IBM Watson and click Apps and Generate API key it shown below like these:



And take a function node and rename it has a temperature and message in the editor.

The screenshot shows the Node-RED web interface in a browser. The main workspace displays a flow named 'Flow 1' with a 'msg.payload' node connected to three function nodes: 'Temperature', 'Humidity', and 'soil moisture'. A 'get/sensor' node is connected to an 'httpfunction' node. The 'Edit function node' panel for the 'Temperature' node is open, showing the following code:

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

The 'debug' console on the right shows a series of log messages, including:

```
{ Temperature: 11, Humidity: 28, Soil: 6 }
{ Temperature: 6, Humidity: 29, Soil: 61 }
{ Temperature: 58, Humidity: 59, Soil: 27 }
{ Temperature: 17, Humidity: 44, Soil: 66 }
```

Now take a function node for humidity and type message in the editor.

The screenshot shows the Node-RED web interface with the 'Edit function node' panel for the 'Humidity' node open. The code in the editor is:

```
1 msg.payload=msg.payload.Humidity
2 global.set('h',msg.payload)
3 return msg;
```

The 'debug' console on the right shows the same log messages as the previous screenshot, including:

```
{ Temperature: 11, Humidity: 28, Soil: 6 }
{ Temperature: 6, Humidity: 29, Soil: 61 }
{ Temperature: 58, Humidity: 59, Soil: 27 }
{ Temperature: 17, Humidity: 44, Soil: 66 }
```

And take a function node and rename it has a Soil moisture and message in the editor

The screenshot shows the Node-RED web interface in a browser. The main workspace displays a flow with an 'IBM IoT' node connected to a 'msg.payload' node, which then branches into three nodes: 'Temperature', 'Humidity', and 'soil moisture'. A 'debug' node is also present. The 'Edit function node' panel is open for the 'soil moisture' node, showing the following code:

```
1 msg.payload=msg.payload,Soil
2 global.set('s',msg.payload)
3 return msg
```

The 'Properties' panel shows the node name as 'soil moisture'. The 'debug' console on the right displays a log of messages, including:

```
{ Temperature: 11, Humidity: 28, Soil: 6 }
```

The bottom of the screen shows a taskbar with various application icons and a system tray displaying '28°C Cloudy' and the time '15:47 14-11-2022'.

Now take temperature gauge meter in the dashboard and give name as temperature and range 0 to 100.

The screenshot shows the Node-RED web interface with the same flow as the previous image. The 'Edit gauge node' panel is open for the 'Temperature' node. The configuration is as follows:

- Group: [Home] Default
- Size: auto
- Type: Gauge
- Label: Temperature
- Value format: {value}
- Units: C
- Range: min 0, max 100
- Colour gradient: A gradient from green to red.
- Sectors: 0, optional, optional, 100
- Class: Optional CSS class name(s) for widget

The 'debug' console on the right shows the same log of messages as the previous image, including:

```
{ Temperature: 11, Humidity: 28, Soil: 6 }
```

The bottom of the screen shows the same taskbar and system tray information as the previous image.

Similarly for humidity u take another gauge meter and range 0 to 100

The screenshot shows the Node-RED web interface in a browser. The main workspace displays a flow named 'Flow 1'. An 'IBM IoT Gateway' node is connected to three function nodes: 'Temperature', 'Humidity', and 'Soil Moisture'. Each function node is connected to a corresponding gauge widget. The 'Humidity' gauge is currently selected, and its configuration is visible in the 'Edit gauge node' panel. The configuration includes: Group '[Home] Default', Size 'auto', Type 'Gauge', Label 'Humidity', Value format '({value})', Units 'C', Range 'min 0 max 80', and a color gradient. The debug console on the right shows a series of messages from the IoT Gateway node, including temperature, humidity, and soil moisture data.

For soil moisture u take another gauge meter and range 0 to 100

The screenshot shows the Node-RED web interface in a browser. The main workspace displays a flow named 'Flow 1'. An 'IBM IoT Gateway' node is connected to three function nodes: 'Temperature', 'Humidity', and 'Soil Moisture'. Each function node is connected to a corresponding gauge widget. The 'Soil Moisture' gauge is currently selected, and its configuration is visible in the 'Edit gauge node' panel. The configuration includes: Group '[Home] Default', Size 'auto', Type 'Gauge', Label 'soil moisture', Value format '({value})', Units 'C', Range 'min 0 max 90', and a color gradient. The debug console on the right shows a series of messages from the IoT Gateway node, including temperature, humidity, and soil moisture data.

Now change the http:// in into the get /sensor

The screenshot shows the Node-RED web interface in a browser. The 'Edit http in node' dialog is open, showing the following properties:

- Method: GET
- URL: /sensor
- Name: Name

The main workspace displays a flow named 'Flow 1'. It starts with a 'msg payload' node connected to three function nodes: 'Temperature', 'Humidity', and 'soil moisture'. These function nodes are then connected to an 'httpfunction' node, which is finally connected to an 'http' node. The debug console on the right shows a series of messages with the following structure:

```
{ Temperature: 11, Humidity: 28, Soil: 6 }
14/11/2022, 15:11:43 node:bc0c47963233920f
iot:
2/hyperweather_device/dweather_indicator/event_1/0
:msg payload: Object
{ Temperature: 6, Humidity: 29, Soil: 61 }
14/11/2022, 15:11:47 node:bc0c47963233920f
iot:
2/hyperweather_device/dweather_indicator/event_1/0
:msg payload: Object
{ Temperature: 58, Humidity: 59, Soil: 27 }
14/11/2022, 15:11:49 node:bc0c47963233920f
iot:
2/hyperweather_device/dweather_indicator/event_1/0
:msg payload: Object
{ Temperature: 17, Humidity: 44, Soil: 66 }
14/11/2022, 15:11:52 node:bc0c47963233920f
iot:
2/hyperweather_device/dweather_indicator/event_1/0
:msg payload: Object
```

Now take http function and type temperature , humidity ,and soil etc.

The screenshot shows the Node-RED web interface. The 'Edit function node' dialog is open, showing the following properties:

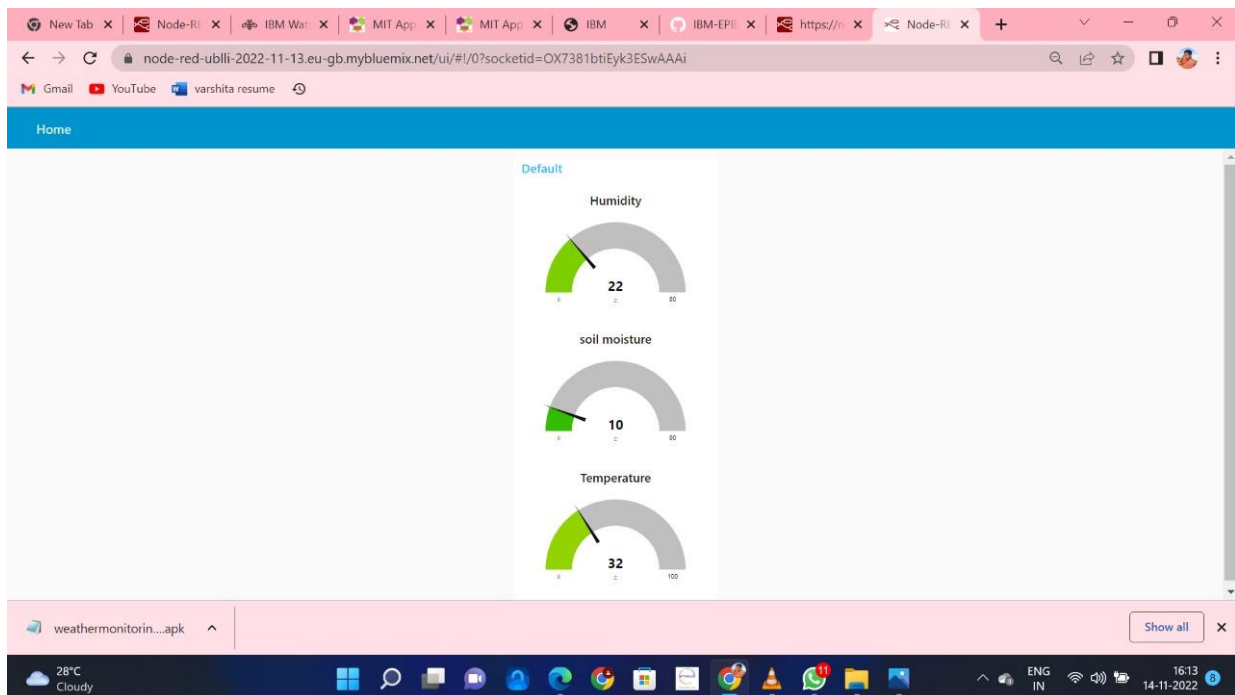
- Name: httpfunction

The 'Setup' tab is selected, and the code editor contains the following code:

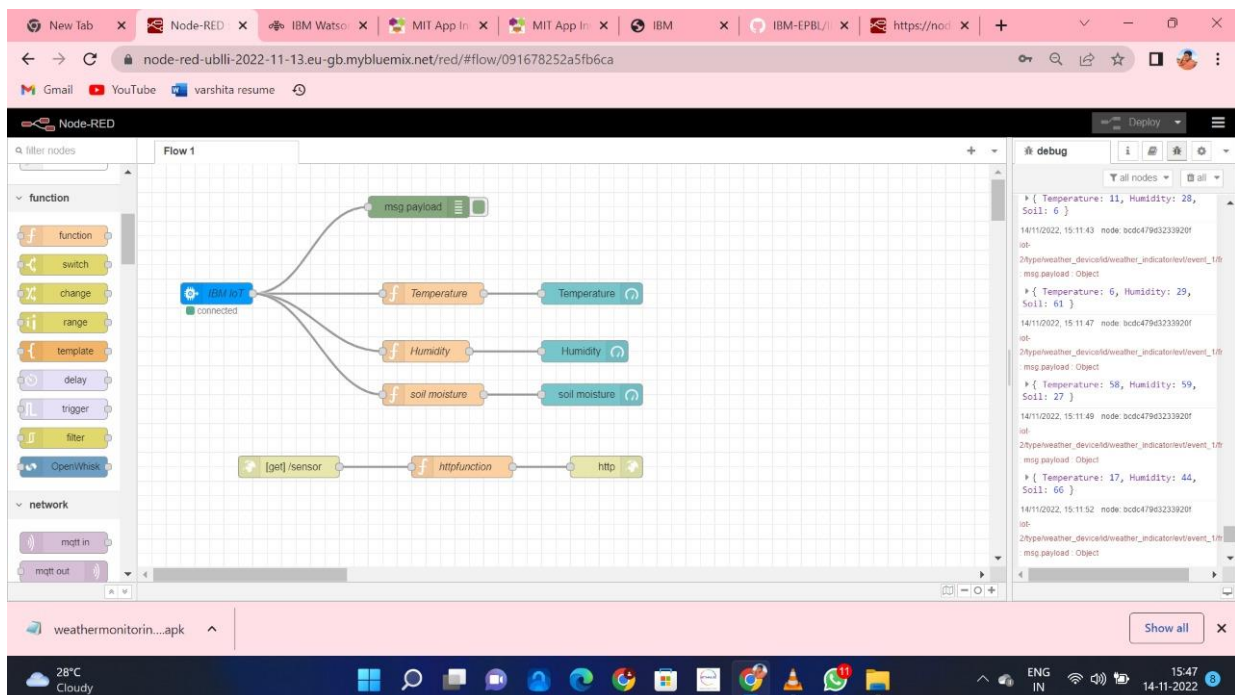
```
1 msg.payload = { "temperature":global.get("t"), "humidity":global.get("h") };
2 return msg;
```

The main workspace and debug console are the same as in the previous screenshot.

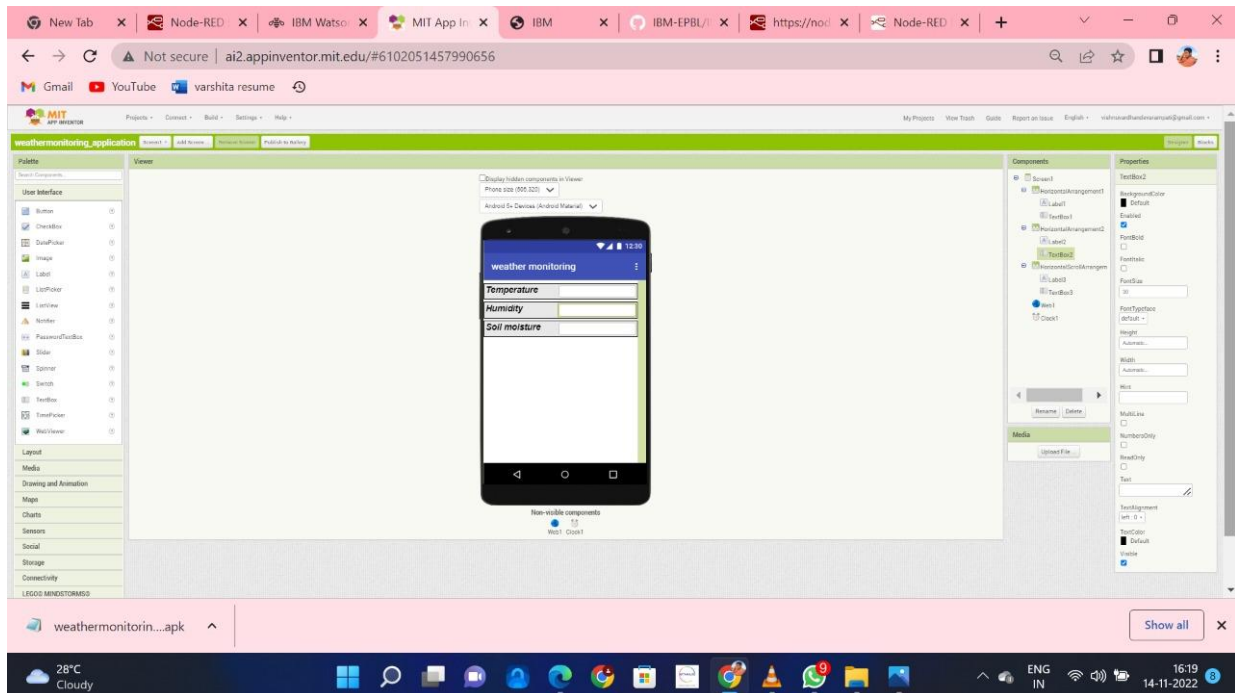
There is an Node-Red dashboard to see the readings of temperature, humidity and soil moisture.



Finally, we can connect as shown in figure below

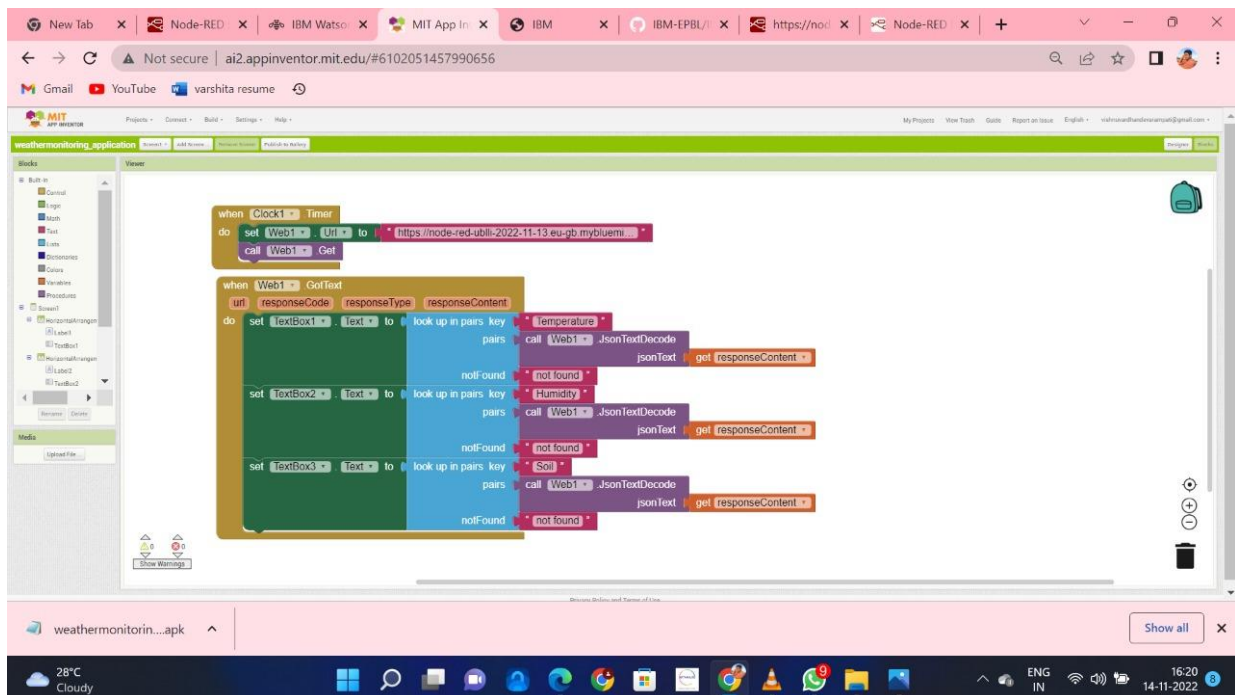


Now developing a mobile application using MIT app inventor can be shown below



As we can observe, we have given three slots namely, temperature, humidity and soil moisture and user can get the values of them in respective allocated spaces .

The blocks of the MIT app inventor is shown below



Now we can observe that Node-Red is connected successfully to the mobile application

