

## Monitoring and Visualization of Sensor Data

**Difficulty Level:** Medium

### Objective

Implement a visual dashboard to monitor the sensor data in a smart factory scenario; produce warning messages when the temperature or humidity reading raises above a certain threshold.

### Achievements

The skills to be acquired at the end of this module:


- Reading sensor data by subscribing to MQTT topics
- Plotting sensor data by using “node-red-dashboard” nodes
- Understanding and adapting the UI layout in Node-RED

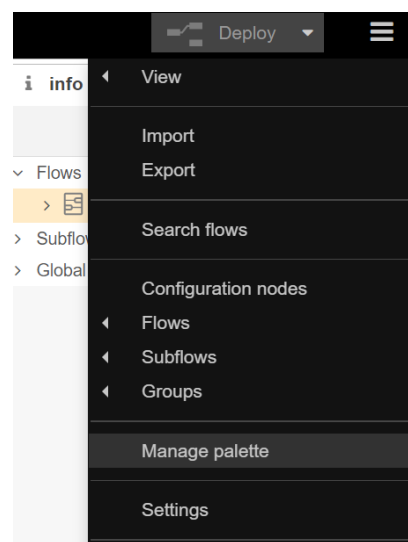
### 1. NodeRED Palette Manager and Adding New Node Types

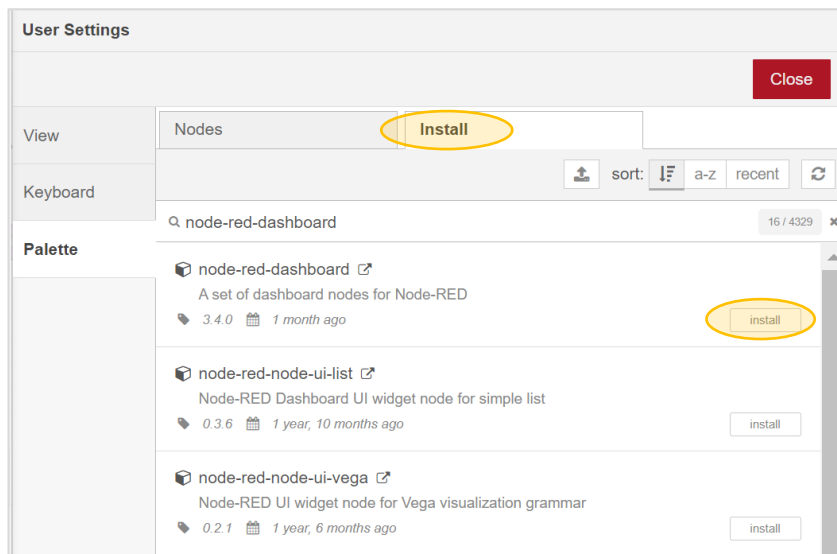
In this module, we will retrieve the temperature and humidity sensor data from the warehouse and product stock. For this, we need to connect to the MQTT broker and subscribe to the corresponding MQTT topics.

Every time a new sensor value is received, we will dynamically visualize it in a graphic plot. For this purpose, we will use the **node-red-dashboard** nodes to easily create user interface (UI) with dynamic plots. Before continuing, we have to install it in the *palette manager*.

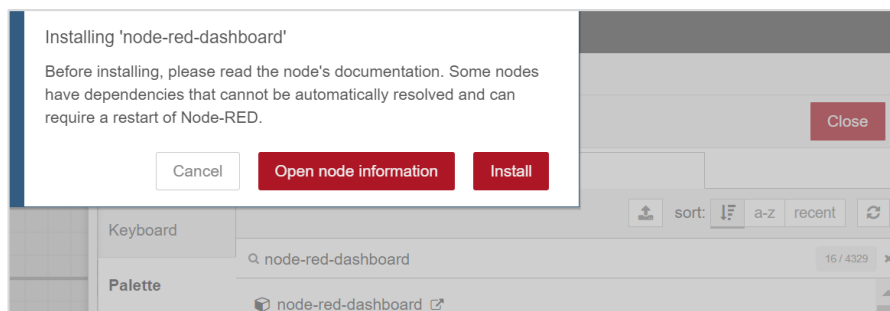
The **palette manager** is a great feature that allows us to extend the functionality of NodeRED. With this, we can add completely new types of nodes to NodeRED, which helps us implement certain functionality with much less complexity and less coding.

- To open the palette manager, click on the icon  at the top right of NodeRED window, and then select the “Manage palette” option from the menu.
- A window will pop up with the User Settings / Palette.
- Go to the second tab called “install”, as depicted below.
- In the search text bow, type “node-red-dashboard”.
- A list of results will be loaded as you type. We will use the first one that should also be named “node-red-dashboard”.
- Click install for this first result, as shown below.





- A warning message will pop up, as depicted below. You can simply click on install, but it is always a good practice first to check its documentation by clicking on “Open node information”. This will open up the page <https://flows.nodered.org/node/node-red-dashboard> in a new browser window.



- When ready, you can go back to the NodeRED window and continue installing *node-red-dashboard*.

When the installation is complete, a message will pop up with a list of all nodes that are added to the palette:

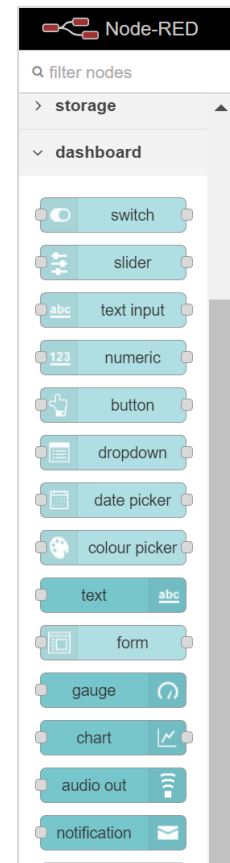


You should now be able to see those new nodes on the left side panel of NodeRED editor. You may need to scroll all the way down or “collapse” some of the existing node groups to find the new node group “dashboard”.

You can use any of those new nodes just like other built-in nodes in Node-RED, simply by dragging and dropping a node to the flow editing area.

You can go ahead and play around to make yourself more familiar with the dashboard nodes.

However, we will actually come back to using those later in this module, in order to create the dynamic plots, after we complete the setup for retrieving the sensor data.



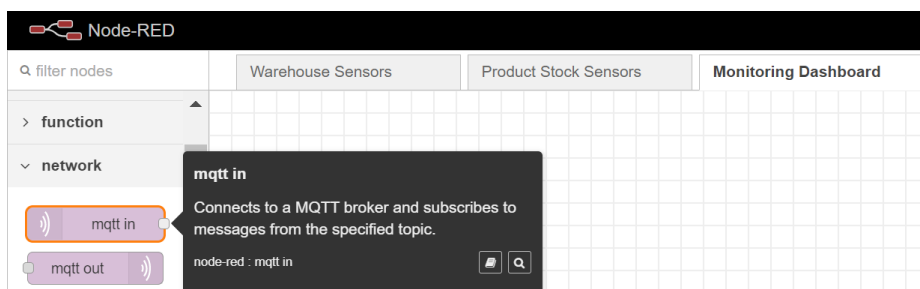
## 2. Retrieving sensor data via MQTT

We start by creating/renaming an empty flow in NodeRED and call it “Monitoring Dashboard”.

Note that we need to retrieve four different sources of sensor data, listed below with the corresponding MQTT topic names:

- Temperature sensor in the parts warehouse (*mqtt topic: “warehouse/temp”*)
- Humidity sensor in the parts warehouse (*mqtt topic: “warehouse/humidity”*)
- Temperature sensor in the product stock (*mqtt topic: “stock/temp”*)
- Humidity sensor in the product stock (*mqtt topic: “stock/humidity”*)

We will use an “**mqtt in**” node in NodeRED to retrieve the data for each of these sensors/topics. You can find this under the “network” node group in the palette (left panel):

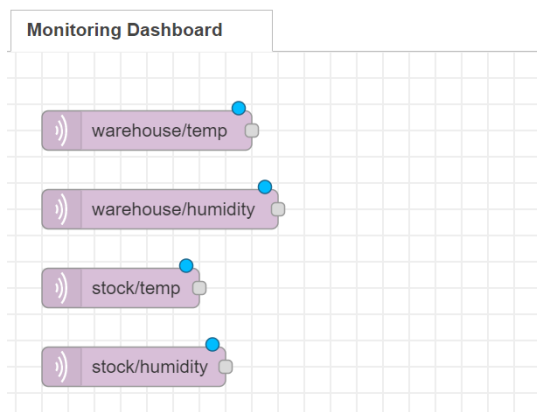


Drag and drop four of those `mqtt in` nodes into your flow.

Double click on each of them one by one to edit the “topic” field, so that each node **subscribes** to one of the four **mqtt-topics** given above.

The “server” setting for each node should be configured to the same MQTT Broker we used in the previous two modules, i.e., *test.mosquitto.org* server with port 1883. See module 3 (Implementing Warehouse Sensors) for details, if needed.

At the end, your flow should look like the following. (Note that omitting the node name for an `mqtt` node results in the `mqtt` topic being displayed as the node name in the flow editor.)



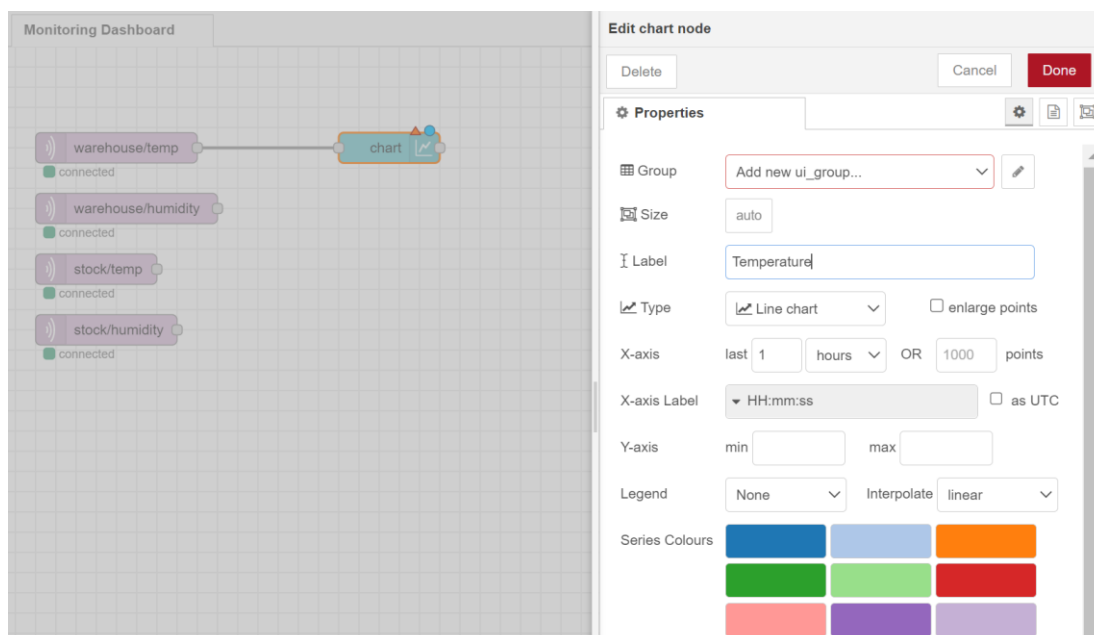
You can click on “Deploy” in NodeRED to test the connection with the MQTT broker. You may also connect a *debug node* to an *mqtt in* node’s output to check the incoming sensor data, as explained in Module 2 (Node-RED as an IoT Application Development Tool).

### 3. Visualizing Sensor Data in Dynamic Charts

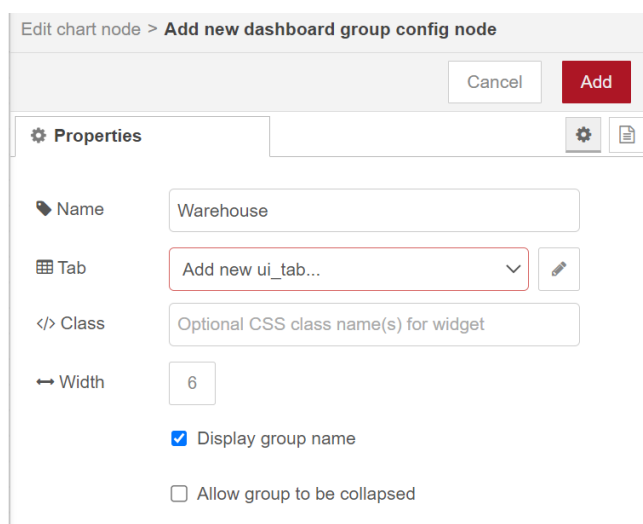
Now we are ready to make use of the incoming sensor data and display them in dynamic charts.

Here is where the dashboard nodes, which we installed in the first step of this module, come in very handy.

- Find the **Chart** node under the dashboard group in the palette (left side panel) in NodeRED.
- Drag and drop a *chart* node next to the first *mqtt in* node and connect the two.
- Double click on the chart node to bring up its settings:

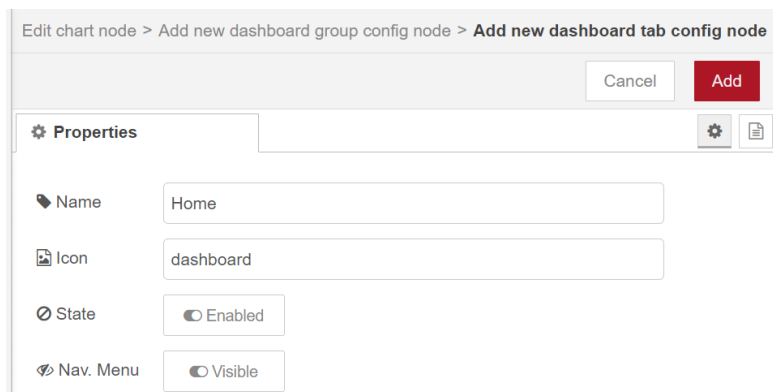


- Change the **Label** from the default value of “Chart” to “Temperature”, as shown in the figure above.
- Notice the first setting named “Group”. Here we first need to create a new *user interface (UI) group* by clicking on the pencil icon next to it.
- This brings up the following window:



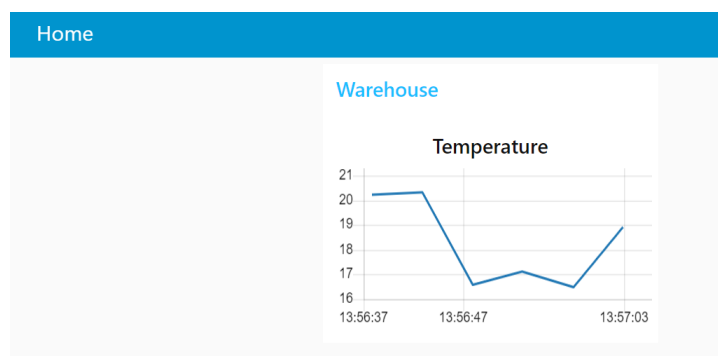
- Change the name to “Warehouse”.
- Here, we have again a setting called “Tab” with a pencil icon next to it. Tabs are highest level UI group elements, which allows creating separate “views” or “pages” on the web interface.
- If we create multiple “*ui tab*” elements, our web dashboard will have a navigation menu to switch between those tabs/pages, and we can place our UI elements (e.g. charts) in any tab.

- For this module, we will place all charts on a single page / single tab. Therefore, we simply click on the pencil icon to create a new tab with the default settings:

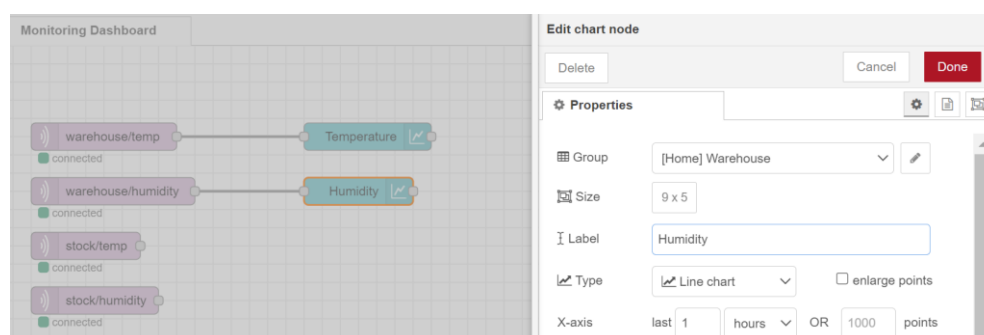


- Click on “Add” to close the tab config window and then once again click on “Add” to close the ui group window.
- This brings us back to the “Edit chart node” window. Click on “Done”.

Now we can already see our chart in action for the warehouse temperature. For that, we need to open the web interface for our dashboard. Assuming that your NodeRED is running at the address <http://localhost:1880>, simply add a “/ui” at the end of that to reach the NodeRED Dashboard web UI, i.e. <http://localhost:1880/ui/>. It should look like the following:



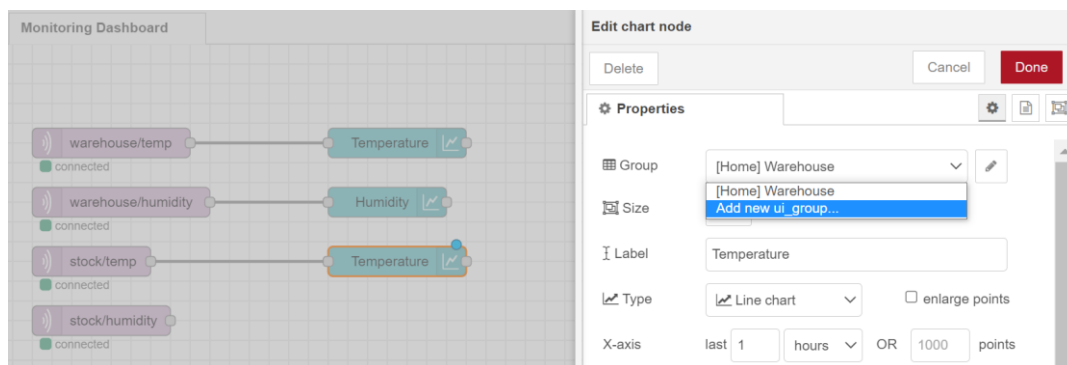
We will now add **three other chart nodes** for the remaining sensor data. Adding the **humidity chart** for the warehouse is quite straightforward. Copy the temperature chart and edit it as follows.



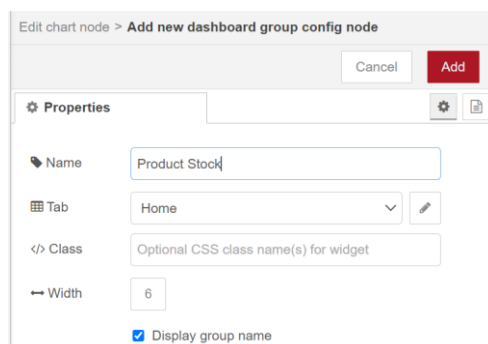
Note that we used the same **Warehouse** group (within the [Home] tab) that we had set up for the first chart. This will result in the two charts being grouped under a Warehouse heading.

Adding the remaining two charts is similar to the above, but this time we will create a new **Product Stock** group (again within the [Home] tab), so that they are separated from the Warehouse charts:

- Create a copy of the first Temperature chart and connect it to the “stock/temp” *mqtt in* node.
- Open the settings of the new chart node and select the option of “Add new ui\_group”, and then click on the pencil icon.

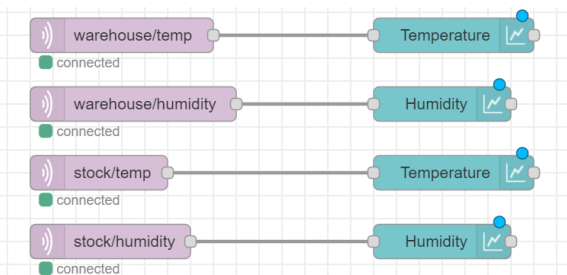


- In the opening window, set the name of the group to “Product Stock”, as shown below.

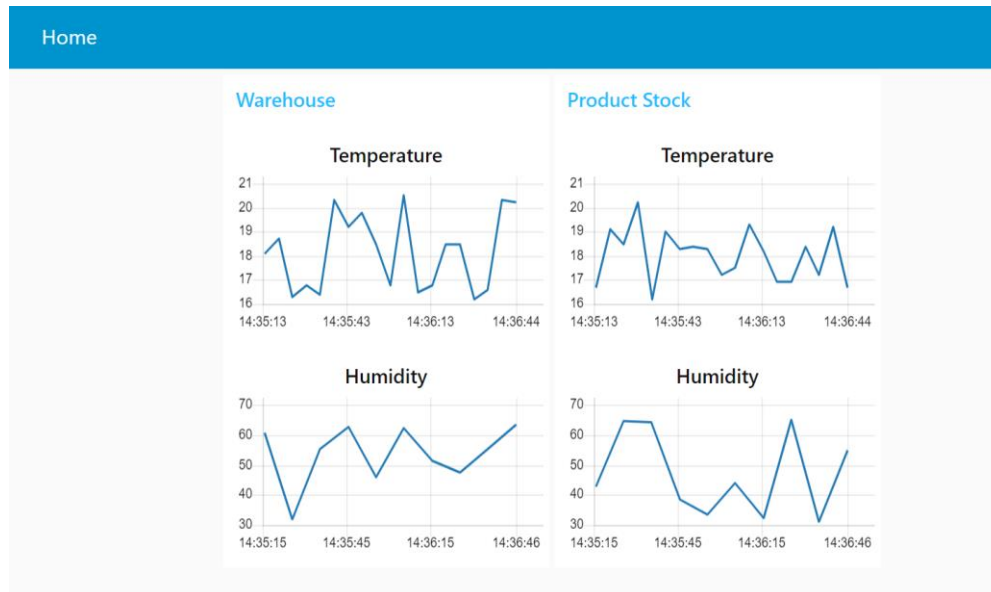


- Note above that we keep the ‘Tab’ name as “Home”, since we are placing all UI elements on a single tab / single page.
- Click on “Add” and then “Done” to complete the configuration of the chart.
- Apply the same above to create the last “Humidity” chart under the “Product Stock” group.

The resulting flow should look like the following:



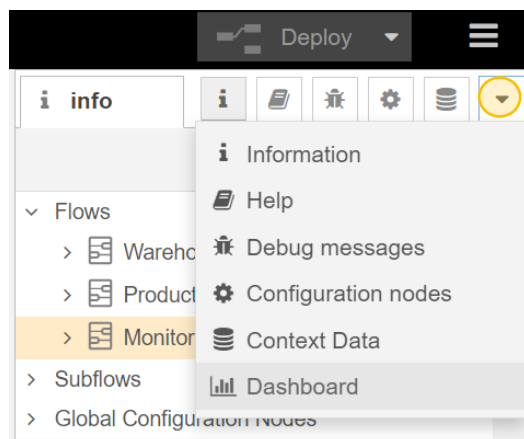
Finally, when we deploy our flow in NodeRED and then visit the web UI of the dashboard (<http://localhost:1880/ui/>), we should see the four charts dynamically showing the sensor data:



#### 4. Making Changes in the UI Layout

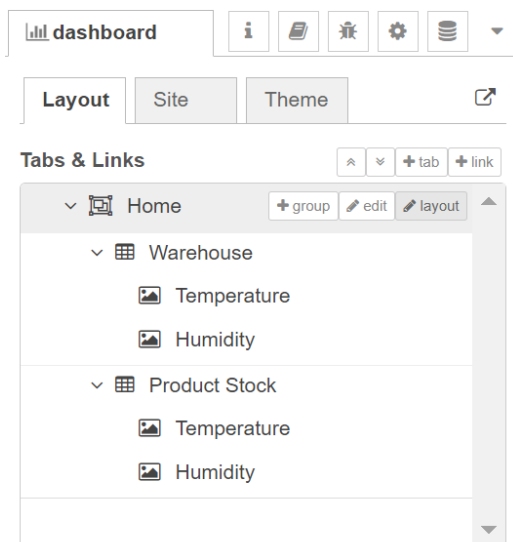
In the previous sections, we didn't make any changes to the default configuration of the UI elements, but NodeRED dashboard provides a lot of flexibility in setting the layout of the web UI and arranging the UI elements on a page.

For this, click on the little down arrow on the right side panel of NodeRED, as shown below, as then select "Dashboard".



This brings up the following with an overview of the UI elements:



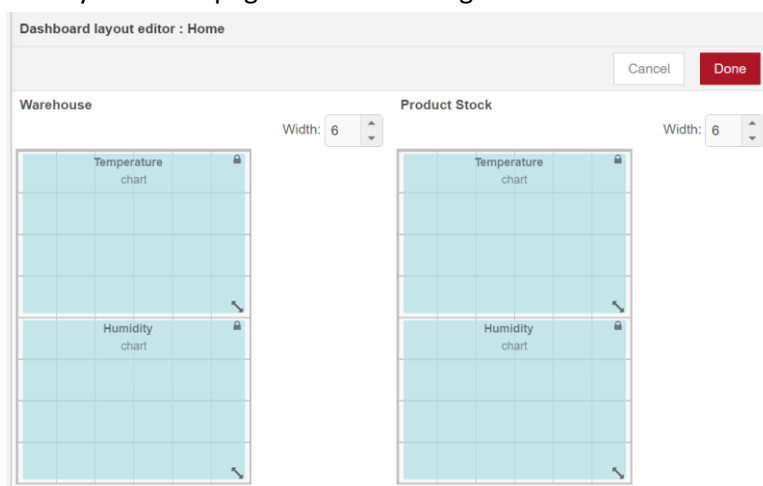


Here we see the list of all UI elements:

- At the top level, we have a single *Tab* (Page) called “Home”
- Under the Home tab, we have two UI *Groups*: “Warehouse” and “Product Stock”
- Finally, under each group, we have the Temperature and Humidity charts.

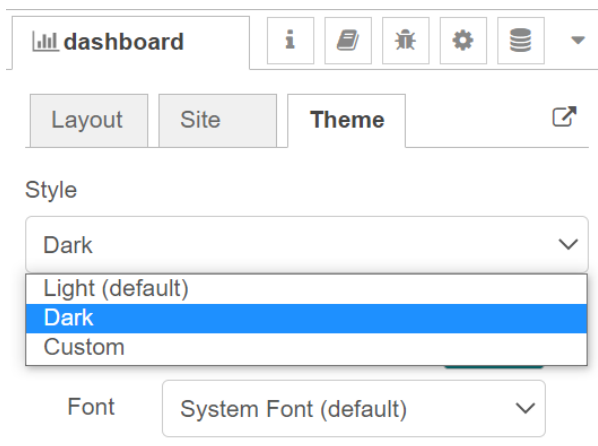
When we hover over each line in this view, some options emerge, as shown in the figure above for the “Home” tab:

- Clicking on “+group”, we can add a new UI group and then add/move certain UI elements, e.g. charts into the new group.
- Clicking on “edit” brings up the node configuration window that we have seen earlier when editing the chart nodes.
- Clicking on “layout” brings up a more interesting and useful window where we can visually see the layout of the page and make changes to it:

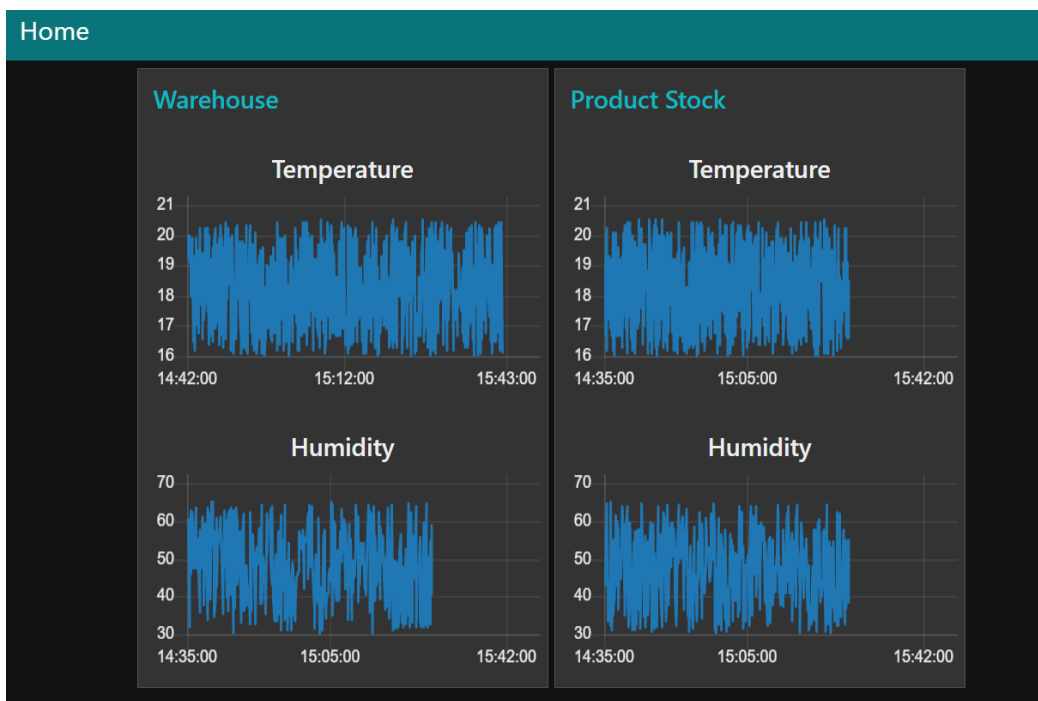


- We see that the default width of each group and chart is 6 “units”, which we can change. After increasing the group width, we can also pick and drag from the corner of a chart area and extend it both horizontally and vertically to increase its size.

Going back to the Dashboard view and switching to the “Theme” page, we can change the style of the page from the default Light to the Dark mode, as depicted below.



Then, the look of the web UI simply changes to the following view. We can also select the Custom option and configure the colors used in the theme.



Congratulations! You have accomplished a web-based application for monitoring IoT sensor data via MQTT and dynamic charts.