ERICSSON

IoT Accelerator Developer Portal

Migrating Labels to Custom Fields with Node-RED

Tutorial

2022-05-31

Contents

1 Purpose and Scope

This document is in the scope of IoT Accelerator Service.

Purpose of the document is to describe how to use Node-RED and a pre-configured flow to migrate the content of the legacy "Labels" into the new "Custom Fields", leveraging on Custom Field API.

The Node-RED flow logic will query all the Subscriptions of a specified Company, read the calues of the legacy "Labels" and transfer the values into Custom Fields, with a field name that is configurable (see "newLabel" variable described below).

A Custom Field deletion flow is also provided, meant to delete the created Custom Field. This is provided for testing purpose so to retry the tutorial a number of times if needed.

Please Note custom field does not support special chars except: **_ (Underscore)** **- (dash)** **( ) (round brackets)**
All invalid characters in the Label field will be converted to underscore during the copy process in the custom label

2 NodeRed Setup

The following chapters describe how to install Node-RED, load the preconfigured flow and run it on your local PC.

2.1 Download & install Node-RED® Client

Read how to install Node-RED® from here.

Launch Node-RED®.

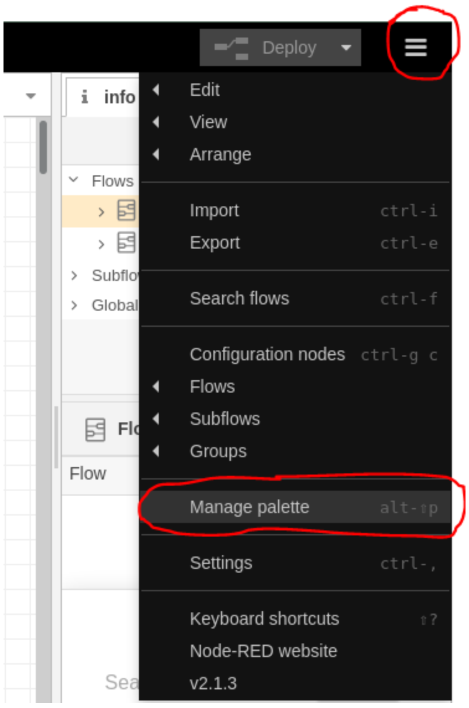
2.2 Configure the environment

The following nodes have to be installed on your NodeRed instance:

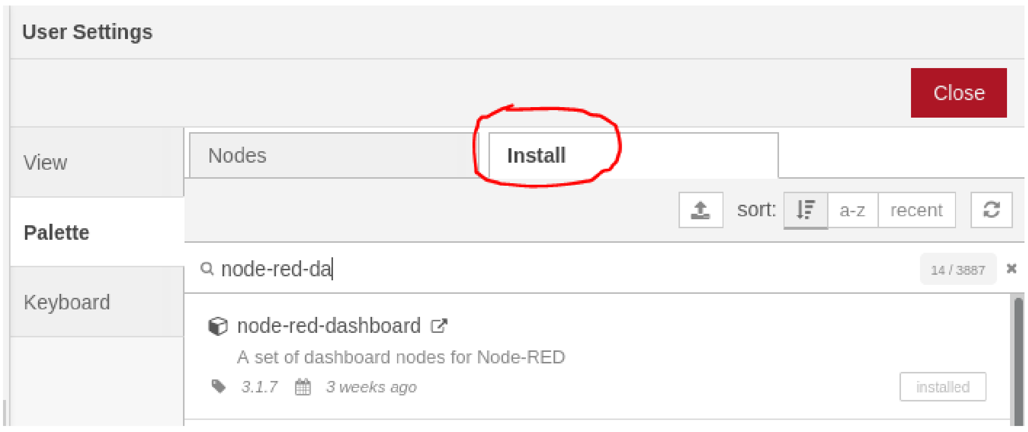
Node	Min. Version
node-red	2.1.4
node-red-dashboard	3.1.2

node-red-node-ui-table	0.3.12

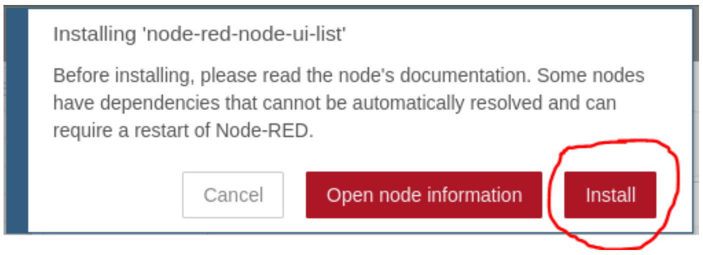
You can install nodes (node-red-dashboard and node-red-node-ui-table) directly within the editor by selecting the Manage Palette option from the main menu (Three lines on top right) to open the Palette Manager.



Select "Install" tabs than type in "Search modules" "node-red-dashboard" when the result has shown click "Install" button



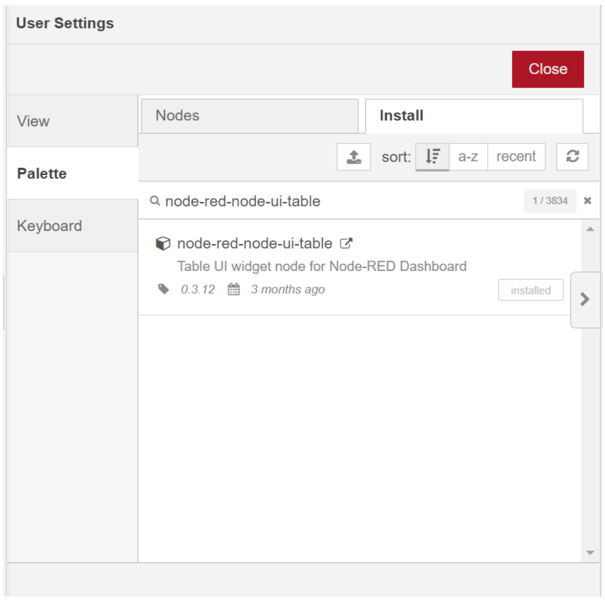
A new pop-up will ask to confirm the installation, click on the red button "Install" to complete the installation process


Installing 'node-red-node-ui-list'

Before installing, please read the node's documentation. Some nodes have dependencies that cannot be automatically resolved and can require a restart of Node-RED.

Make the same to search node-red-node-ui-table.

The 'Nodes' tab lists all of the modules which you have installed, the ones you are using and whether a SW update is available for any of them.



node-red-node-ui-table 1 / 3834

node-red-node-ui-table

Table UI widget node for Node-RED Dashboard

0.3.12 3 months ago installed

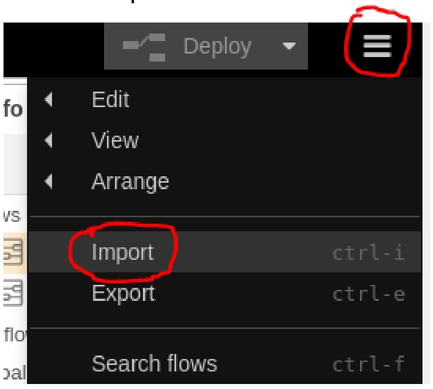
Figure 1

N.B. Click on Close Red button to close the Palette and return to Node-Red Editor.

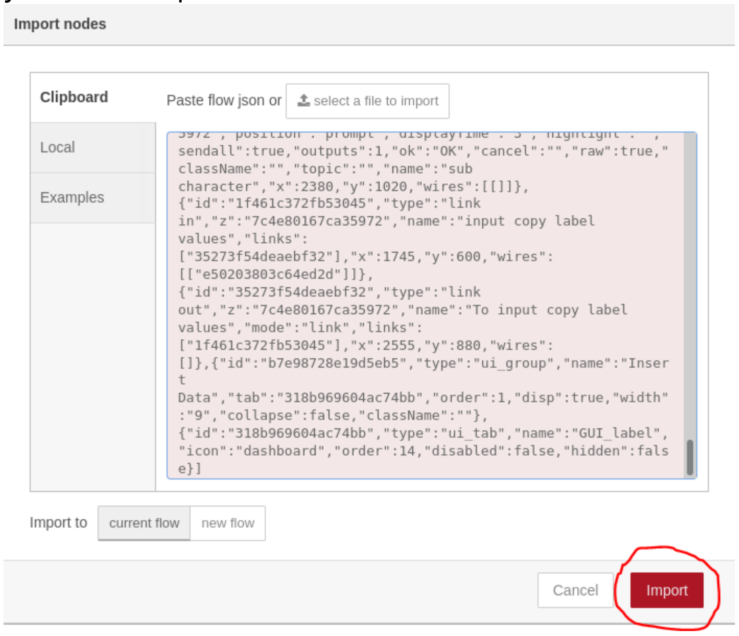
2.3 Import the Flow

- Download the JSON configuration files that correspond to the flow you want to import and extract it to a folder on your desktop.

- In Node-RED®, click on the "*Menu*" icon in the Top-Right Corner then select "Import".



- In the popup window, click on "*select a file to Import*" and select the file you want to upload.


Import nodes

Clipboard

Local

Examples

Paste flow json or select a file to import

```
sendall":true,"outputs":1,"ok":"OK","cancel":"","raw":true,"
className":"","topic":"","name":"sub
character","x":2380,"y":1020,"wires":[[]]},
{"id":"1f461c372fb53045","type":"link
in","z":"7c4e80167ca35972","name":"input copy label
values","links":
["35273f54deaebf32"],"x":1745,"y":600,"wires":
[["e50203803c64ed2d"]]},
{"id":"35273f54deaebf32","type":"link
out","z":"7c4e80167ca35972","name":"To input copy label
values","mode":"link","links":
["1f461c372fb53045"],"x":2555,"y":880,"wires":
[]},{"id":"b7e98728e19d5eb5","type":"ui_group","name":"Inser
t
Data","tab":"318b969604ac74bb","order":1,"disp":true,"width"
:"9","collapse":false,"className":""},
{"id":"318b969604ac74bb","type":"ui_tab","name":"GUI_label",
"icon":"dashboard","order":14,"disabled":false,"hidden":fals
e}]
```

Import to [✓ current flow] new flow

Cancel Import

After the import a new Tab will be shown on the Editor called "Move label to new label"

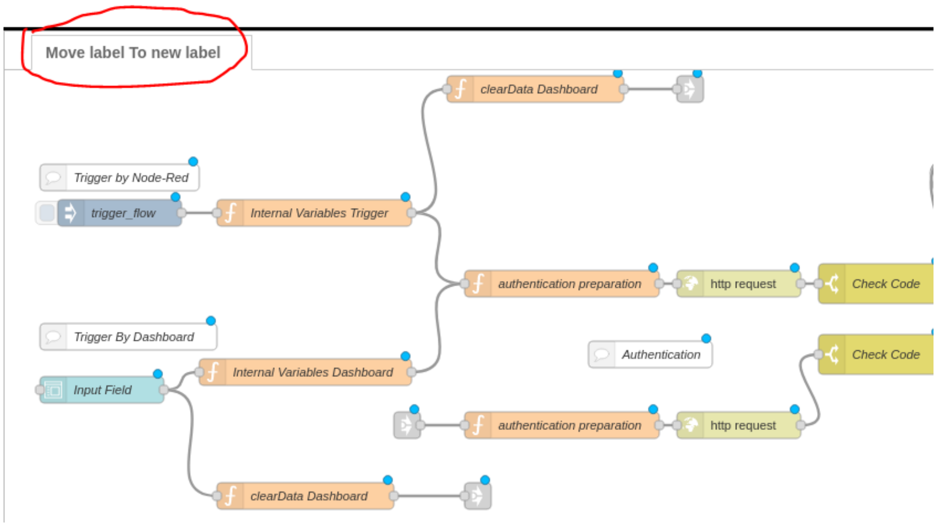


Figure 2

- After importing the file, the Flow will be available in the Navigation Panel on top of the application. Click on "*Deploy*" in the right upper corner of the screen to save and deploy the imported flows to your instance.

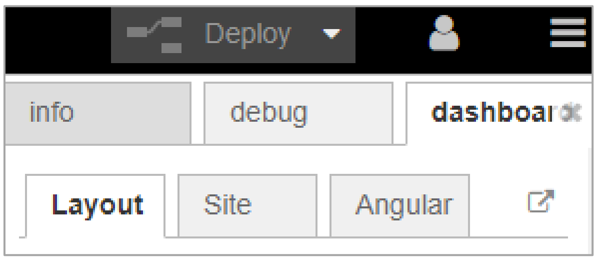
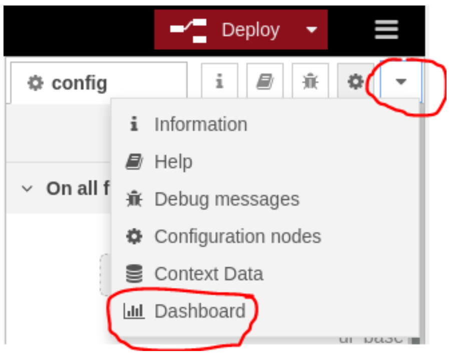
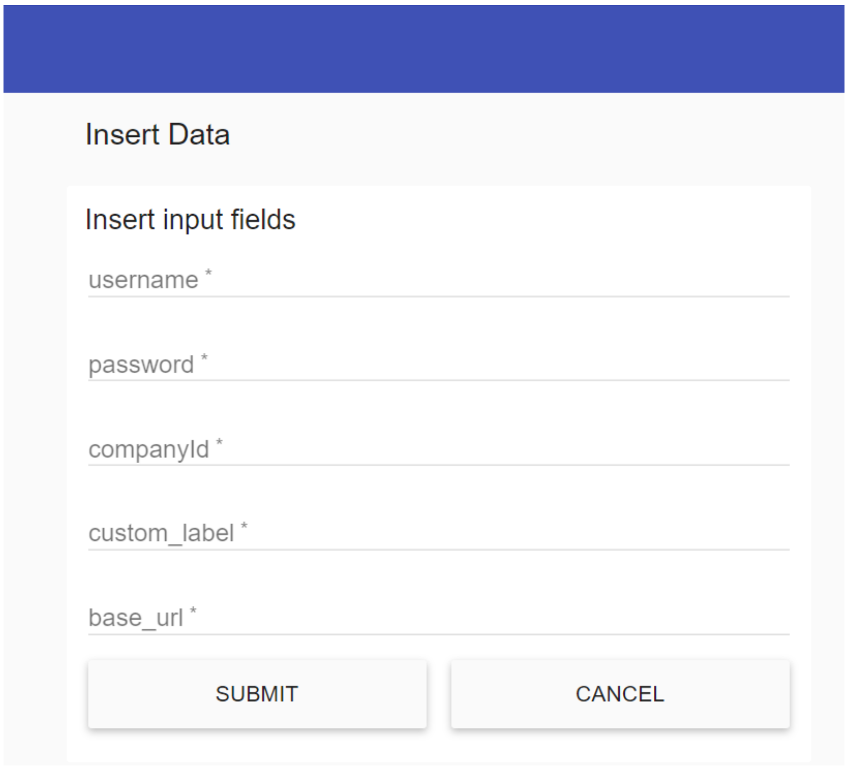


Figure 3

3 Running the flow from the Service Dashboard

A Graphical User Interface is available to the user for configuring the parameters required by the flow to run, and to execute the flow. To open the dashboard, click on the "Dashboard" tab in the right sidebar of the GUI, then on the icon.




Insert Data

Insert input fields

username *

password *

companyId *

custom_label *

base_url *

SUBMIT

CANCEL

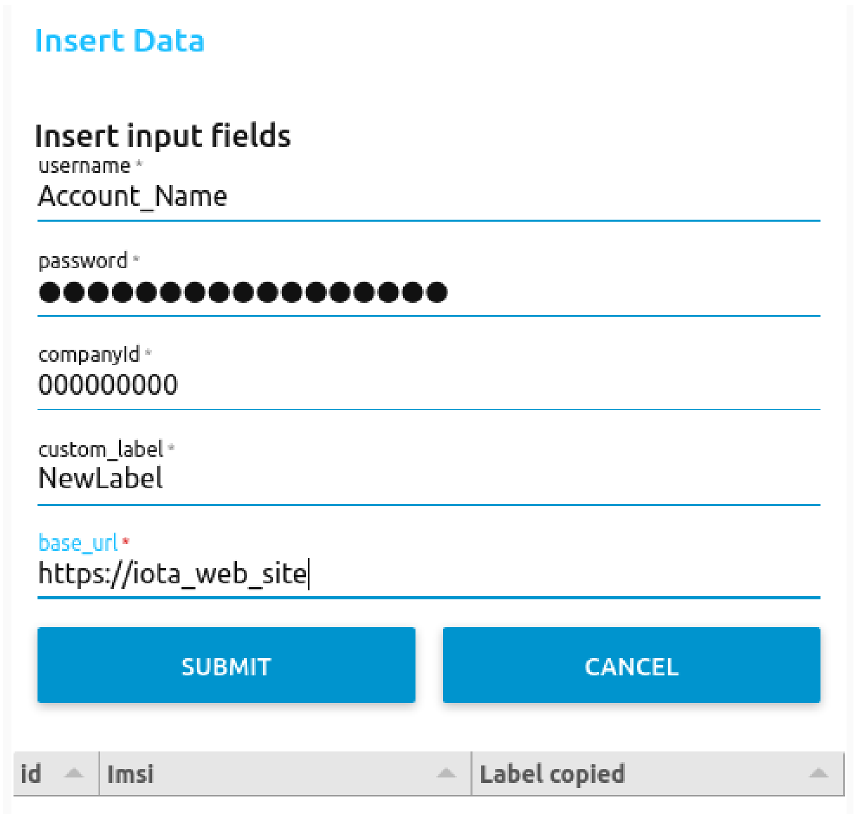
Figure 4

The following table describes the parameters required by the dashboard:

username	username of your IoTA account for consuming APIs
password	password of your IoTA account for consuming API
companyId	Customer number of your Enterprise
Custom_label	The name of the new Custom field to be create
Base_url	Ericsson IoTA https address path

Table 1

Example:


Insert Data

Insert input fields

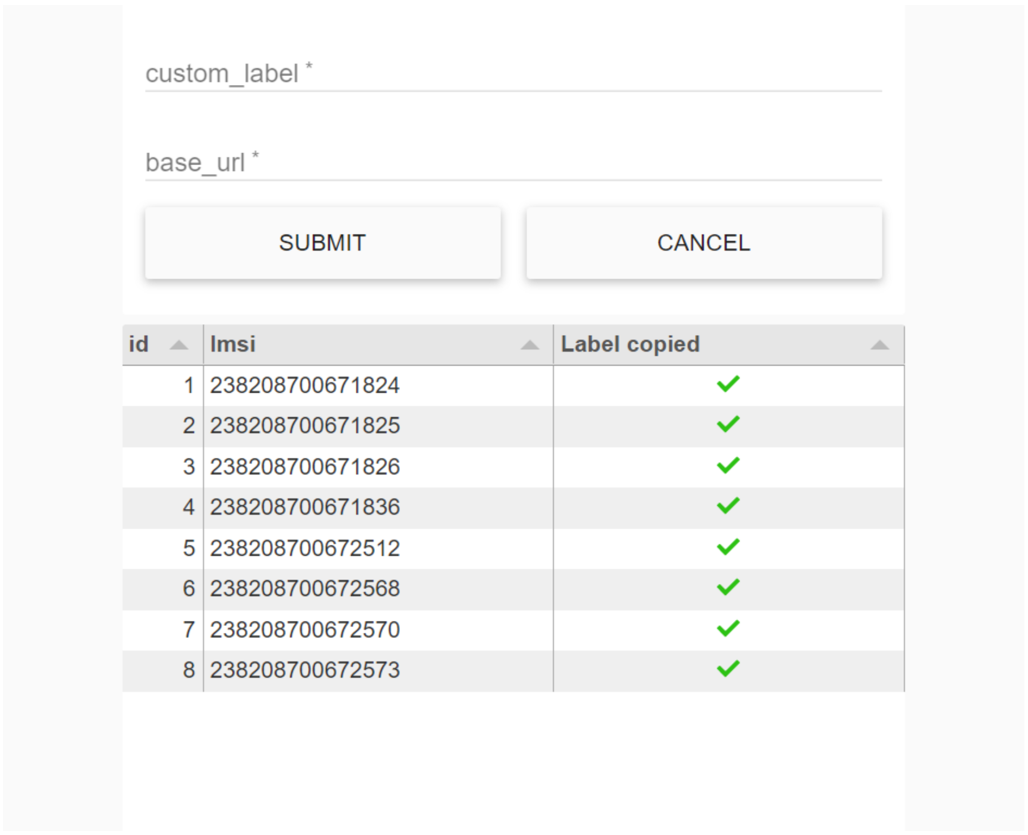
username *	Account_Name
password *	●●●●●●●●●●●●●●●●●
companyId *	000000000
custom_label *	NewLabel
base_url *	https://iota_web_site

id	Imsi	Label copied

Figure 5

Clicking on the SUBMIT button, the flow will run, and all the labels will be migrated into the defined Custom field (custom_label).

All the updated subscriptions will be printed below the Dashboard.



id	Imsi	Label copied
1	238208700671824	✓
2	238208700671825	✓
3	238208700671826	✓
4	238208700671836	✓
5	238208700672512	✓
6	238208700672568	✓
7	238208700672570	✓
8	238208700672573	✓

Figure 6

4 Running the flow from the Node-RED flow

It is also possible to trigger the flow without the help of the dashboard.

To do this you need to configure the parameters in the node

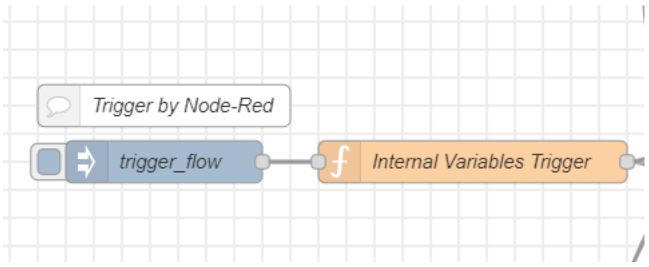
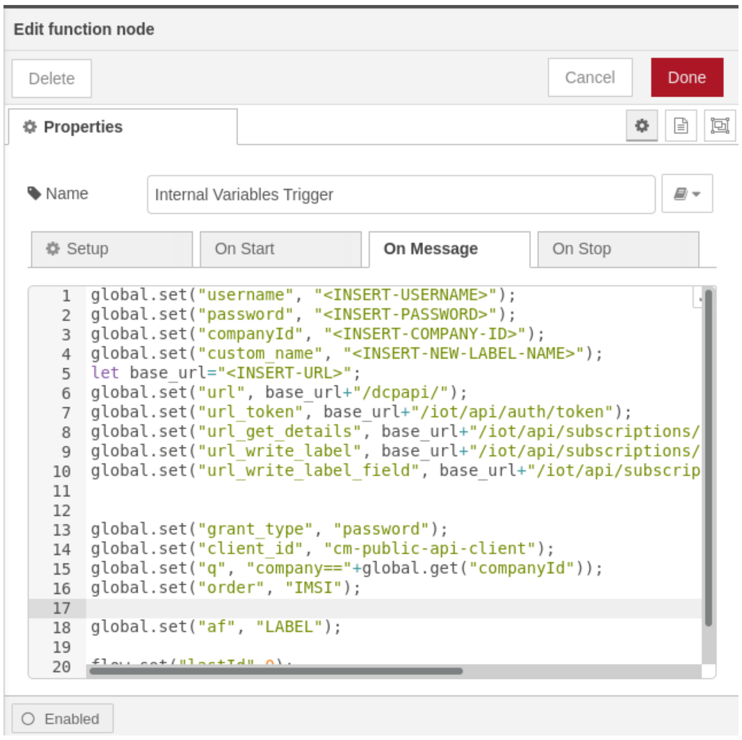


Figure 7

Clicking-on on the **Internal Variables Trigger** node will open a window with the values to be set:



Name: Internal Variables Trigger

```
global.set("username", "<INSERT-USERNAME>");
global.set("password", "<INSERT-PASSWORD>");
global.set("companyId", "<INSERT-COMPANY-ID>");
global.set("custom_name", "<INSERT-NEW-LABEL-NAME>");
let base_url="<INSERT-URL>";
global.set("url", base_url+"/dcpapi/");
global.set("url_token", base_url+"/iot/api/auth/token");
global.set("url_get_details", base_url+"/iot/api/subscriptions/
global.set("url_write_label", base_url+"/iot/api/subscriptions/
global.set("url_write_label_field", base_url+"/iot/api/subscrip


global.set("grant_type", "password");
global.set("client_id", "cm-public-api-client");
global.set("q", "company=="+global.get("companyId"));
global.set("order", "IMSI");

global.set("af", "LABEL");

flow.set("lastId", 0);
```

Figure 8

You need to configure your account data as in below example (in blue).

username	`global.set("username", "`**`user01.1@ericsson.com`**`");`
password	`global.set("password", "`**`xxxxx`**`");`
companyId	`global.set("companyId", "`**`00000000`**`");`
Custom_label	`global.set("custom_name", "`**`NewLabelName`**`");`
Base_url	`Ericsson IoTA https address path`

Table 2

Once the data has been configured, just click on the node **trigger_flow** to start the flow

5 Description of the Node-Red Flow

The flow can be described by the following blocks (they are not described here in logical sequence):

Triggering

As described, the triggering can be done in 2 ways: through the dashboard or through the Node-Red flow. In both cases, some variables need to be configured since are required by the APIs.

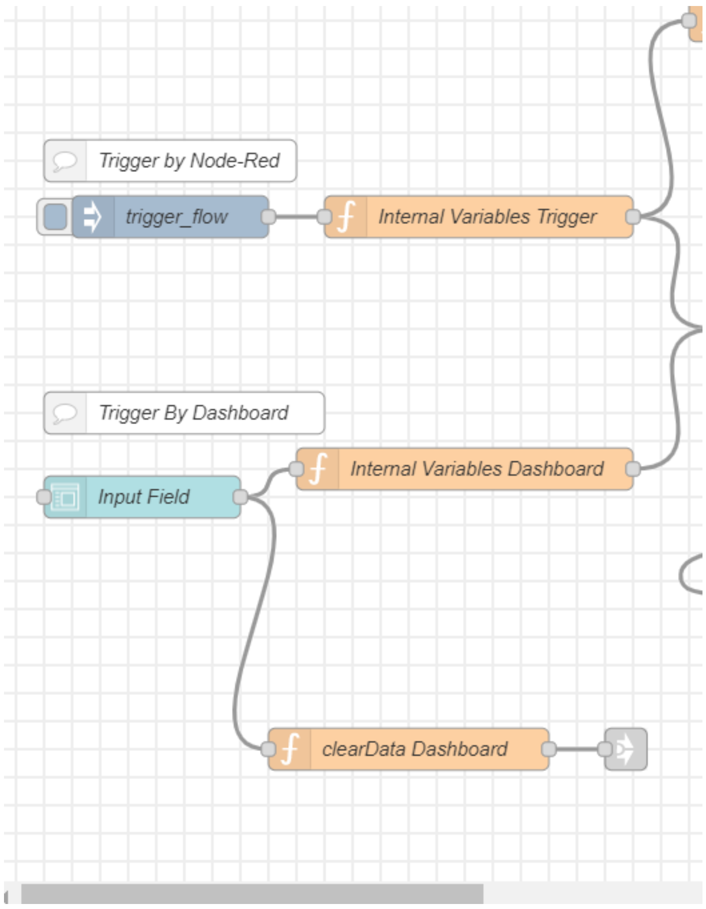


Figure 9

Here is a list of variables used by the flow:

Name	Example	Description
username	`global.set("username", "user01.1@ericsson.com");`	Username IOT portal
password	`global.set("password", "xxxxx");`	Password IOT portal
companyId	`global.set("companyId", "00000000");`	Enterprise Id Iot Portal
custom_label	`global.set("custom_name", "NewLabelName");`	Name new custom Field
base_url	https://IoTa_web_URL	Ericsson IoTA https address path
url_token	`global.set("url_token", <"base_url”>/iot/api/auth/token");`	url for request token
url_get_details	`global.set("url_get_details", <"base_url”>/iot/api/subscriptions/details");`	url for get subscriptions detail
url_write_label	`global.set("url_write_label", <"base_url”>/iot/api/subscriptions/custom-fields");`	url for create new label
url_write_label_field	`global.set("url_write_label_field", <"base_url”>/iot/api/subscriptions/custom-fields/requests");`	url for copy the label value

Table 3

Authentication

Authentication is performed to get the Access Token to be then reused in API requests.

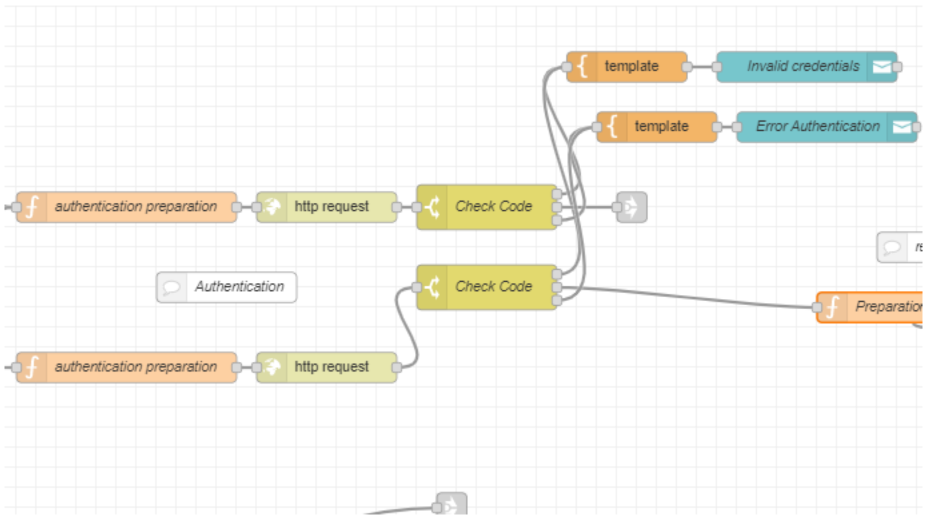


Figure 10

The TOKEN is requested to the following endpoint:

```
url: "<"base_url”>/iot/api/auth/token"
```

where the headers and payload of the API is as follows:

```
headers: object

Content-Type: "application/x-www-form-urlencoded"
payload: grant_type=password&client_id=undefined&username=user01.1@er
icsson.com&password=xxxx
```

Errors are communicated to the user via dashboard alerts.

Check if name label already exists than create of new custom Label

In this part of the flow a new label is created after checking if it already exists, or if the maximum number of custom labels per companyId (EnterpriseId) has been reached.

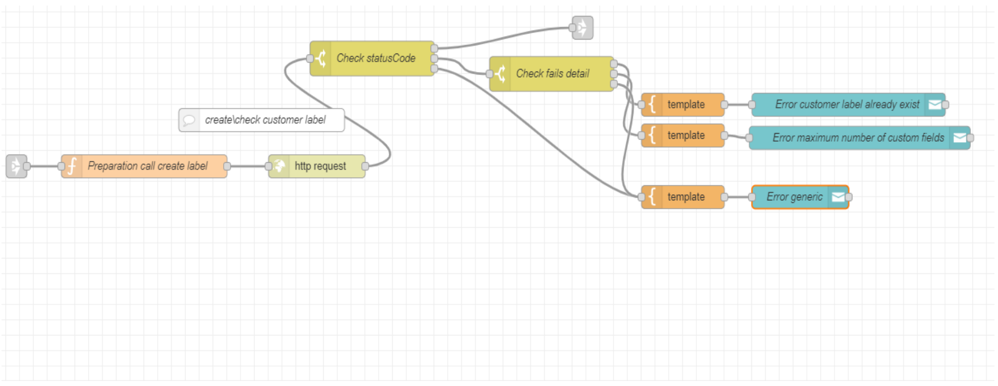

Check statusCode
Check fails detail
create\check customer label
Preparation call create label
http request
template
Error customer label already exist
template
Error maximum number of custom fields
template
Error generic

Figure 11

This API endpoint is used to perform the Custom Label Change:

`url: <"base url”>/iot/api/subscriptions/custom-fields`

with the following parameters in the payload:

Name	Example	Description
companyId	`0000000`	The ID of the organization, .
fieldName	`NewLabelName`	Custom field name, up to 60 characters.
type	`FREE_TEXT`	Custom field type, FREE_TEXT

If the creation of the customer was successful, the API answers with:

201 OK Response

(Only the response status code is returned).

If any problems may occur, the HTTP response status code and the problem details object provide information about the problem as per below table:

Name	Description
type	An absolute URI that identifies the problem type. When dereferenced, it might provide human-readable documentation for the problem type in HTML format. The URI is a constant, for example, https://www.iana.org/assignments/http-status-codes.
title	A short summary of the general problem type. Written in English and readable for engineers (usually not suited for non-technical stakeholders and not localized).
status	The HTTP status code generated by the origin server for this occurrence of the problem.
detail	A human-readable explanation specific to this occurrence of the problem.
instance	An absolute URI that identifies the specific occurrence of the problem. It might yield further information if dereferenced.

Table 4

In particular, if the label already exists or if the number of possible labels (10) for that CompanyId has been exceeded, the user is notified with error messages on the Dashboard.

Read CompanyId subscriptions

In this part of the flow all the subscriptions of the CompanyId are queried in order to understand if they have a legacy "Label" associated.

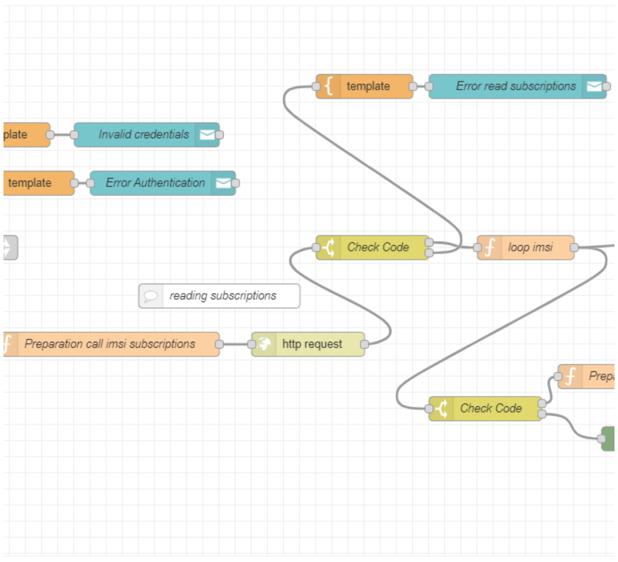


Figure 12

This API endpoint is used to get the Label information for all the subscriptions of a Company:

url: <"base_url”>/iot/api/subscriptions/details?q=company==00000000&order=IMSI&additionalFields=LABEL

Errors, if any, are communicated to the user via dashboard alerts.

Copy the LABEL field value in the Customer Label created previously

This part of the flow is performed as many times as the number of subscriptions which have a **Label** field assigned.

If the label Field value contains a character like *~`!@#$%^&*+={}[];:'"<>,/* it will be replaced with the character _.

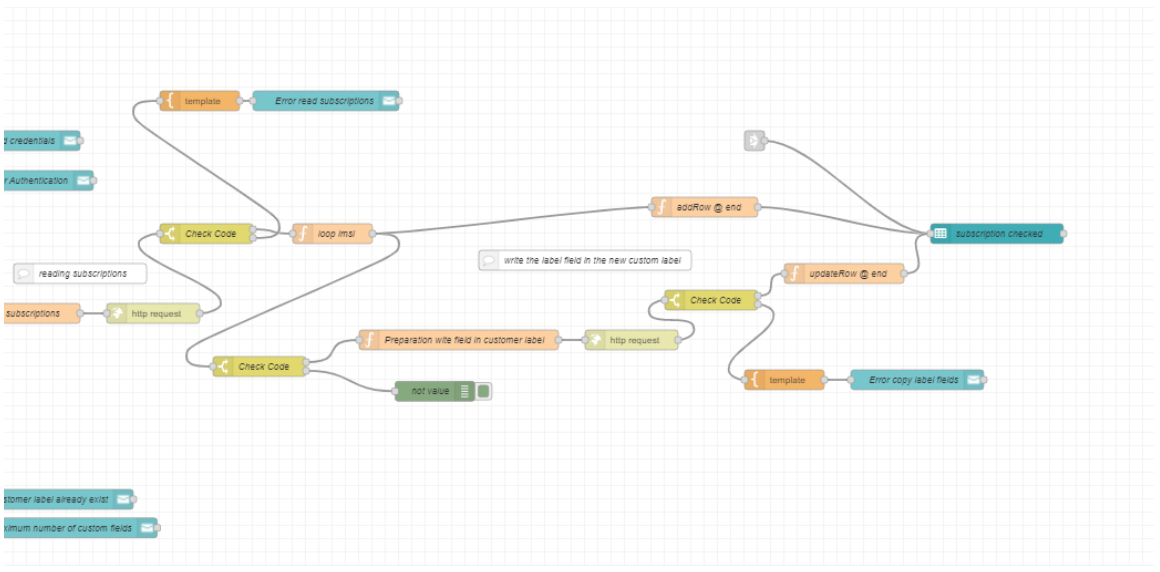


Figure 13

For every subscription this API endpoint is used:

url: <"base_url”>/iot/api/subscriptions/custom-fields/requests

with the following parameters in the payload:

Name	Example	Description
identifierType	IMSI	The type of the identifier, one of the following: IMSI

Name	Example	Description
`identifiers`	`["123456789012345"]`	Imsi of the subscription
`customFields`	-	Contains the attached custom fields.
`fieldName`	`-newLabelName`	The name of the customer label
`fieldValue`	`-test_value_label`	Custom field value

If the copy of label value is successful , the API reply will be:

201 OK Response

Only the response status code is returned. Errors are communicated to the user via dashboard alerts.