



**VIT®**  
Vellore Institute of Technology  
(Deemed to be University under section 3 of UGC Act, 1956)

**SCHOOL OF ELECTRONICS ENGINEERING  
B.TECH. ECE/ECM  
ECE3502 – IoT DOMAIN ANALYST**

**LAB MANUAL**

**List of Experiments**

S. no	Title of the Experiment	Page
1.	Display Light Intensity using Node-Red and MQTT	3
2	Web Based Application (HTML) Form Creation & Submission in Node-Red	6
3.	Online Media Monitoring Using Node-Red   Feedparser Node	13
4.	Front End for Node-Red (FRED) a cloud-based Node-Red	20
5.	Weather Report Analysis using Openweathermap node in Node-Red and FRED	26
6.	ThingsBoard – NodeRed Posting Data in Dashboard through HTTP	32
7.	Email using Node-Red	36
8.	Telemedicine using Node-Red and FRED	41
9.	KNIME Analytics Platform – Sales Analytics	48
10.	Data Manipulation in Knime: String Manipulation, Math Formula, Rule Engine	51
11.	K Means Clustering: Unsupervised Machine Learning	56

## Experiment 1

# Display Light Intensity using Node-Red and MQTT

Aim: Using appropriate nodes in Node Red measure the light intensity in the room and output data to the web API. Display the output in Debug Monitor, UI and send the data through MQTT and display the output in HiveMQ.

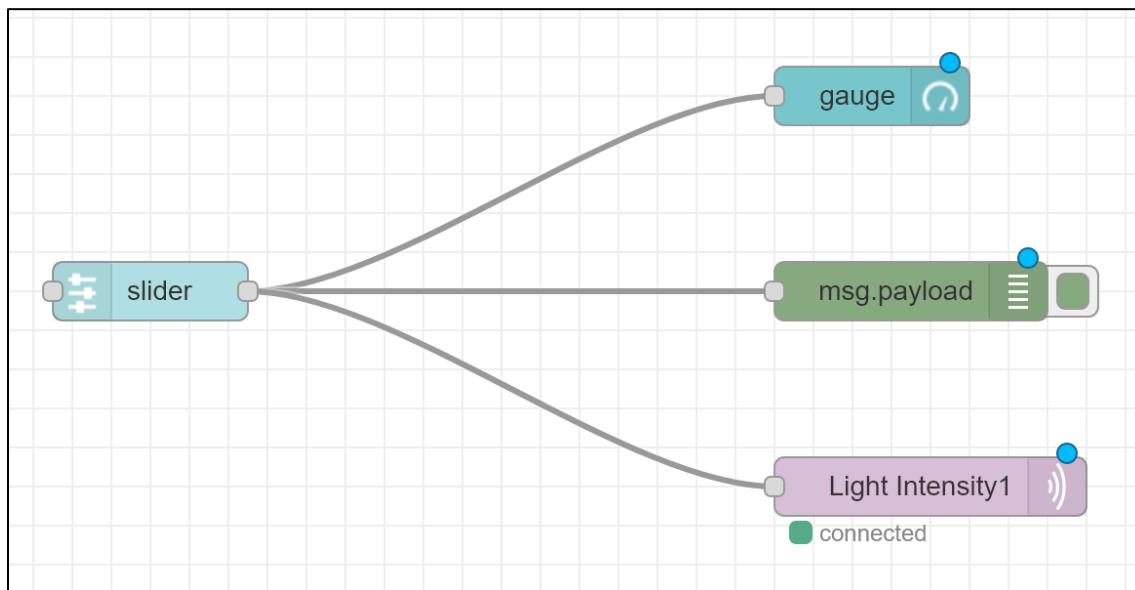
Theory:

- MQTT (Message Queuing Telemetry Transport) is an open OASIS and ISO standard (ISO/IEC 20922) lightweight, publish-subscribe network protocol that transports messages between devices. The protocol usually runs over TCP/IP; however, any network protocol that provides ordered, lossless, bi-directional connections can support MQTT. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited.
- MQTT is a publish-subscribe-based messaging protocol used in Internet of Things. The goal is to provide a protocol, which is bandwidth-efficient and uses little battery power.

Procedure:

- Trigger node red in cmd using node-red -v
- <http://127.0.0.1:1880/>
- Arrange nodes according to the flow and assign properties
- Open HiveMQ and get the output
- <http://www.hivemq.com/demos/websocket-client/>

Flow:



## Properties:

**Edit slider node**

**Properties**

- Group: [Home] Light Intensity
- Size: auto
- Label: slider
- Tooltip: optional tooltip
- Range: min 0, max 200, step 1
- Output: only on release
- If msg arrives on input, set slider to new payload value:
- When changed, send:
  - Payload: Current value
  - Topic:
  - Name:

**Edit gauge node**

**Properties**

- Group: [Home] Light Intensity
- Size: auto
- Type: Gauge
- Label: gauge
- Value format: {{value}}
- Units: units
- Range: min 0, max 200
- Colour gradient: (green, cyan, purple)
- Sectors: 0 ... optional ... optional ... 200
- Name:

**Edit debug node**

**Properties**

- Output: msg.payload
- To:
  - debug window
  - system console
  - node status (32 characters)
- Name: Name

**Edit mqtt out node**

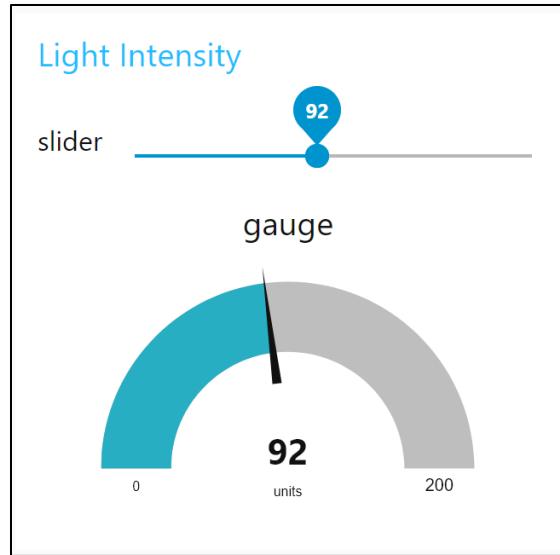
**Properties**

- Server: 181107@broker.mqttdashboard.com:1883
- Topic: Light Intensity1
- QoS: 1, Retain: false
- Name: Name

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

Output:

Dashboard:



MQTT:

The figure shows the MQTT node configuration in Node-RED. It includes tabs for 'Connection' (status: connected), 'Publish' (Topic input, QoS 0, Retain checkbox, Publish button), 'Subscriptions' (Add New Topic Subscription button, listed subscription for Qos: 2 Light Intensity1), and 'Messages' (two messages received: 92 and 97, both from Topic: Light Intensity1).

Inference: A slider node is used to control the light intensity by the user. This is measured and published to the MQTT server using MQTT out node

Result: Hence, Light Intensity has been measured, displayed and published using MQTT and Dashboard

## Experiment 2

# Web Based Application (HTML) Form Creation & Submission in Node-Red

Aim: To create a web-based application using form creation and submission with HTML in node-red

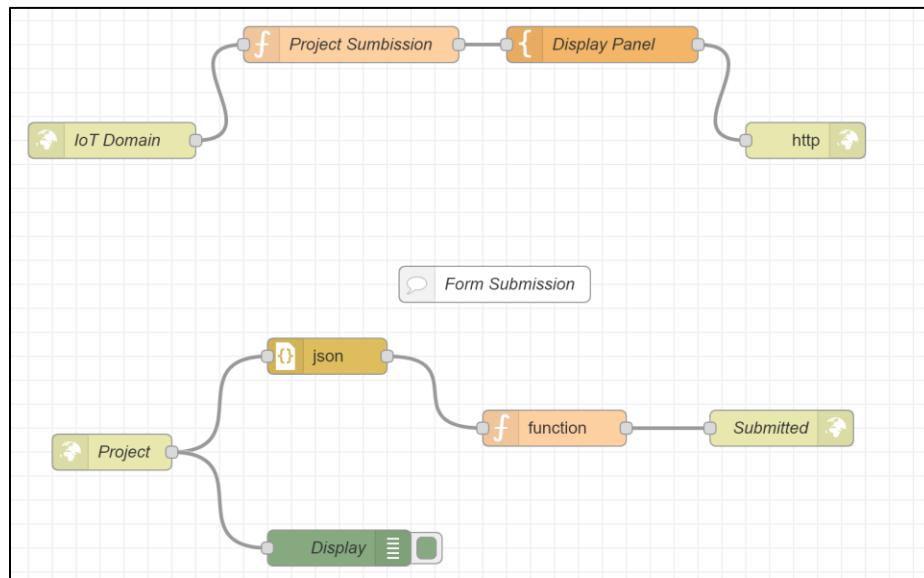
### Theory:

- Web API: Web API as the name suggests, is an API over the web which can be accessed using HTTP protocol.
- Mustache: Mustache is a simple web template system. Mustache is described as a logic-less template engine because it does not have any explicit control flow statements, such as if and else conditionals or for loops. Looping and conditional evaluation can be achieved using section tags processing lists and lambdas.

### Procedure:

- Trigger node red in cmd using node-red -v
- Arrange nodes as shown in the flow and configure properties
- Run the web app in the browser and insert values in the form

### Flow:



### Properties:

The image displays four separate dialog boxes from a node-based configuration interface:

- Edit http in node**: A configuration dialog for an HTTP request node. It includes fields for Method (GET), URL (/IoTDomain), and Name (IoT Domain). Buttons for Delete, Cancel, and Done are at the top.
- Edit http response node**: A configuration dialog for an HTTP response node. It includes fields for Name (Name) and Status code (msg.statusCode). Buttons for Delete, Cancel, and Done are at the top.
- Edit function node**: A configuration dialog for a function node named "Project Sumbission". It has tabs for Setup, Function, and Close. The Function tab contains the following code:

```

1 msg.url="Project";
2 return msg;

```
- Edit template node**: A configuration dialog for a template node named "Display Panel". It includes a "Property" dropdown set to "msg.payload" and a "Template" code editor. The template code is:

```

1 <!DOCTYPE html>
2
3 <html>
4   <head>
5     <h1 style="background-color:DodgerBlue;">IoT Domain</h1>
6   </head>
7   <body>
8     <p style="background-color:Tomato;">
9
10    <OL>
11      <LI> Enter your Name.
12      <LI> Registration Number.
13    </OL>
14
15    <h4>
16      <a href="https://projectmark.com/"> Project Mark</a>
17
18  </h4>
19
20  <form method="post" action="/{{url}}">

```

HTML Code:

```

<!DOCTYPE html>

<html>
<head>
  <h1 style="background-color:DodgerBlue;">IoT Domain</h1>
</head>
<body>

```

```
<p style="background-color:Tomato;">

<OL>
<LI> Enter your Name.
<LI> Registration Number.
</OL>
<h4>
    <a href="https://projectmark.com/"> Project Mark</a>

</h4>
<form method="post" action="/{{url}}">
    <label for="name">First name:</label><br>
    <input type="text" id="fname" name="fname"><br>

    <label for="reg">Reg No:</label><br>
    <input type="text" id="reg" name="reg" ><br><br>

    <label for="topic">Project Title:</label><br>
    <input type="text" id="topic" name="Project Topic" ><br><br>

    <input type="submit" value="Submit">

    <input type="reset" value="Reset" >
</form>

</body>
</html>
```

a

**Edit http in node**

- Method: POST
- Accept file uploads?
- URL: /Project
- Name: Project

**Edit json node**

- Action: Convert between JSON String & Object
- Property: msg. payload
- Name: Name
- Format JSON string

**Edit function node**

- Name: Name
- Setup Function:

```
1 msg.payload = "Data Submitted and is available in debug window: "+msg.payload
2
3 return msg;
```

**Edit http response node**

- Name: Submitted
- Status code: msg.statusCode

**Edit debug node**

- Output: msg. payload
- To:
  - debug window
  - system console
  - node status (32 characters)
- Name: Display

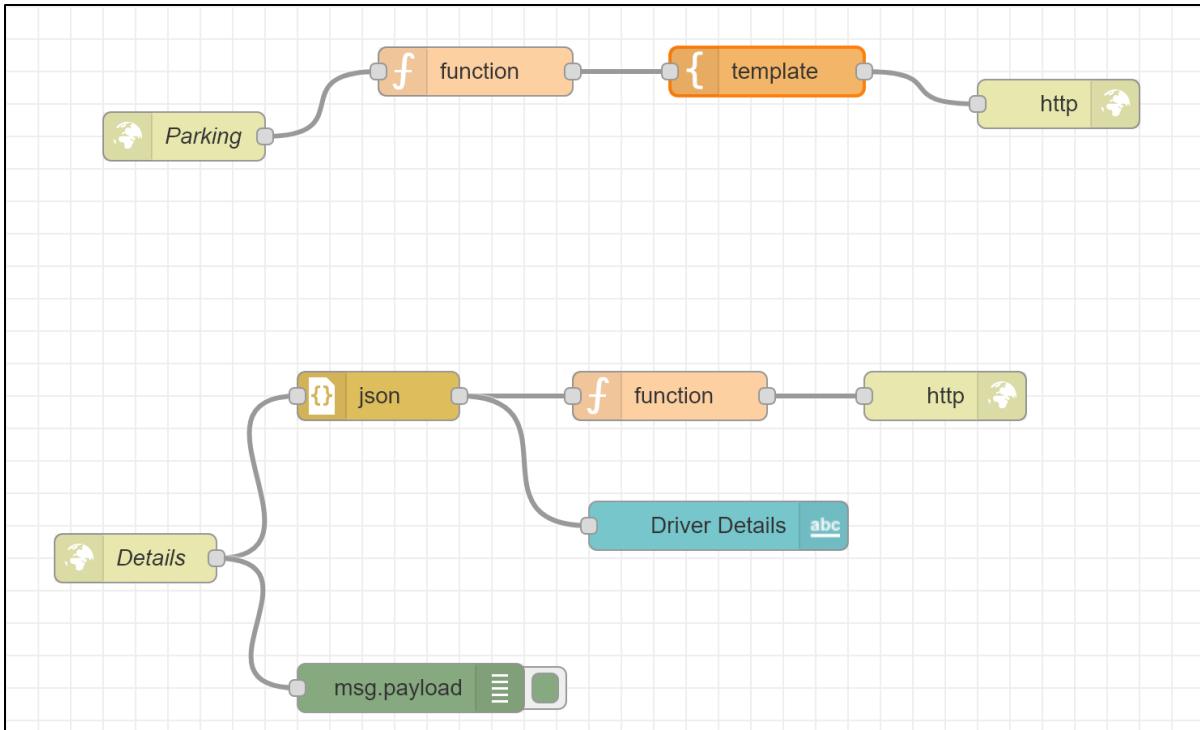
Web Application (Form)

The screenshot shows a web browser window with the URL 127.0.0.1:1880/IoTDomain. The title bar says "IoT Domain". The page content includes instructions: "1. Enter your Name.  
2. Registration Number." Below this is a section titled "Project Mark". It contains fields for "First name:" with value "Anila", "Reg No:" with value "18BLC1107", and "Project Title:" with value "Air Quality Monitoring". At the bottom are "Submit" and "Reset" buttons.

The screenshot shows a web browser window with the URL 127.0.0.1:1880/Project. The title bar says "Project". The page content displays the message "Data Submitted and is available in debug window; {"fname":"Anila","reg":"18BLC1107","Project Topic":"Air Quality Monitoring"}".

**Task:** Using Node-Red build a web-based application to automate a smart parking system in a metro station. Create a form through HTML and CSS to get car owner details to automate the parking system. Display the received details in the debug monitor and UI. Create a hyperlink to show the parking rates of the metro station

Flow:



HTML Code:

```

<!DOCTYPE html>

<html>
  <head>
    <h1 style="background-color:Green;text-align:center;font-family:century gothic;color:white">Parking</h1>

    <style>
      h4 {text-align: center; font-family:century gothic}
      form {text-align: center; font-family:century gothic}
      body {text-align: center; font-family:century gothic}

    </style>
  </head>
  <body>
    <p style="background-color:Tomato;">

    <OL>
      <LI> Enter your Name.
      <LI> License Plate Number.
    </OL>

    <h4>
      <a href="https://chennaimetrorail.org/wp-content/uploads/2018/05/Parking-Tariff-stations.pdf"> Check Fare Rates</a>
    </h4>
  
```

```

<form method="post" action="/{{url}}">

    <label for="name">Name:</label><br>
    <input type="text" id="fname" name="fname"><br><br>

    <label for="reg">License Plate No:</label><br><br>
    <input type="text" id="reg" name="reg" ><br><br>

    <label for="topic">Phone number:</label><br><br>
    <input type="text" id="topic" name="Project Topic" ><br><br>

    <input type="submit" value="Submit">

    <input type="reset" value="Reset" >

</form>

</body>
</html>

```

Output:

Form

**Parking**

1. Enter your Name.  
2. License Plate Number.

[Check Fare Rates](#)

Name:

License Plate No:

Phone number:

Data Submitted and is available in debug window; {"fname": "Anila", "reg": "TN 04 M 1234", "Project Topic": "9876543210"}

Hyperlink

Parking Tariff for All stations		
Parking Time	Car	Bike
0-6hrs.	20	10
6-12hrs.	30	15
More than 12hrs.	40	20
Beyond service hours, (24:00 hrs to 05	100	50
Monthly parking pass for day	500	250
Monthly parking pass for 24hrs	2000	1000

Note : Above Tariff not applicable for Airport, Guindy, Shenoy nagar, Pachaiyapas, St. Thomas amount, Vadapalani)

Parking tariff for Airport station		
Parking Time	Car	Bike
10min-2hrs	75	15
	Rs.75 + Rs.10 per hour	Rs.15 + Rs.10 per hour
2hrs - 7hrs	Rs.350	Rs.100
Beyond 7 hrs-24hrs	Rs.700	Rs.200
Beyond 24 hrs	Rs.3500	Rs.500
Monthly parking pass for day		

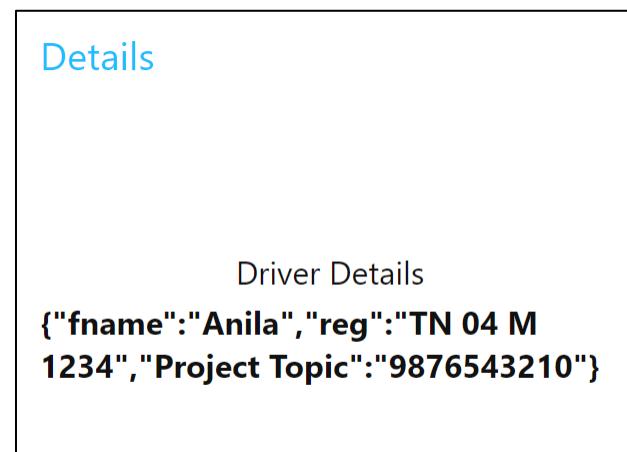
Debug Window

```

02/03/2021, 21:52:43  node: 6bb2b3e4.13575c
msg.payload : Object
▶ { fname: "Anila", reg: "TN 04 M 1234", Project
Topic: "9876543210" }

```

Dashboard



Inference: HTTP in node has been used to do GET and POST operations. The template node is used to write the HTML code for the form submission and provide hyperlinks. The input information is then posted in the debug and dashboard windows

Result: Thus, we have performed and observed a Web Based Application (HTML) Form Creation and Submission for a Project Title Submission and a Smart Parking System (Task) in Node-Red and have successfully obtained all the outputs for the same.

### Experiment 3

## Online Media Monitoring Using Node-Red | Feedparser Node

Aim: To monitor online news feeds and display headlines using Feedparser node in Node-Red

Theory:

RSS (RDF Site Summary or Really Simple Syndication)

- It refers to files easily read by a computer called XML files that automatically update information. An RSS feed takes the headlines, summaries, and update notices, and then links back to articles on the website's page.
- RSS is a web feed that allows users and applications to access updates to websites in a standardized, computer-readable format.

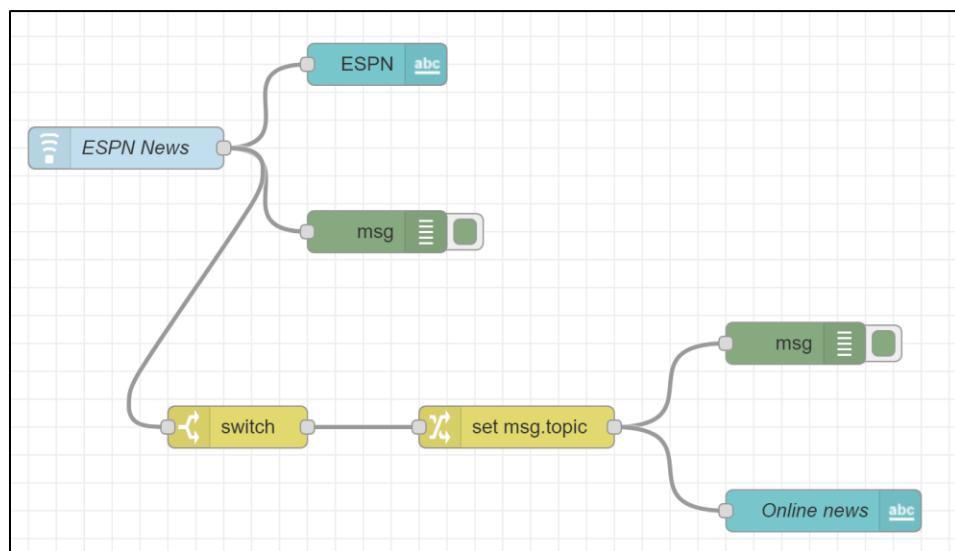
RSS Feed:

- In an RSS feed, these updates and notifications are gathered, organized and updated in real-time into one convenient dashboard.
- This information is fetched by a user's RSS feed reader that converts the files and the latest updates from websites into an easy-to-read format.

Procedure:

- Trigger node-red in cmd using node-red -v
- Connect the flow and configure their properties as shown
- Copy the RSS feed link to the Feedparser node to monitor news feeds
- Deploy onto dashboard and debug window

Flow:



## Properties:

**Edit feedparser node**

Delete Cancel Done

**Properties**

Feed url: <https://www.espn.com/espn/rss/news>

Refresh: 15 minutes

Name: ESPN News

**Edit text node**

Delete Cancel Done

**Properties**

Group: [Home] Details

Size: 6 x 6

Label: ESPN

Value format: {{msg.payload}}

Layout:

label value	label value	label value
label value	label value	

**Edit debug node**

Delete Cancel Done

**Properties**

Output: complete msg object

To:

- debug window
- system console
- node status (32 characters)

Name: Name

**Edit switch node**

Delete Cancel Done

**Properties**

Name: Name

Property: msg. article.title

Rules:

- matches regex: ^a\_z prince → 1  ignore case

**Edit change node**

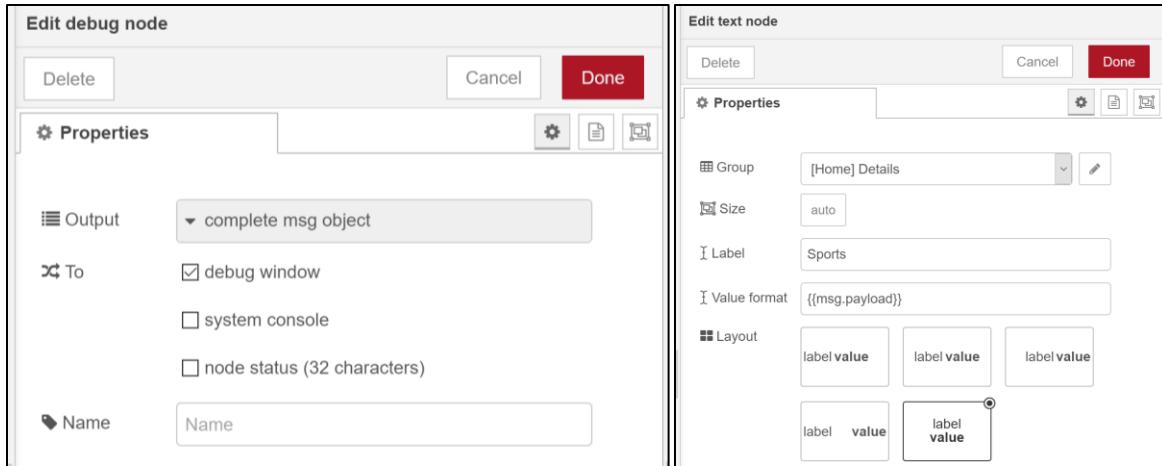
Delete Cancel Done

**Properties**

Name: Name

Rules:

- Set msg. topic to ^a\_z article title

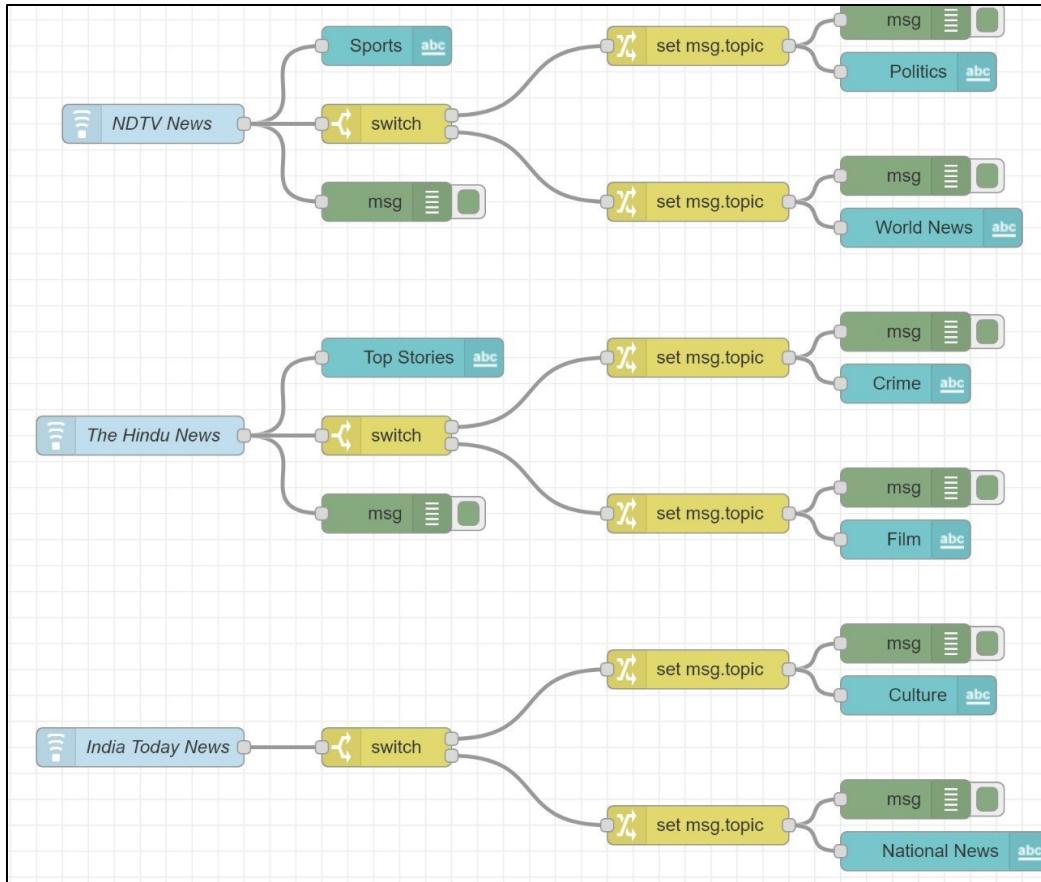


Output:

Dashboard	Debug window
<p><b>Details</b></p> <p>ESPN</p> <p><b>MLS expansion side Austin FC is continuing the league's trend of combing South America for young talent and skilled veterans.</b></p> <p>Sports</p> <p><b>The NIT is downsizing from 32 to 16 teams this season, and the entire tournament will be played in the Dallas-Fort Worth area.</b></p>	<p>02/03/2021, 02:30:44 node: 548e8310.a1f10c  <a href="https://www.espn.com/mens-college-basketball/story/_id/30984916/teddy-allen-nebraska-cornhuskers-leading-scorer-abruptly-leaves-program?device=featurephone">https://www.espn.com/mens-college-basketball/story/_id/30984916/teddy-allen-nebraska-cornhuskers-leading-scorer-abruptly-leaves-program?device=featurephone</a> : msg : Object</p> <pre>▶ { topic: "https://www.espn.com/mens-college-basketball/story/_id/30984916/teddy-allen-nebraska-cornhuskers-leading-scorer-abruptly-leaves-program?device=featurephone", payload: "Teddy Allen, Nebraska's leading scorer abruptly leaves program?", article: object, _msgid: "3433e1bd.9a6a1e" }</pre> <p>02/03/2021, 02:30:44 node: 548e8310.a1f10c  <a href="https://www.espn.com/womens-college-basketball/story/_id/30985241/texas-aggies-reach-best-ever-ranking-no-2-women-ap-poll-uconn-huskies-remain-no-1?device=featurephone">https://www.espn.com/womens-college-basketball/story/_id/30985241/texas-aggies-reach-best-ever-ranking-no-2-women-ap-poll-uconn-huskies-remain-no-1?device=featurephone</a> : msg : Object</p> <pre>▶ { topic: "https://www.espn.com/womens-college-basketball/story/_id/30985241/texas-aggies-reach-best-ever-ranking-no-2-women-ap-poll-uconn-huskies-remain-no-1?device=featurephone", payload: "Texas A&amp;M moved up to No. 2 in the AP poll.", article: object, _msgid: "ec1f8259.c057e" }</pre>

**Task:** Create a web API using template node and design a form to get reader details. On submitting reader details, make the reader read the article. Using feedparser node, create an online media monitoring system from 3 RSSFEEDS. Filter the news category with at least 3 verticals. Display the output in dashboard and debug monitor.

News Reader Flow:



## Properties:

**Edit feedparser node**

Delete	Cancel	Done
<b>Properties</b>		
Feed url	https://feeds.feedburner.com/ndtvnews-top-stories	
Refresh	15	minutes
Name	NDTV News	

**Edit switch node**

Delete	Cancel	Done
<b>Properties</b>		
Name	Name	
Property	msg. article.title	
<ul style="list-style-type: none"> <li>matches regex <math>a_z</math> Congress <math>\rightarrow 1</math></li> <li>matches regex <math>a_z</math> Kim jong <math>\rightarrow 2</math></li> </ul>		

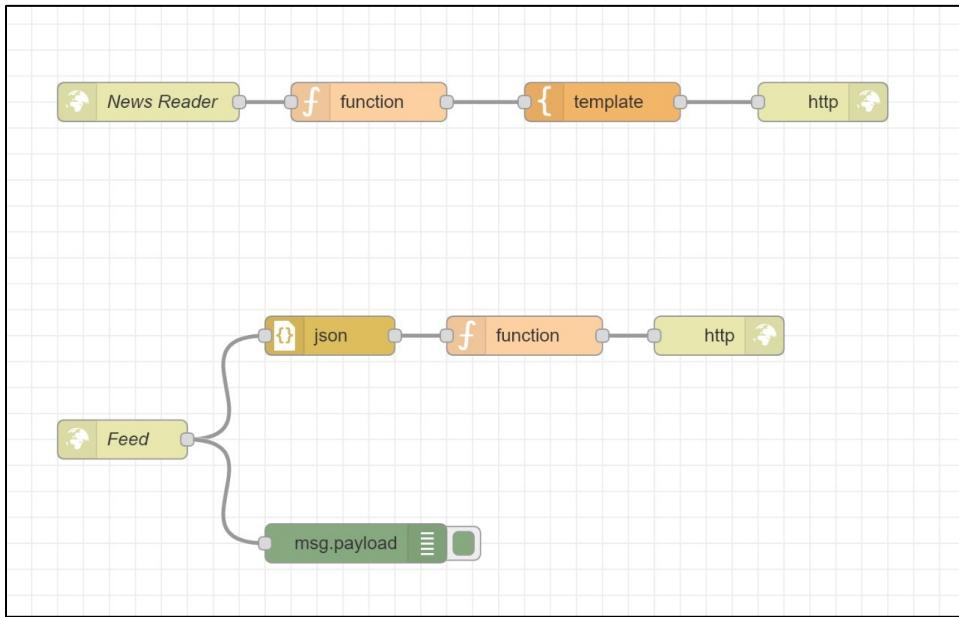
**Edit change node**

Delete	Cancel	Done
<b>Properties</b>		
Name	Name	
<b>Rules</b>		
Set	msg. topic	to $a_z$ article title

**Edit text node**

Delete	Cancel	Done
<b>Properties</b>		
Group	[Home] Details	
Size	6 x 3	
Label	Politics	
Value format	{{msg.payload}}	

## Form Flow:



## Properties:

<b>Edit http in node</b>	<b>Edit function node</b>						
<input type="button" value="Delete"/> <input type="button" value="Cancel"/> <input type="button" value="Done"/> <b>Properties</b> <b>Method</b> : GET <b>URL</b> : /News <b>Name</b> : News Reader	<input type="button" value="Delete"/> <input type="button" value="Cancel"/> <input type="button" value="Done"/> <b>Properties</b> <b>Name</b> : Name <table border="1" style="margin-top: 10px;"> <tr> <td>Setup</td> <td>Function</td> <td>Close</td> </tr> <tr> <td colspan="3"> <pre>1 msg.url="Feed"; 2 return msg;</pre> </td> </tr> </table>	Setup	Function	Close	<pre>1 msg.url="Feed"; 2 return msg;</pre>		
Setup	Function	Close					
<pre>1 msg.url="Feed"; 2 return msg;</pre>							
<b>Edit http in node</b>	<b>Edit function node</b>						
<input type="button" value="Delete"/> <input type="button" value="Cancel"/> <input type="button" value="Done"/> <b>Properties</b> <b>Method</b> : POST <input type="checkbox"/> Accept file uploads? <b>URL</b> : /Feed <b>Name</b> : Feed	<input type="button" value="Delete"/> <input type="button" value="Cancel"/> <input type="button" value="Done"/> <b>Properties</b> <b>Name</b> : Name <table border="1" style="margin-top: 10px;"> <tr> <td>Setup</td> <td>Function</td> <td>Close</td> </tr> <tr> <td colspan="3"> <pre>1 msg.payload = "Data Submited and is available in debug window; "+msg.payload; 2 3 return msg;</pre> </td> </tr> </table>	Setup	Function	Close	<pre>1 msg.payload = "Data Submited and is available in debug window; "+msg.payload; 2 3 return msg;</pre>		
Setup	Function	Close					
<pre>1 msg.payload = "Data Submited and is available in debug window; "+msg.payload; 2 3 return msg;</pre>							

## HTML code for template node:

```

<!DOCTYPE html>

<html>
  <head>
    <h1 style="background-color:Tomato;font-family:century gothic">News Reader</h1>
  
```

```

</head>
<body>
<p style="background-color:Tomato;">

<OL>
<LI> Enter your Name.
<LI> Enter Phone number and Email.
</OL>

<h4>
<a href="https://feeds.feedburner.com/ndtvsports-latest"> NDTV</a>
<a href="https://www.thehindu.com/business/Economy/feeder/default.rss"> The Hindu</a>
<a href="https://www.indiatoday.in/rss/1206504"> India Today</a>

</h4>

<form method="post" action="/{{url}}">

<label for="name">Name:</label><br>
<input type="text" id="fname" name="fname"><br><br>

<label for="reg">Phone no:</label><br>
<input type="text" id="reg" name="reg" ><br><br>

<label for="topic">Email:</label><br>
<input type="text" id="email" name="email" ><br><br>

<input type="submit" value="Submit">

<input type="reset" value="Reset" >

</form>

</body>
</html>

```

News Reader Output:

Dashboard:

NDTV	The Hindu	India Today
Sports <b>India vs England: Rishabh Pant's work behind the stumps in the 2nd Test against England earned him praise from one of the finest wicketkeepers in world cricket -- Adam Gilchrist.</b>	Top Stories <b>He had stolen a policeman's car after threatening him with a sword</b>	Culture <b>An offering of a saree made of gold, weighing over 2 kg, for goddess Yellamma at a temple in Hyderabad is among the many programmes planned for Telangana Chief Minister K Chandrashekhar Rao's...</b>
Politics <b>The Congress party today made a clean sweep of Punjab's seven municipal corporations in the state local body polls. The party bagged all the bodies for which the results were declared today: Moga...</b>	Crime <b>The 58-year-old ex-serviceman has been arrested</b>	National News <b>Journalist Priya Ramani was acquitted by a Delhi court Wednesday afternoon over a defamation case filed by former Union Minister MJ Akbar, whom she had accused of sexual misconduct in 2018. A year...</b>
World News <b>The wife of North Korean leader Kim Jong Un made her first public appearance in a year, ending an unusual absence that stoked speculation about her condition.</b>	Film <b>In the northeastern city of Barcelona, thousands of protestors set trash cans on fire and threw rocks at the police.</b>	

### Debug Window:

17/02/2021, 18:03:54 node: dcf0c9c.a3f85b8 <a href="https://www.thehindu.com/news/national/congress-urges-speaker-not-to-bypass-rajya-sabha-by-declaring-7-key-bills-as-money-bills/article33861373.ece">https://www.thehindu.com/news/national/congress-urges-speaker-not-to-bypass-rajya-sabha-by-declaring-7-key-bills-as-money-bills/article33861373.ece</a> : msg : Object ► { topic: "https://www.thehindu.com/news/", payload: "Budget session of Parliament i...", article: object, _msgid: "8521e112.477a6" }	17/02/2021, 18:03:54 node: dcf0c9c.a3f85b8 <a href="https://www.thehindu.com/news/national/metoo-women-lawyers-activists-hail-historic-judgement-acquitting-priya-ramani/article33861358.ece">https://www.thehindu.com/news/national/metoo-women-lawyers-activists-hail-historic-judgement-acquitting-priya-ramani/article33861358.ece</a> : msg : Object ► { topic: "https://www.thehindu.com/news/", payload: "'What a resounding victory for...', article: object, _msgid: "d41851aa.484bc" }
17/02/2021, 18:03:54 node: dcf0c9c.a3f85b8 <a href="https://www.thehindu.com/news/national/rail-roko-railways-deploys-20-additional-rpsf-companies-focus-on-punjab-haryana-up/article33861362.ece">https://www.thehindu.com/news/national/rail-roko-railways-deploys-20-additional-rpsf-companies-focus-on-punjab-haryana-up/article33861362.ece</a> : msg : Object ► { topic: "https://www.thehindu.com/news/", payload: "The SKM had said that the rail...", article: object, _msgid: "84378219.fa921" }	17/02/2021, 18:03:54 node: dcf0c9c.a3f85b8 <a href="https://www.thehindu.com/news/national/tamil-nadu/man-shoots-son-kills-him-in-vellore/article33861277.ece">https://www.thehindu.com/news/national/tamil-nadu/man-shoots-son-kills-him-in-vellore/article33861277.ece</a> : msg : Object ► { topic: "https://www.thehindu.com/news/", payload: "The 58-year-old ex-serviceman ...", article: object, _msgid: "d17cc302.d4f46" }

### News Form:

**News Reader**

1. Enter your Name.  
2. Enter Phone number and Email.

**NDTV The Hindu India Today**

Name:

Phone no:

Email:

127.0.0.1:1880/Feed

Data Submitted and is available in debug window: {"fname": "Anila", "reg": "9876543210", "email": "anila123@gmail.com"}

Hyperlinks:

**NDTV News - Topstories** 

syndicated content powered by FeedBurner

FeedBurner makes it easy to receive content updates in My Yahoo!, Newsgator, Bloglines, and other news readers.

[Learn more about syndication and FeedBurner...](#)

**Subscribe Now!**

...with web-based news readers. Click your choice below:

[MY YAHOO!](#) [feedly](#) [netvibes](#) [SubToMe](#)

...with other readers:

(Choose Your Reader)

**Current Feed Content**

-  [29 Lakh Registered On Day 1 Of Public Rollout Of Vaccines: Harsh Vardhan](#)

Posted: March 2, 2021 01:13 AM  
Over 29 lakh people had registered - online or via the Aarogya Setu app - by 8.30 pm on Monday, the first day of the public rollout of the coronavirus vaccines, Union Health Minister Dr Harsh Vardhan...

Inference: Feedparser node is used to get RSS feeds of news websites like ESPN, NDTV, the Hindu, etc. The results are displayed in the dashboard using text node, and in the debug window using debug node.

Results: Hence the given tasks to monitor online news feeds using feedparser node have been carried out in node-red.

#### Experiment 4

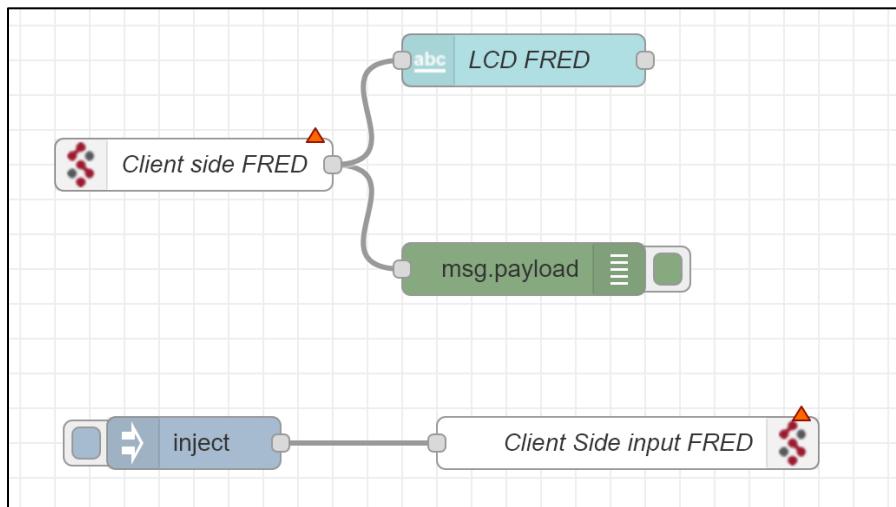
## Front End for Node-RED (FRED) a cloud-based Node-RED

Aim: To use FRED, a cloud-based node-red to communicate between client and admin side.

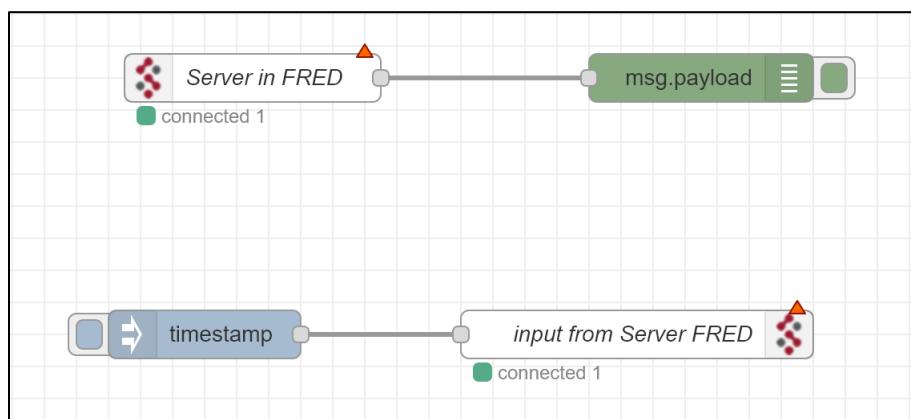
Procedure:

- Trigger node red in cmd using node-red -v
- Configure the flow as shown according to the properties on node-red and in FRED
- Deploy to see results in debug window

Node-red Flow:



FRED flow:



Properties:

Client side/Node-red:

**Edit fred in node**

**Properties**

- Endpoint: MIoT
- Name: Client side FRED

**Edit text input node**

**Properties**

- Group: [Home] Details
- Size: auto
- Label:
- Tooltip: optional tooltip
- Mode: text input, Delay (ms): 300

**Edit inject node**

**Properties**

- Name: Name
- msg. payload = Successfully connected with FRED

**Edit fred out node**

**Properties**

- Endpoint: MIoT
- Name: Client Side input FRED

### Server side/FRED:

**Edit fred in node**

**Properties**

- Endpoint: MIoT
- Name: Server in FRED

**Edit debug node**

**Properties**

- Output: msg. payload
- To: debug window

**Edit inject node**

**Properties**

- Name: Name
- msg. payload = Successfully connected with client

**Edit fred out node**

**Properties**

- Endpoint: MIoT
- Name: Name

### Client-side output:

```
02/03/2021, 03:46:07 node: 63fa7ce7.3c1b24
msg.payload : string[32]
"Successfully connected with FRED"
```

### Server-side Output:

Debug Window

```
24/02/2021, 16:37:39 node: c2f3b0cd.f976b8  
msg.payload : string[24]  
"I am server nodered FRED"
```

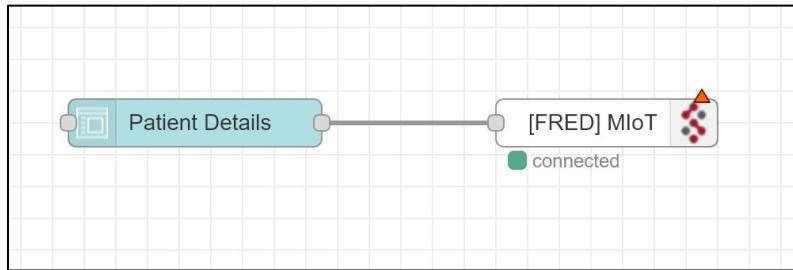
### Dashboard

LCD FRED

I am server nodered FRED

**Task:** Create a hospital-patient service using cloud-based FRED and node-red to patient details like name, temperature, heart-rate, etc. and suggest appropriate treatment.

Client/Patient Side:



Dashboard (form Input):

**Details**

**Patient Details**

Name \*  
Anila

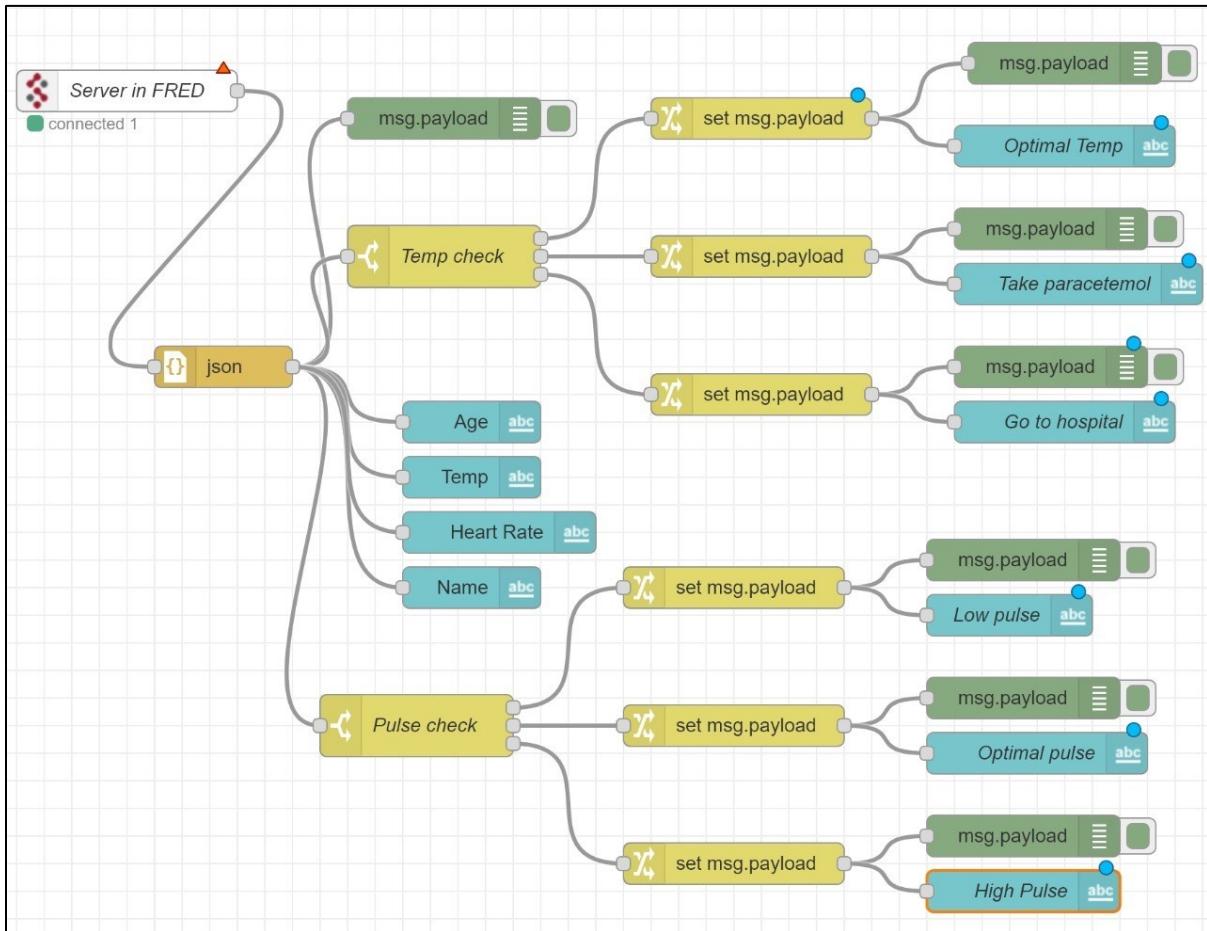
Age \*  
21

Heart Rate \*  
78

Temperature \*  
101

**SUBMIT**    **CANCEL**

Server/Hospital Admin side:



## Properties:

**Edit text node**

**Properties**

- Group: [Home] Patient Details
- Size: auto
- Label: Name
- Value format: {{msg.payload.name}}
- Layout: label value (with three columns)

**Edit switch node**

**Properties**

- Name: Temp check
- Property: msg.payload.temp

Rules (3 steps):

- < 99 → 1
- is between 99 and 101 → 2
- > 101 → 3

**Edit switch node**

**Properties**

- Name: Pulse check
- Property: msg.payload.pulse

Rules (3 steps):

- < 62 → 1
- is between 62 and 90 → 2
- > 90 → 3

**Edit change node**

**Properties**

- Name: Name

**Rules**

Set msg.payload to Take paracetemol

## Output:

Debug Window

```

24/02/2021, 18:04:16 node:
f12d9241.6d32c8
msg.payload : Object
▶ { name: "Anila", age: 21,
pulse: 78, temp: 101 }

24/02/2021, 18:04:17 node:
f0af3763.2149a
msg.payload : string[16]
"Take paracetemol"

24/02/2021, 18:04:18 node:
11868167.b364e7
msg.payload : string[27]
"Optimal pulse, Stay
Healthy"

```

## Dashboard

Temperature	Pulse	Patient Details	
Take paracetemol	Optimal pulse, Stay Healthy	Age	21
		Heart Rate	78
		Temp	101
		Name	Anila

Inference: FRED in node is used to get input from the other side via cloud and FRED out is used to send data. In the task, form node (form in dashboard) is used to get patient details on the client side. This info is sent in json format to the server/hospital admin side, where appropriate treatment is suggested based on the symptoms using switch and change nodes and displayed in the debug and dashboard windows.

Result: Hence, cloud-based FRED and node-red have been used to perform server-client operations and display outputs.

## Experiment 5

# Weather Report Analysis using OpenWeatherMap node in Node-Red and FRED

Aim: To create a weather report analysis system using OpenWeatherMap node in Node-Red and FRED

Theory:

OpenWeatherMap Nodes – Two nodes that get the weather report from OpenWeatherMap.

- o Input Node
- o Query node

Input Node:

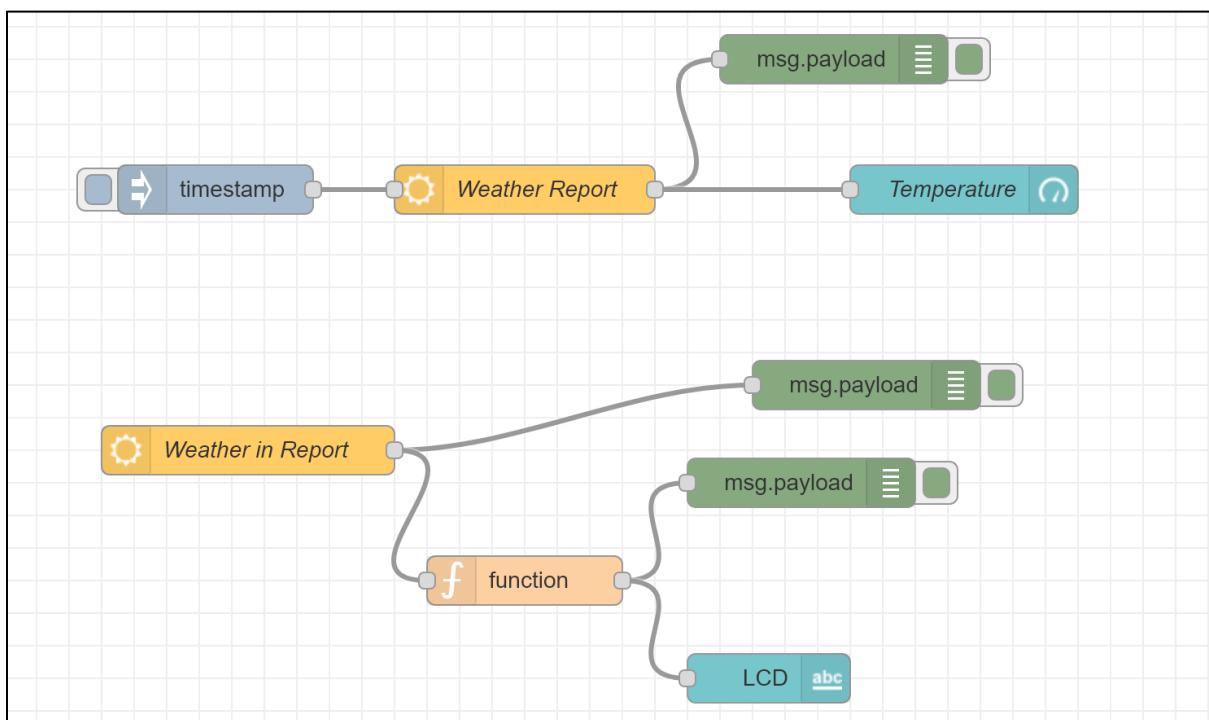
- Fetches the current weather or day forecast at a location specified by city and country or latitude and longitude every 10 minutes.
- Outputs a message if something has changed.

Query Node: Accepts an input to trigger fetching the current weather either from a specific city and country or latitude and longitude or passed in on.

Procedure:

- Trigger node red in cmd using node-red -v
- Configure the flow as shown according to the properties on node-red and in FRED
- Deploy to see results in debug window

Flow:



Properties:

The image displays four separate configuration panels from a node editor, likely for a platform like Node-RED:

- Edit openweathermap node**: A configuration panel for an "openweathermap" node. It includes fields for API Key (redacted), Language (English), Current weather for (City, Country), Location (City: Chennai, Country: India), and Name (Weather Report).
- Edit gauge node**: A configuration panel for a "gauge" node. It includes fields for Group ([Weather] Weather), Size (auto), Type (Gauge), Label (Temperature in Celcius), Value format ({{msg.payload.tempc}}), and Units (units).
- Edit openweathermap in node**: A configuration panel for an "openweathermap" node, similar to the first one but with a different name (Weather in Report).
- Edit function node**: A configuration panel for a "function" node. It includes a setup tab with a code editor containing the following Node.js code:

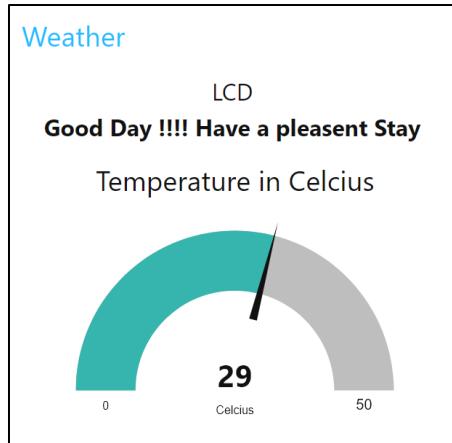
```

1 if(msg.payload.weather == "Clear")
2 {
3   msg.payload = "Good Day !!!! Have a pleasent Stay"
4   return msg;
5 }
6 else if(msg.payload.weather == "Rain")
7 {
8   msg.payload = "Stay Home!!Stay Safe!! It is Raining"
9   return msg;
10}
11
12 return null;

```
- Edit text node**: A configuration panel for a "text" node. It includes fields for Group ([Weather] Weather), Size (auto), Label (LCD), Value format ({{msg.payload}}), and Layout (with options for label and value).

**Output:**

Dashboard



Debug Window

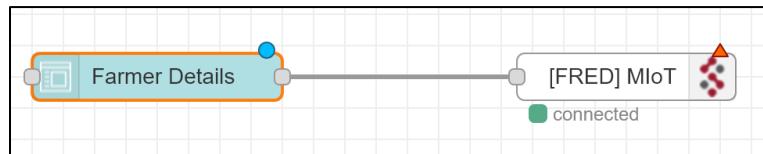
```
03/03/2021, 16:32:48 node: 7184ff9.1e42c8
msg.payload : Object
  ▼ object
    id: 801
    weather: "Clouds"
    detail: "few clouds"
    icon: "02d"
    tempk: 304.15
    tempc: 31
    temp_maxc: 31
    temp_minc: 31
    humidity: 35
    pressure: 1011
    maxtemp: 304.15
    mintemp: 304.15
    windspeed: 3.6
    winddirection: 160
    location: "Chennai"
    sunrise: 1614732838
    sunset: 1614775684
    clouds: 20
    description: "The weather in Chennai at
coordinates: 13.0878, 80.2785 is Clouds (few
clouds)."
03/03/2021, 16:32:48 node: 4324cb7b.5612ec
msg.payload : string[34]
"Good Day !!!! Have a pleasant Stay"
```

**Task:** Design a cloud-based weather analysis and forecasting automation through FRED. Retrieve details from <https://openweathermap.org>, perform analysis which helps the farmer for their agriculture and send appropriate forecast details

Client Device: Get farmer details in local Node-Red and send to the server

Server Device: in FRED, display all weather details and produce a visual representation of all parameters retrieved from OpenWeatherMap which may help farmers visualize the current state of weather. Enable audio output so that farmers with poor eyesight can hear the details. Display the output in UI and Debug window

Client-side Flow:



Dashboard Input

**Weather**

**Farmer Details**

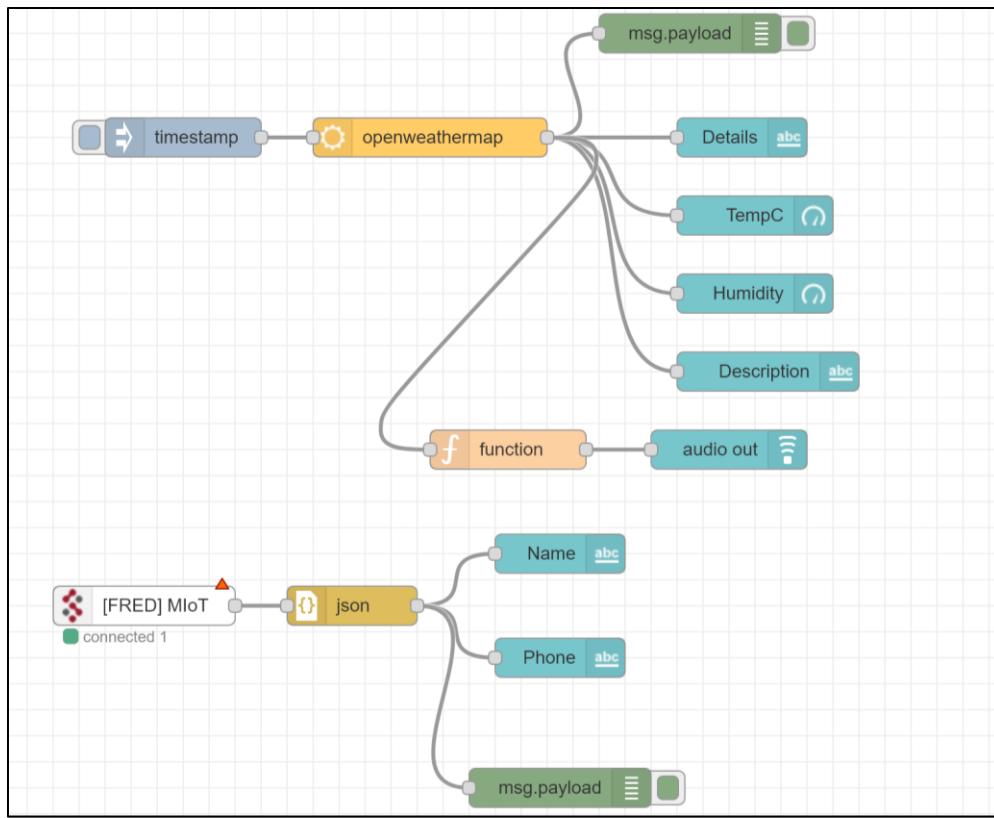
Name \*  
Bob

Phone \*  
9876543210

SUBMIT CANCEL

The form contains fields for Name and Phone number, both marked with asterisks indicating they are required. There are 'SUBMIT' and 'CANCEL' buttons at the bottom.

Server-Side Flow:



Function for audio:

```

msg.payload="Temperature: " + msg.payload.tempc + "Humidity: " + msg.payload.humidity +
"Description: " + msg.payload.description
return msg;
  
```

Output:

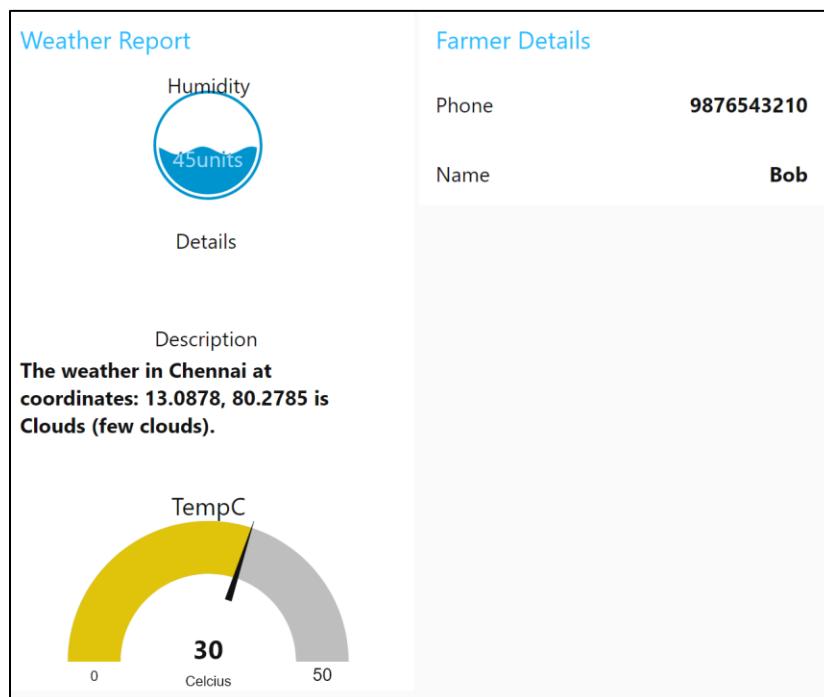
Debug Window

```

03/03/2021, 17:03:55 node: 9e41be94.620cc8
msg.payload : Object
  ▼ object
    name: "Bob"
    phone: 9876543210
03/03/2021, 17:07:31 node: ce94cb92.f3b298
msg.payload : Object
  ▼ object
    weather: "Clouds"
    detail: "few clouds"
    tempk: 303.15
    tempc: 30
    temp_maxc: 30
    temp_minc: 30
    humidity: 45
    maxtemp: 303.15
    mintemp: 303.15
    windspeed: 4.12
    winddirection: 120
    location: "Chennai"
    sunrise: 1614732838
    sunset: 1614775684
    clouds: 20
    description: "The weather in Chennai at
coordinates: 13.0878, 80.2785 is Clouds (few
clouds)."

```

## Dashboard



Inference: The openweathermap node is used to retrieve weather details of a city like temperature, humidity, etc. from <https://openweathermap.org>. These details are then displayed in the dashboard using text and gauge nodes. In the task, farmer details are retrieved using the form node through dashboard and sent to the server side through FRED out node. In the server side, weather details are retrieved from the website and displayed along with the farmer details in the debug window and dashboard. Audio output is also enabled using the audio output node.

Results: Hence, OpenWeatherMap node has been successfully used to perform weather analysis in Node-Red and FRED.

## Experiment 6

# ThingsBoard – NodeRed Posting Data in Dashboard through HTTP

Aim: To use ThingsBoard to post device data through HTTP

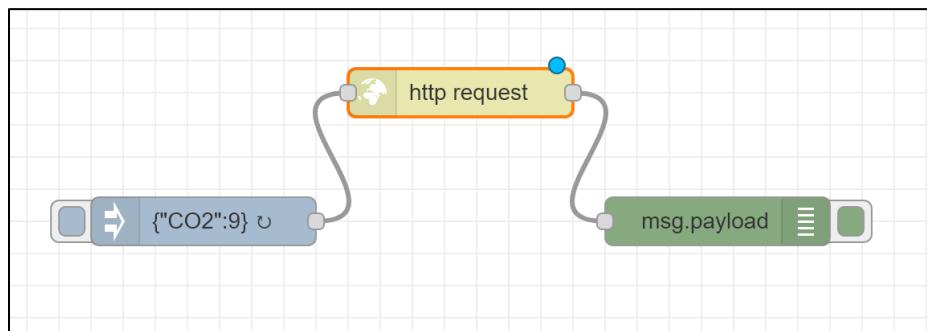
Theory: ThingsBoard

- Open-source IoT Platform
- Device management, data collection, processing and visualization for IoT solutions.
- It enables device connectivity via industry standard
- IoT protocols - MQTT, COAP and HTTP and supports both cloud and on-premises deployments.
- ThingsBoard combines scalability, fault-tolerance and performance so we will never lose our data.

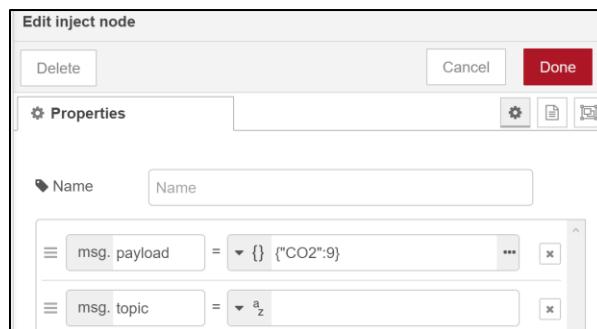
Procedure:

- Trigger node red in cmd using node-red -v
- Configure the flow as shown according to the properties on node-red and in FRED
- Deploy to see results in debug window

Flow:



Properties:



**Edit http request node**

Delete      Cancel      Done

**Properties**

Method: POST

URL: <https://demo.thingsboard.io/api/v1/VyjT8oJU04Vu>

Enable secure (SSL/TLS) connection

Use authentication

Enable connection keep-alive

Use proxy

Return: a UTF-8 string

Name: Name

17/03/2021, 16:31:45 node: 8cf947f5.c6e71  
msg.payload : string[0]  
""

17/03/2021, 16:31:48 node: 8cf947f5.c6e71  
msg.payload : string[0]  
""

17/03/2021, 16:31:51 node: 8cf947f5.c6e71  
msg.payload : string[0]  
""

17/03/2021, 16:31:54 node: 8cf947f5.c6e71  
msg.payload : string[0]  
""

17/03/2021, 16:31:57 node: 8cf947f5.c6e71  
msg.payload : string[0]  
""

Key: VyjT8oJU04Vu1BGNToh

Output:

**Debug Window**

17/03/2021, 16:59:19 node: a1e7e0b4.ecf758  
msg.payload : string[0]  
""

17/03/2021, 16:59:20 node: a1e7e0b4.ecf758  
msg.payload : string[0]  
""

ThingsBoard

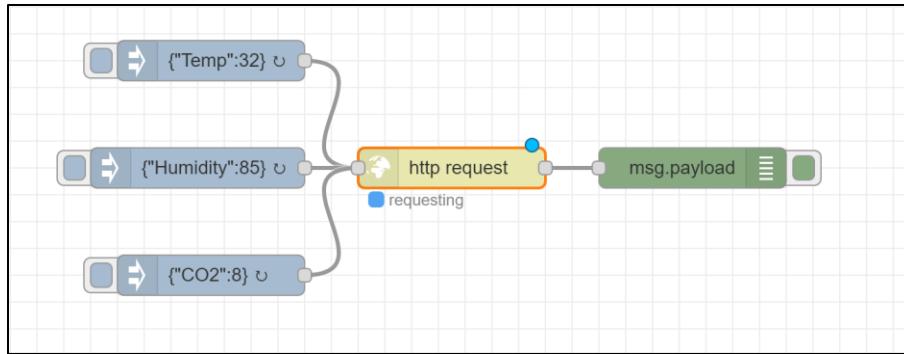
The screenshots show the ThingsBoard IoT Car device details interface. The top screenshot displays the 'Latest telemetry' tab, which lists the most recent data update time and value. The bottom screenshot displays the 'Cards' tab, which contains a single card showing the current CO2 level and temperature.

ThingsBoard Dashboard

The screenshot shows the ThingsBoard dashboard with the 'IoT Car' board selected. The board displays a single orange card with the text 'CO2' and '9 °C'.

**Task:** Using node-red create a flow which performs automatic plant health monitoring. Using appropriate sensors simulate various parameters like air temperature, humidity, CO2, etc. Display the visualization in ThingsBoard Dashboard Widgets.

Flow:



### Debug

17/03/2021, 16:59:17	node: a1e7e0b4.ecf758	<input type="button" value="▼"/>
msg.payload : string[0]		
""		
17/03/2021, 16:59:18	node: a1e7e0b4.ecf758	
msg.payload : string[0]		
""		
17/03/2021, 16:59:19	node: a1e7e0b4.ecf758	
msg.payload : string[0]		
""		
17/03/2021, 16:59:20	node: a1e7e0b4.ecf758	
msg.payload : string[0]		
""		

Key for Task: rX9ZwBM3pLECgUxafSxI

Output:

ThingsBoard

Plant Health

Device details

?

X

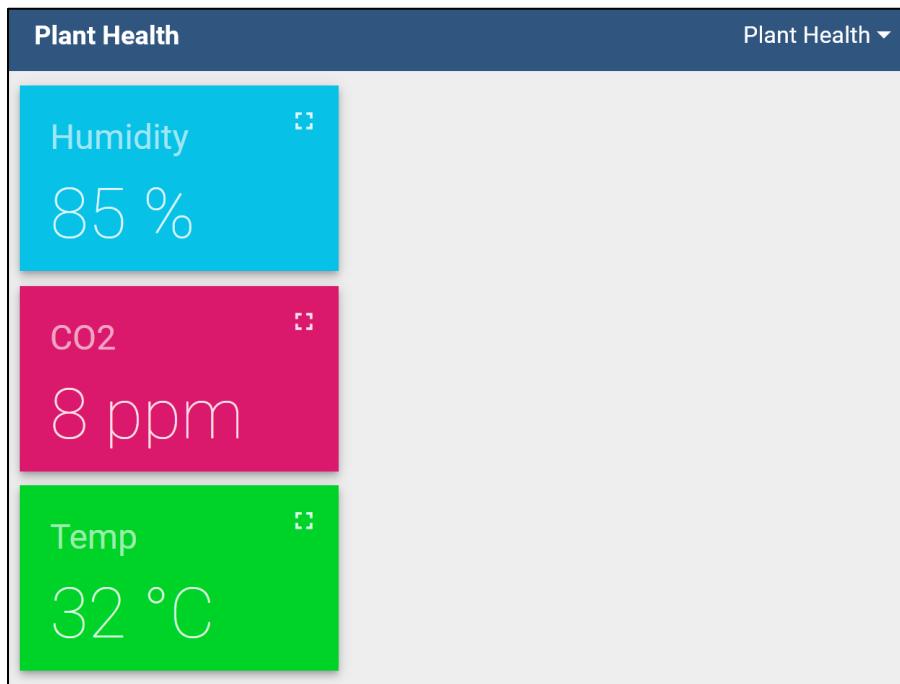
Details Attributes Latest telemetry Alarms Events Relations

Latest telemetry

Last update time Key Value

Last update time	Key	Value
2021-03-17 16:54:39	CO2	8
2021-03-17 16:54:39	Humidity	85
2021-03-17 16:54:39	Temp	32

ThingsBoard Dashboard



Inference: ThingsBoard is used to monitor sensor values. The sensor values are provided through the inject node. Then a device profile is created in ThingsBoard and the sensor values are displayed through widgets.

Results: Hence sensor data monitoring through http has been done in Node-Red and ThingsBoard and displayed in ThingsBoard Dashboard

## Experiment 7

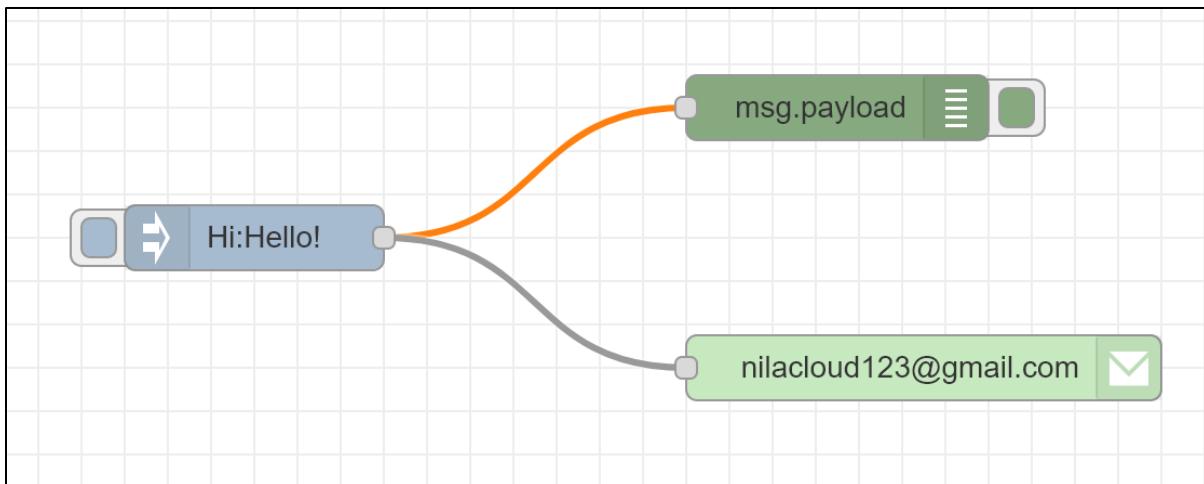
# Email using Node-Red

Aim: To send emails via email node in Node-Red

Procedure:

- Trigger Node-Red in cmd using node-red -v
- Open Node-Red and arrange flow as shown and configure properties
- Enter email and password. Then run

Flow:



Properties:

The image shows two configuration dialogs side-by-side.

**Edit inject node** (Left):

- Properties tab selected.
- Name: "Name" (input field).
- msg. payload: "msg. payload" (input field) = "Hello!" (output field).
- msg. topic: "msg. topic" (input field) = "Hi" (output field).

**Edit debug node** (Right):

- Properties tab selected.
- Output: "msg. payload" (selected dropdown).
- To:
  - debug window
  - system console
  - node status (32 characters)
- Name: "Name" (input field).

Edit email node

Delete Cancel Done

**Properties**

To: nilacloud123@gmail.com

Server: smtp.gmail.com

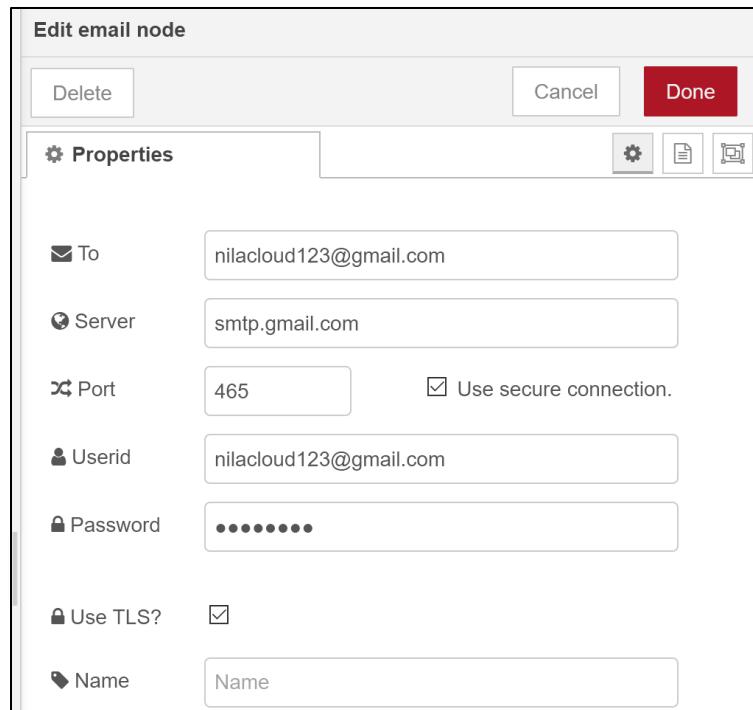
Port: 465       Use secure connection.

Userid: nilacloud123@gmail.com

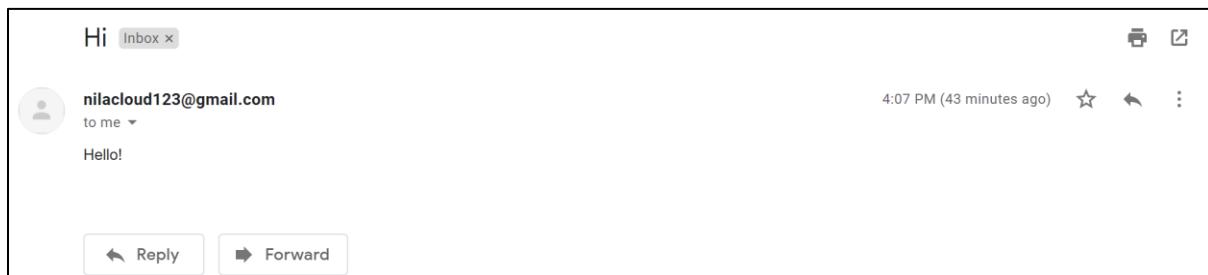
Password: \*\*\*\*\*

Use TLS?

Name: Name

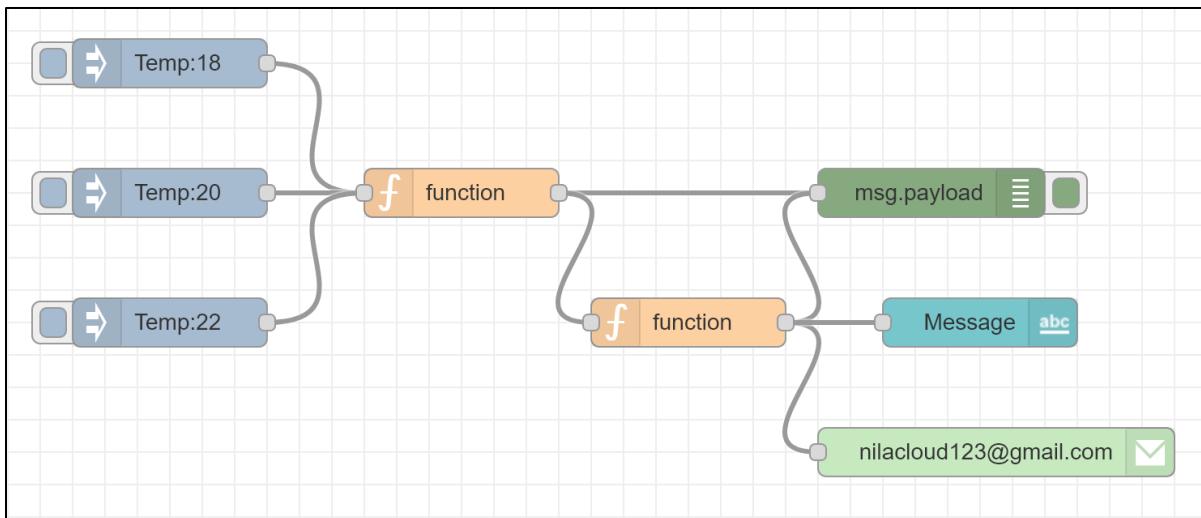


Output:

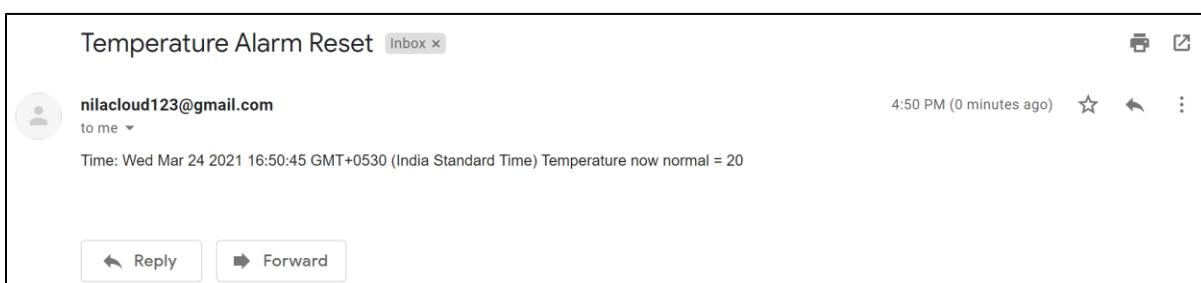
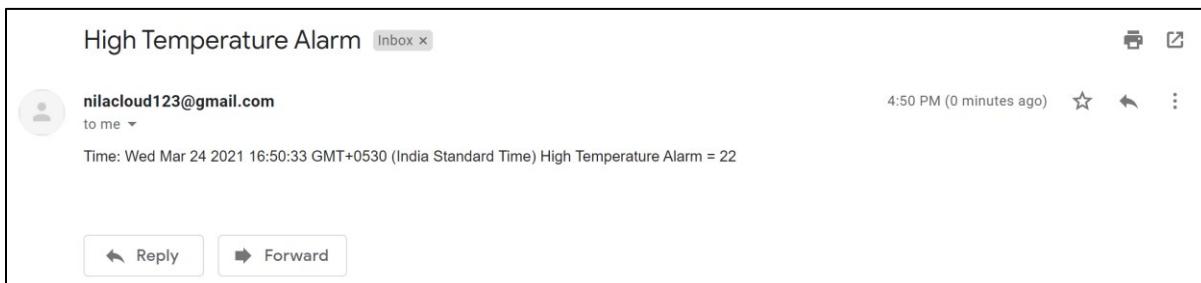


Task 1:

Flow:



Properties:



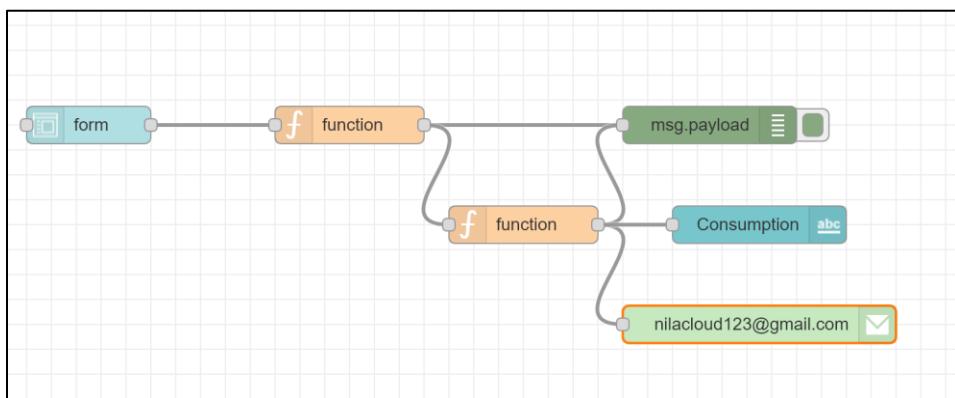
Output:

**Temp Monitor**  
Message  
**Time: Wed Mar 24 2021 16:53:00**  
**GMT+0530 (India Standard Time)**  
**High Temperature Alarm = 22**

**Temp Monitor**  
Message  
**Time: Wed Mar 24 2021 16:50:45**  
**GMT+0530 (India Standard Time)**  
**Temperature now normal = 20**

Task 2:

Flow:



Properties:

```

1 var hr=msg.payload.hr;
2 var sp=msg.payload.sp;
3 var bp=msg.payload.bp;
4 var cons;
5
6 var alarm_flag=context.get("alarm_flag");
7 if(typeof alarm_flag=="undefined")
8 alarm_flag=false;
9
10 if(hr<60 && sp<80 && bp<90)
11 {
12     msg.payload.cons="Low";
13 }
14 else if (hr>90 && sp>90 && bp>120)
15 {
16     msg.payload.cons="High";
17 }
18 else
19 {
20     msg.payload.cons="Medium";
21 }
22 return msg;

```

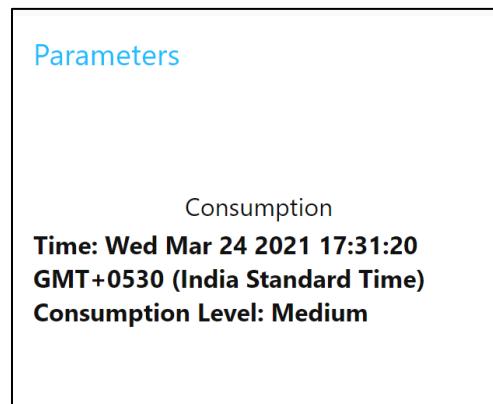
  

```

1 var temp=msg.payload;
2 msg.to="nilacloud123@gmail.com";
3 msg.from="nilacloud123@gmail.com";
4
5 var d=new Date();
6 var message="";
7 if(msg.payload.cons=="High")
8 {
9     msg.topic="Drug Consumption Level Reg";
10    message=" Consumption Level: High ";
11 }
12 else if (msg.payload.cons=="Low")
13 {
14     msg.topic="Drug Consumption Level Reg";
15    message=" Consumption Level: Low ";
16 }
17 else
18 {
19     msg.topic="Drug Consumption Level Reg";
20    message=" Consumption Level: Medium ";
21 }
22 msg.payload="Time: "+d+message+msg.payload;
23 return msg;

```

Output:



Output with Form Node in Dashboard:

Consumption Levels

Consumption

**Time: Wed Mar 24 2021 17:51:38  
GMT+0530 (India Standard Time)  
Consumption Level: High**

Parameters

HR *	100
SpO2 *	100
BP *	130

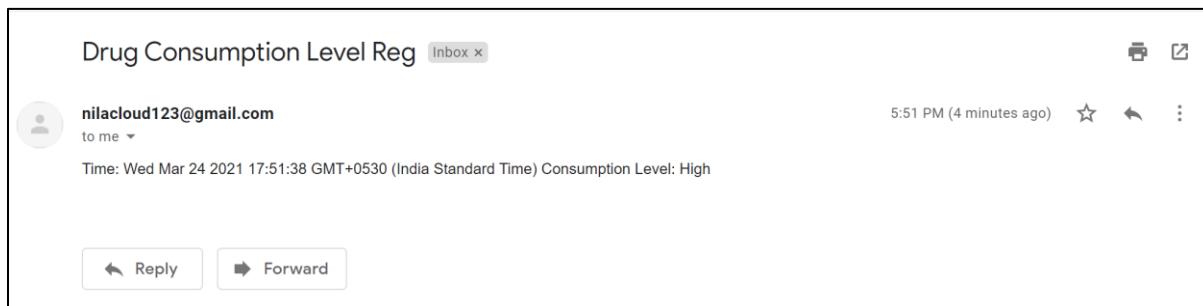
**SUBMIT**    **CANCEL**

Debug Window:

```
24/03/2021, 17:51:38 node: 2cf2f8d1.241748
msg.payload : Object
▶ { hr: 100, sp: 100, bp: 130, cons:
"High" }

24/03/2021, 17:51:39 node: 2cf2f8d1.241748
Drug Consumption Level Reg : msg.payload : string[86]
"Time: Wed Mar 24 2021 17:51:38
GMT+0530 (India Standard Time)
Consumption Level: High "
```

Email:



Inference: The email node was configured in node-red to send emails to the required email address. The emails sent were according to the constraints set by the function node.

Result: Hence, sending emails via Node-red was successfully executed

## Experiment 8

# Telemedicine using Node-Red and FRED

Aim: To collect the patient's symptoms for COVID-19 and display the doctor's decision in dashboard and debug window of node-red.

Task: The Patients who have corona will have the following symptoms

Most common symptoms:

- Fever
- dry cough
- tiredness

Less common symptoms:

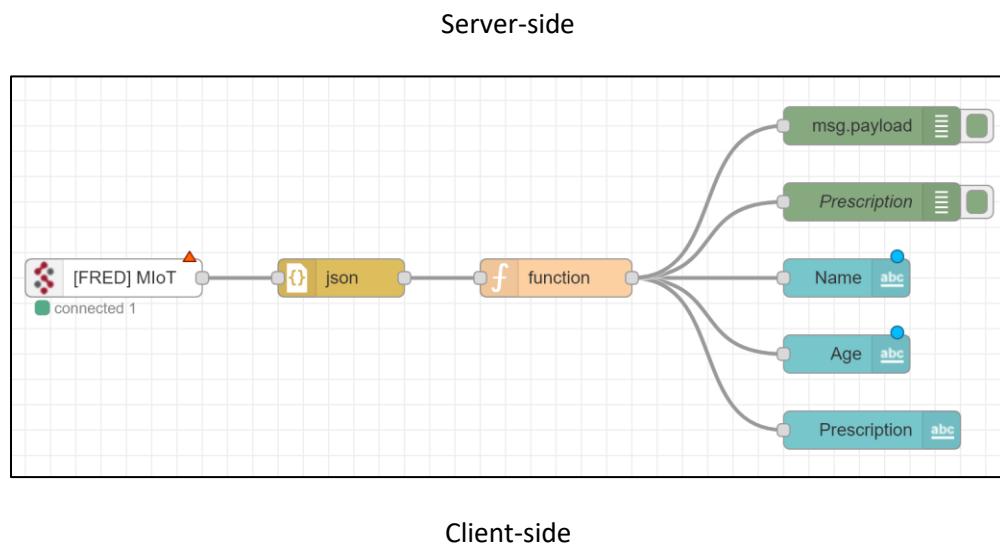
- headache
- aches and pains
- sore throat
- diarrhoea
- conjunctivitis
- loss of taste or smell
- a rash on skin, or discolouration of fingers or toes

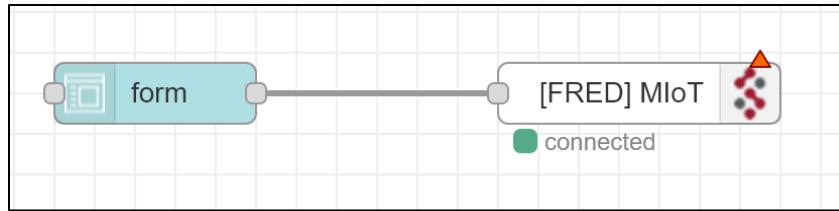
Serious symptoms:

- difficulty breathing or shortness of breath
- chest pain or pressure
- loss of speech or movement

Assume the patient is seeking for telemedicine, For the serious symptoms doctor may suggest the patient to be admitted in the hospital, For the other symptoms doctor may prescribe medicine. Display the doctor's decision in dashboard and debug window of Node red.

Flow:





Properties:

### Client-side

Label	Name	Type	Required	Rows	Remove
Name	name	Text	<input checked="" type="checkbox"/>		
Age	age	Number	<input checked="" type="checkbox"/>		
Breathing Difficult	breath	Checkbox	<input checked="" type="checkbox"/>		
Chest pain/press	chest	Checkbox	<input checked="" type="checkbox"/>		
Loss of speech/n	loss	Checkbox	<input checked="" type="checkbox"/>		

Label	Name	Type	Required	Rows	Remove
Dry Cough/Sore	dough	Checkbox	<input checked="" type="checkbox"/>		
Rash/Skin discolor	skin	Checkbox	<input checked="" type="checkbox"/>		
Loss of taste/sm	taste	Checkbox	<input checked="" type="checkbox"/>		
Conjunctivitis	conj	Checkbox	<input checked="" type="checkbox"/>		
Diarrhoea	diar	Checkbox	<input checked="" type="checkbox"/>		

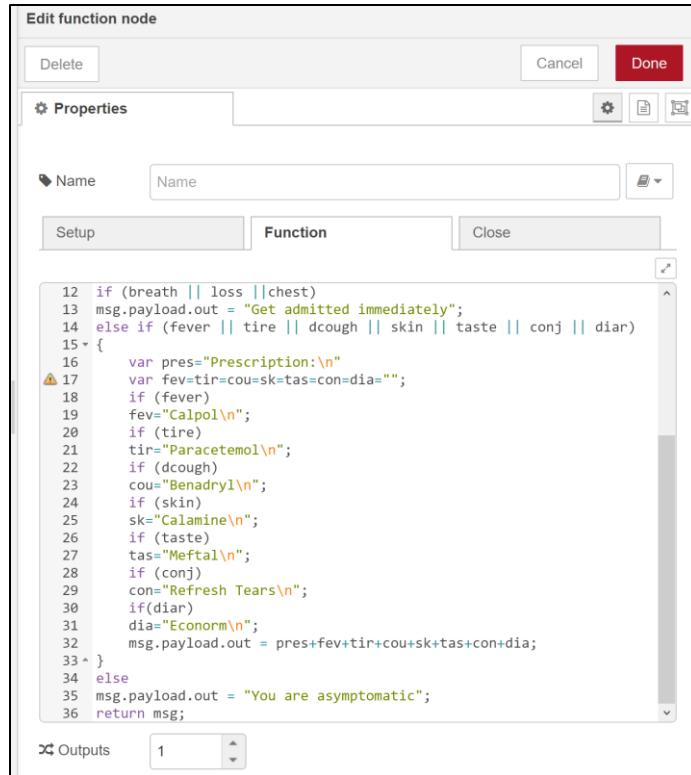
**Edit fred out node**

Properties	
Endpoint	MIoT
Name	Name

### Server-side

**Edit fred in node**

Properties	
Endpoint	MIoT
Name	Name



Code:

```

var breath=msg.payload.breath;
var loss=msg.payload.loss;
var chest=msg.payload.chest;
var fever=msg.payload.fever;
var tire=msg.payload.tire;
var dcough=msg.payload.dcough;
var skin=msg.payload.skin;
var taste=msg.payload.taste;
var conj=msg.payload.conj;
var diar=msg.payload.diar;
var out;
if (breath || loss || chest)
msg.payload.out = "Get admitted immediately";
else if (fever || tire || dcough || skin || taste || conj || diar)
{
    var fev=tir=cou=sk=tas=con=dia="";
    if (fever)
        fev="Calpol\n";
    if (tire)
        tir="Paracetemol\n";
    if (dcough)
        cou="Benadryl\n";
    if (skin)

```

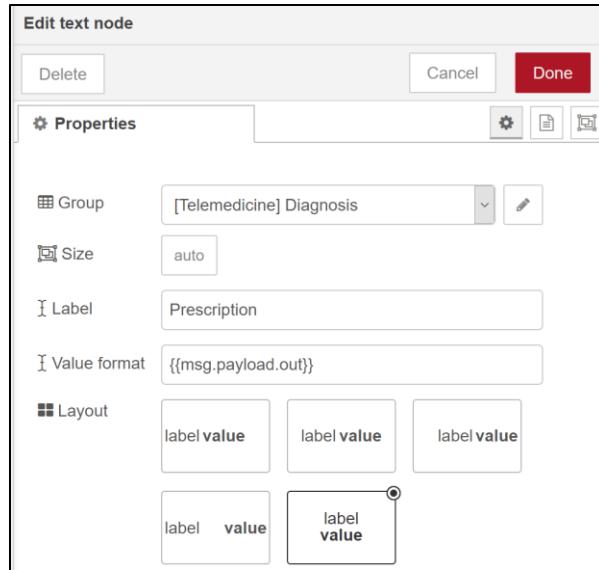
```

sk="Calamine\n";
if (taste)
tas="Meftal\n";
if (conj)
con="Refresh Tears\n";
if(diar)
dia="Econorm\n";
msg.payload.out = fev+tir+cou+sk+tas+con+dia;
}
else
msg.payload.out = "You are asymptomatic / Do not have COVID";
return msg;

```

<div style="border: 1px solid #ccc; padding: 10px;"> <p><b>Edit debug node</b></p> <p><b>Properties</b></p> <p>Output: msg. payload.out</p> <p>To:</p> <p><input checked="" type="checkbox"/> debug window</p> <p><input type="checkbox"/> system console</p> <p><input type="checkbox"/> node status (32 characters)</p> <p>Name: Prescription</p> </div>	<div style="border: 1px solid #ccc; padding: 10px;"> <p><b>Edit debug node</b></p> <p><b>Properties</b></p> <p>Output: msg. payload</p> <p>To:</p> <p><input checked="" type="checkbox"/> debug window</p> <p><input type="checkbox"/> system console</p> <p><input type="checkbox"/> node status (32 characters)</p> <p>Name: Name</p> </div>
--	--

<div style="border: 1px solid #ccc; padding: 10px;"> <p><b>Edit text node</b></p> <p><b>Properties</b></p> <p>Group: [Telemedicine] Diagnosis</p> <p>Size: auto</p> <p>Label: Name</p> <p>Value format: {{msg.payload.name}}</p> <p>Layout:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 33%;">label value</td> <td style="width: 33%;">label value</td> <td style="width: 33%;">label value</td> </tr> <tr> <td>label value</td> <td>label value</td> <td>label value</td> </tr> </table> </div>	label value	<div style="border: 1px solid #ccc; padding: 10px;"> <p><b>Edit text node</b></p> <p><b>Properties</b></p> <p>Group: [Telemedicine] Diagnosis</p> <p>Size: auto</p> <p>Label: Age</p> <p>Value format: {{msg.payload.age}}</p> <p>Layout:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 33%;">label value</td> <td style="width: 33%;">label value</td> <td style="width: 33%;">label value</td> </tr> <tr> <td>label value</td> <td>label value</td> <td>label value</td> </tr> </table> </div>	label value										
label value	label value	label value											
label value	label value	label value											
label value	label value	label value											
label value	label value	label value											



Output:

1. Less severe symptoms

Client-side Patient Info

Server- side Debug Window

**Form**

Name \*

Age \*

Breathing Difficulty

Chest pain/pressure

Loss of speech/movement

Fever

Tiredness/Headache /Bodyache and pains

Dry Cough/Sore

Throat/Rash/Skin discolourations

Loss of taste/smell

Conjunctivitis

Diarrhoea

**SUBMIT**

**CANCEL**

```
31/03/2021, 17:33:39 node: Prescription
msg.payload.out : string[35]
▼ string[35]
  Paracetemol
  Calamine
  Refresh Tears
```

```
31/03/2021, 17:33:40 node: 22450b8c.6e591c
msg.payload : Object
▼ object
  name: "Alice"
  age: 33
  breath: false
  chest: false
  loss: false
  fever: false
  tire: true
  dcough: false
  skin: true
  taste: false
  conj: true
  diar: false
▼ out: string
  Paracetemol
  Calamine
  Refresh Tears
```

Server-side Doctor's decision

Prescription	
Name	<b>Alice</b>
Age	<b>33</b>
Prescription	
<b>Paracetemol Calamine Refresh Tears</b>	

## 2. Severe Symptoms

Client-side Patient Info	Server- side Debug Window
<p><b>Form</b></p> <p>Name * Jen</p> <p>Age * 41</p> <p> <input checked="" type="checkbox"/> Breathing Difficulty  <input type="checkbox"/> Chest pain/pressure  <input checked="" type="checkbox"/> Loss of speech/movement  <input type="checkbox"/> Fever  <input type="checkbox"/> Tiredness/Headache /Bodyache and pains  <input type="checkbox"/> Dry Cough/Sore Throat  <input type="checkbox"/> Rash/Skin discolourations  <input type="checkbox"/> Loss of taste/smell  <input type="checkbox"/> Conjunctivitis  <input type="checkbox"/> Diarrhoea         </p> <p style="text-align: center;"><b>SUBMIT</b>      <b>CANCEL</b></p>	<p>31/03/2021, 17:42:11 node: Prescription msg.payload.out : string[24]</p> <p><b>"Get admitted immediately"</b></p> <p>31/03/2021, 17:42:11 node: 22450b8c.6e5 msg.payload : Object</p> <p>▼ object</p> <pre> name: "Jen" age: 41 breath: true chest: false loss: true fever: false tire: false dcough: false skin: false taste: false conj: false diar: false out: "Get admitted immediately" </pre>

Server-side Doctor's decision

Prescription	
Name	Jen
Age	41
Prescription	
<b>Get admitted immediately</b>	

Inference: The form node is used in the client side to get patient details and symptoms, which are sent to the server-side using FRED out. Then, on the server-side, the symptoms' information is extracted through the FRED in node and a prescription is formed. For less severe symptoms, medicines are prescribed by the doctor. For the serious symptoms, the patient is advised to get admitted.

Results: Hence patient symptom info is collected and doctor's decision is displayed in dashboard and debug window using node-red and FRED.

## Experiment 9

# KNIME Analytics Platform – Sales Analytics

Aim: To do Sales Analysis on KNIME Analytics Platform and visualize the data with Stacked Area Chart and Pie Chart.

Software Used: KNIME, the Konstanz Information Miner (<https://www.knime.com/>).

Theory:

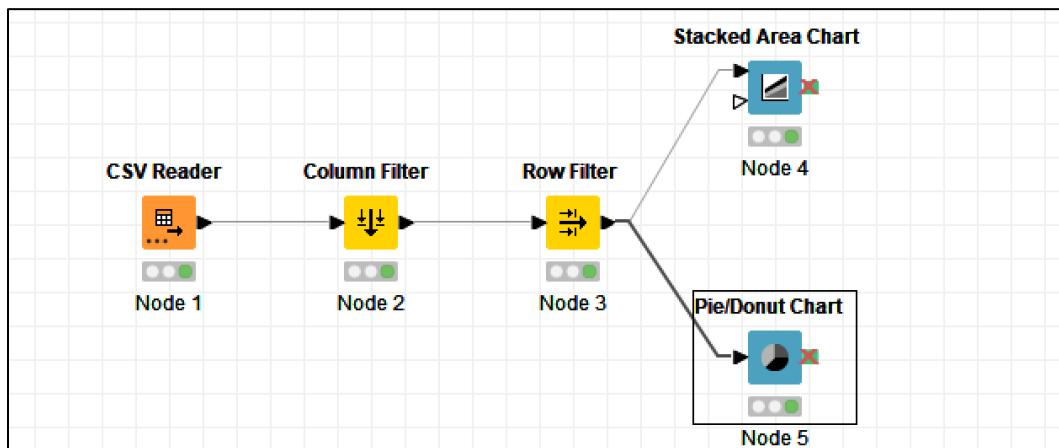
About KNIME:

- KNIME Analytics Platform is a free, open-source software used in data science.
- It is helpful in discovering potentials hidden in the data, mining for fresh insights, or predicting new features.

Dataset Used:

- sales\_data.csv file used for Sales Analytics
- ([https://files.knime.com/sites/default/files/sales\\_data.csv](https://files.knime.com/sites/default/files/sales_data.csv))

Workflow:



Configurations and Outputs:

**Column Filter Dialog:** This dialog shows two lists of columns: 'Exclude' (product, quantity, card, Cust\_ID) and 'Include' (country, date, amount). The 'Exclude' list is highlighted with a red border, and the 'Include' list is highlighted with a green border. The 'OK' button is at the bottom.

Row ID	country	date	amount
Row0	unknown	2008-12-12	3
Row1	China	2009-04-10	160
Row2	China	2009-04-10	160
Row3	China	2009-05-10	160
Row4	USA	2009-05-20	1600
Row5	Brazil	2009-06-08	1200
Row6	USA	2009-07-04	70
Row7	USA	2009-07-14	70
Row8	USA	2009-08-20	1600

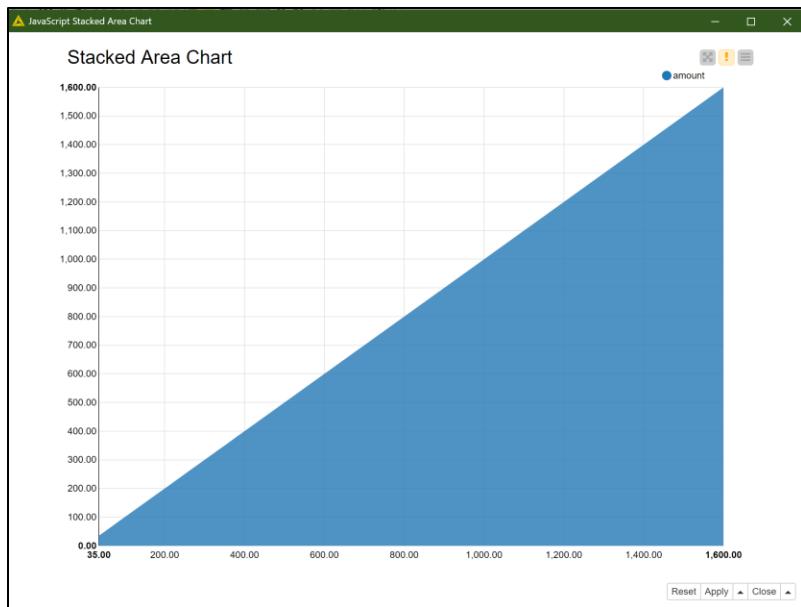
**Filtered table - 3:2 - Column Filter:** This screenshot shows the resulting filtered data table with 47 rows. The columns are Row ID, country, date, and amount. The data is identical to the table above.

The screenshot shows two windows side-by-side. On the left is the 'Dialog - 3:3 - Row Filter' dialog, which contains settings for filtering rows based on a column value ('country'). It includes options for case sensitivity, wildcards, regular expressions, range checking, and missing values. On the right is a table titled 'Filtered - 3:3 - Row Filter' showing 46 rows of data with columns for Row ID, country, date, and amount.

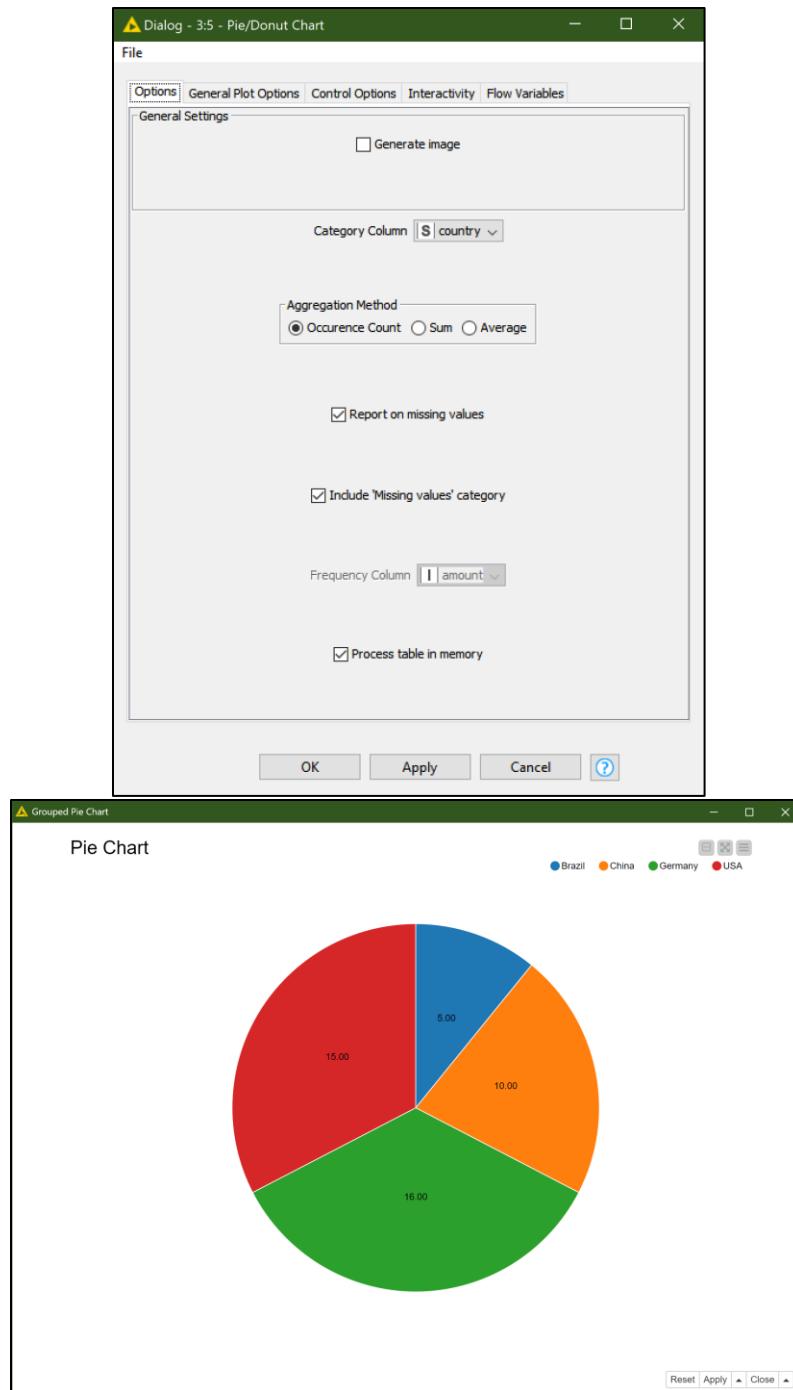
Row ID	country	date	amount
Row1	China	2009-04-10	160
Row2	China	2009-04-10	160
Row3	China	2009-05-10	160
Row4	USA	2009-05-20	1600
Row5	Brazil	2009-06-08	1200
Row6	USA	2009-07-04	70
Row7	USA	2009-07-14	70
Row8	USA	2009-08-20	1600
Row9	Germany	2009-11-02	600

Stacked Area Chart:

The screenshot shows the 'Dialog - 3:4 - Stacked Area Chart' dialog. It includes general settings like generating an image and specifying a maximum number of rows (2,500). Below these are sections for x-axis column selection and sorting. Two filter selection panes are displayed: one on the left with a red border labeled 'Filter' and another on the right with a green border also labeled 'Filter'. Both panes show a list of columns ('amount') and include buttons for moving items between lists and checkboxes for 'Enforce exclusion' (left) and 'Enforce inclusion' (right). The 'Manual Selection' radio button is selected at the top of the pane area.



Pie Chart:



Inference: Knime aids in pre-processing the data using column and row filters, removing redundant or unwanted data (data cleaning) and visualizing the data via graphs

Result: Thus, we have performed Sales Analysis on Knime Analytics Platform and visualized the data with Stacked Area Chart and Pie Chart and observed the outcome successfully.

## Data Manipulation in Knime: String Manipulation, Math Formula and Rule Engine

Aim: To do Sales Analysis and Data Manipulation on Knime Analytics Platform and analyze the data with Rule Engine, Math Formula and String Manipulation.

Software Used: KNIME, the Konstanz Information Miner (<https://www.knime.com/>).

Theory:

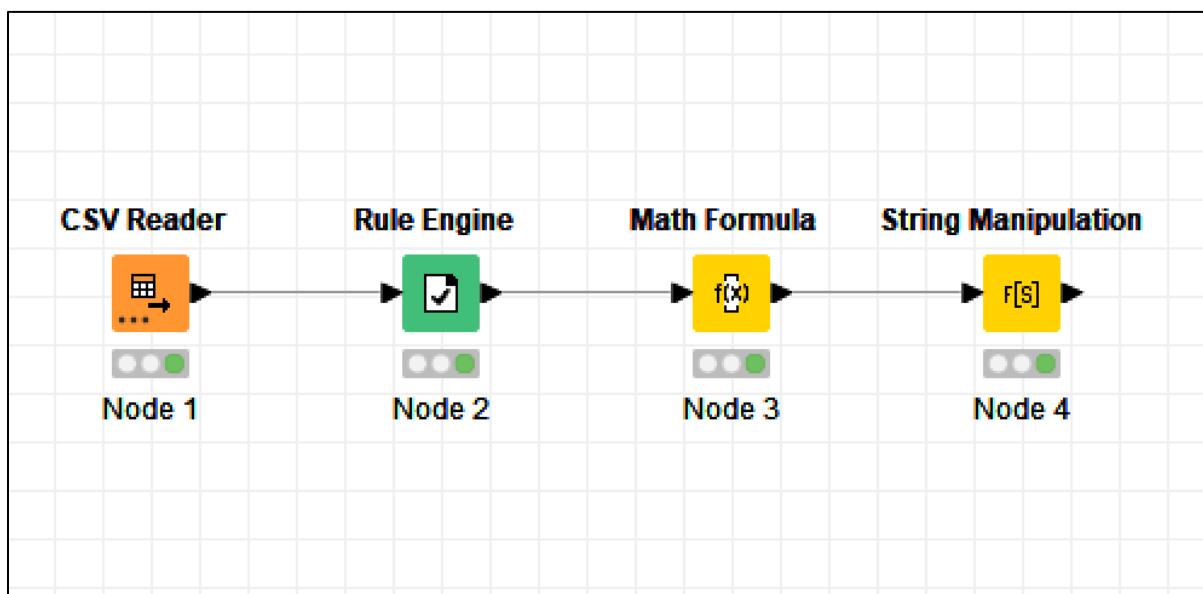
About KNIME: KNIME Analytics Platform is a free, open-source software used in data science. It is helpful in discovering potentials hidden in the data, mining for fresh insights, or predicting new features.

Data Manipulation Nodes:

- Rule Engine Node: This node takes a list of user-defined rules and tries to match them to each row in the input table. If a rule matches, its outcome value is added into a new column. The first matching rule in order of definition determines the outcome.
- Math Formula Node: This node evaluates a mathematical expression based on the values in a row. The computed results can be either appended as a new column or be used to replace an input column. This node uses JEP, the Java Math Expression Parser.
- String Manipulation Node: Manipulates strings like search and replace, capitalize or remove leading and trailing white spaces.

Dataset used: sales\_data.csv file used for Sales Analytics

Flow:



Configurations:

CSV Reader:

File Table - 3:1 - CSV Reader								
File Edit Hilite Navigation View								
Table "default" - Rows: 47 Spec - Columns: 7 Properties Flow Variables								
Row ID	\$ product	\$ country	\$ date	I quantity	I amount	\$ card	\$ Cust_ID	\$
Row0	prod_4	unknown	2008-12-12	1	3	?	Cust_8	
Row1	prod_3	China	2009-04-10	2	160	N	Cust_2	
Row2	prod_3	China	2009-04-10	2	160	Y	Cust_5	
Row3	prod_3	China	2009-05-10	2	160	?	Cust_2	
Row4	prod_3	USA	2009-05-20	20	1600	?	Cust_3	
Row5	prod_3	Brazil	2009-06-08	15	1200	?	Cust_7	
Row6	prod_1	USA	2009-07-04	2	70	Y	Cust_3	
Row7	prod_1	USA	2009-07-14	2	70	?	Cust_6	
Row8	prod_3	USA	2009-08-20	20	1600	?	Cust_3	
Row9	prod_2	Germany	2009-11-02	15	600	?	Cust_1	
Row10	prod_2	Germany	2009-11-22	15	600	N	Cust_1	
Row11	prod_1	Germany	2009-12-02	1	35	Y	Cust_1	
Row12	prod_1	China	2009-12-12	1	35	Y	Cust_2	
Row13	prod_3	USA	2010-01-03	20	1600	?	Cust_3	

Rule Engine:

```
Expression
?
1 // enter ordered set of rules, e.g.:
?
2 // $double column name$ > 5.0 => "large"
?
3 // $string column name$ LIKE "*blue*" => "small and blue"
?
4 // TRUE => "default outcome"
S
5 $quantity$ =1 => "Need based customer"
S
6 $quantity$ >=10 => "Impulse customer"
S
7 TRUE => "Loyal Customer"
```

Classified values - 3:2 - Rule Engine

File Edit Hilite Navigation View

Table "default" - Rows: 47 Spec - Columns: 8 Properties Flow Variables

Row ID	product	country	date	quantity	amount	card	Cust_ID	Customer Status
Row0	prod_4	unknown	2008-12-12	1	3	?	Cust_8	Need based customer
Row1	prod_3	China	2009-04-10	2	160	N	Cust_2	Loyal Customer
Row2	prod_3	China	2009-04-10	2	160	Y	Cust_5	Loyal Customer
Row3	prod_3	China	2009-05-10	2	160	?	Cust_2	Loyal Customer
Row4	prod_3	USA	2009-05-20	20	1600	?	Cust_3	Impulse customer
Row5	prod_3	Brazil	2009-06-08	15	1200	?	Cust_7	Impulse customer
Row6	prod_1	USA	2009-07-04	2	70	Y	Cust_3	Loyal Customer
Row7	prod_1	USA	2009-07-14	2	70	?	Cust_6	Loyal Customer
Row8	prod_3	USA	2009-08-20	20	1600	?	Cust_3	Impulse customer
Row9	prod_2	Germany	2009-11-02	15	600	?	Cust_1	Impulse customer
Row10	prod_2	Germany	2009-11-22	15	600	N	Cust_1	Impulse customer
Row11	prod_1	Germany	2009-12-02	1	35	Y	Cust_1	Need based customer
Row12	prod_1	China	2009-12-12	1	35	Y	Cust_2	Need based customer
Row13	prod_3	USA	2010-01-03	20	1600	?	Cust_3	Impulse customer
Row14	prod_1	Germany	2010-01-10	1	35	N	Cust_1	Need based customer
Row15	prod_3	Germany	2010-01-13	1	80	?	Cust_4	Need based customer
Row16	prod_2	Germany	2010-01-15	25	1000	?	Cust_1	Impulse customer

Math Formula Node:

Expression

```
1 pow($amount$,2)
```

Append Column: Sales Gain

Replace Column: **S** Customer Status

Output data - 3:3 - Math Formula

File Edit Hilite Navigation View

Table "default" - Rows: 47 Spec - Columns: 9 Properties Flow Variables

Row ID	product	country	date	quantity	amount	card	Cust_ID	Customer Status	Sales Gain
Row0	prod_4	unknown	2008-12-12	1	3	?	Cust_8	Need based customer	9
Row1	prod_3	China	2009-04-10	2	160	N	Cust_2	Loyal Customer	25,600
Row2	prod_3	China	2009-04-10	2	160	Y	Cust_5	Loyal Customer	25,600
Row3	prod_3	China	2009-05-10	2	160	?	Cust_2	Loyal Customer	25,600
Row4	prod_3	USA	2009-05-20	20	1600	?	Cust_3	Impulse customer	2,560,000
Row5	prod_3	Brazil	2009-06-08	15	1200	?	Cust_7	Impulse customer	1,440,000
Row6	prod_1	USA	2009-07-04	2	70	Y	Cust_3	Loyal Customer	4,900
Row7	prod_1	USA	2009-07-14	2	70	?	Cust_6	Loyal Customer	4,900
Row8	prod_3	USA	2009-08-20	20	1600	?	Cust_3	Impulse customer	2,560,000
Row9	prod_2	Germany	2009-11-02	15	600	?	Cust_1	Impulse customer	360,000
Row10	prod_2	Germany	2009-11-22	15	600	N	Cust_1	Impulse customer	360,000
Row11	prod_1	Germany	2009-12-02	1	35	Y	Cust_1	Need based customer	1,225

String Manipulation Node:

Expression	
1	upperCase(\$country\$)
<input type="radio"/> Append Column: <input type="text" value="new column"/> <input type="checkbox"/> Insert Missing As Null	
<input checked="" type="radio"/> Replace Column: <input type="text" value="S country"/> <input checked="" type="checkbox"/> Syntax check on close	

⚠ Appended table - 3:4 - String Manipulation

File Edit Hilite Navigation View

Table "default" - Rows: 47 Spec - Columns: 9 Properties Flow Variables

Row ID	S product	S country	S date	I quantity	I amount	S card	S Cust_ID	S Customer Status	D Sales Gain
Row0	prod_4	UNKNOWN	2008-12-12	1	3	?	Cust_8	Need based customer	9
Row1	prod_3	CHINA	2009-04-10	2	160	N	Cust_2	Loyal Customer	25,600
Row2	prod_3	CHINA	2009-04-10	2	160	Y	Cust_5	Loyal Customer	25,600
Row3	prod_3	CHINA	2009-05-10	2	160	?	Cust_2	Loyal Customer	25,600
Row4	prod_3	USA	2009-05-20	20	1600	?	Cust_3	Impulse customer	2,560,000
Row5	prod_3	BRAZIL	2009-06-08	15	1200	?	Cust_7	Impulse customer	1,440,000
Row6	prod_1	USA	2009-07-04	2	70	Y	Cust_3	Loyal Customer	4,900
Row7	prod_1	USA	2009-07-14	2	70	?	Cust_6	Loyal Customer	4,900
Row8	prod_3	USA	2009-08-20	20	1600	?	Cust_3	Impulse customer	2,560,000
Row9	prod_2	GERMANY	2009-11-02	15	600	?	Cust_1	Impulse customer	360,000
Row10	prod_2	GERMANY	2009-11-22	15	600	N	Cust_1	Impulse customer	360,000
Row11	prod_1	GERMANY	2009-12-02	1	35	Y	Cust_1	Need based customer	1,225
Row12	prod_1	CHINA	2009-12-12	1	35	Y	Cust_2	Need based customer	1,225

Task:

Math Formula:

Expression	
1	pow(\$amount\$,2)
2	\$quantity\$+\$amount\$
3	\$quantity\$*5
<input checked="" type="radio"/> Append Column: <input type="text" value="New 1"/>	
<input type="radio"/> Replace Column: <input type="text" value="S Customer Status"/>	
<input type="checkbox"/> Convert to Int	

Output data - 3:3 - Math Formula

File Edit Hilite Navigation View

Table "default" - Rows: 47 Spec - Columns: 9 Properties Flow Variables

Row ID	product	country	date	quantity	amount	card	Cust_ID	Customer Status	New1
Row0	prod_4	unknown	2008-12-12	1	3	?	Cust_8	Need based customer	24
Row1	prod_3	China	2009-04-10	2	160	N	Cust_2	Loyal Customer	52,800
Row2	prod_3	China	2009-04-10	2	160	Y	Cust_5	Loyal Customer	52,800
Row3	prod_3	China	2009-05-10	2	160	?	Cust_2	Loyal Customer	52,800
Row4	prod_3	USA	2009-05-20	20	1600	?	Cust_3	Impulse customer	51,360,000
Row5	prod_3	Brazil	2009-06-08	15	1200	?	Cust_7	Impulse customer	21,690,000
Row6	prod_1	USA	2009-07-04	2	70	Y	Cust_3	Loyal Customer	10,500
Row7	prod_1	USA	2009-07-14	2	70	?	Cust_6	Loyal Customer	10,500
Row8	prod_3	USA	2009-08-20	20	1600	?	Cust_3	Impulse customer	51,360,000
Row9	prod_2	Germany	2009-11-02	15	600	?	Cust_1	Impulse customer	5,445,000
Row10	prod_2	Germany	2009-11-22	15	600	N	Cust_1	Impulse customer	5,445,000
Row11	prod_1	Germany	2009-12-02	1	35	Y	Cust_1	Need based customer	1,400
Row12	prod_1	China	2009-12-12	1	35	Y	Cust_2	Need based customer	1,400
Row13	prod_3	USA	2010-01-03	20	1600	?	Cust_3	Impulse customer	51,360,000
Row14	prod_1	Germany	2010-01-10	1	35	N	Cust_1	Need based customer	1,400

### String Manipulation:

Expression

```
1 reverse(upperCase($country$))
```

Append Column: New2  Insert Missing As Null

Replace Column:  Syntax check on close

OK Apply Cancel ?

Appended table - 3:4 - String Manipulation

File Edit Hilite Navigation View

Table "default" - Rows: 47 Spec - Columns: 10 Properties Flow Variables

Row ID	product	country	date	quantity	amount	card	Cust_ID	Customer Status	New1	New2
Row0	prod_4	unknown	2008-12-12	1	3	?	Cust_8	Need based customer	24	NWONKNU
Row1	prod_3	China	2009-04-10	2	160	N	Cust_2	Loyal Customer	52,800	ANIHIC
Row2	prod_3	China	2009-04-10	2	160	Y	Cust_5	Loyal Customer	52,800	ANIHIC
Row3	prod_3	China	2009-05-10	2	160	?	Cust_2	Loyal Customer	52,800	ANIHIC
Row4	prod_3	USA	2009-05-20	20	1600	?	Cust_3	Impulse customer	51,360,000	ASU
Row5	prod_3	Brazil	2009-06-08	15	1200	?	Cust_7	Impulse customer	21,690,000	LIZARB
Row6	prod_1	USA	2009-07-04	2	70	Y	Cust_3	Loyal Customer	10,500	ASU
Row7	prod_1	USA	2009-07-14	2	70	?	Cust_6	Loyal Customer	10,500	ASU
Row8	prod_3	USA	2009-08-20	20	1600	?	Cust_3	Impulse customer	51,360,000	ASU
Row9	prod_2	Germany	2009-11-02	15	600	?	Cust_1	Impulse customer	5,445,000	YNAMREG
Row10	prod_2	Germany	2009-11-22	15	600	N	Cust_1	Impulse customer	5,445,000	YNAMREG
Row11	prod_1	Germany	2009-12-02	1	35	Y	Cust_1	Need based customer	1,400	YNAMREG
Row12	prod_1	China	2009-12-12	1	35	Y	Cust_2	Need based customer	1,400	ANIHIC
Row13	prod_3	USA	2010-01-03	20	1600	?	Cust_3	Impulse customer	51,360,000	ASU
Row14	prod_1	Germany	2010-01-10	1	35	N	Cust_1	Need based customer	1,400	YNAMREG
Row15	prod_3	Germany	2010-01-13	1	80	?	Cust_4	Need based customer	6,800	YNAMREG
Row16	prod_2	Germany	2010-01-15	25	1000	?	Cust_1	Impulse customer	25,125,000	YNAMREG

Inference: Rule Engine node is used to add columns based on user-defined rules. Math Formula node is used to execute mathematical expressions on the data, and String Manipulation node is used to manipulate string values in the dataset.

Result: Thus, we have performed sales analysis and data manipulation on Knime Analytics Platform and analysed the data with Rule Engine, Math Formula and String Manipulation successfully.

## **K Means Clustering in Knime: Unsupervised Machine Learning**

Aim: To cluster data and classify them under respective features on Knime Analytics Platform with the help of Color Manger & Shape Manager and analyze the data using k-Means cluster node and Cluster assigner node.

Software Used: KNIME, the Konstanz Information Miner (<https://www.knime.com/>).

Theory:

About KNIME: KNIME Analytics Platform is a free, open-source software used in data science. It is helpful in discovering potentials hidden in the data, mining for fresh insights, or predicting new features.

Clustering:

- Unsupervised Learning.
- Requires data, but no labels.
- Detect patterns e.g., in
  - Group emails or search results.
  - Customer shopping patterns.
  - Region of images.
- Useful when you don't know what you are looking for.
- Basic Idea: Group Together similar instances.
- Example: 2D point patterns.

k-Means Algorithm:

- An iterative clustering algorithm.
- Initialize: Pick k random points as cluster centers.
- Alternate: (1) Assign data points to the closest cluster center. (2) Change the cluster center to the average of its assigned points.
- Stop when no points assignment changes.

Nodes:

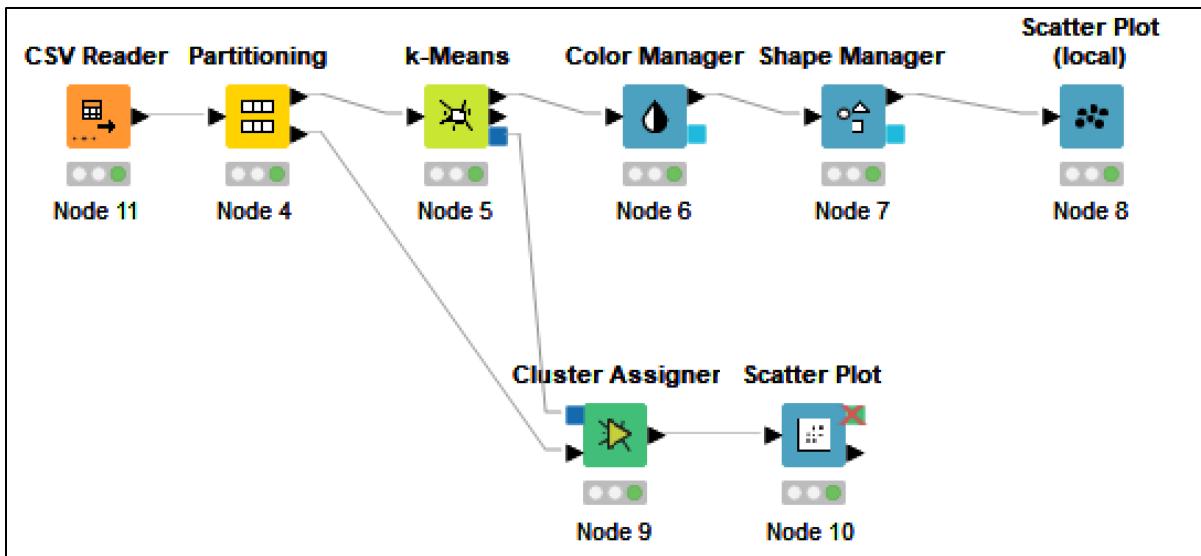
- k-Means Node:
  - This node outputs the cluster centers for a predefined number of clusters (no dynamic number of clusters).
  - K-means performs a crisp clustering that assigns a data vector to exactly one cluster.
  - The Algorithm terminates when the cluster assignments do not change anymore.
  - The clustering algorithm uses the Euclidean distance on the selected attributes.
  - The data is not normalized by the node.

- Cluster Assigner Node:
  - This node assigns new data to an existing set of prototypes, which are obtained e.g., by a k-means clustering.
  - Each data point is assigned to its nearest prototype.
- Color Manager Node:
  - Colors can be assigned for either nominal (possible values have to be available) or numeric columns (with lower and upper bounds).
- Shape Manager Node:
  - Assigns (different) shapes for each attribute value of one nominal column, i.e., for each possible value. Supporting views then render data points with the shape associated with the corresponding attribute value.
- Scatter Plot Node (local):
  - Creates a scatterplot of two selectable attributes. Then each datapoint is displayed as a dot at its corresponding place, dependent on its values of the selected attributes.
  - The dots are displayed in the color defined by the Color Manager, the size defined by the Size Manager, and the shape defined by the Shape Manager.
- Scatter Plot:
  - A scatter plot using a JavaScript based charting library. The view can be accessed either via the "interactive view" action on the executed node or in KNIME Server web portal page.
  - The configuration of the node lets you choose the size of a sample to display and to enable certain controls, which are then available in the view.

**Dataset Used:** The Iris dataset consists of 50 samples from each of three classes of iris flowers. There are five attributes in the dataset:

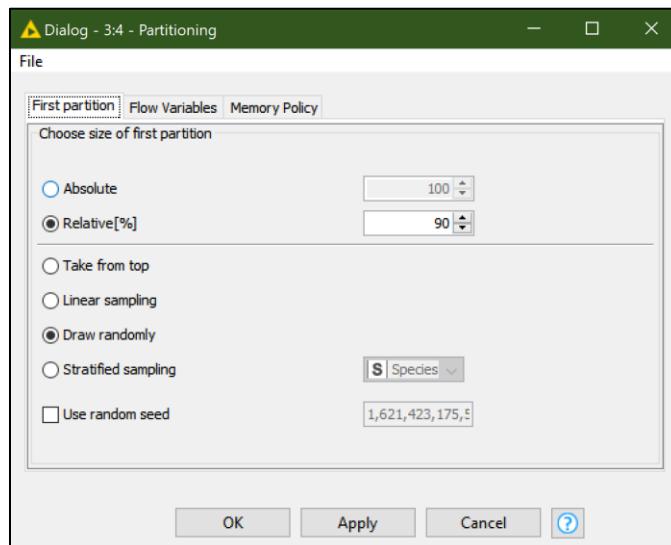
- sepal length in cm,
- sepal width in cm,
- petal length in cm,
- petal width in cm, and
- class: Iris Setosa, Iris Versicolour, and Iris Virginica.

**Flow:**

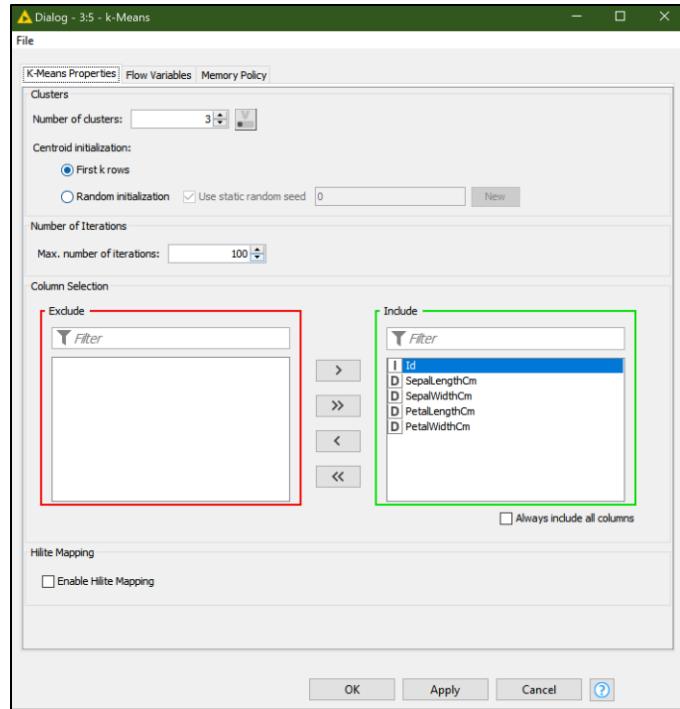


Configurations:

Partitioning:



K-Means:



**Clusters - 3:5 - k-Means**

File Edit Hilite Navigation View

Table "default" - Rows: 3 Spec - Columns: 5 Properties Flow Variables

Row ID	D Id	D SepalLe...	D SepalW...	D PetalLe...	D PetalWi...
cluster_0	24.674	5	3.405	1.467	0.253
cluster_1	74.864	5.961	2.768	4.23	1.295
cluster_2	124.646	6.552	2.981	5.517	2.004

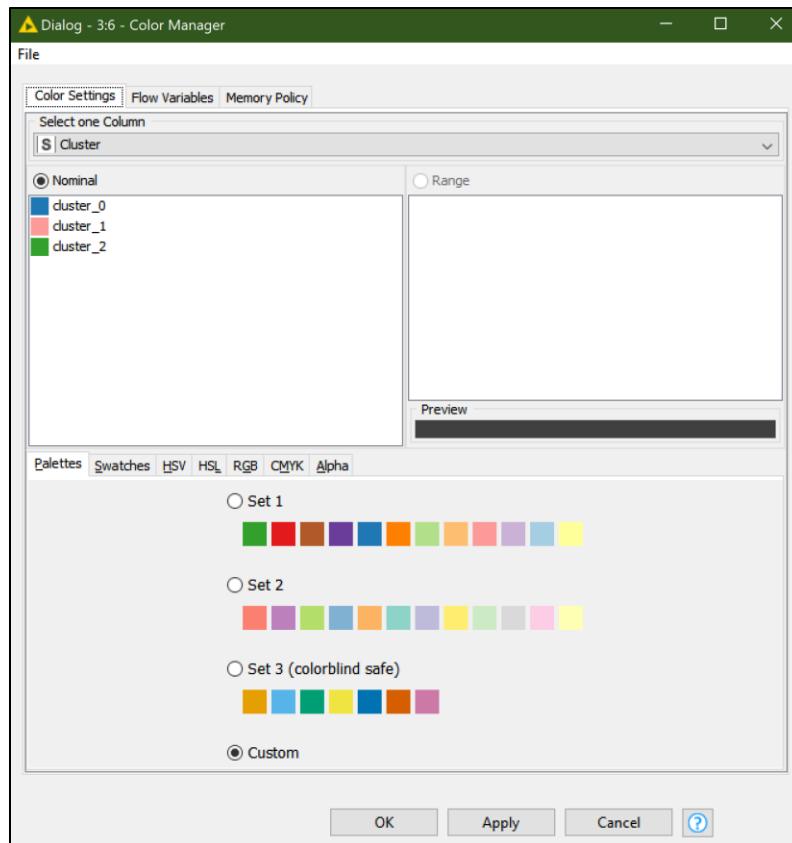
**Labeled input - 3:5 - k-Means**

File Edit Hilite Navigation View

Table "default" - Rows: 135 Spec - Columns: 7 Properties Flow Variables

Row ID	I Id	D SepalLe...	D SepalW...	D PetalLe...	D PetalWi...	S Species	S Cluster
Row0	1	5.1	3.5	1.4	0.2	Iris-setosa	cluster_0
Row1	2	4.9	3	1.4	0.2	Iris-setosa	cluster_0
Row2	3	4.7	3.2	1.3	0.2	Iris-setosa	cluster_0
Row3	4	4.6	3.1	1.5	0.2	Iris-setosa	cluster_0
Row4	5	5	3.6	1.4	0.2	Iris-setosa	cluster_0
Row5	6	5.4	3.9	1.7	0.4	Iris-setosa	cluster_0
Row6	7	4.6	3.4	1.4	0.3	Iris-setosa	cluster_0
Row7	8	5	3.4	1.5	0.2	Iris-setosa	cluster_0
Row9	10	4.9	3.1	1.5	0.1	Iris-setosa	cluster_0
Row10	11	5.4	3.7	1.5	0.2	Iris-setosa	cluster_0
Row11	12	4.8	3.4	1.6	0.2	Iris-setosa	cluster_0

Color Manager:



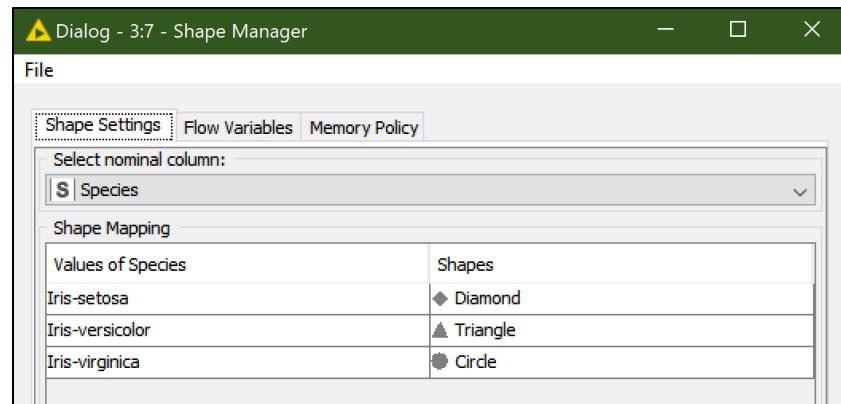
**Table with Colors - 3:6 - Color Manager**

File Edit Hilite Navigation View

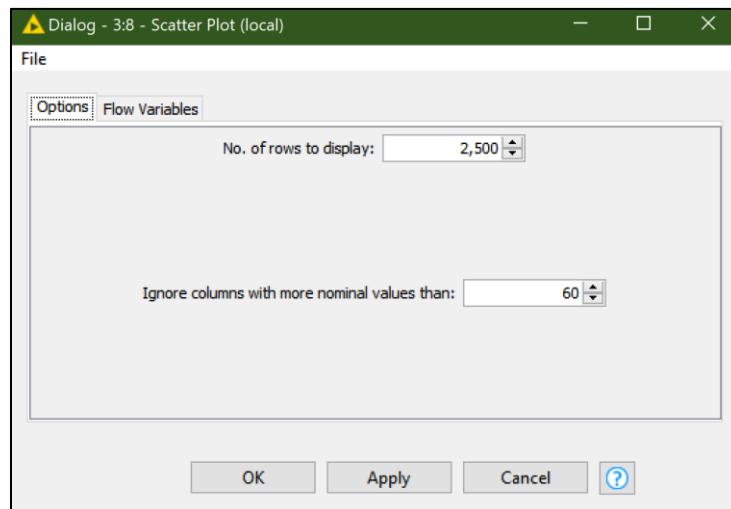
Table "default" - Rows: 135 Spec - Columns: 7 Properties Flow Variables

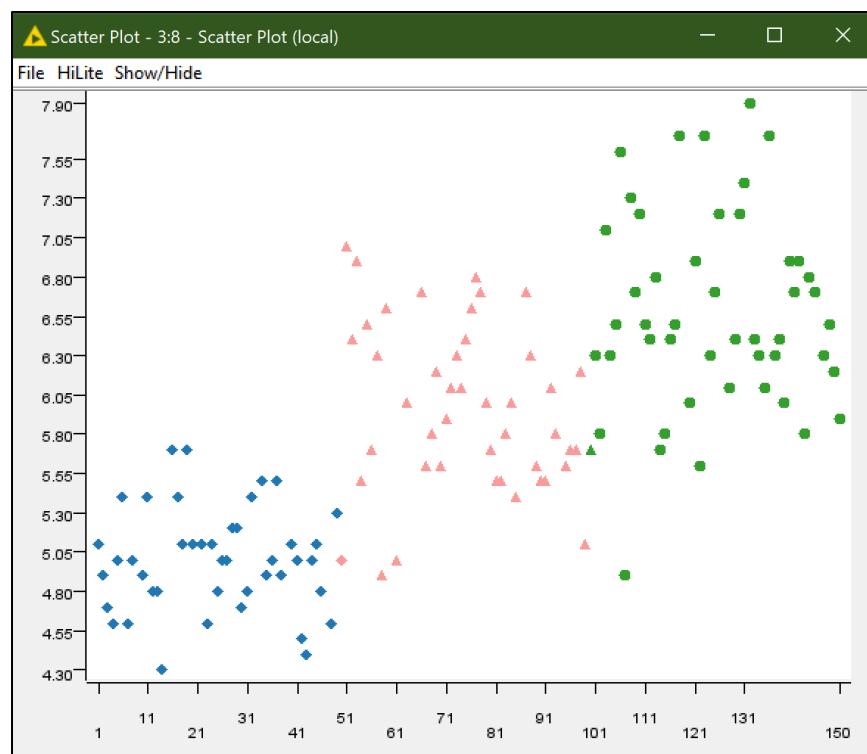
Row ID	I   Id	D   SepalLe...	D   SepalW...	D   PetalLe...	D   PetalWi...	S   Species	S   Cluster
Row41	42	4.5	2.3	1.3	0.3	Iris-setosa	cluster_0
Row42	43	4.4	3.2	1.3	0.2	Iris-setosa	cluster_0
Row43	44	5	3.5	1.6	0.6	Iris-setosa	cluster_0
Row44	45	5.1	3.8	1.9	0.4	Iris-setosa	cluster_0
Row45	46	4.8	3	1.4	0.3	Iris-setosa	cluster_0
Row47	48	4.6	3.2	1.4	0.2	Iris-setosa	cluster_0
Row48	49	5.3	3.7	1.5	0.2	Iris-setosa	cluster_0
Row49	50	5	3.3	1.4	0.2	Iris-setosa	cluster_1
Row50	51	7	3.2	4.7	1.4	Iris-versicolor	cluster_1
Row51	52	6.4	3.2	4.5	1.5	Iris-versicolor	cluster_1
Row52	53	6.9	3.1	4.9	1.5	Iris-versicolor	cluster_1
Row53	54	5.5	2.3	4	1.3	Iris-versicolor	cluster_1

Shape Manager:

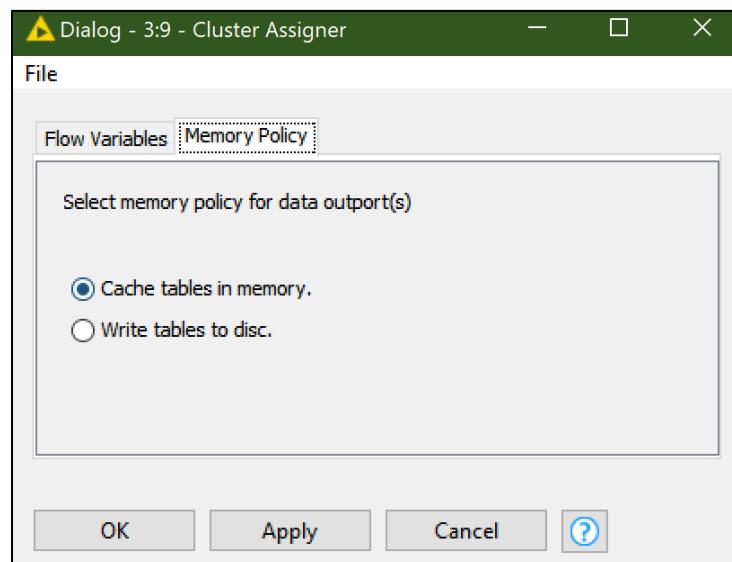


Scatter Plot (Local):





Cluster Assigner:



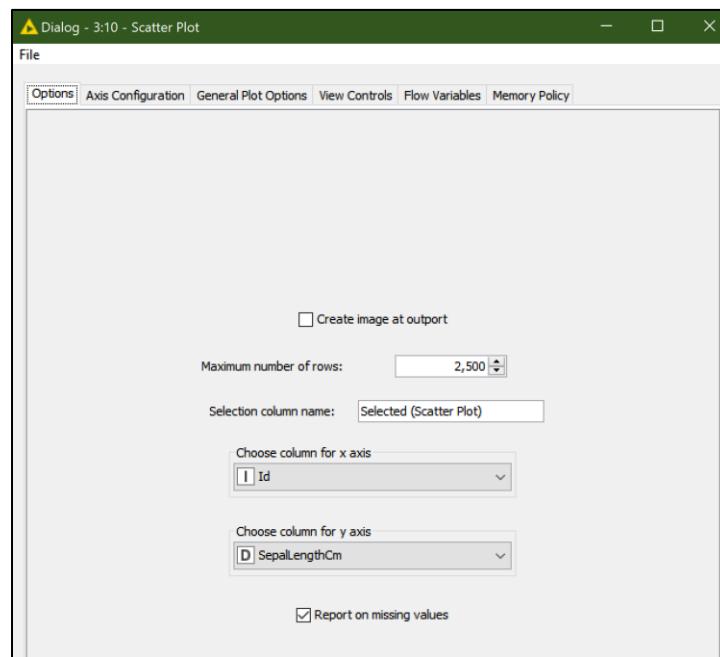
**Assigned Data - 3:9 - Cluster Assigner**

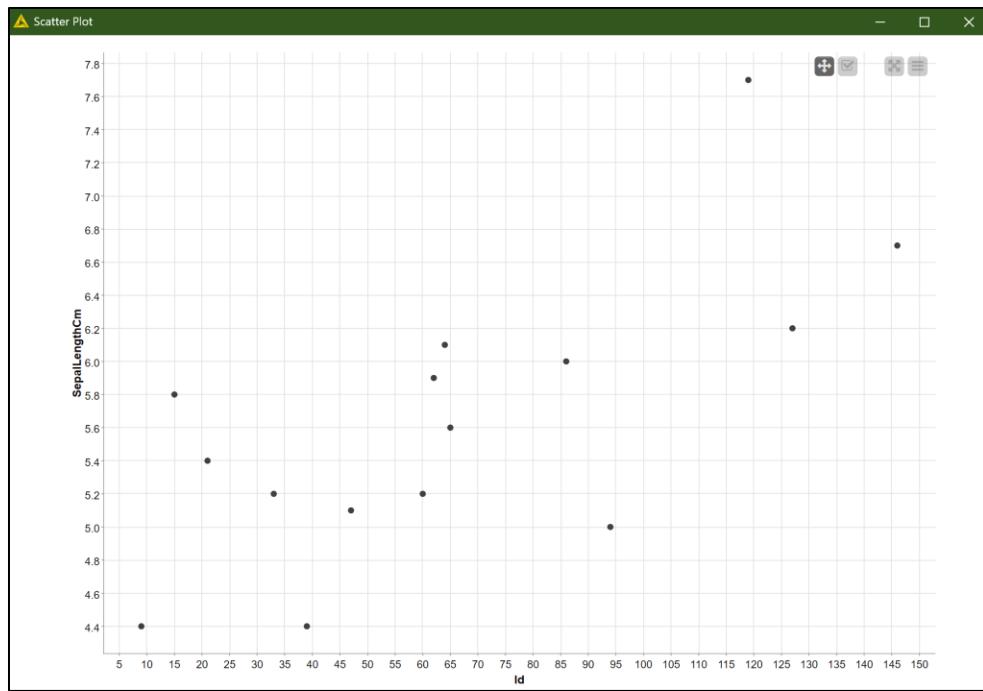
File Edit Hilite Navigation View

Table "default" - Rows: 15 Spec - Columns: 7 Properties Flow Variables

Row ID	I Id	D SepalLe...	D SepalW...	D PetalLe...	D PetalWi...	S Species	S Cluster
Row8	9	4.4	2.9	1.4	0.2	Iris-setosa	cluster_0
Row14	15	5.8	4	1.2	0.2	Iris-setosa	cluster_0
Row20	21	5.4	3.4	1.7	0.2	Iris-setosa	cluster_0
Row32	33	5.2	4.1	1.5	0.1	Iris-setosa	cluster_0
Row38	39	4.4	3	1.3	0.2	Iris-setosa	cluster_0
Row46	47	5.1	3.8	1.6	0.2	Iris-setosa	cluster_0
Row59	60	5.2	2.7	3.9	1.4	Iris-versicolor	cluster_1
Row61	62	5.9	3	4.2	1.5	Iris-versicolor	cluster_1
Row63	64	6.1	2.9	4.7	1.4	Iris-versicolor	cluster_1
Row64	65	5.6	2.9	3.6	1.3	Iris-versicolor	cluster_1
Row85	86	6	3.4	4.5	1.6	Iris-versicolor	cluster_1
Row93	94	5	2.3	3.3	1	Iris-versicolor	cluster_1
Row118	119	7.7	2.6	6.9	2.3	Iris-virginica	cluster_2

Scatter Plot:

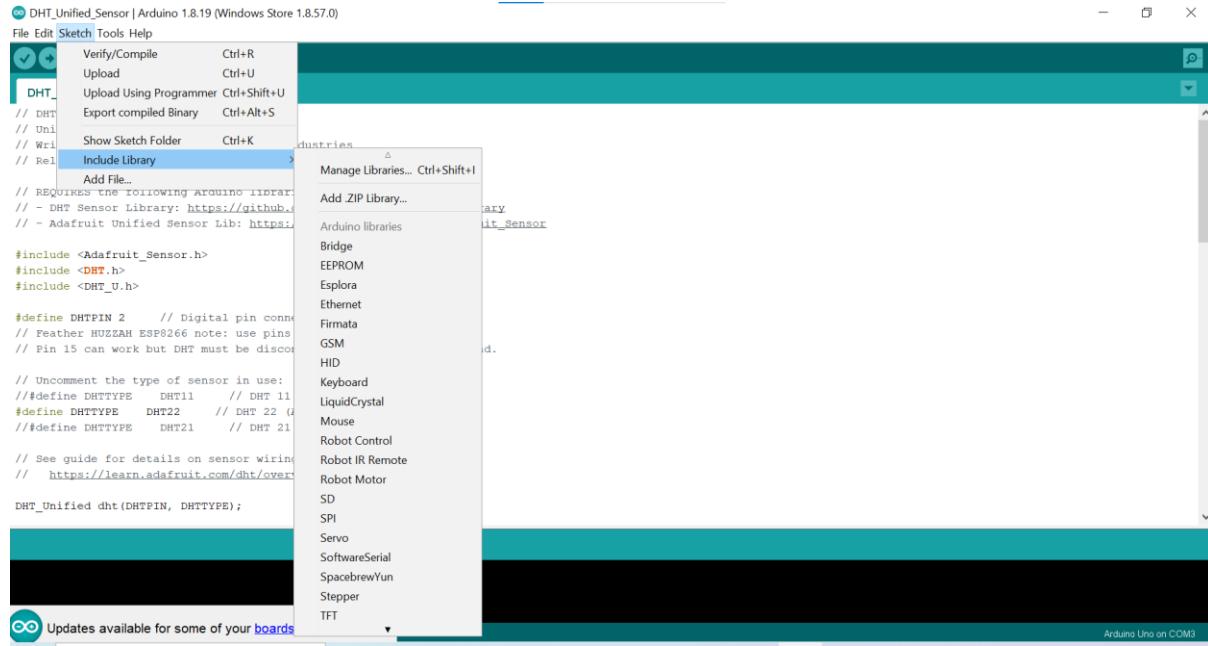




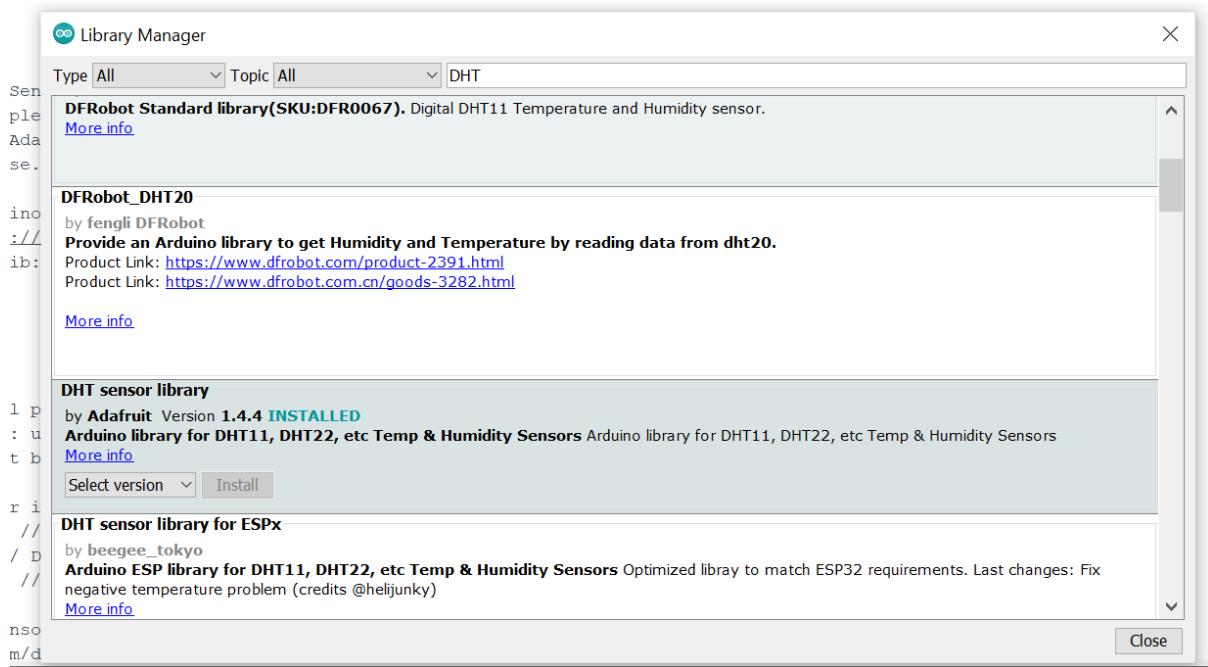
Inference: The K Means node is used to perform k-means clustering on the dataset. The Color Manager and Shape Manager nodes are used to assign colours and shapes to the clusters. The Scatter Plot node is used to view the clusters as scatter plots.

Result: Thus, we have clustered data and classified them under respective features on Knime Analytics Platform with the help of Color Manger & Shape Manager and analyzed the data using k-Means cluster node and Cluster assigner node.

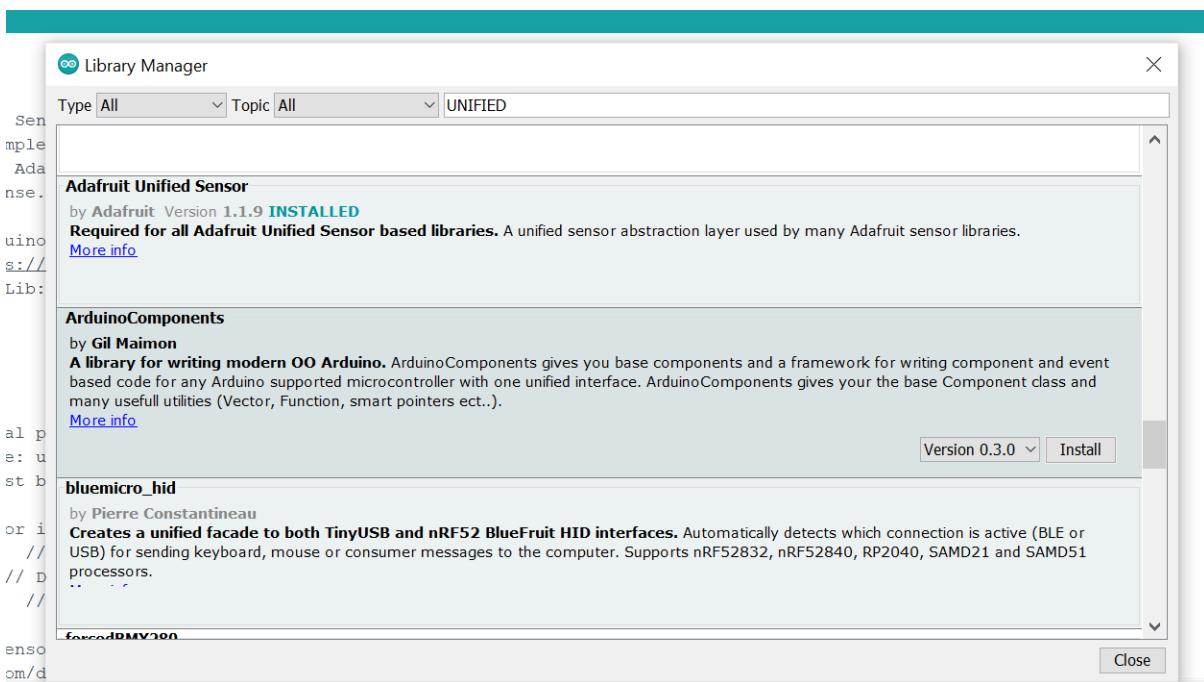
Temperature, humidity sensing with Arduino and node red dashboard



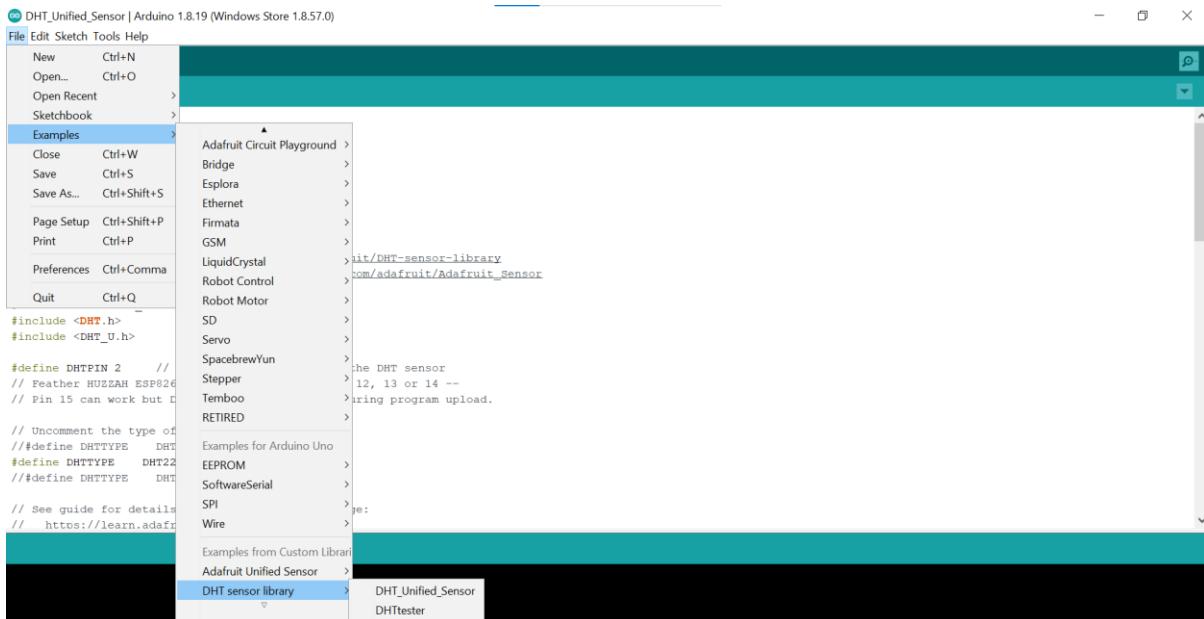
Click manage libraries



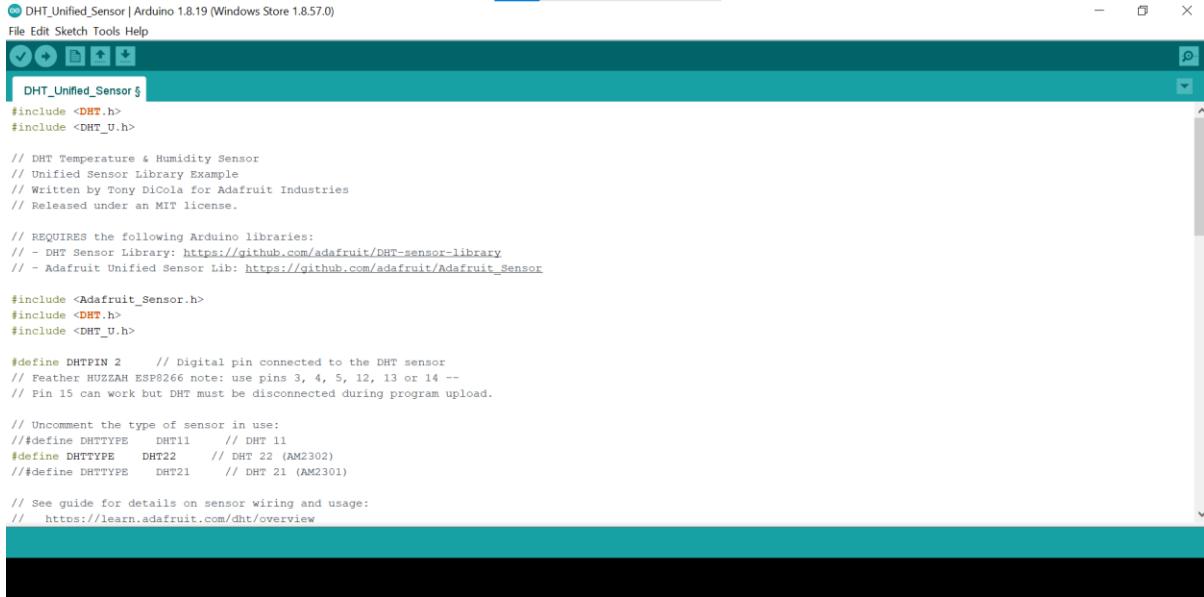
INSTALL DHT SENSOR LIBRARY



## Make sure adafruit unified sensor is installed



## Choose DHT unified sensor



DHT\_Unified\_Sensor | Arduino 1.8.19 (Windows Store 1.8.57.0)

```
#include <DHT.h>
#include <DHT_U.h>

// DHT Temperature & Humidity Sensor
// Unified Sensor Library Example
// Written by Tony DiCola for Adafruit Industries
// Released under an MIT license.

// REQUIRES the following Arduino libraries:
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit\_Sensor

#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2      // Digital pin connected to the DHT sensor
// Feather HUZZAH ESP8266 note: use pins 3, 4, 5, 12, 13 or 14 --
// Pin 15 can work but DHT must be disconnected during program upload.

// Uncomment the type of sensor in use:
//#define DHTTYPE DHT11    // DHT 11
#define DHTTYPE DHT22    // DHT 22 (AM2302)
//#define DHTTYPE DHT21    // DHT 21 (AM2301)

// See guide for details on sensor wiring and usage:
// https://learn.adafruit.com/dht/overview
```

I am using DHT11 so



DHT\_Unified\_Sensor | Arduino 1.8.19 (Windows Store 1.8.57.0)

```
#include <DHT.h>
#include <DHT_U.h>

// DHT Temperature & Humidity Sensor
// Unified Sensor Library Example
// Written by Tony DiCola for Adafruit Industries
// Released under an MIT license.

// REQUIRES the following Arduino libraries:
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit\_Sensor

#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2      // Digital pin connected to the DHT sensor
// Feather HUZZAH ESP8266 note: use pins 3, 4, 5, 12, 13 or 14 --
// Pin 15 can work but DHT must be disconnected during program upload.

// Uncomment the type of sensor in use:
#define DHTTYPE DHT11    // DHT 11
//#define DHTTYPE DHT22    // DHT 22 (AM2302)
//#define DHTTYPE DHT21    // DHT 21 (AM2301)

// see guide for details on sensor wiring and usage:
// https://learn.adafruit.com/dht/overview
```

Done compiling.

Click serial monitor

```

DHT_Unified_Sensor | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
Serial Monitor
DHT_Unified_Sensor $ 
#include <DHT.h>
#include <DHT_U.h>

// DHT Temperature & Humidity Sensor
// Unified Sensor Library Example
// Written by Tony DiCola for Adafruit Industries
// Released under an MIT license.

// REQUIRES the following Arduino libraries:
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit\_Sensor

#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2      // Digital pin connected to the DHT sensor
// Feather HUZZAH ESP8266 note: use pins 3, 4, 5, 12, 13 or 14 --
// Pin 15 can work but DHT must be disconnected during program upload.

// Uncomment the type of sensor in use:
#define DHTTYPE DHT11    // DHT 11
// #define DHTTYPE DHT22    // DHT 22 (AM2302)
// #define DHTTYPE DHT21    // DHT 21 (AM2301)

// See guide for details on sensor wiring and usage:
// https://learn.adafruit.com/dht/overview

```

```

COM3
Send
Humidity: 76.00%
Temperature: 32.20°C
Humidity: 76.00%
Autoscroll  Show timestamp
Newline 9600 baud Clear output
#include <DHT_U.h>

#define DHTPIN 2      // Digital pin connected to the DHT sensor
// Feather HUZZAH ESP8266 note: use pins 3, 4, 5, 12, 13 or 14 --
// Pin 15 can work but DHT must be disconnected during program upload.

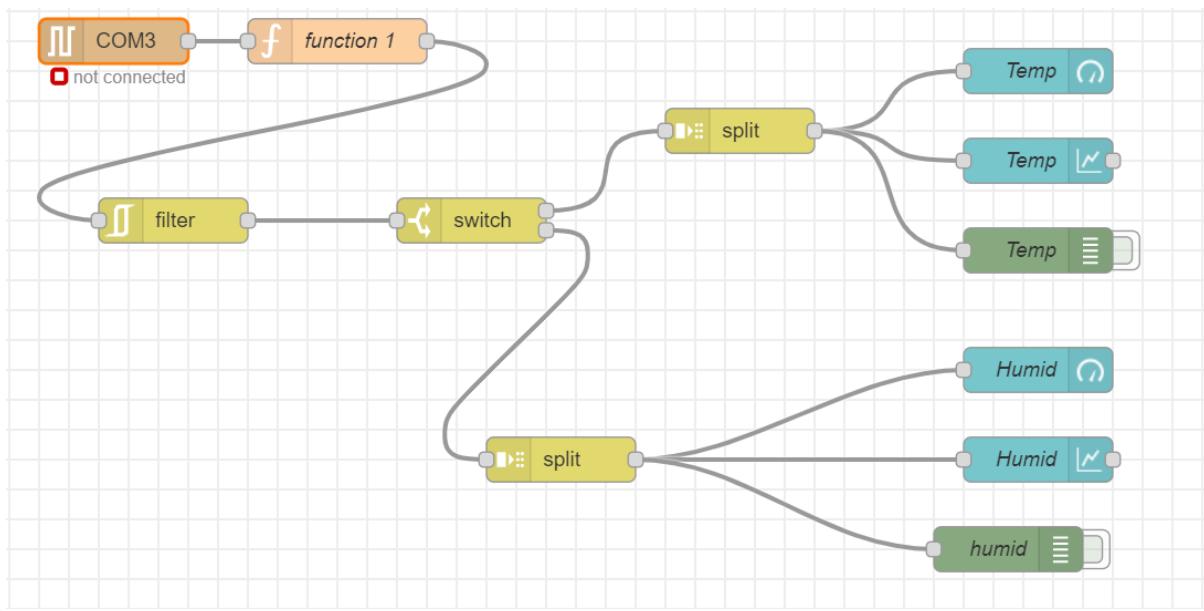
// Uncomment the type of sensor in use:
#define DHTTYPE DHT11    // DHT 11
// #define DHTTYPE DHT22    // DHT 22 (AM2302)
// #define DHTTYPE DHT21    // DHT 21 (AM2301)

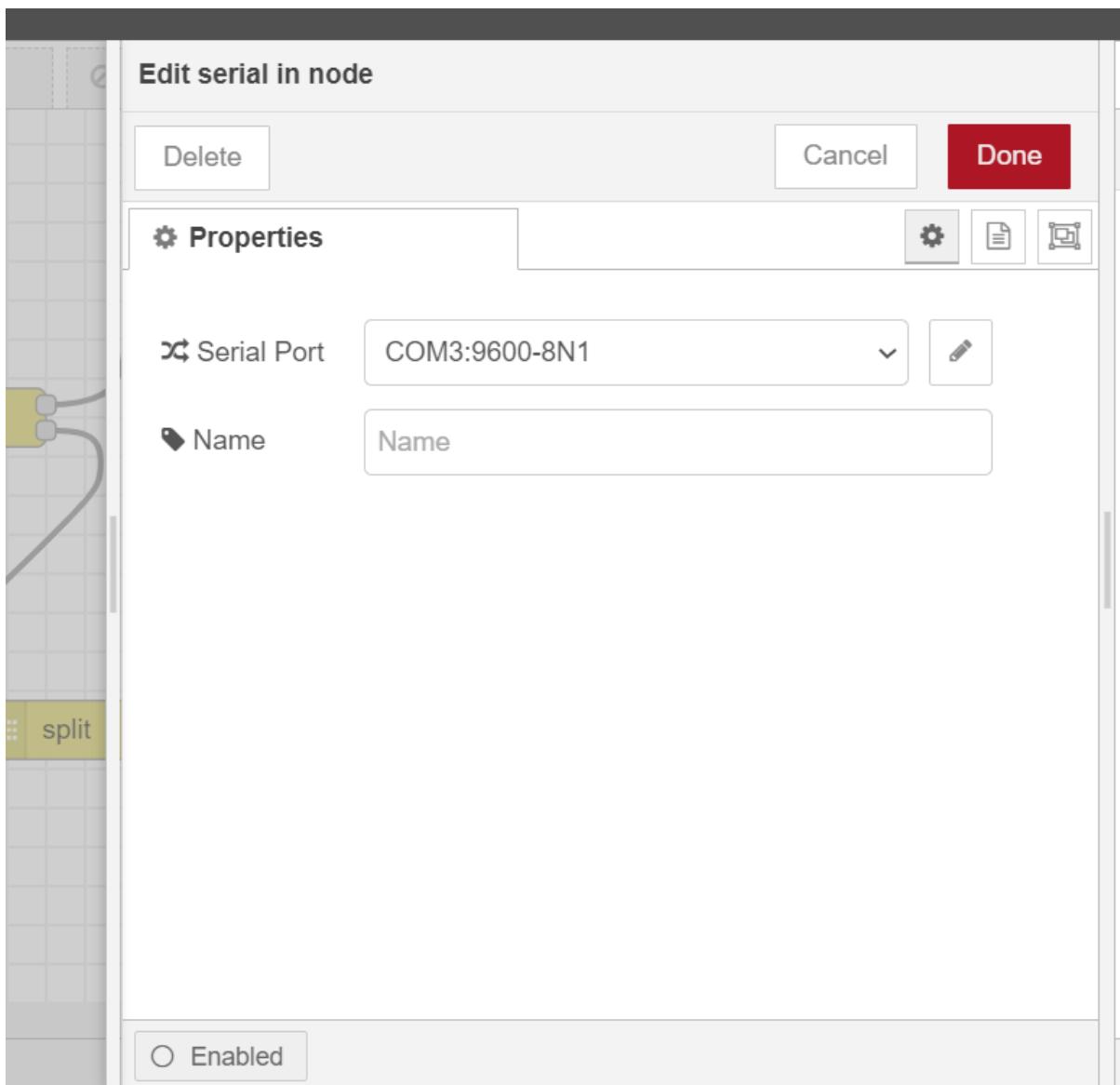
// See guide for details on sensor wiring and usage:
// https://learn.adafruit.com/dht/overview

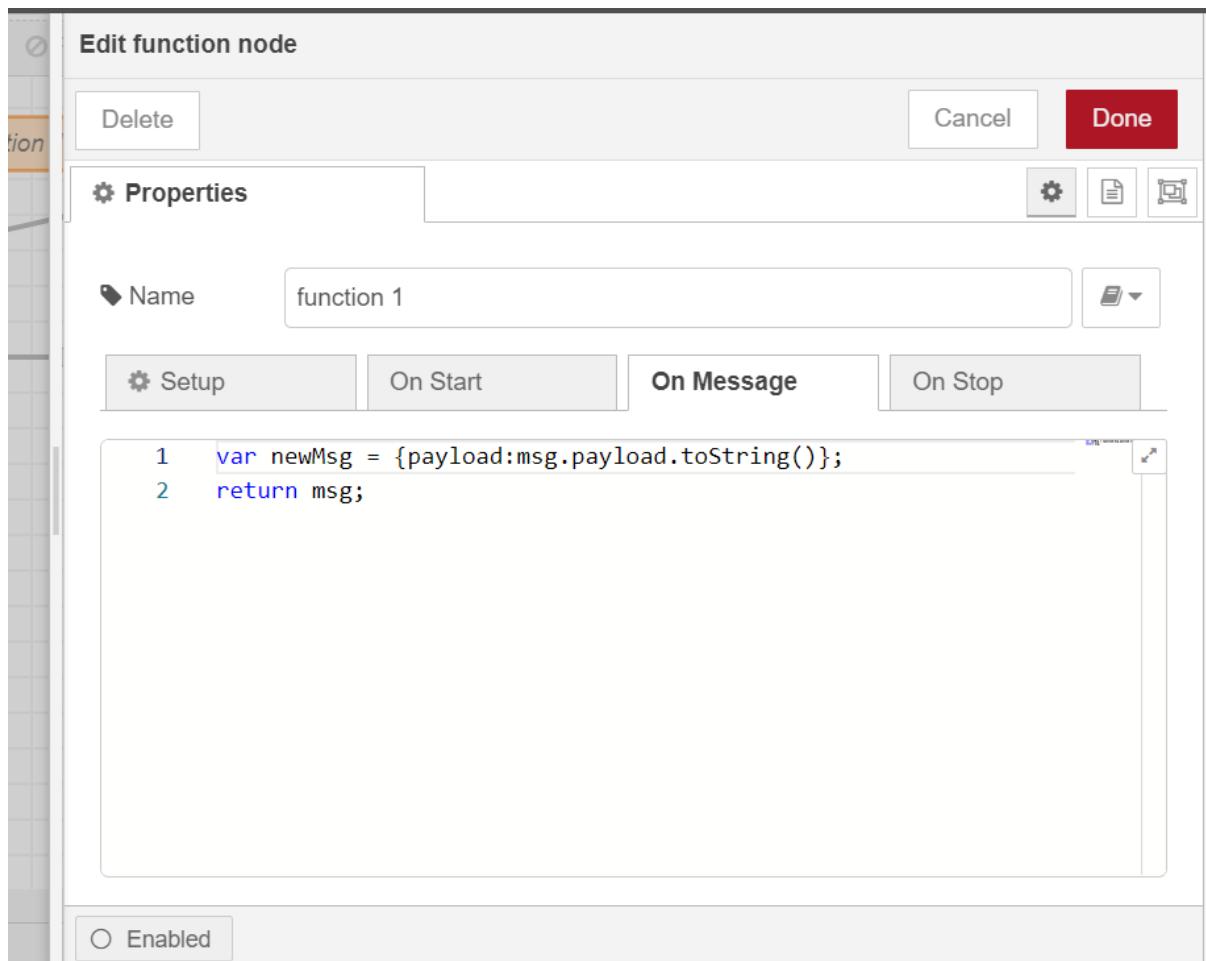
```

svrduude: ser\_open(): can't open device "\.\COM3": Access is denied.

Go to node red

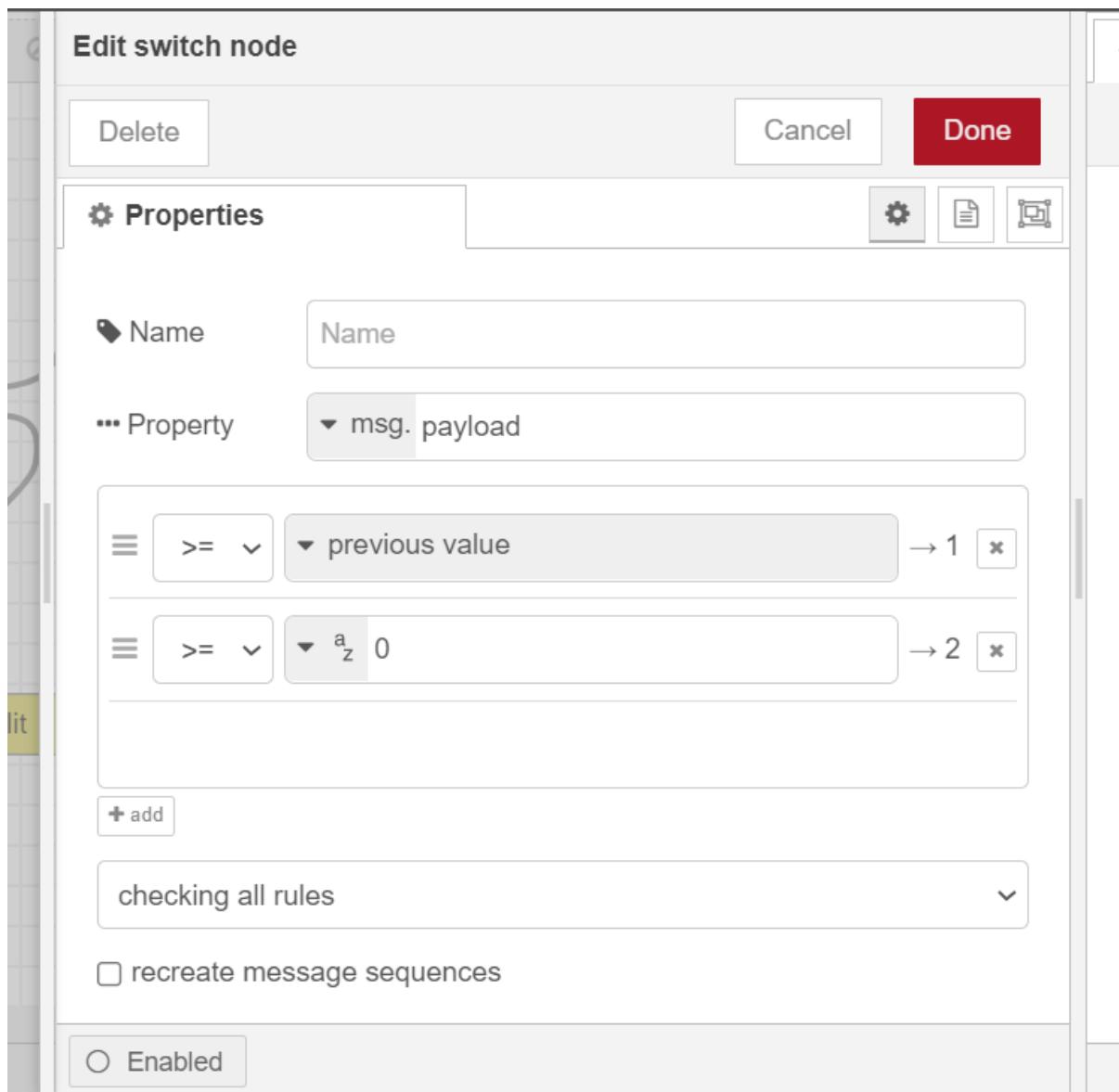


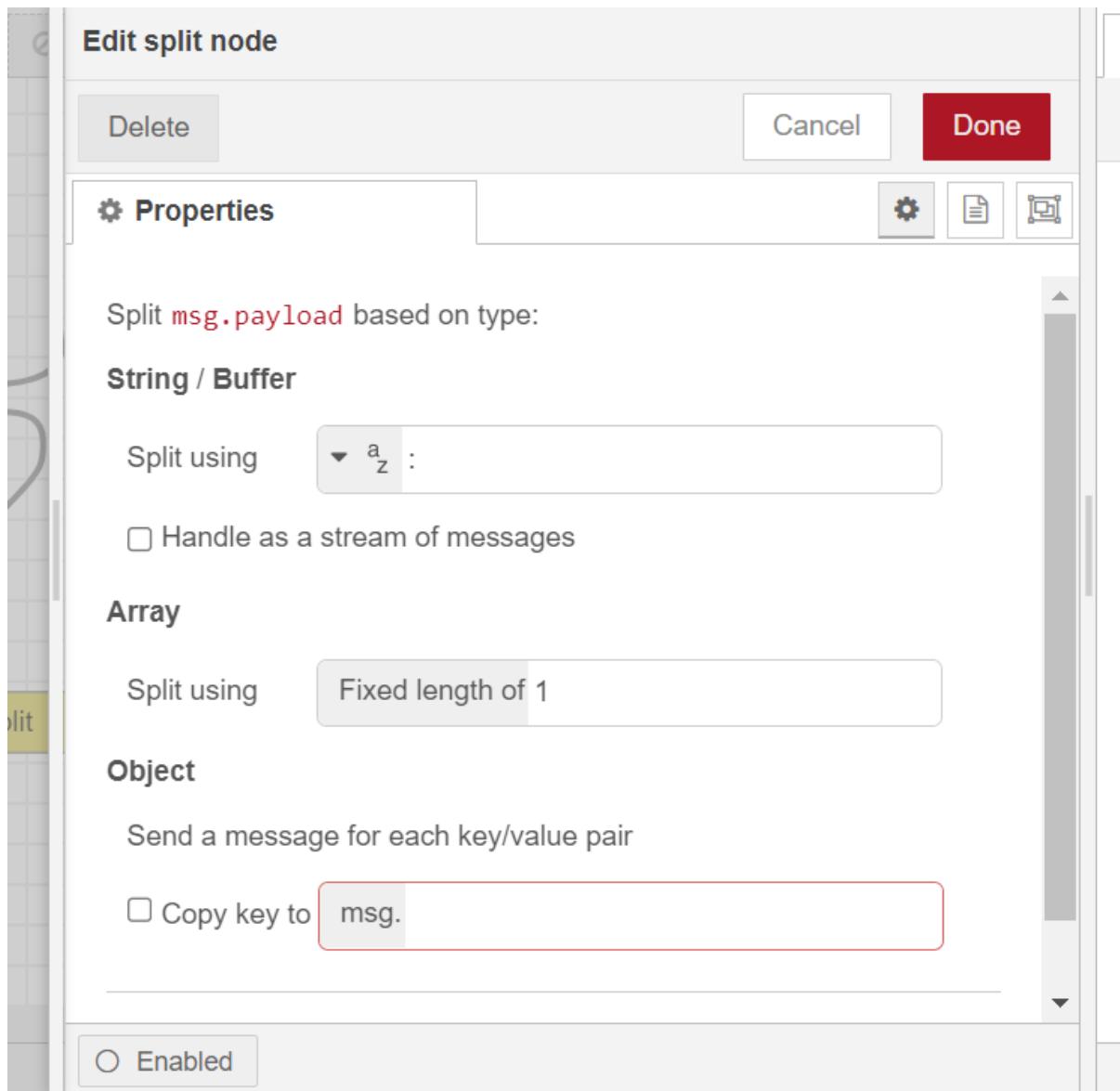


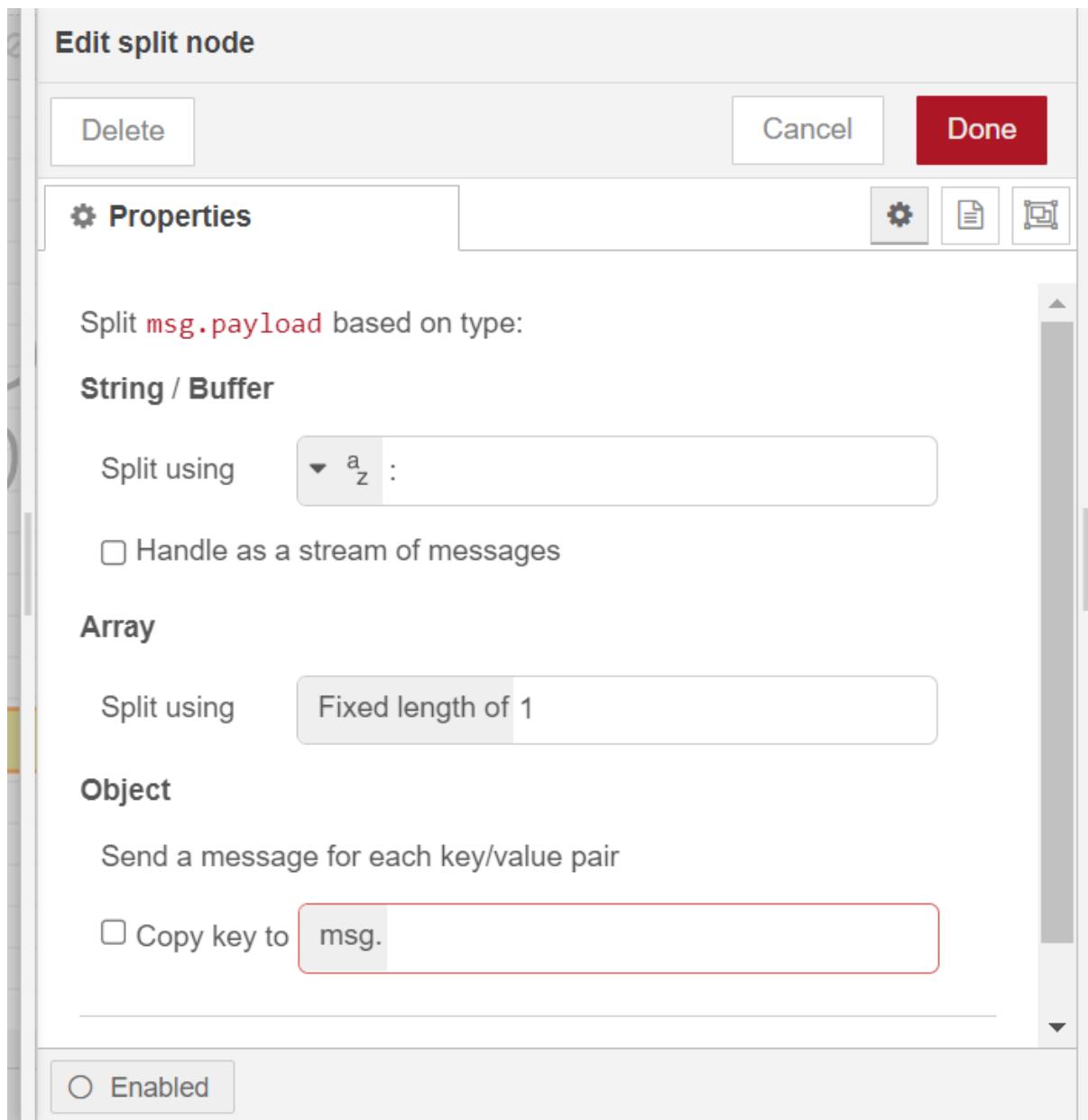


```
var newMsg = {payload:msg.payload.toString()};

return msg;
```







### Edit gauge node

[Delete](#) [Cancel](#) [Done](#)

#### Properties

[Group](#) [tempsensor] tempsenor

[Size](#) auto

[Type](#) Donut

[Label](#) gauge

[Value format](#) {{value}}

[Units](#) units

Range min  max

Colour gradient 

Enabled

### Edit chart node

[Delete](#) [Cancel](#) [Done](#)

#### Properties

Group: [tempsensor] tempsenor

Size: auto

Label: chart

Type: Line chart  enlarge points

X-axis: last 10  OR  points

X-axis Label: ▾ HH:mm:ss  as UTC

Y-axis: min  max

Legend: None  Interpolate: linear

Enabled

### Edit gauge node

Delete      Cancel      Done

#### Properties

Group: [tempsensor] tempsenor

Size: auto

Type: Donut

Label: gauge

Value format: {{value}}

Units: units

Range: min 0 max 100

Colour gradient: 

Enabled:

### Edit chart node

Delete      Cancel      Done

#### Properties

Group: [tempsensor] tempsenor

Size: auto

Label: chart

Type: Line chart  enlarge points

X-axis: last 10 second OR 1000 points

X-axis Label: HH:mm:ss  as UTC

Y-axis: min max

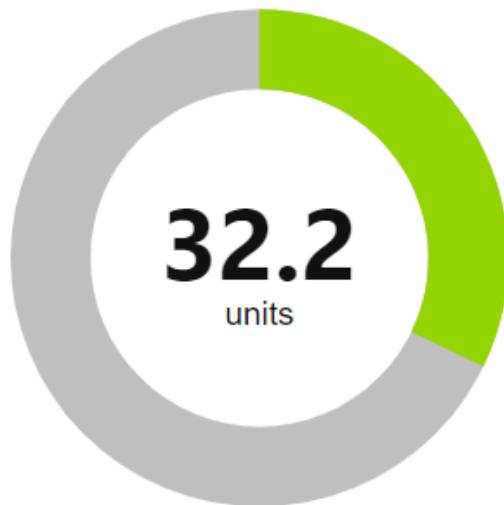
Legend: None Interpolate linear

Enabled

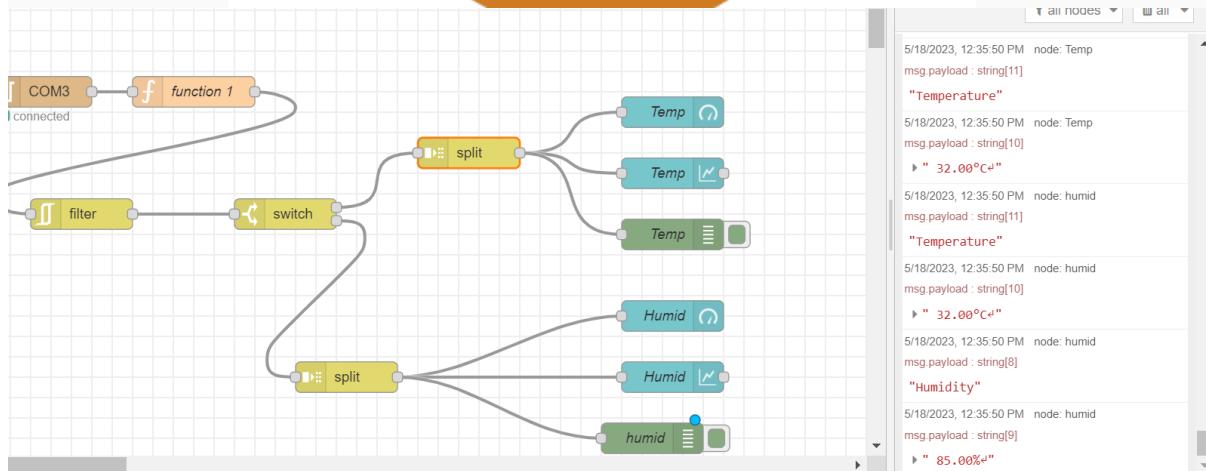
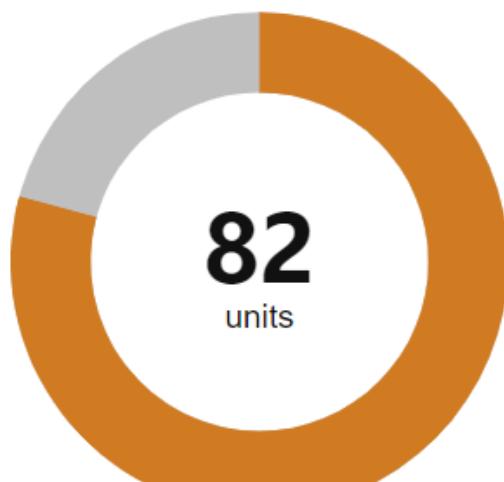
Close the com output in arduino

Deploy and ui

gauge



gauge



**chart**



**chart**

