Lab 3: End to end project with Node-Red

Objective: The objective of this laboratory exercise is to develop practical skills in implementing end-to-end IoT solutions using Node-Red, Arduino, and cloud integration. Through hands-on tasks, you will gain experience in utilizing Node-Red to establish serial communication with an Arduino device to control and synchronize physical and virtual components such as LEDs and push-buttons.

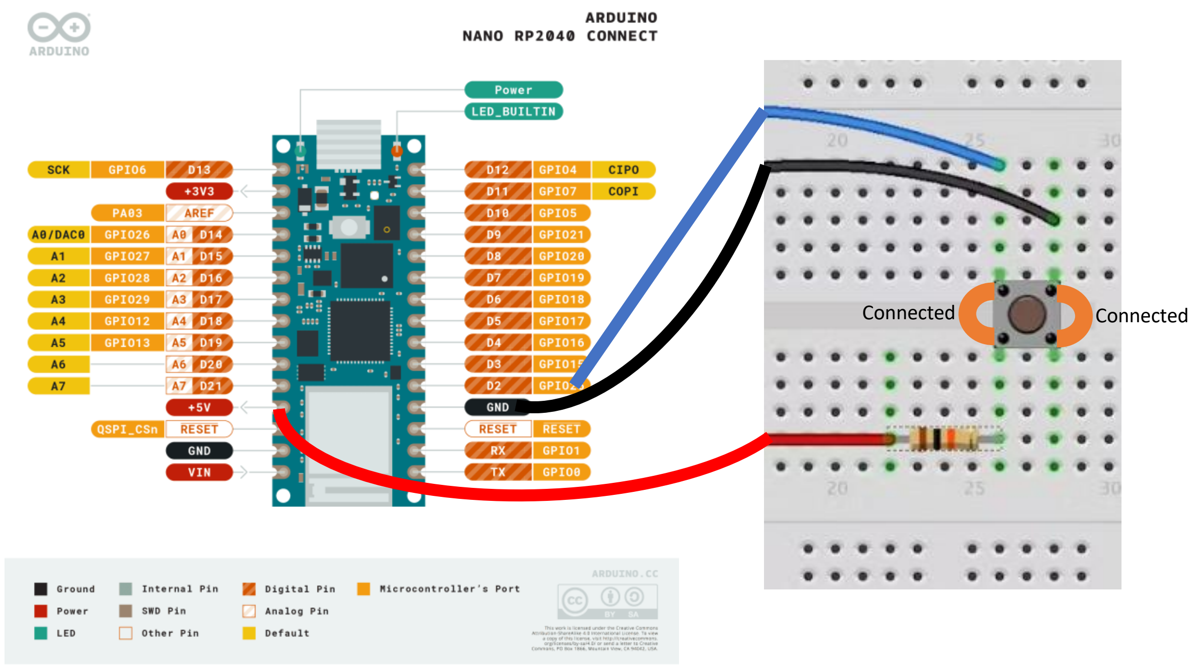
You will design dashboards for real-time monitoring and control of devices, enabling data visualization and interaction between physical and virtual components. Additionally, you will integrate a temperature sensor with Arduino to collect and display real-time temperature data on Node-Red dashboards, with dynamic color-coding based on predefined ranges.

The exercise also involves working with MQTT protocols by configuring HiveMQ and integrating with an IoT-cloud platform (DATACAKE) to visualize, monitor, and analyze data trends over time. This laboratory session focuses on practical workflows, including hardware-software integration, data transmission via MQTT, and cloud-based visualization, helping you build a comprehensive understanding of IoT systems.

Everybody prepares an individual report!

Exercise 1

Build an Arduino circuit like the one in the visualization below. For this, keep in mind that the LED is powered directly by the Arduino and that both the LED and the button are independent. It is not a connected circuit.



Please be aware that there are different kinds of buttons existing!

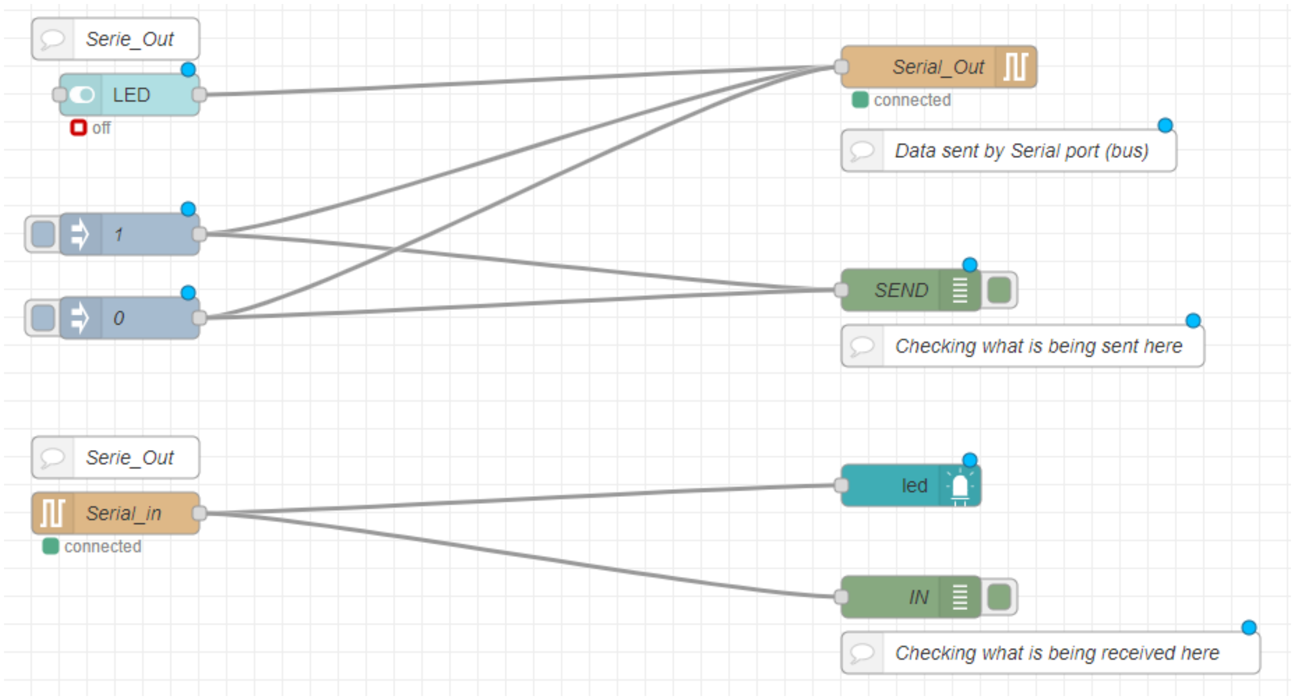
On the visualization above, the button has 4 contacts, while the button you will use has only two contacts. However, in the button from the visualization, both left contacts are connected together, the same is with two contacts on the right side of the button. Which means that from the schematic point of view, button with 4 contacts (as on the visualization) and with two contacts (as you will use) are similar.

In the Arduino environment, implement a program that reads data through the serial port and turns on the LED if it receives a '1' and turns it off if the value is '0'. You must also work with the value of the push-button by sending it through the serial port if the value is '1' (button pressed) or '0' (button not pressed) – this will be used by the Node-Red implementation to turn on/off the virtual LED (read further description below).

How do you use an external button? There are several different options, please check the provided visualization and also other options and information provided at:

https://docs.arduino.cc/built-in-examples/digital/Button

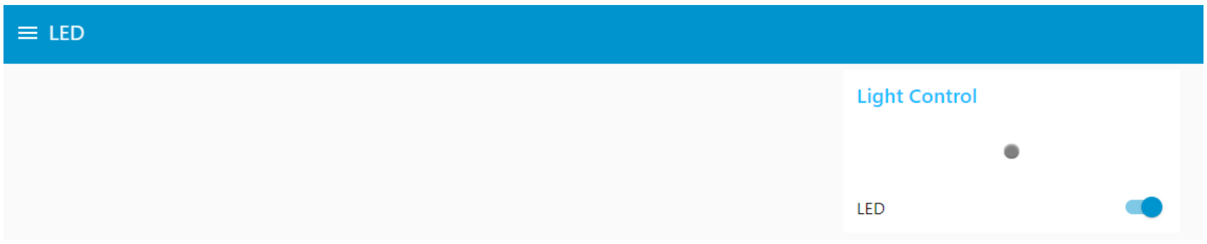
For the implementation of the exercise on Node-Red, you should build a flow like the following.


Serie_Out
LED
off
1
0
Serial_Out
connected
Data sent by Serial port (bus)
SEND
Checking what is being sent here
Serie_Out
Serial_in
connected
led
IN
Checking what is being received here

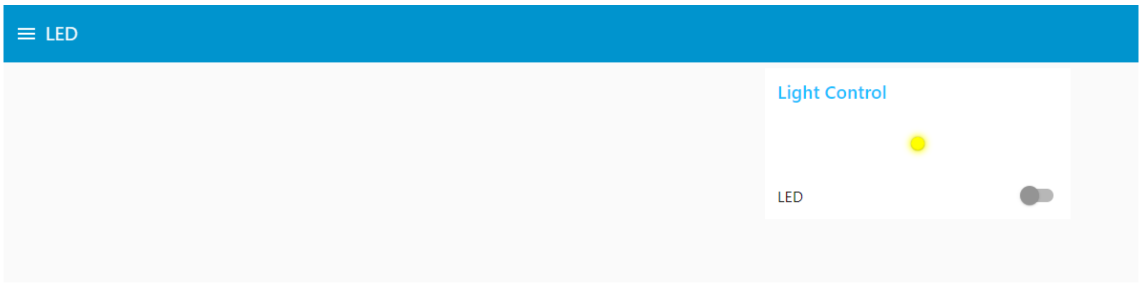
For the configuration of the serial nodes, the value must be 'ASCII string' and the 'Baud Rate' must be in accordance with the value in the Arduino environment, that is, 9600.

The dashboard should look like the dashboard in the image below (for the implementation of the dashboard, we simply use the node *node-red-contrib-ui-led*)

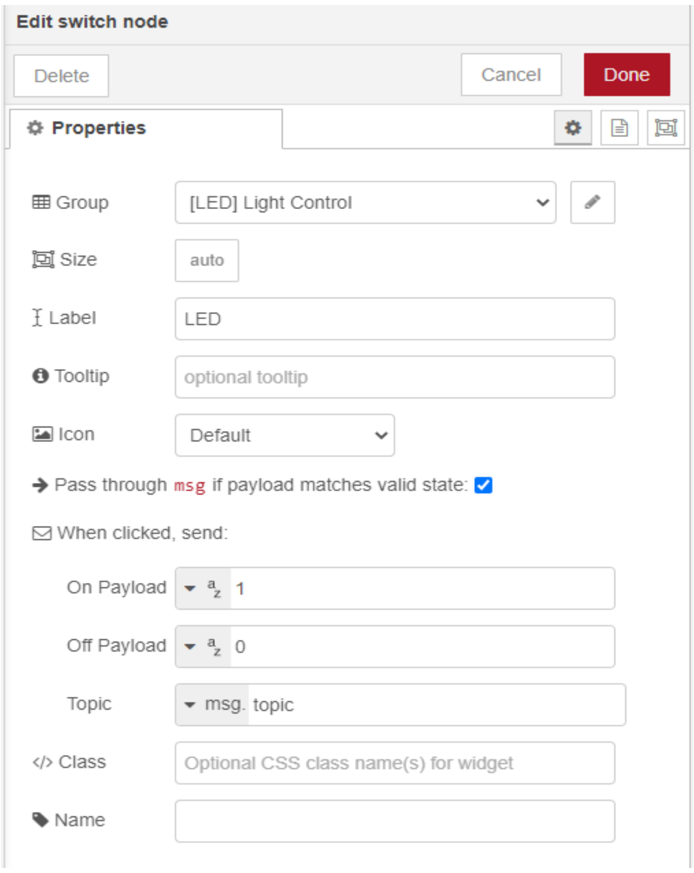
- When you activate the button on the dashboard, the physical led on the Arduino lights up.



- When you activate the physical push-button on the breadboard, the virtual LED lights up.



The configuration for the LED node should be something like this.



Group: [LED] Light Control
Size: auto
Label: LED
Tooltip: optional tooltip
Icon: Default
Pass through msg if payload matches valid state: ✓
When clicked, send:
On Payload: 1
Off Payload: 0
Topic: msg. topic
Class: Optional CSS class name(s) for widget
Name:

Could you also synchronize both LEDs (virtual and on Arduino) so that both of them are turned on/turned off if virtual or real button are pressed?

*Needed nodes: *node-red-node-arduino, node-red-node-serialport, node-red-contrib-ui-led, node-red-dashboard.*

Please include in your report the short description of the performed tasks with screenshots showing the nodes, dashboard and nodes configuration. Also add a photo of the Arduino + breadboard (with the button) + connections as well as the Arduino code.

Would it be possible to use wifi to send and receive the data to turn on the led? Take a look and try the following example (must not be included into the report):

https://docs.arduino.cc/tutorials/nano-rp2040-connect/rp2040-web-server-rgb

Exercise 2: Temperature dashboard

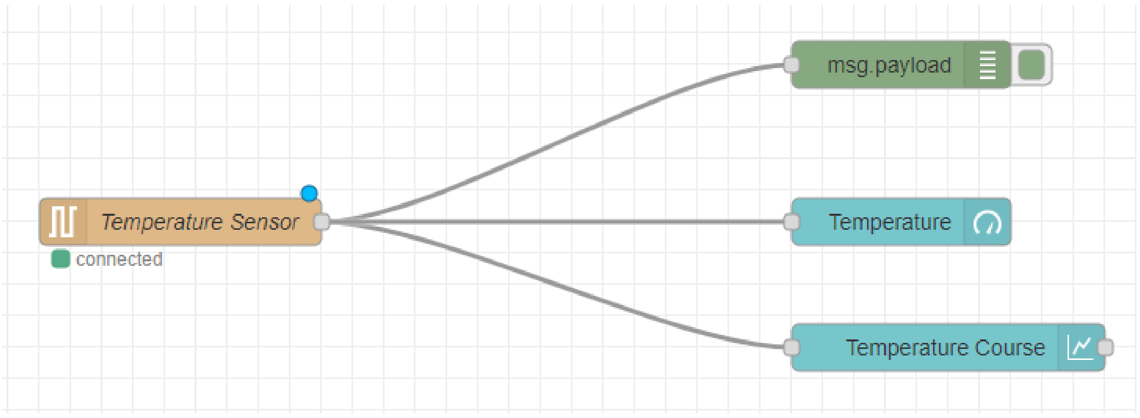
This exercise aims to build a dashboard that shows the temperature from the data collected by a temperature sensor. A program will be implemented on the Arduino IDE to read the temperature value. This value is sent to Node-Red through the serial port and finally displayed on a dashboard. In the node configuration for the dashboard display, some ranges must be used:

- For temperature values greater than 25ºC -> Red
- For temperature values lower than 15ºC -> Blue
- For temperature values between 15ºC and 25ºC -> Green

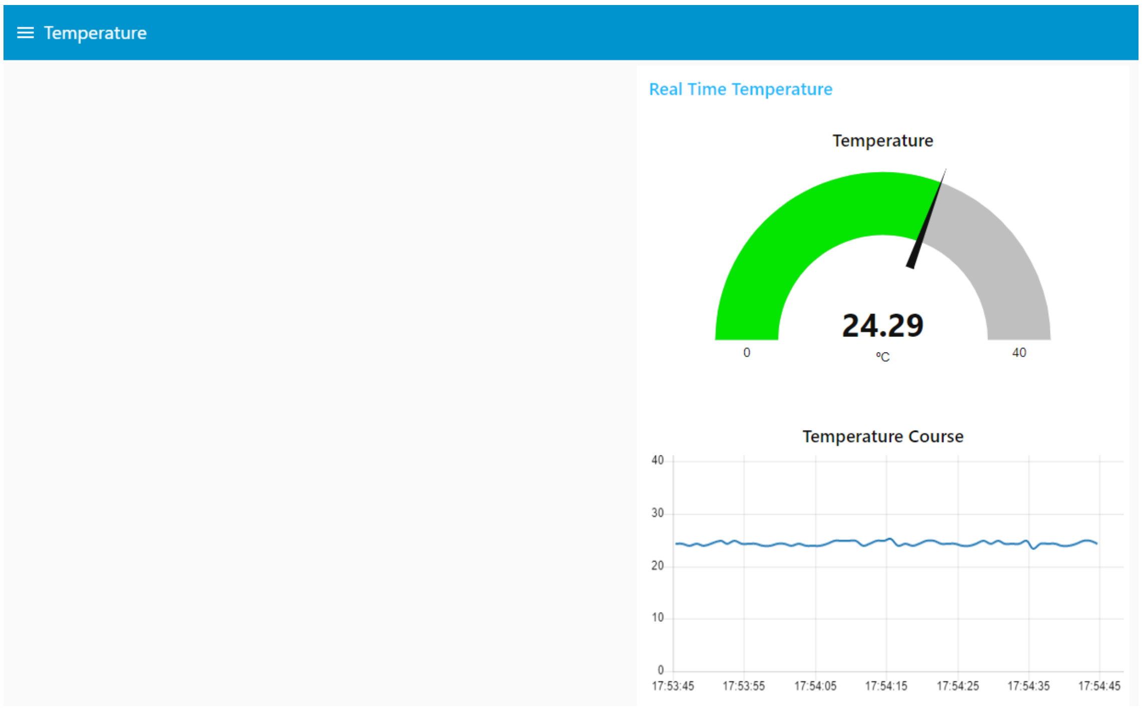
It is important to stop running Node-Red to free the serial port to be able to verify that the Arduino collects the temperature data correctly.

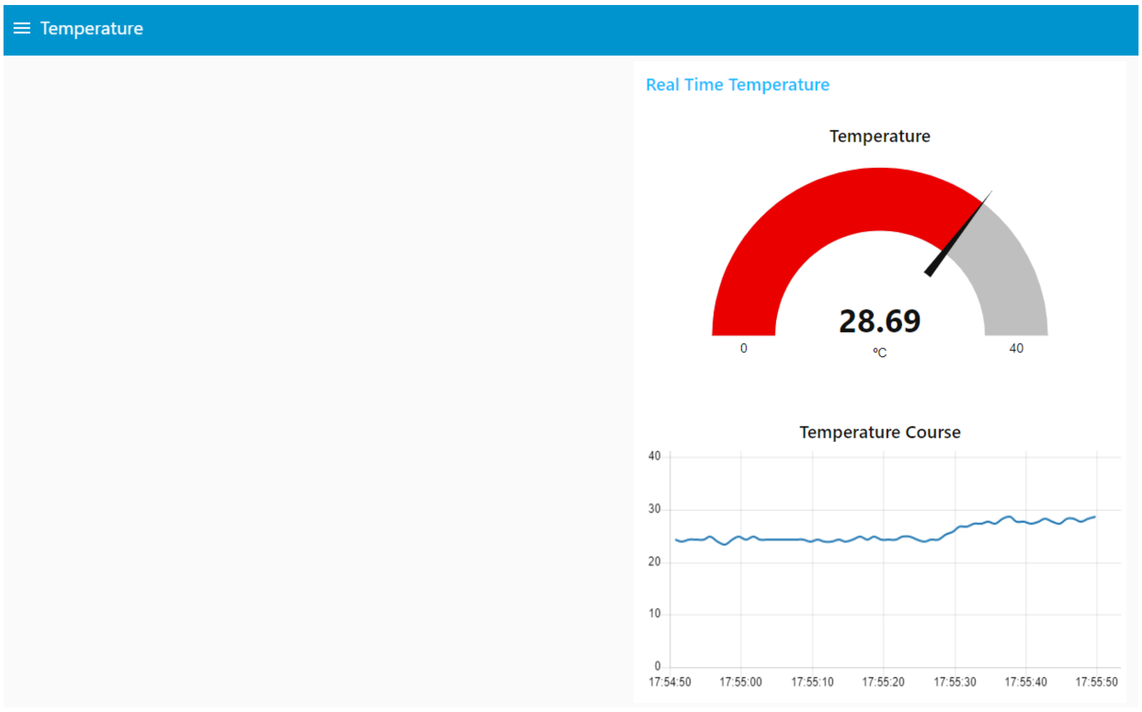
Develop an Arduino program that collects the temperature data in degrees Celsius and sends it through the serial port.

In Node-Red, you have to implement a flow like the following, which reads the values from the serial port and displays them on a dashboard.



Below you can see two screenshots taken from the dashboards. Real-time temperature and temperature courses are displayed. Both nodes used to generate the dashboard must be configured to achieve this result. To see temperature changes, place your fingers around the temperature sensor.





Please include in your report the short description of the performed tasks with screenshots showing the nodes, dashboard and nodes configuration as well as the Arduino code.

Exercise 3: Dashboard, MQTT and Cloud

In this exercise, we will work with the MQTT protocol to send the temperature data from Arduino over Node-Red to a broker that will be configured using the HiveMQ software tool. Once the broker is up and running, the data will be sent to an IoT-cloud service (DATACAKE) to be able to view it and perform some statistics with on it.

Configure the MQTT broker so that it can send and receive data. To do this, register in HiveMQ and create MQTT Client credentials to be used in Node-Red.

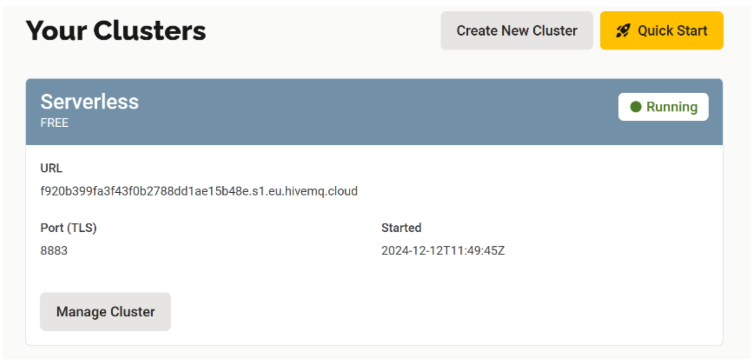
https://www.hivemq.com/company/get-hivemq/

Use HiveMQ Cloud for this exercice.

If you are not familiar with MQTT, read some basic instructions:

HiveMQ Introduction :: HiveMQ Documentation

Once this step is done, you should see something like the images below.

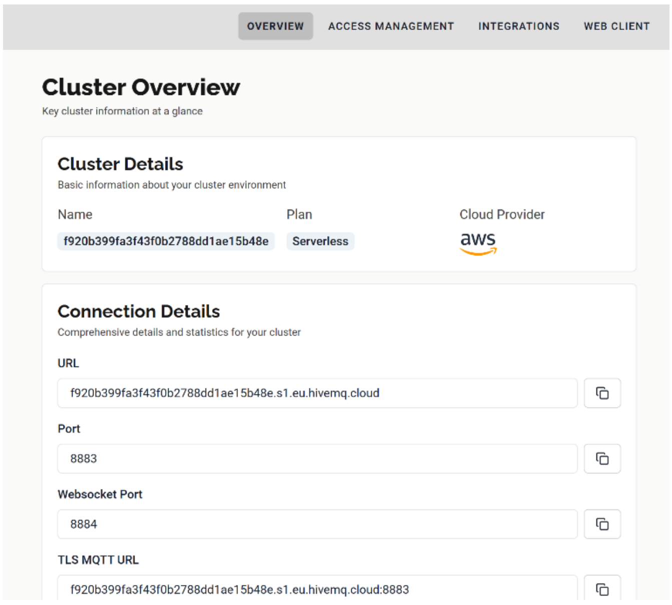

Your Clusters

Serverless
FREE

URL
f920b399fa3f43f0b2788dd1ae15b48e.s1.eu.hivemq.cloud

Port (TLS)
8883

Started
2024-12-12T11:49:45Z


Cluster Overview
Key cluster information at a glance

Cluster Details
Basic information about your cluster environment

Name: f920b399fa3f43f0b2788dd1ae15b48e
Plan: Serverless
Cloud Provider: aws

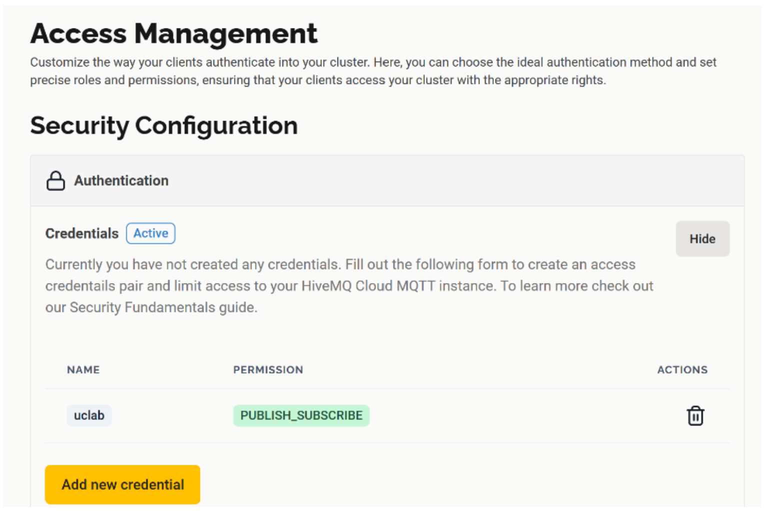
Connection Details
Comprehensive details and statistics for your cluster

URL
f920b399fa3f43f0b2788dd1ae15b48e.s1.eu.hivemq.cloud

Port
8883

Websocket Port
8884

TLS MQTT URL
f920b399fa3f43f0b2788dd1ae15b48e.s1.eu.hivemq.cloud:8883



Access Management

Customize the way your clients authenticate into your cluster. Here, you can choose the ideal authentication method and set precise roles and permissions, ensuring that your clients access your cluster with the appropriate rights.

Security Configuration

Authentication

Credentials Active

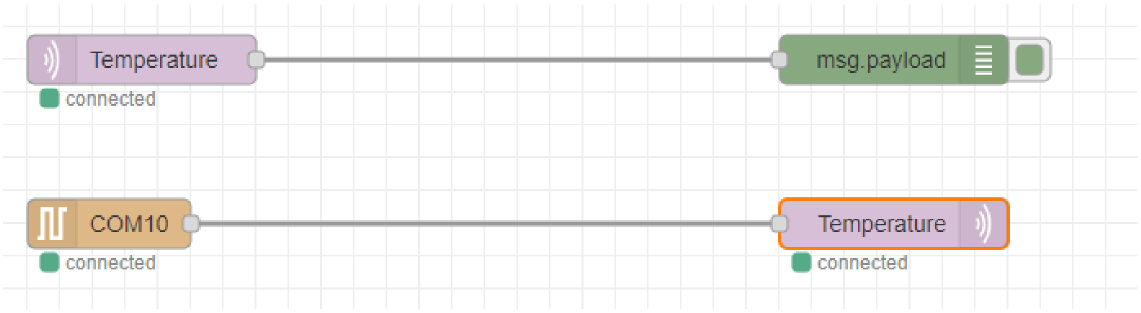
Hide

Currently you have not created any credentials. Fill out the following form to create an access credentials pair and limit access to your HiveMQ Cloud MQTT instance. To learn more check out our Security Fundamentals guide.

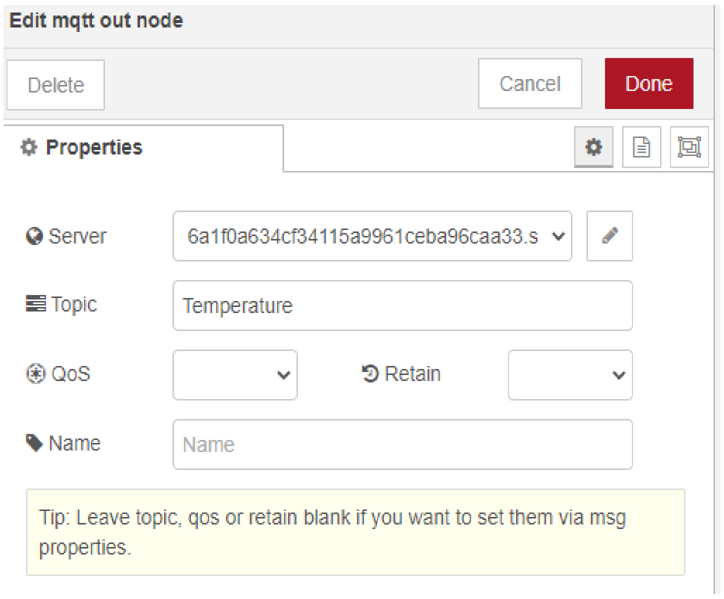
NAME	PERMISSION	ACTIONS
uclab	PUBLISH_SUBSCRIBE	

Add new credential

Create a flow in Node-Red like the one in the image below. To do this, make use of the mqtt-in and mqtt-out nodes and configure them.



Indicate the name of the server, the security data with the MQTT credentials and do not forget to enter the name of the server in the TSL configuration.



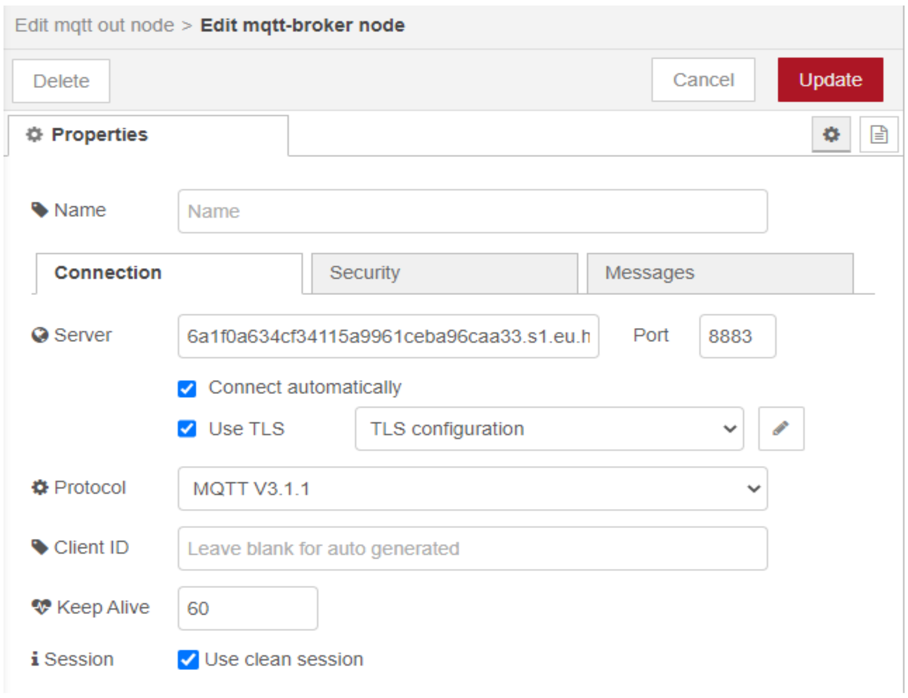
Edit mqtt out node

Delete Cancel Done

Properties

Server	6a1f0a634cf34115a9961ceba96caa33.s
Topic	Temperature
QoS	
Retain	
Name	Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.



Name: Name

Server: 6a1f0a634cf34115a9961ceba96caa33.s1.eu.h Port: 8883

☑ Connect automatically

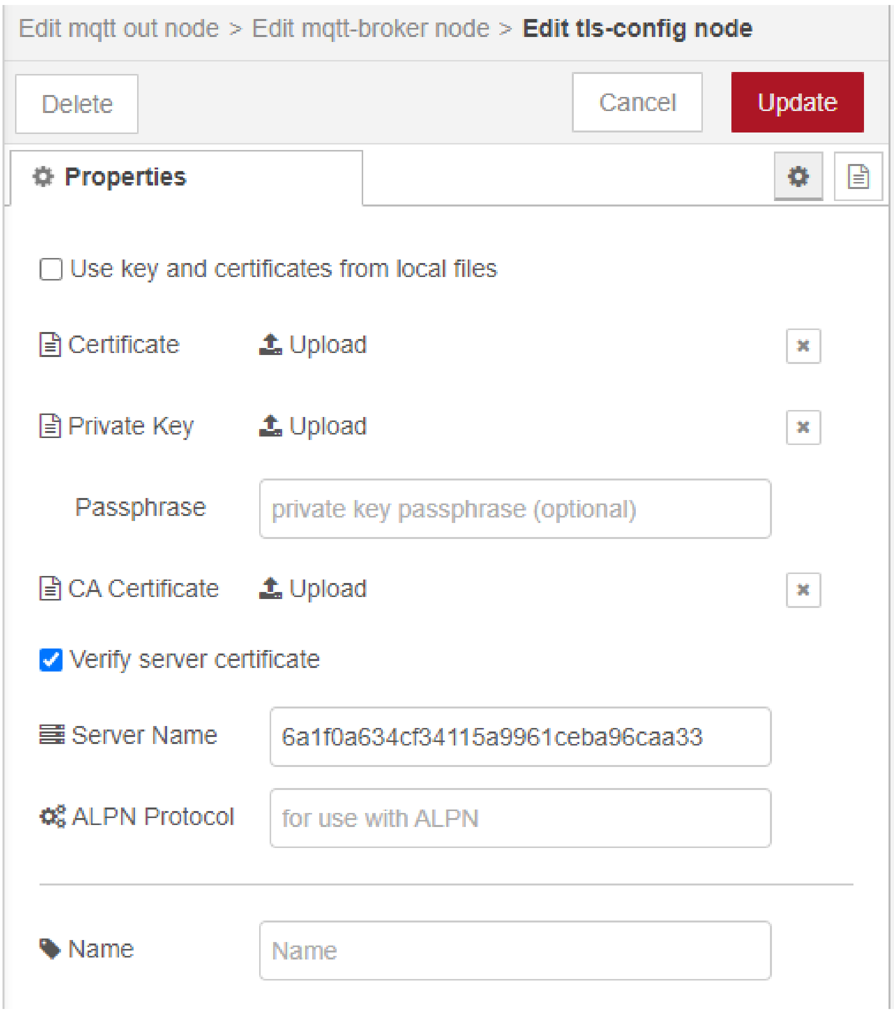
☑ Use TLS: TLS configuration

Protocol: MQTT V3.1.1

Client ID: Leave blank for auto generated

Keep Alive: 60

Session: ☑ Use clean session



Edit mqtt out node > Edit mqtt-broker node > Edit tls-config node
Delete
Cancel
Update
Properties

☐ Use key and certificates from local files

Certificate: Upload

Private Key: Upload

Passphrase: private key passphrase (optional)

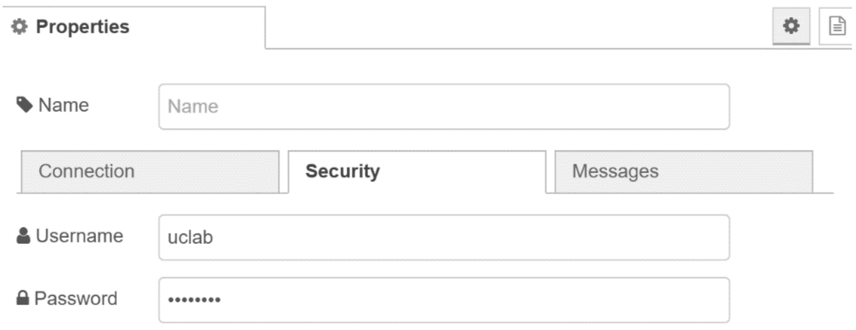
CA Certificate: Upload

☑ Verify server certificate

Server Name: 6a1f0a634cf34115a9961ceba96caa33

ALPN Protocol: for use with ALPN

Name: Name



Properties

Name: Name

Connection | Security | Messages

Username: uclab

Password: ••••••••

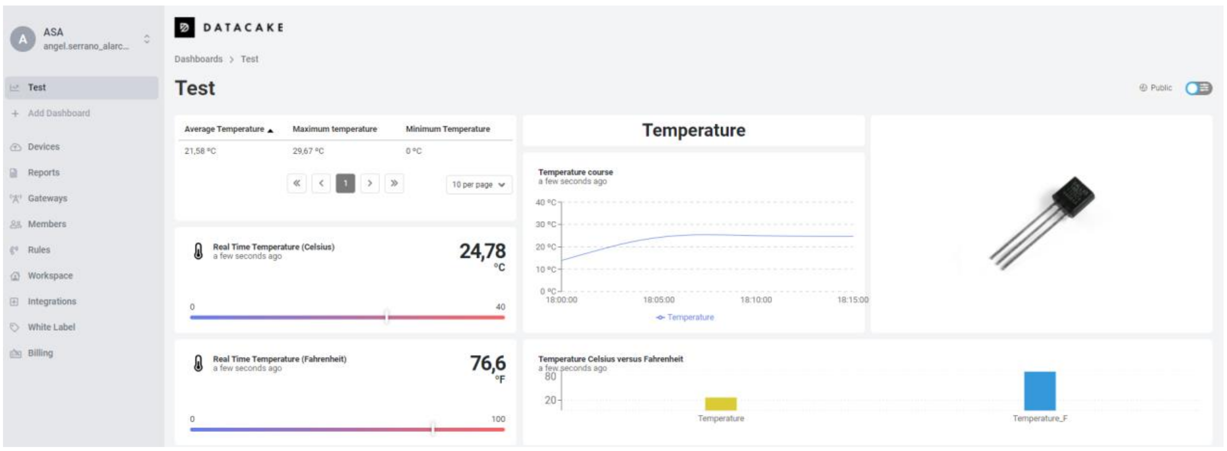
Visualize in a Node-Red Dashboard the temperature from Arduino received over MQTT (acquired from Arduino using serial, published with Node-Red and then subscribed/received with Node-Red using suitable nodes) !

Create an account on DATACAKE and follow the following steps to perform the MQTT integration.

- Create a free account on DATACAKE: https://datacake.co/
- MQTT integration: https://docs.datacake.de/integrations/mqtt

  Use MQTTS as the protocol for connection.

When the integration with DATACAKE is successful, you can work with the data received by the device as you did before. Please visualize the current temperature and its changes over time.



Test

Average Temperature ▲	Maximum temperature	Minimum Temperature
21,58 °C	29,67 °C	0 °C

10 per page

Real Time Temperature (Celsius)
a few seconds ago
24,78 °C
0 40

Real Time Temperature (Fahrenheit)
a few seconds ago
76,6 °F
0 100

Temperature course
a few seconds ago

Temperature Celsius versus Fahrenheit
a few seconds ago

Please include in your report the short description of the performed tasks with screenshots showing the nodes, dashboard and nodes configuration. Furthermore, the screenshots from your MQTT configuration on HiveMQ as well as the output of the DATACAKE dashboard are to be presented in the report.