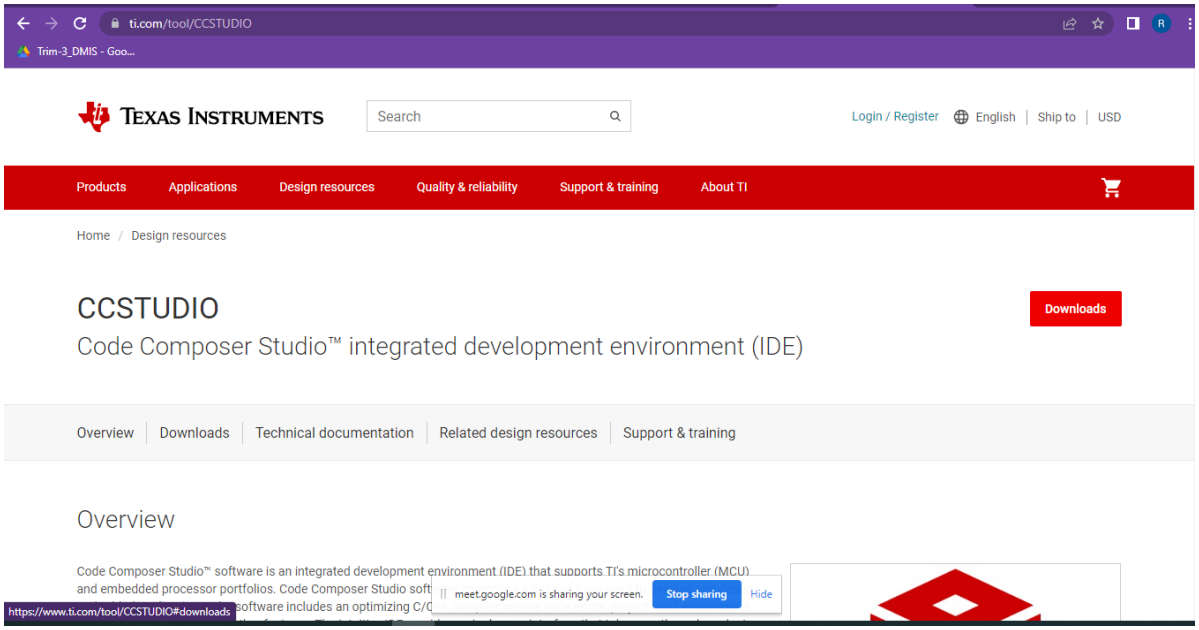
INDEX

Practical 1

CCS Installation, User Interface tour, Project creation, Compiling and debugging with MSP432

1) CCS Installation

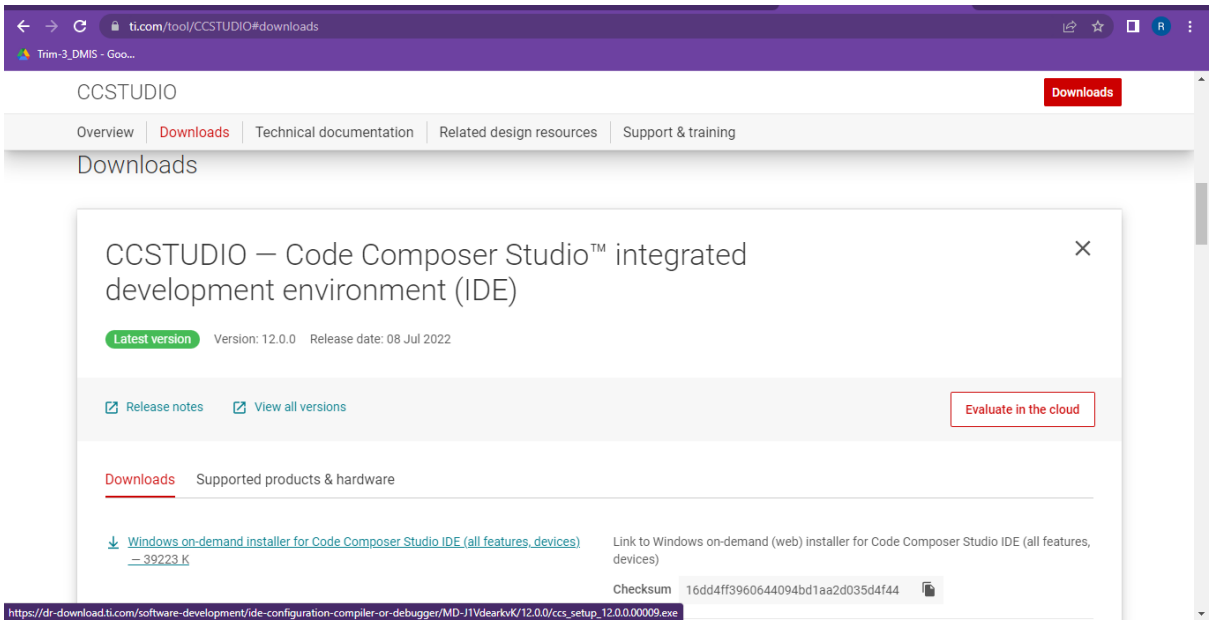
Search for CCSTUDIO



Code Composer Studio software is an integrated development environment (IDE) that supports TI's microcontroller (MCU) and embedded processor portfolios.

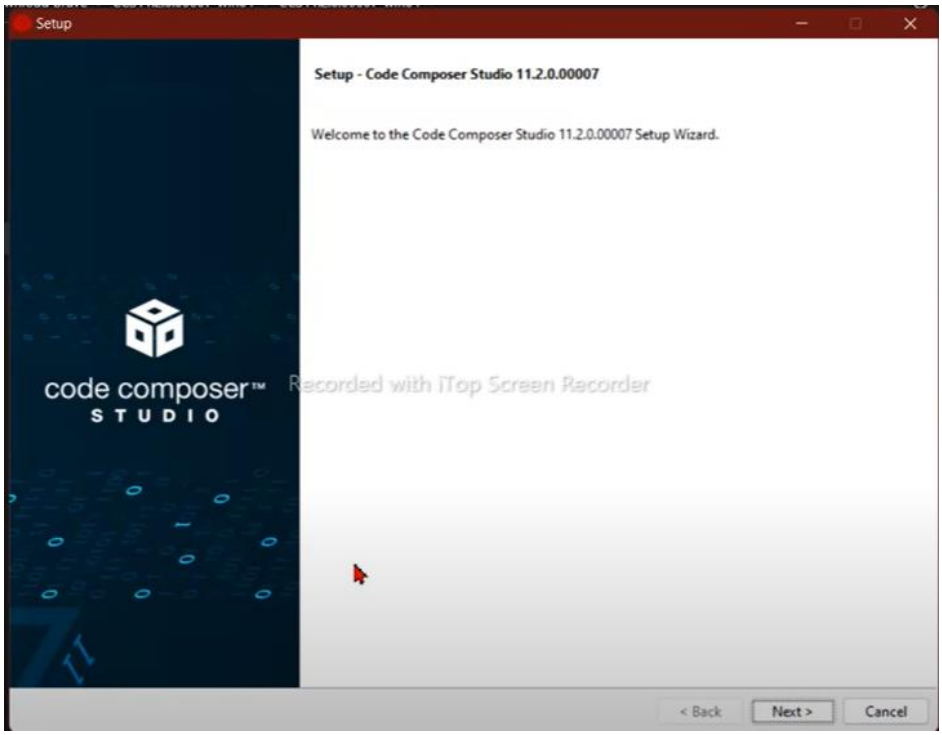
Code Composer Studio software comprises a suite of tools used to develop and debug embedded applications. The software includes an optimizing C/C++ compiler, source code editor, project build environment, debugger, profiler and many other features.

Download the Latest version

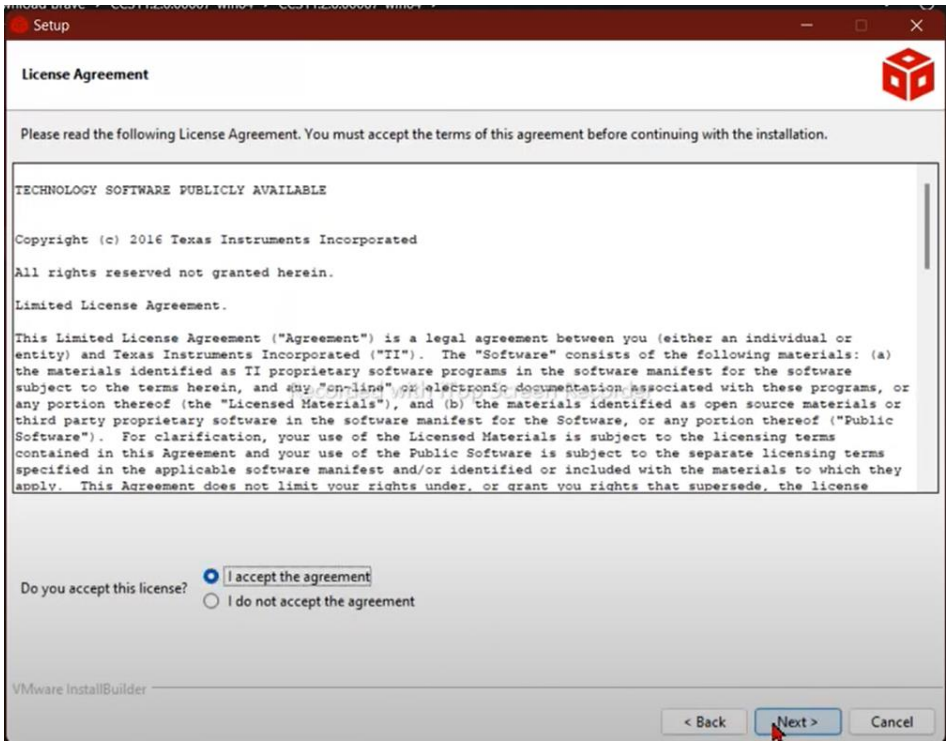


Once download is complete

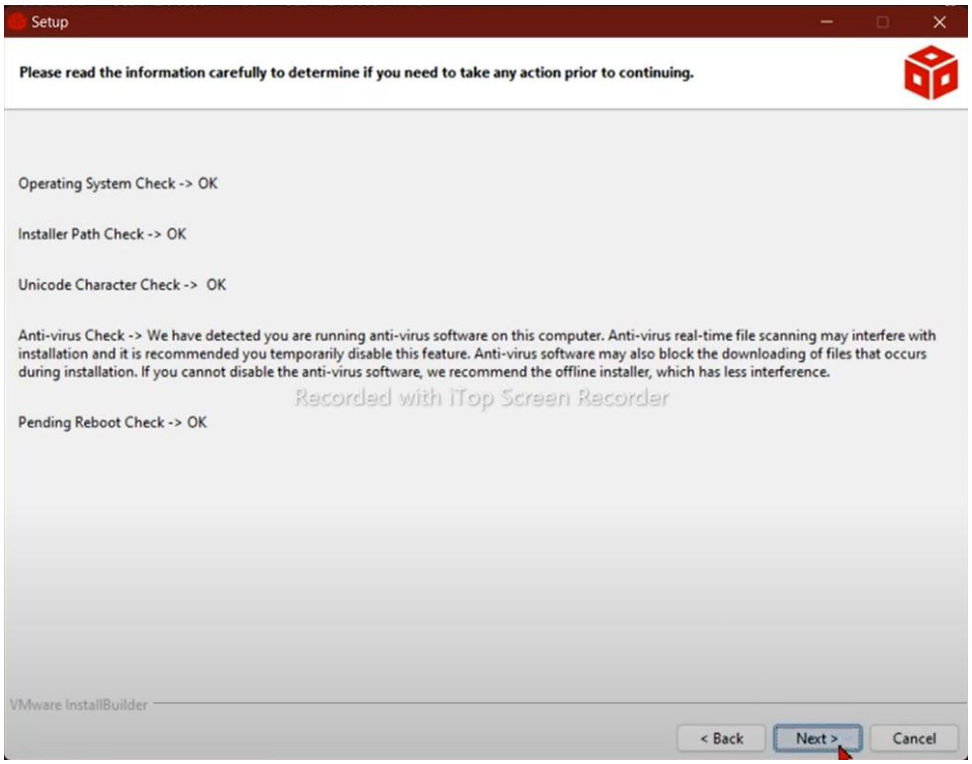
Click on Next



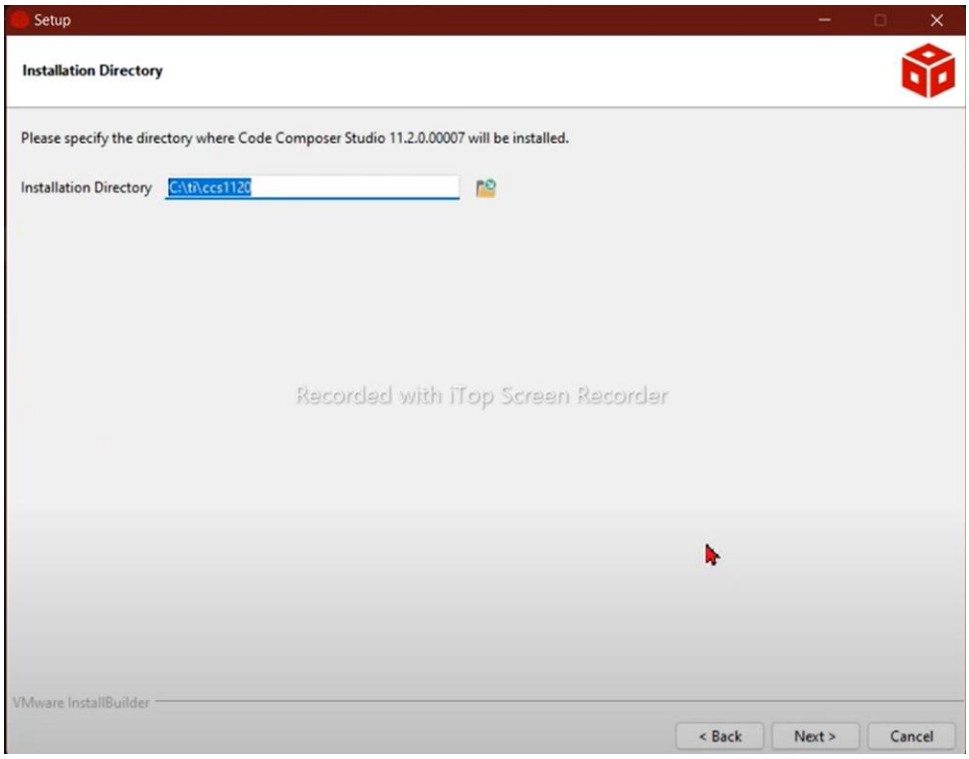
Accept Terms And condition



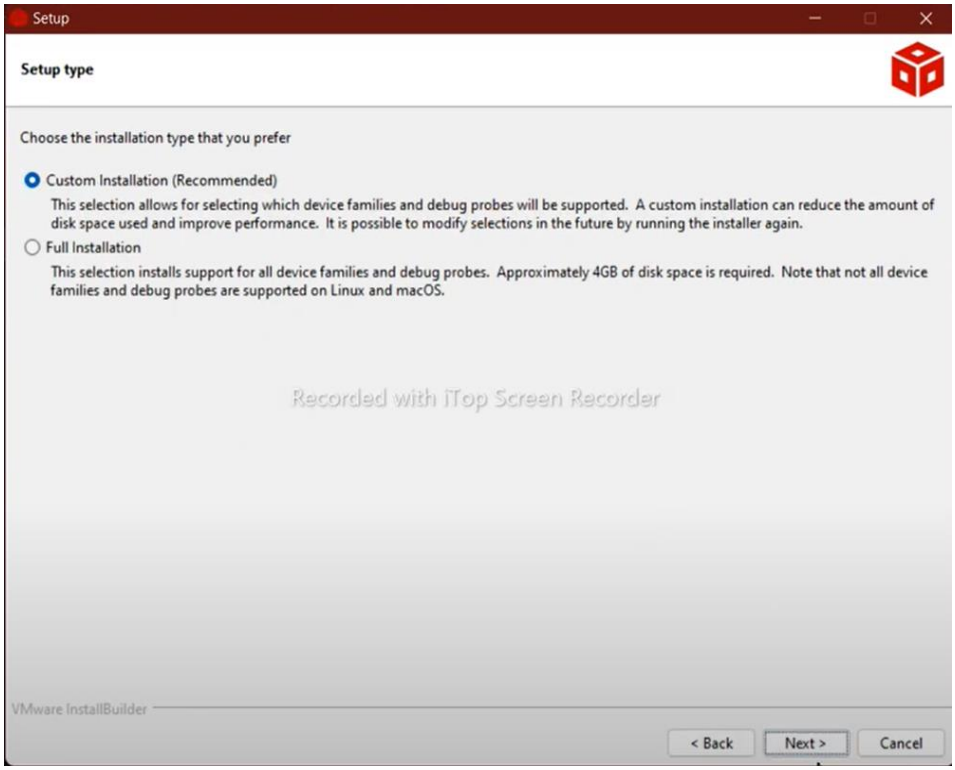
Click Next



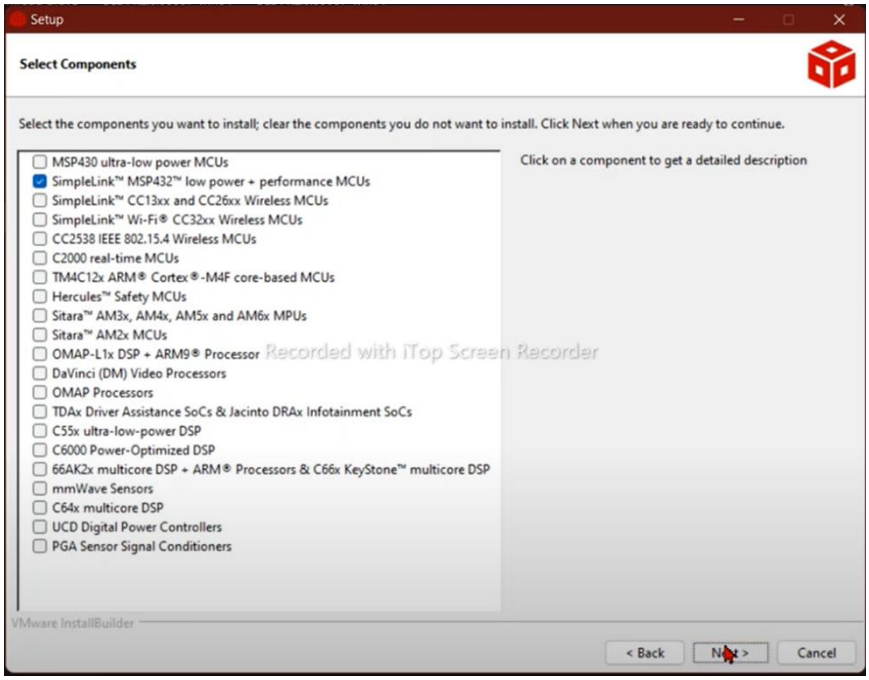
Select Directory Where Code Composer Studio Will be install



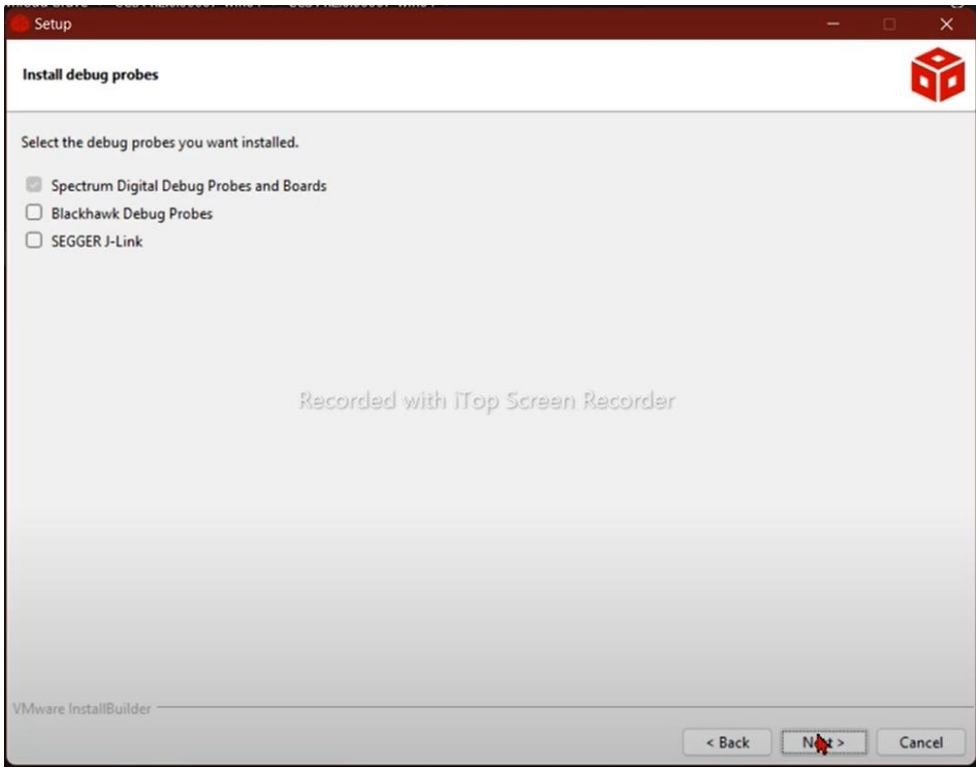
Click On Custom Installation

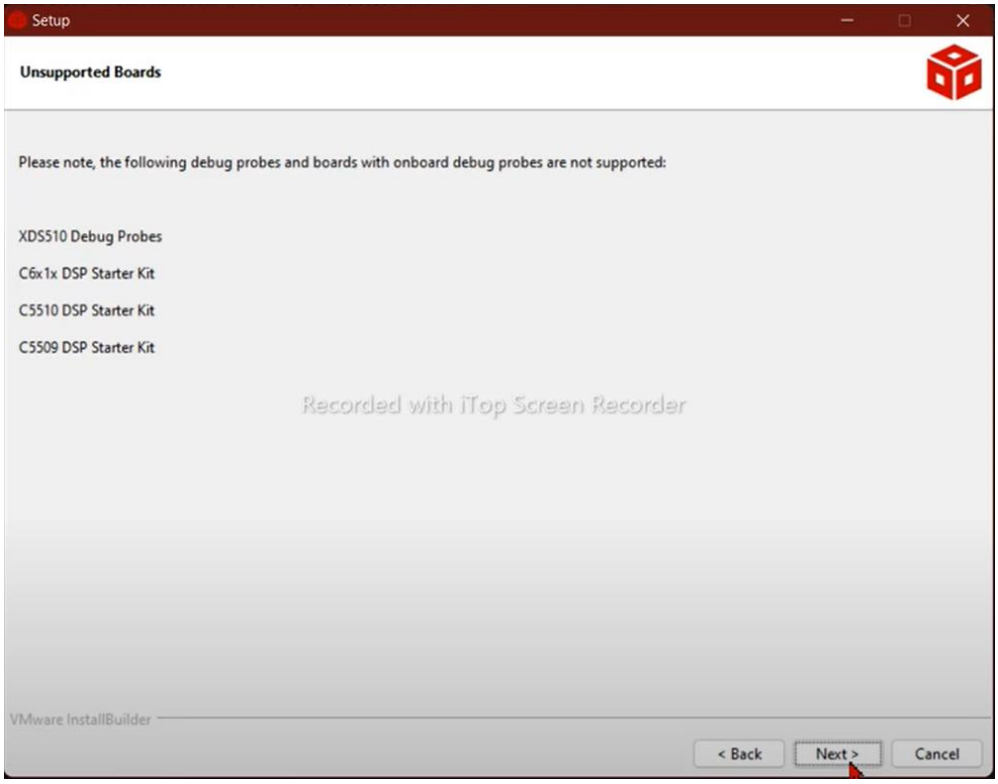


Select Simplelink MSP432 Low power+ performance MCUs

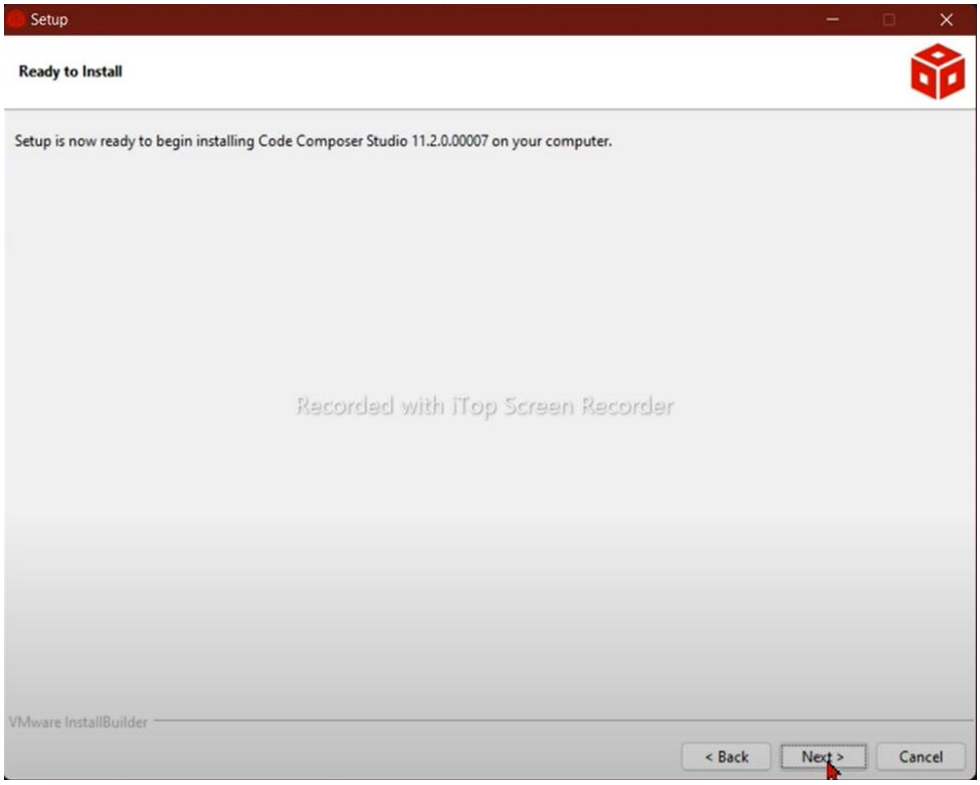


Click on Next

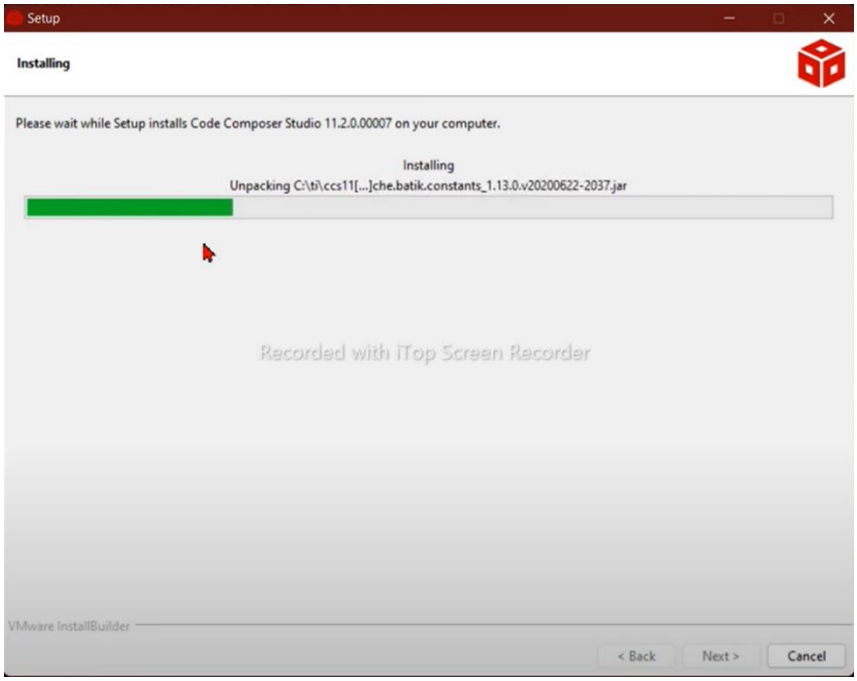


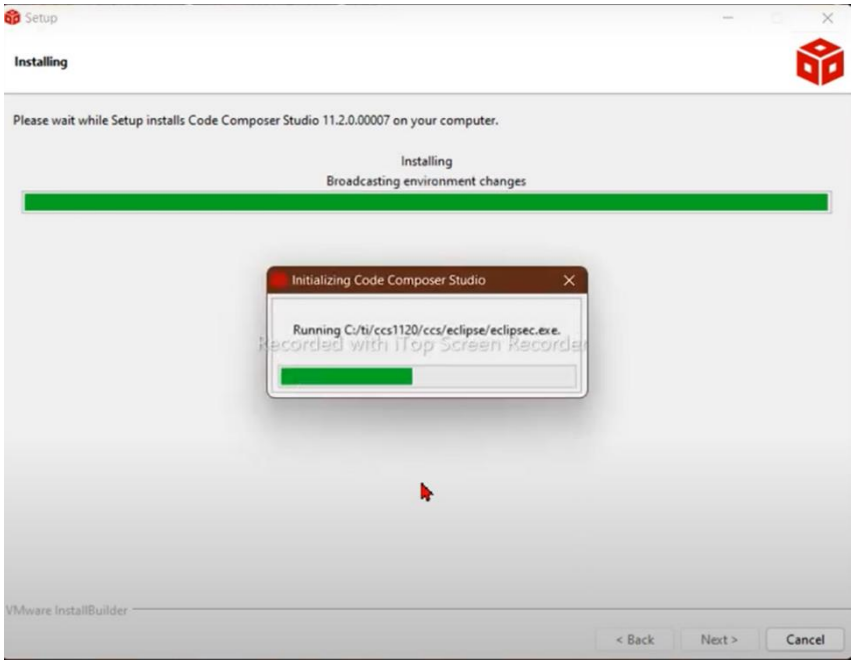


Ready to install and click on Next

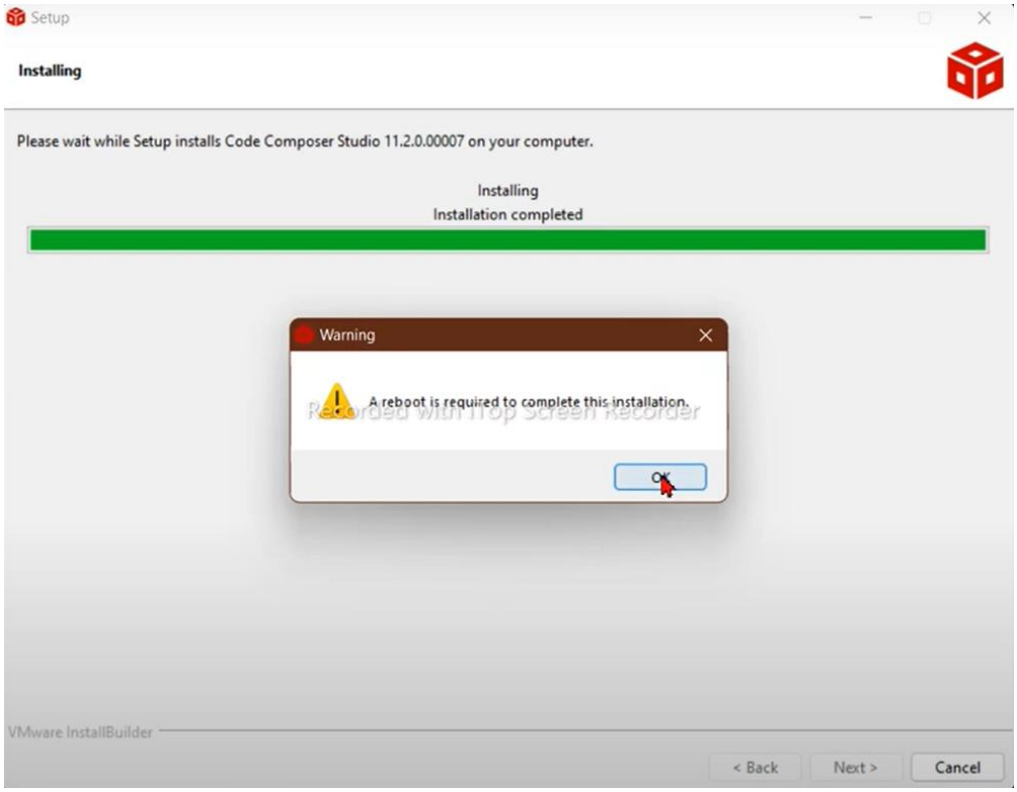


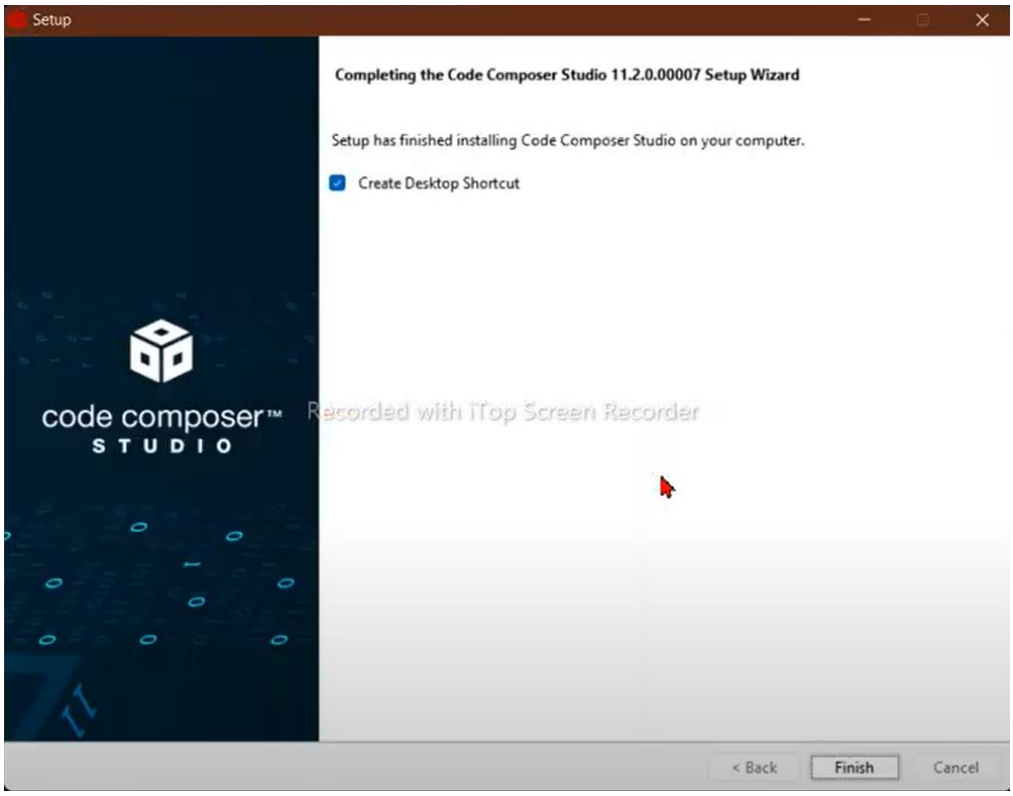
Installing



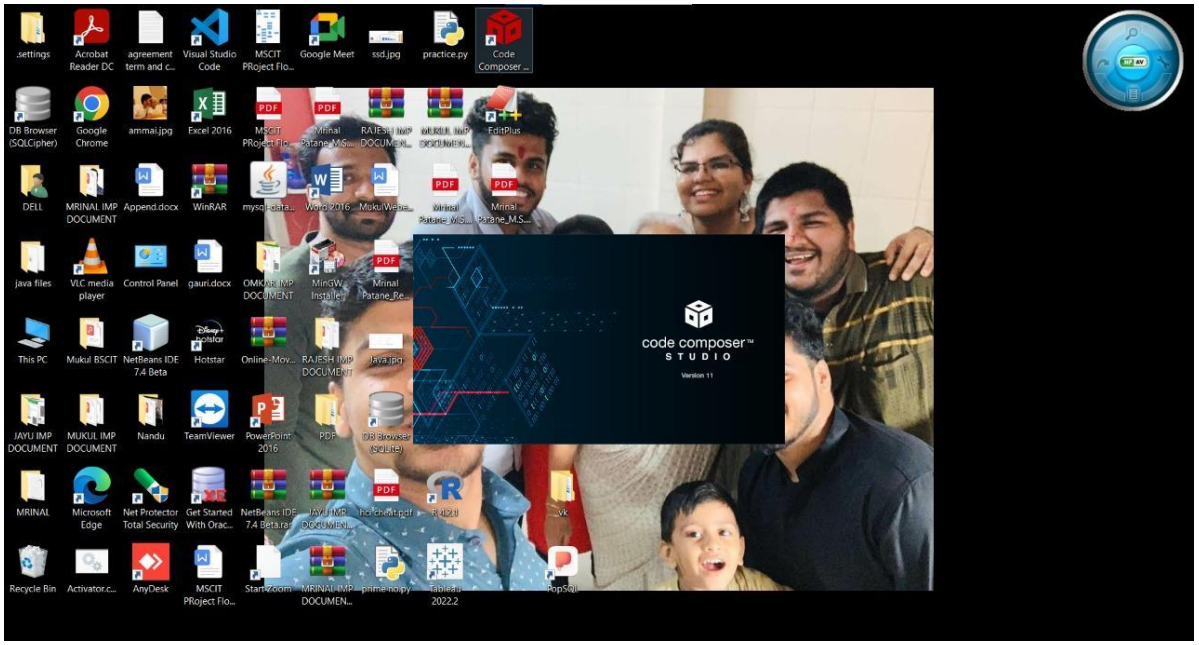


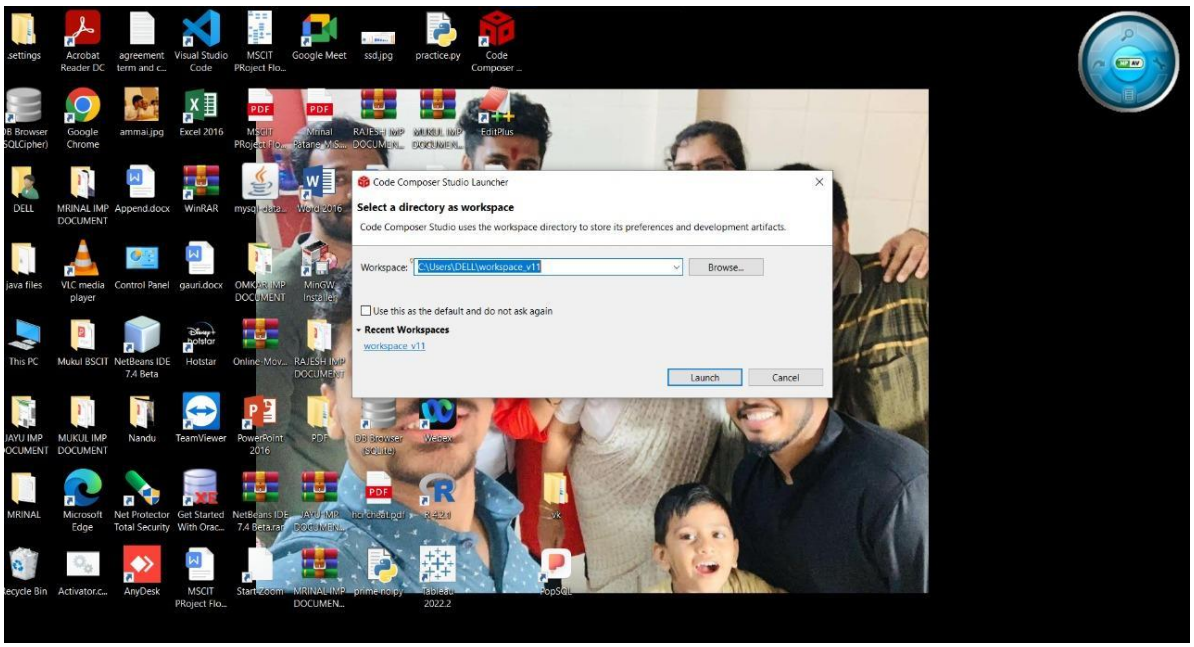
Restart System Once installation completes


Setup
Installing
Please wait while Setup installs Code Composer Studio 11.2.0.00007 on your computer.
Installing
Installation completed
Warning
A reboot is required to complete this installation.
OK
VMware InstallBuilder
< Back
Next >
Cancel

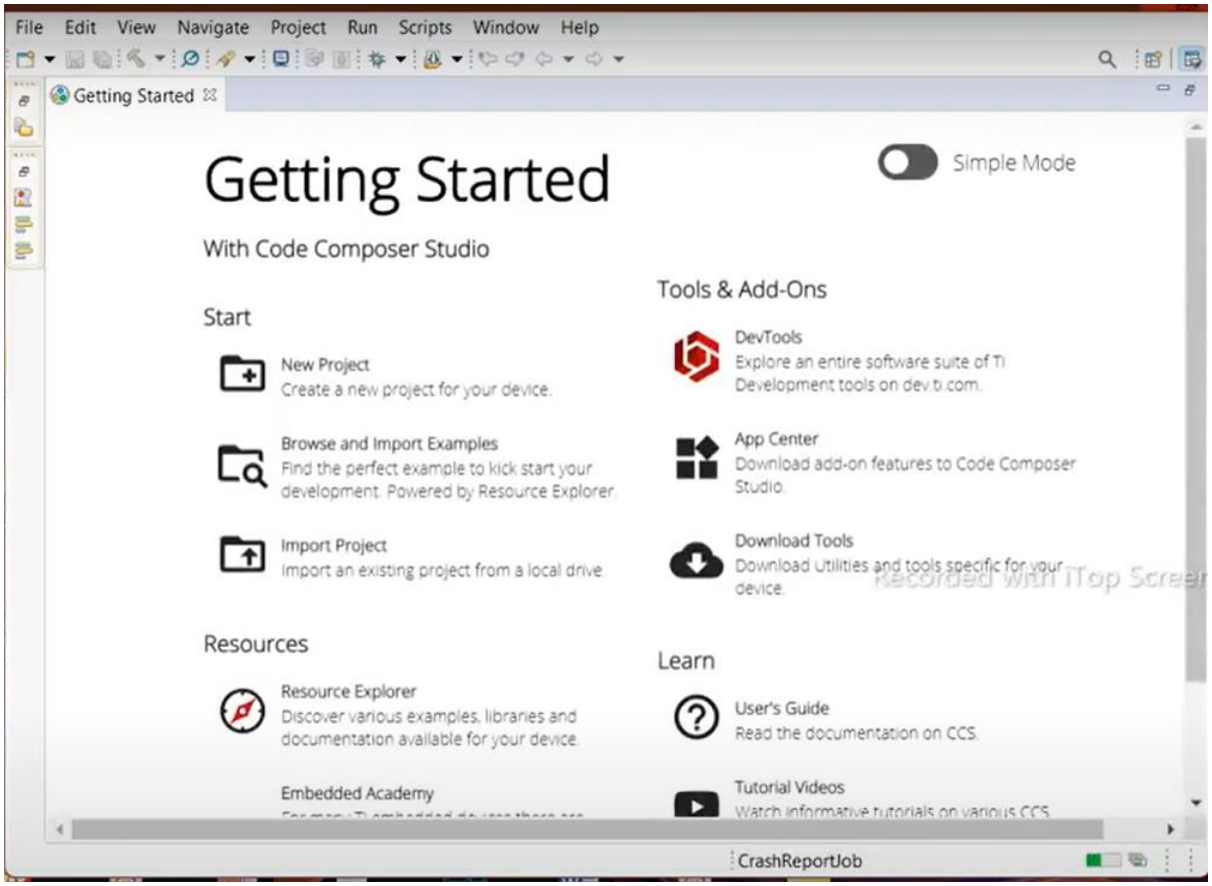

Setup
Completing the Code Composer Studio 11.2.0.00007 Setup Wizard
Setup has finished installing Code Composer Studio on your computer.
Create Desktop Shortcut
code composer™
STUDIO
< Back
Finish
Cancel

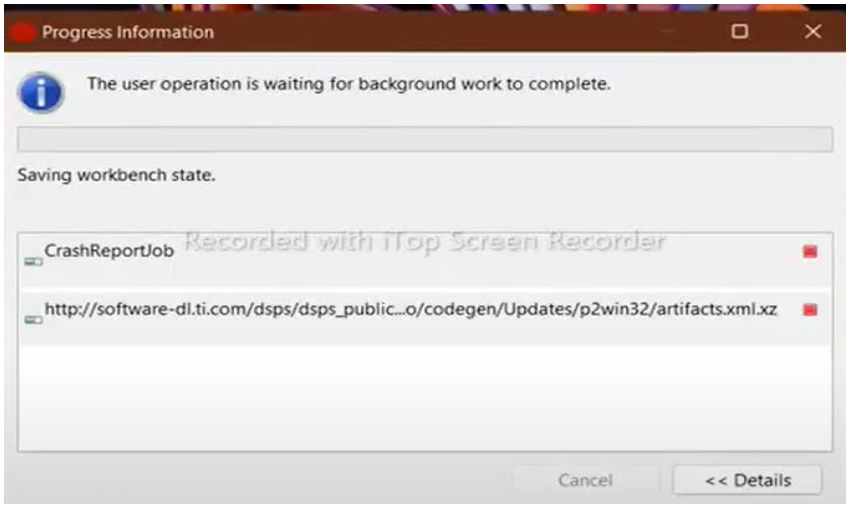
Open CCD





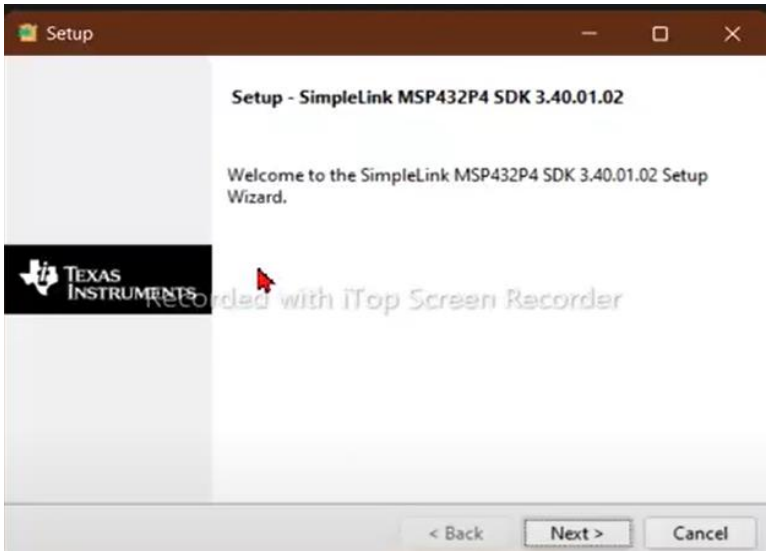
First Page of CCD



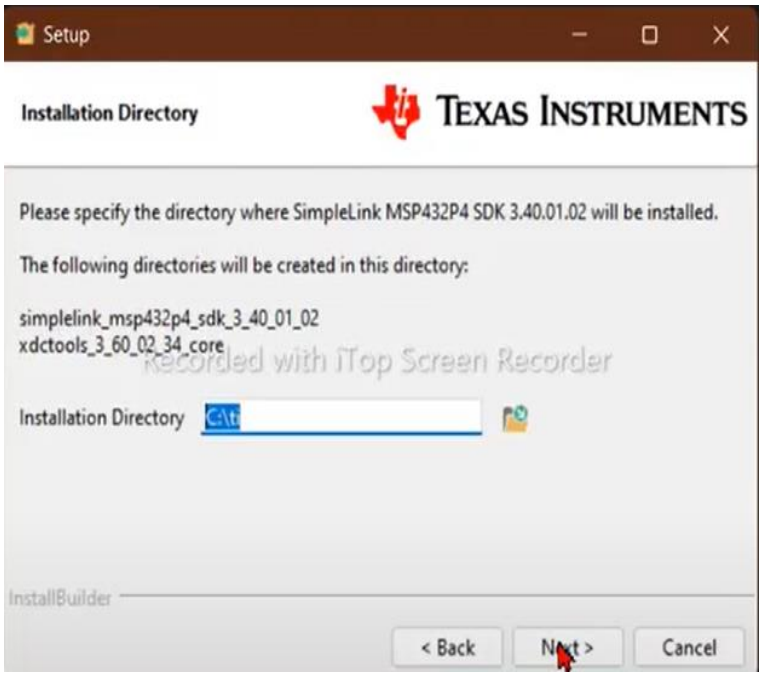


With this, the Code Composer Studio is all set to work.

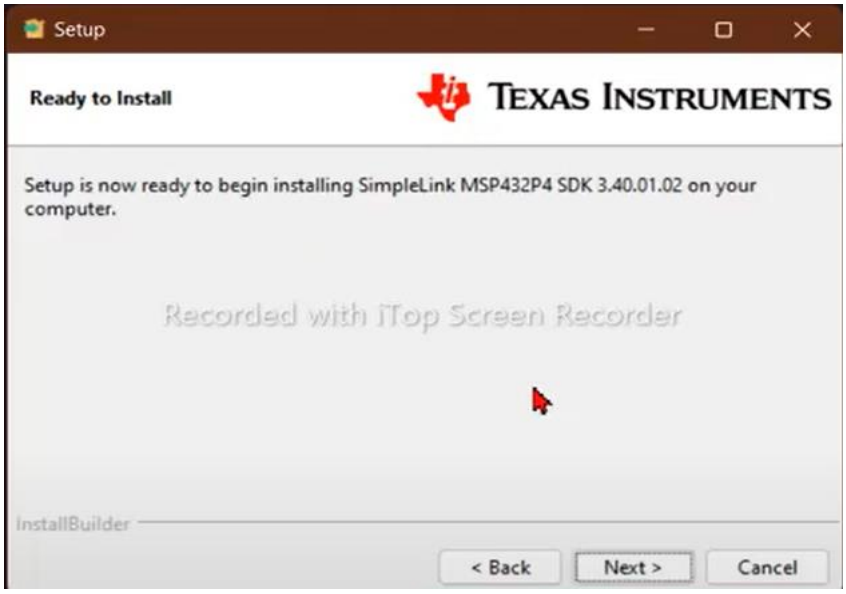
Install SimpleLink MSP432P4

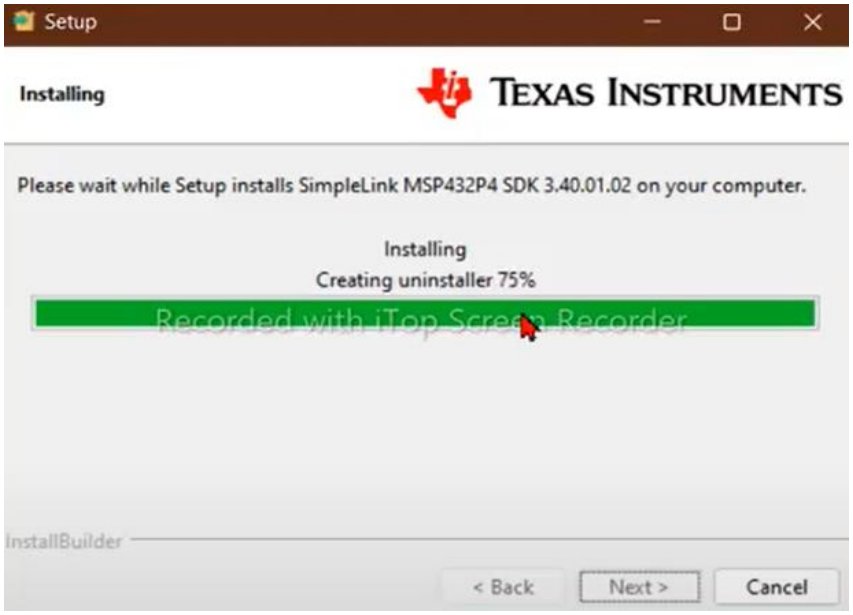


Select installation Directory (By default Select C)

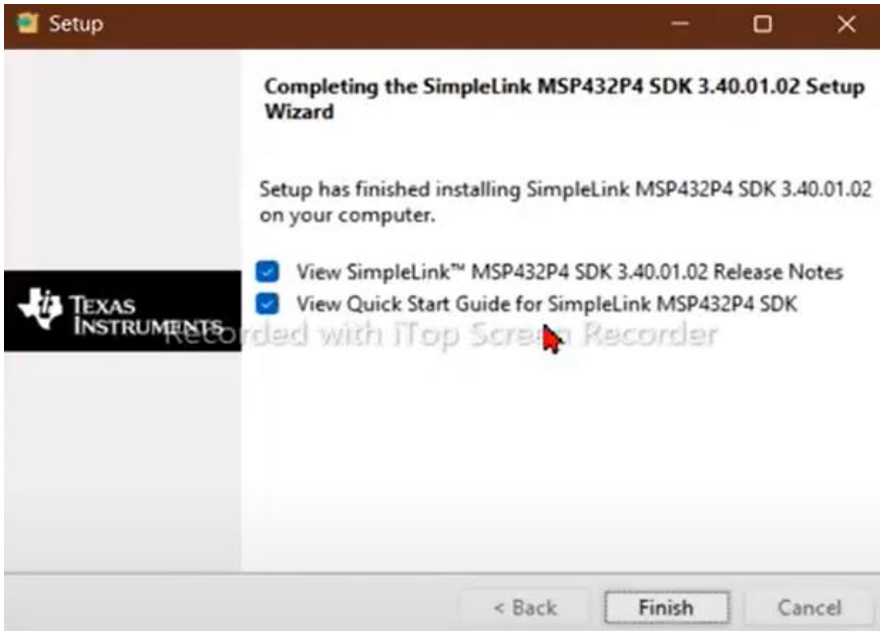


Click On Next

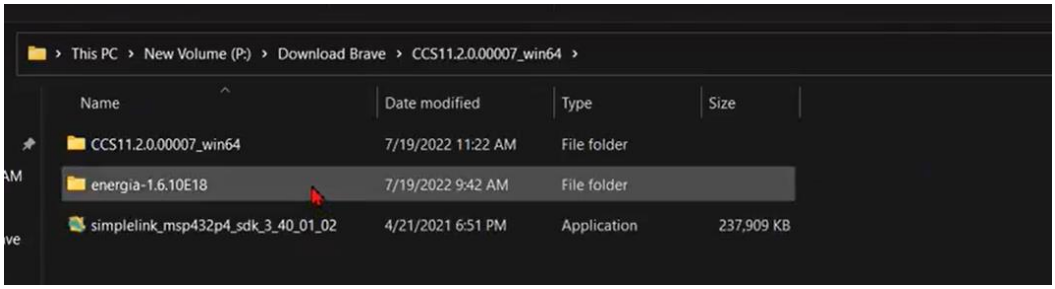




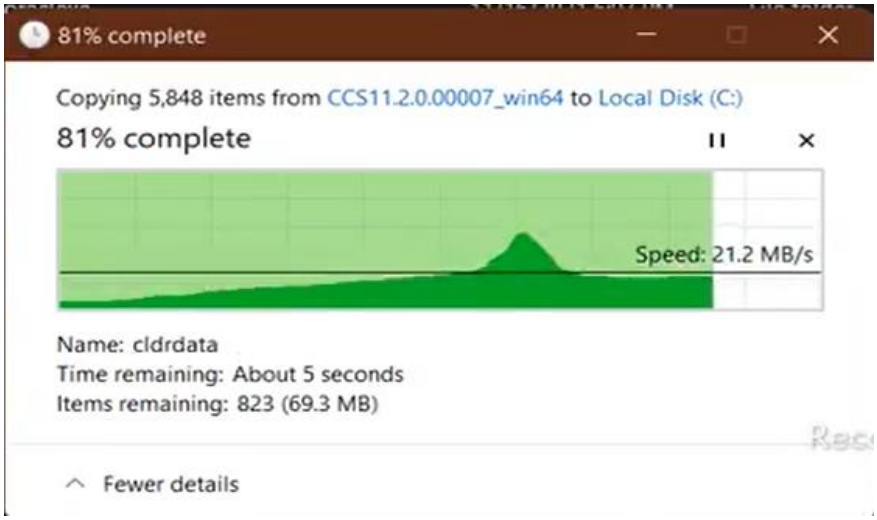
Uncheck both Checkbox And click Next



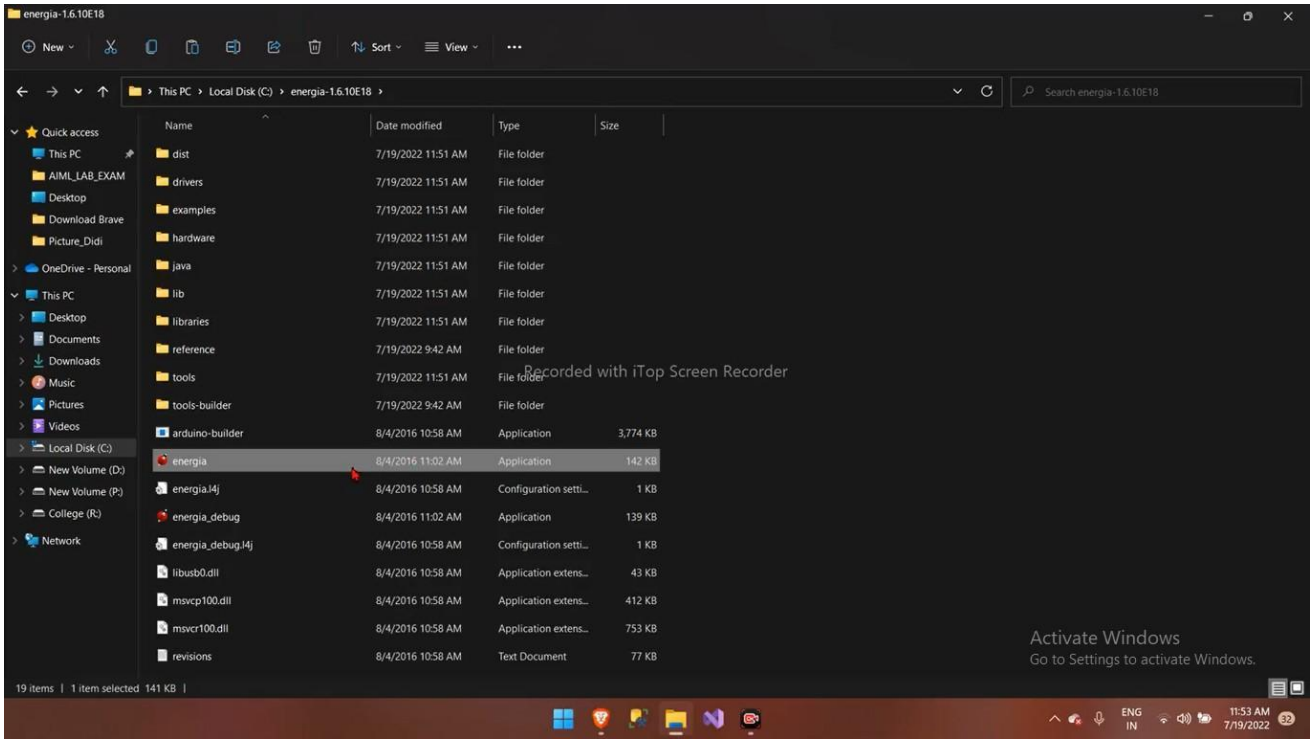
Energic Installation With board manger details



Copy the folder into C drive

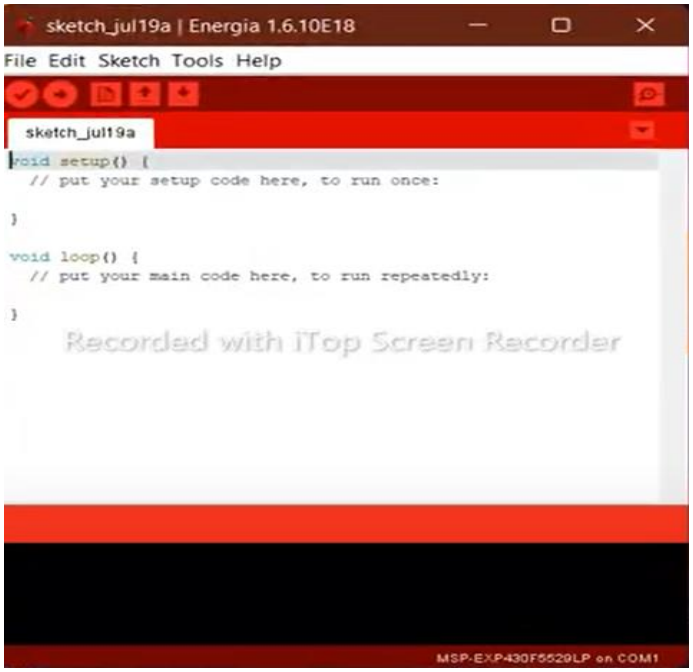


Launch the energia.exe file

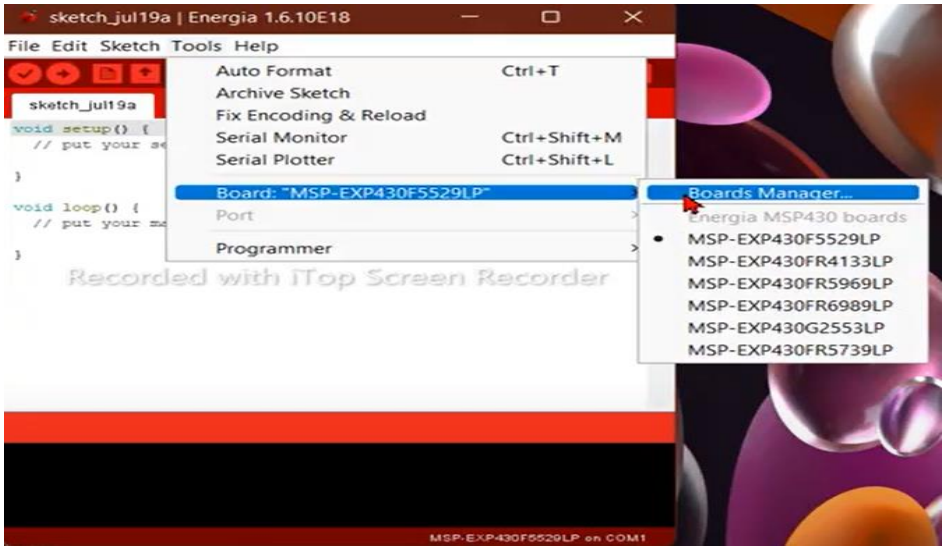

energia-1.6.10E18
New
Sort
View
This PC › Local Disk (C:) › energia-1.6.10E18 ›
Search energia-1.6.10E18
Quick access
This PC
AIML_LAB_EXAM
Desktop
Download Brave
Picture_Didi
OneDrive - Personal
This PC
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
New Volume (D:)
New Volume (P:)
College (R:)
Network
Name
Date modified
Type
Size
dist 7/19/2022 11:51 AM File folder
drivers 7/19/2022 11:51 AM File folder
examples 7/19/2022 11:51 AM File folder
hardware 7/19/2022 11:51 AM File folder
java 7/19/2022 11:51 AM File folder
lib 7/19/2022 11:51 AM File folder
libraries 7/19/2022 11:51 AM File folder
reference 7/19/2022 9:42 AM File folder
tools 7/19/2022 11:51 AM File folder
tools-builder 7/19/2022 9:42 AM File folder
arduino-builder 8/4/2016 10:58 AM Application 3,774 KB
energia 8/4/2016 11:02 AM Application 142 KB
energia.l4j 8/4/2016 10:58 AM Configuration setti... 1 KB
energia_debug 8/4/2016 11:02 AM Application 139 KB
energia_debug.l4j 8/4/2016 10:58 AM Configuration setti... 1 KB
libusb0.dll 8/4/2016 10:58 AM Application extens... 43 KB
msvcp100.dll 8/4/2016 10:58 AM Application extens... 412 KB
msvcr100.dll 8/4/2016 10:58 AM Application extens... 753 KB
revisions 8/4/2016 10:58 AM Text Document 77 KB
Recorded with iTop Screen Recorder
Activate Windows
Go to Settings to activate Windows.
19 items | 1 item selected 141 KB |
ENG
IN
11:53 AM
7/19/2022

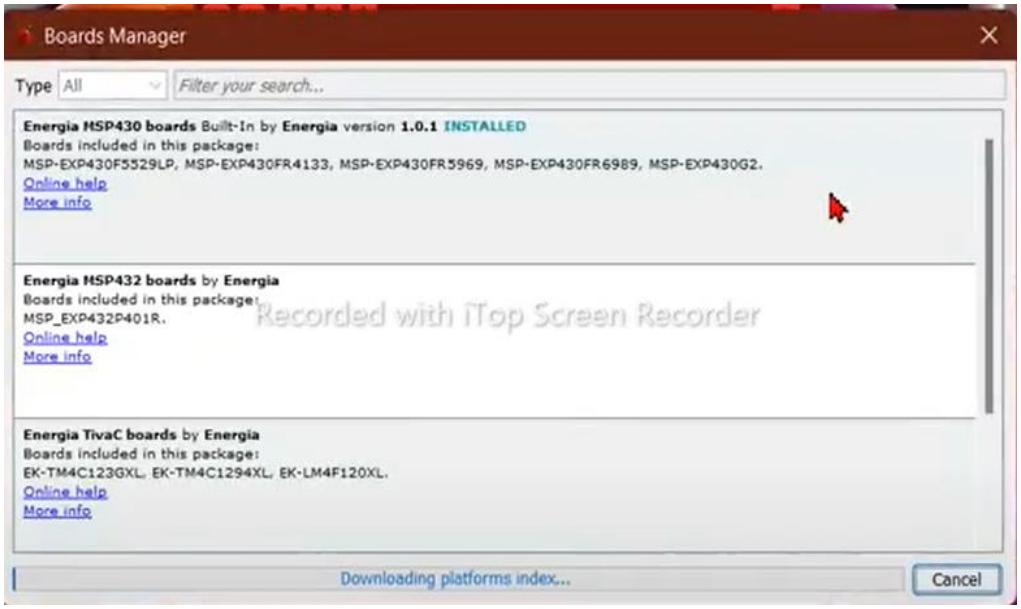

EnergiaMT
Recorded with iTop Screen Recorder
EnergiaMT is a modified version of the Arduino/
Genuino IDE for the Texas Instruments MSP430,
MSP432, Tiva C and CC3200 LaunchPad boards.
This software is not supported by the Arduino LLC.

Main Page of Energia



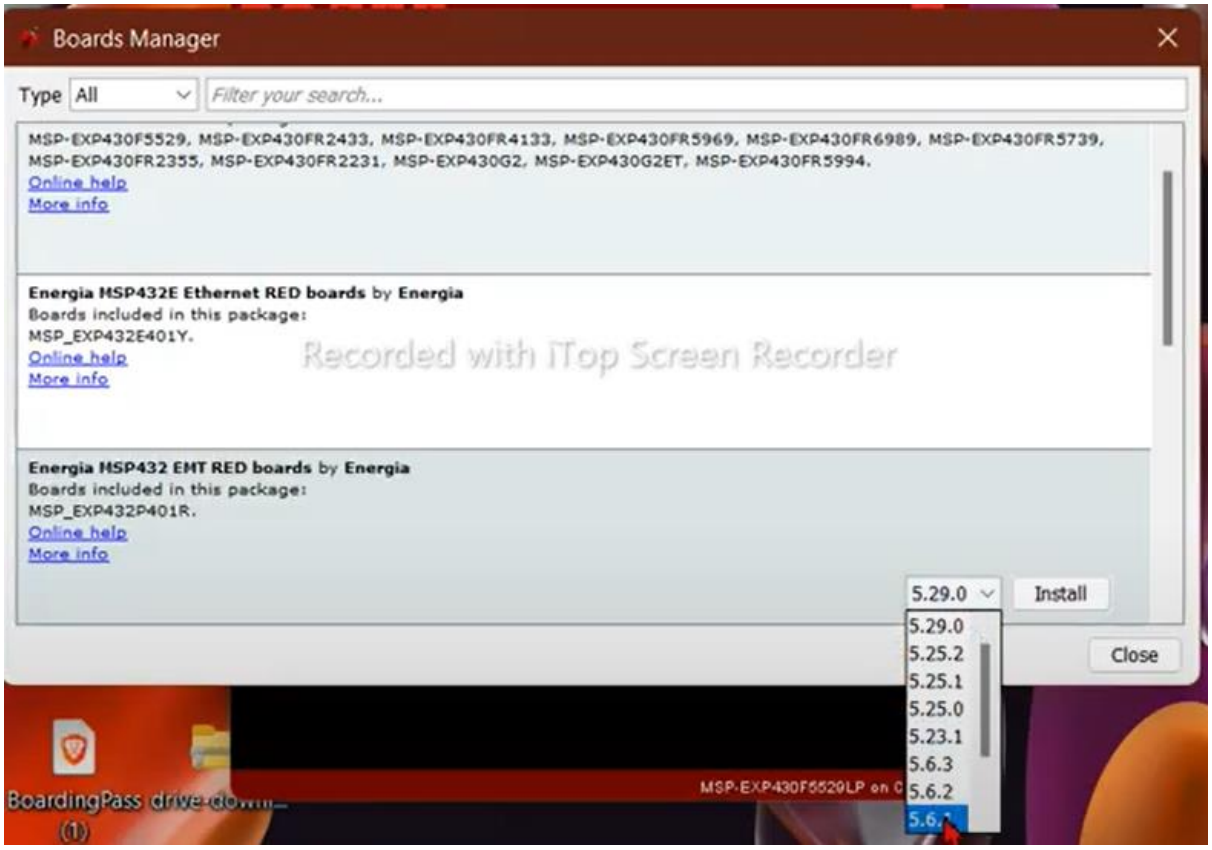
Go to tools and Select Board -> Board Manager



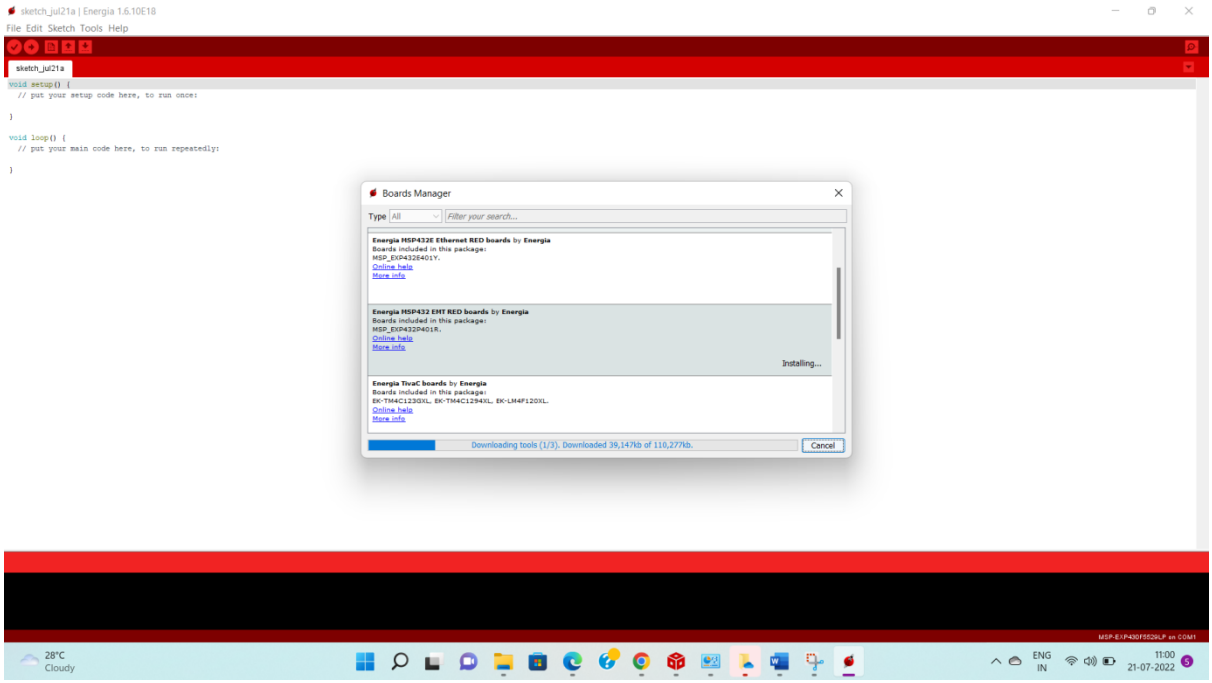


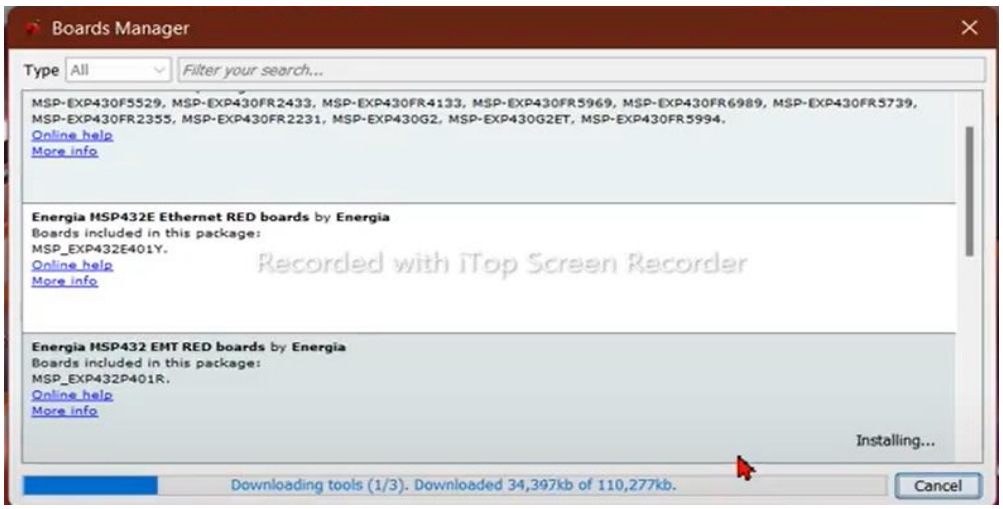
MSP432 SDK installation

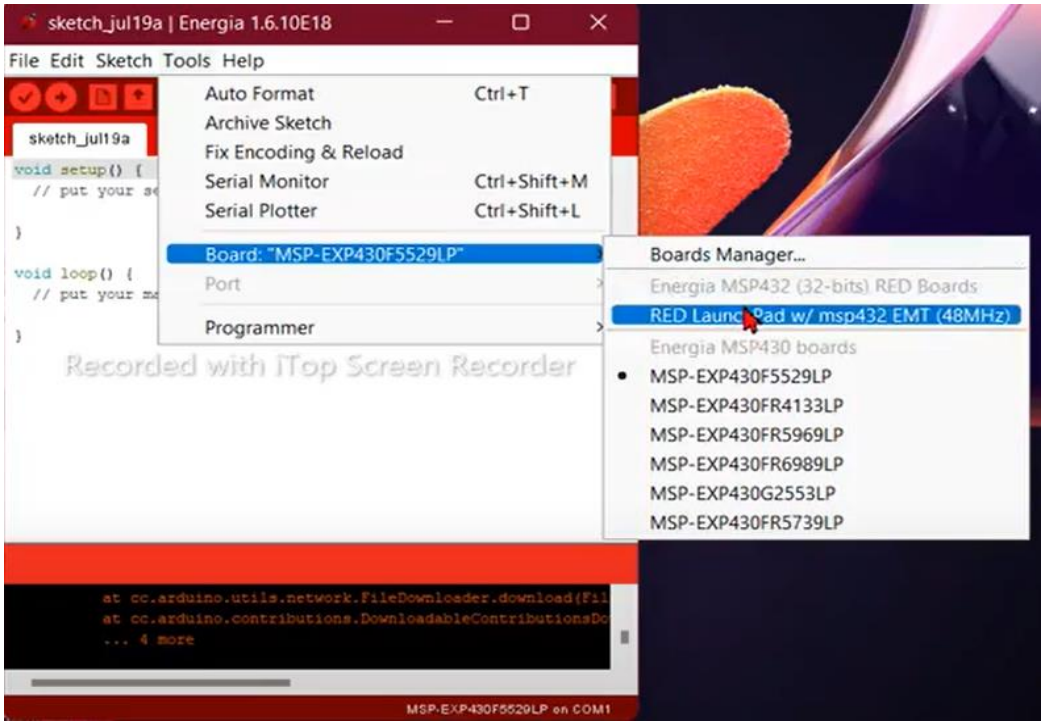
Install MSP432 EMT RED Boards by Energia 5.6.1



Select 5.6.1 version on click on install


sketch_jul21a | Energia 1.6.10E18
File Edit Sketch Tools Help
sketch_jul21a
void setup() {
// put your setup code here, to run once:
}
void loop() {
// put your main code here, to run repeatedly:
}
Boards Manager
Type All
Filter your search...
Energia MSP432E Ethernet RED boards by Energia
Boards included in this package:
MSP_EXP432E401Y.
Online help
More info
Energia MSP432 EMT RED boards by Energia
Boards included in this package:
MSP_EXP432P401R.
Online help
More info
Installing...
Energia TivaC boards by Energia
Boards included in this package:
EK-TM4C123GXL, EK-TM4C1294XL, EK-LM4F120XL.
Online help
More info
Downloading tools (1/3). Downloaded 39,147kb of 110,277kb.
Cancel
28°C
Cloudy
ENG
IN
11:00
21-07-2022


Boards Manager
Type All
Filter your search...
MSP-EXP430F5529, MSP-EXP430FR2433, MSP-EXP430FR4133, MSP-EXP430FR5969, MSP-EXP430FR6989, MSP-EXP430FR5739, MSP-EXP430FR2355, MSP-EXP430FR2231, MSP-EXP430G2, MSP-EXP430G2ET, MSP-EXP430FR5994.
Online help
More info
Energia MSP432E Ethernet RED boards by Energia
Boards included in this package:
MSP_EXP432E401Y.
Online help
More info
Recorded with iTop Screen Recorder
Energia MSP432 EMT RED boards by Energia
Boards included in this package:
MSP_EXP432P401R.
Online help
More info
Installing...
Downloading tools (1/3). Downloaded 34,397kb of 110,277kb.
Cancel

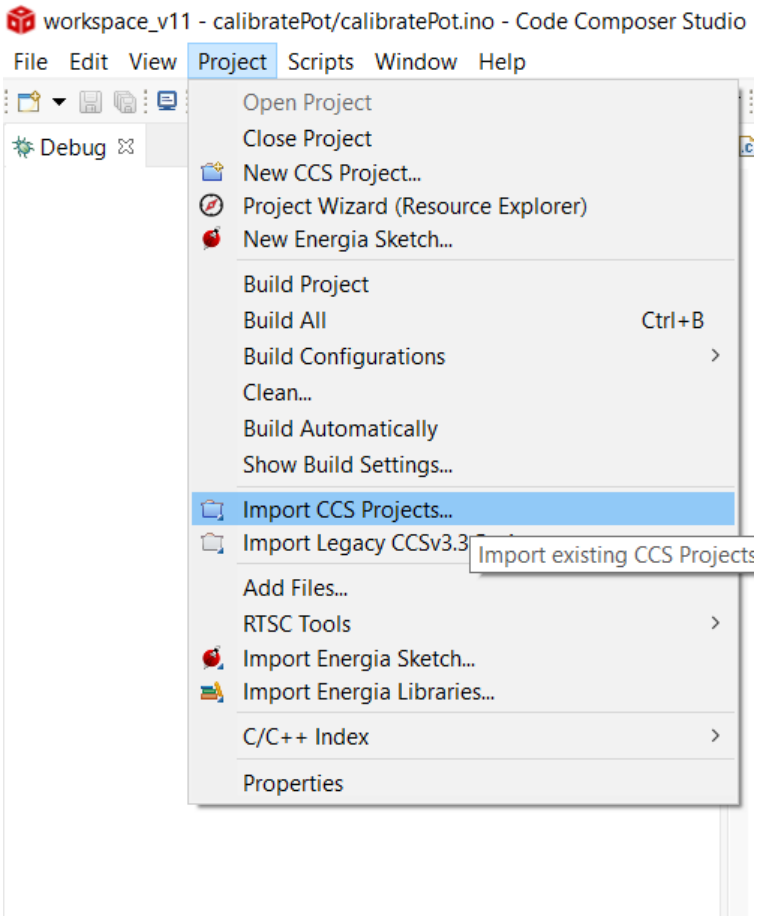


This completes the installation of all the components.

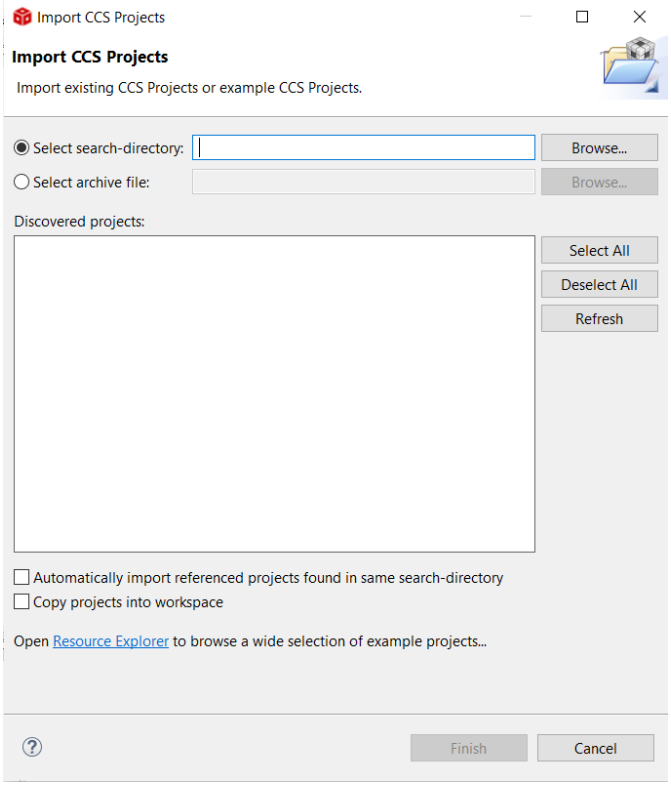
Practical 2

Hello world of Embedded Systems

Open Code Composer Studio (CCS). Under Project Menu Select Import CCS Projects

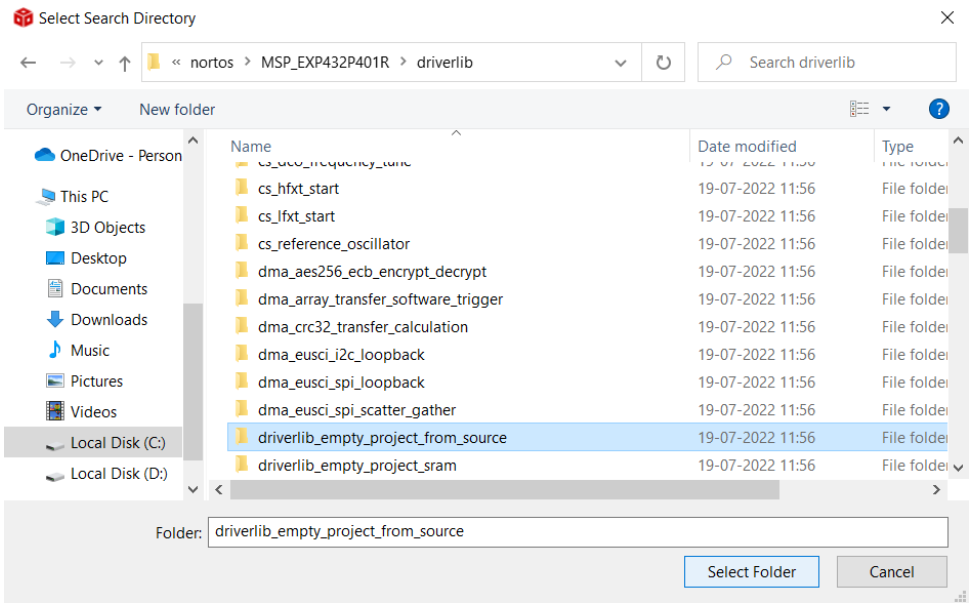


Select Browse Menu

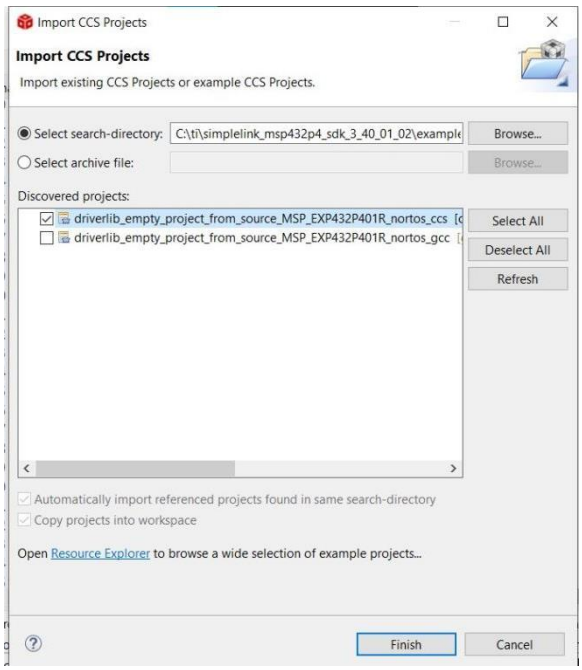


Go to the below path:

C:\ti\simplelink_msp432p4_sdk_3_40_01_02\examples\nortos\MSP_EXP432P401R\driverlib\
And Select Folder - **driverlib_empty_project_from_source**



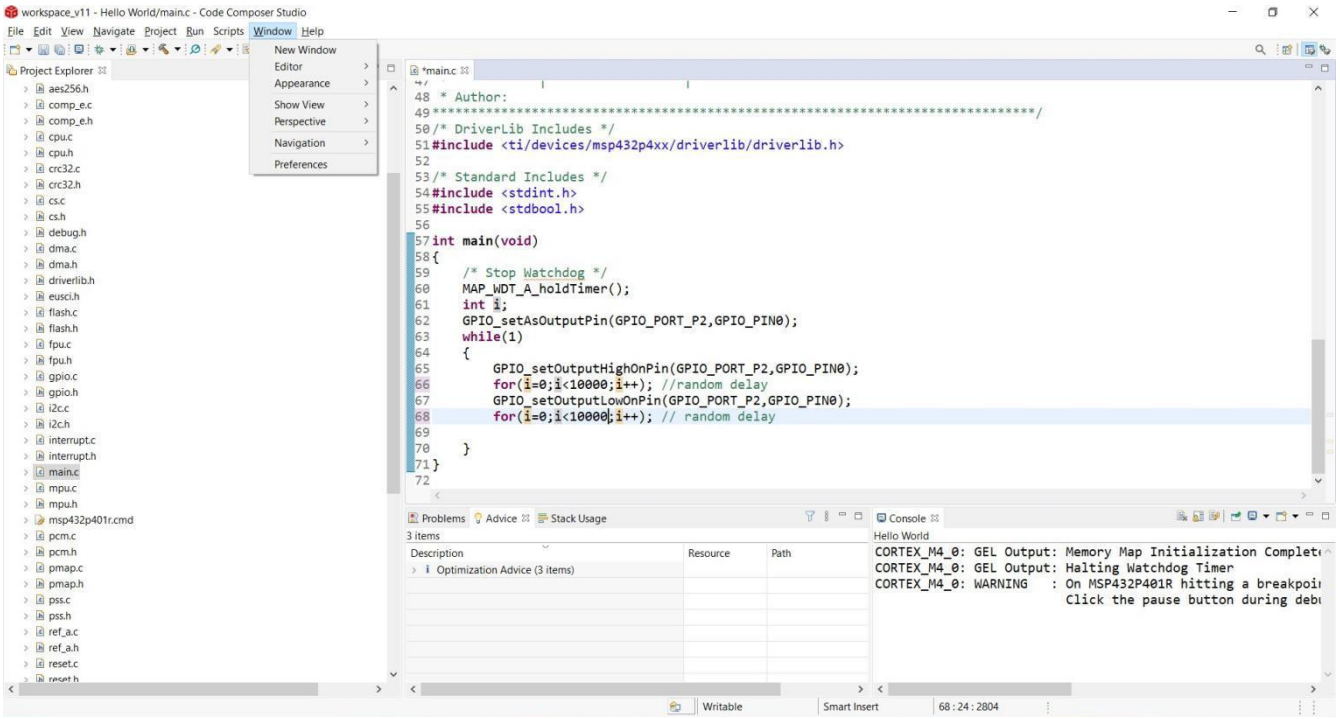
Check the first option - ccs and click on finish



Code:

```
/* DriverLib Includes */

#include <ti/devices/msp432p4xx/driverlib/driverlib.h>

/* Standard Includes */
#include <stdint.h> #include
<stdbool.h>

int main(void)

{

    /* Stop Watchdog */
    MAP_WDT_A_holdTimer(); int i;

    GPIO_setAsOutputPin(GPIO_PORT_P2,GPIO_PIN0);


    while(1)
```

```
    {
        GPIO_setOutputHighOnPin(GPIO_PORT_P2,GPIO_PIN0); for(i=0;i<10000;i++); //random delay
        GPIO_setOutputLowOnPin(GPIO_PORT_P2,GPIO_PIN0); for(i=0;i<10000;i++); // random delay

            }
}
```



Once Done, Compile the code to check for any errors. If the Build is successful connect the board.

```
#include <WiFi.h>
#include <SPI.h>

//#define SSID "octanauts"
//#define PASSWORD "aditya30"
IPAddress shieldIP,subnetMask,gatewayIP;
uint8_t rssi;
uint8_t networkId;
byte macAddr[6]; //mac address is 6 bytes
byte encryptionType;

char ssid[]="S";//my cell phone


void setup() {
  // put your setup code here, to run once:
   Serial.begin(115200);
   Serial.print("Connecting to WiFi..");
   while(WiFi.begin(ssid)!=WL_CONNECTED)  //for cell phone  with password:- while(Wi
   {
       Serial.print(".");
       delay(1);
   }
```

Console

CDT Build Console [wifi_test]

```
**** Build of configuration Debug for project wifi_test ****

"C:\\ti\\ccs1120\\ccs\\utils\\bin\\gmake" -k -j 4 all -O

gmake[1]: 'wifi_test.out' is up to date.

**** Build Finished ****
```

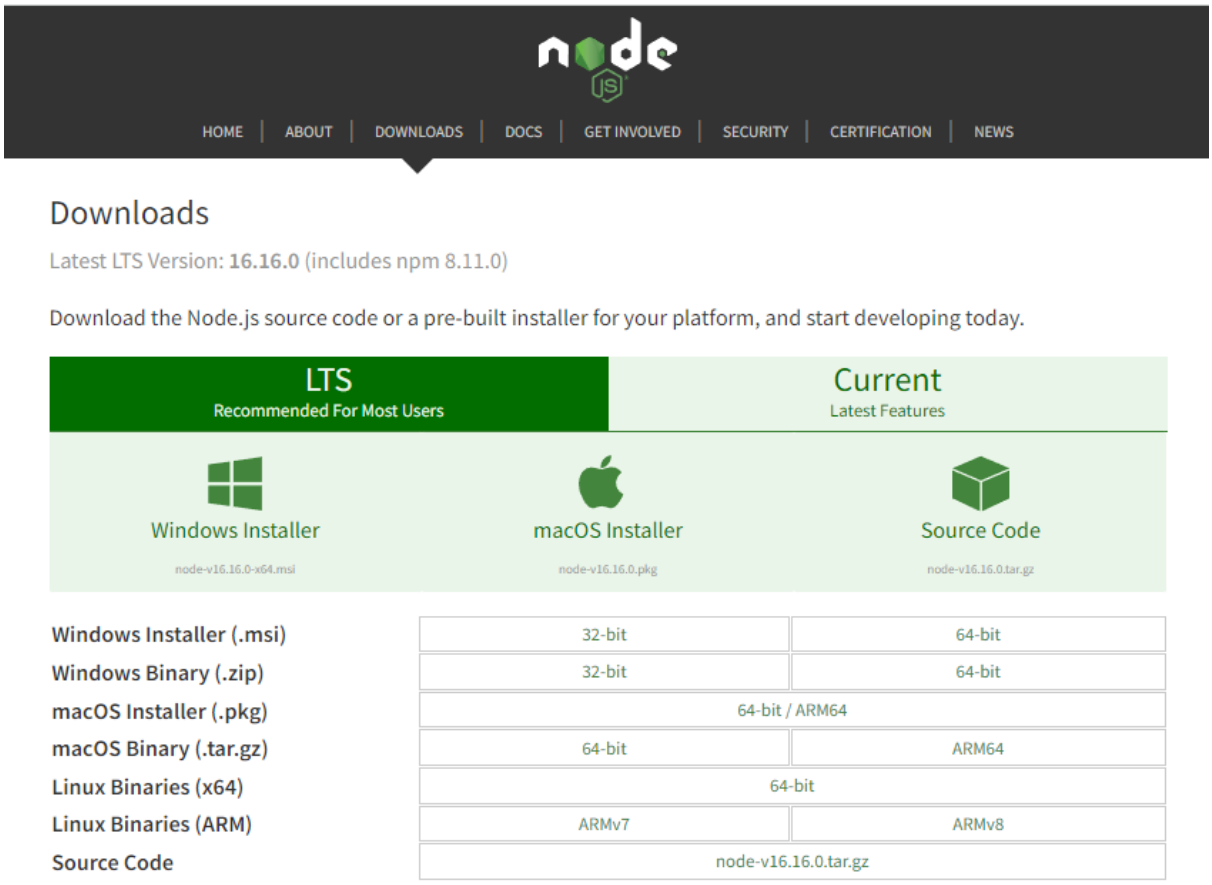
Once Build successfully, connect the board.

Reduce the value and look at the changes in the LED blinking pattern.

Practical 3

Installing Node.js, Node-Red and Node Red interface

Start with installation of Node.js, Node-red.



Once node.js is installed, check whether installation was done correctly by opening command prompt and using the following command:

node -version && npm -version

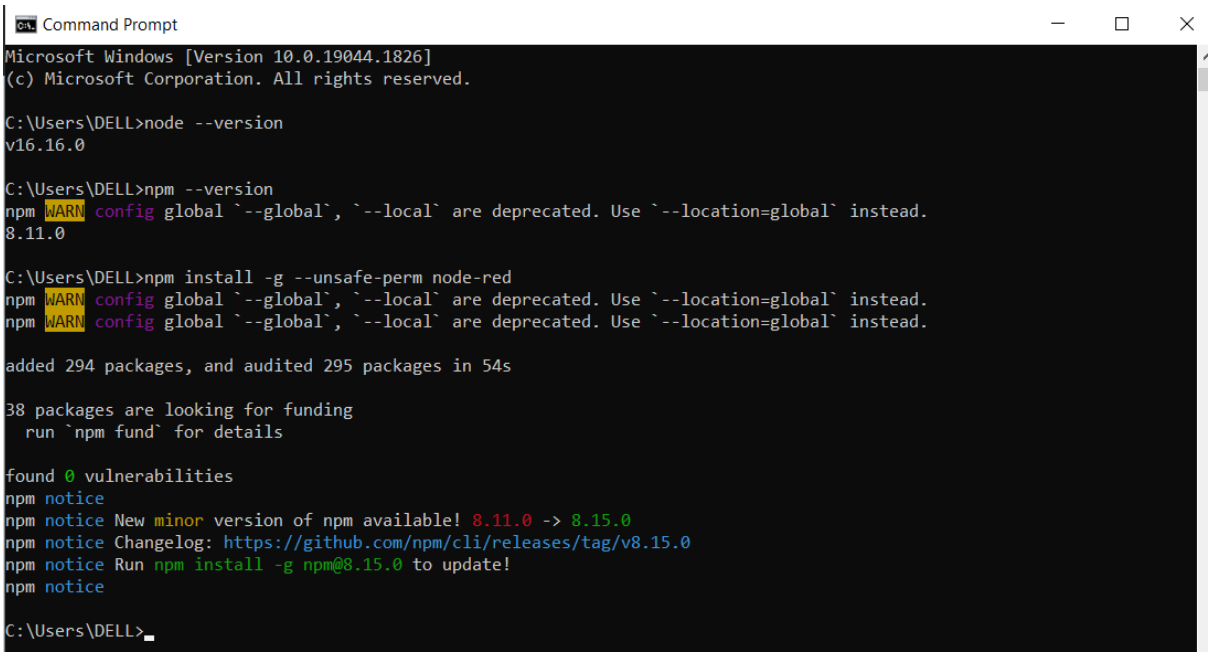
If the installation was successful, it will prompt the version.

Once this is confirmed, node red installation can be done using command:

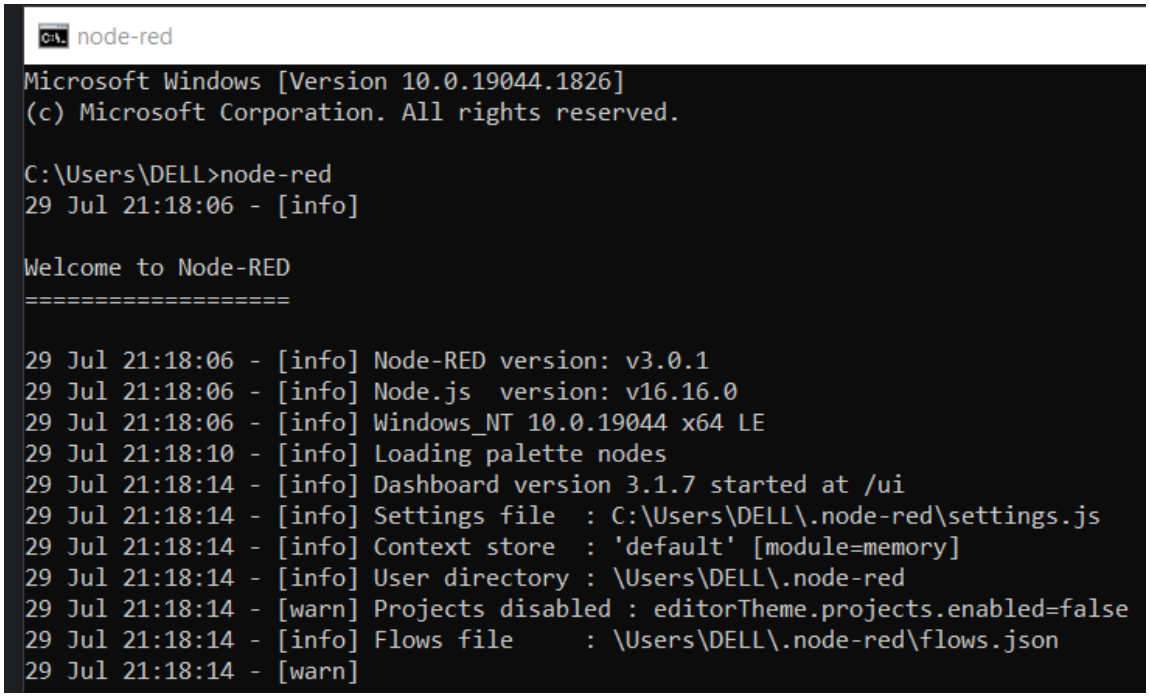
npm install -g -unsafe-perm node-red

If an error is encountered during installation, clear the cache using command:

npm cache clean -force



Once the node red is installed, use command **node-red** to start the flows as shown in below snapshot-

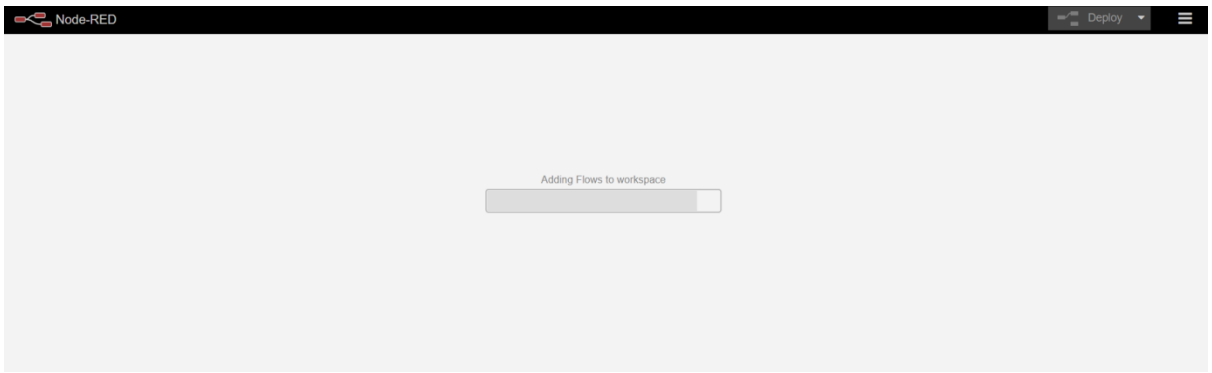


```
---------------------------------------------------------------------
Your flow credentials file is encrypted using a system-generated key.

If the system-generated key is lost for any reason, your credentials
file will not be recoverable, you will have to delete it and re-enter
your credentials.

You should set your own key using the 'credentialSecret' option in
your settings file. Node-RED will then re-encrypt your credentials
file using your chosen key the next time you deploy a change.
---------------------------------------------------------------------

29 Jul 21:18:14 - [info] Server now running at http://127.0.0.1:1880/
29 Jul 21:18:14 - [info] Starting flows
29 Jul 21:18:15 - [info] Started flows
```

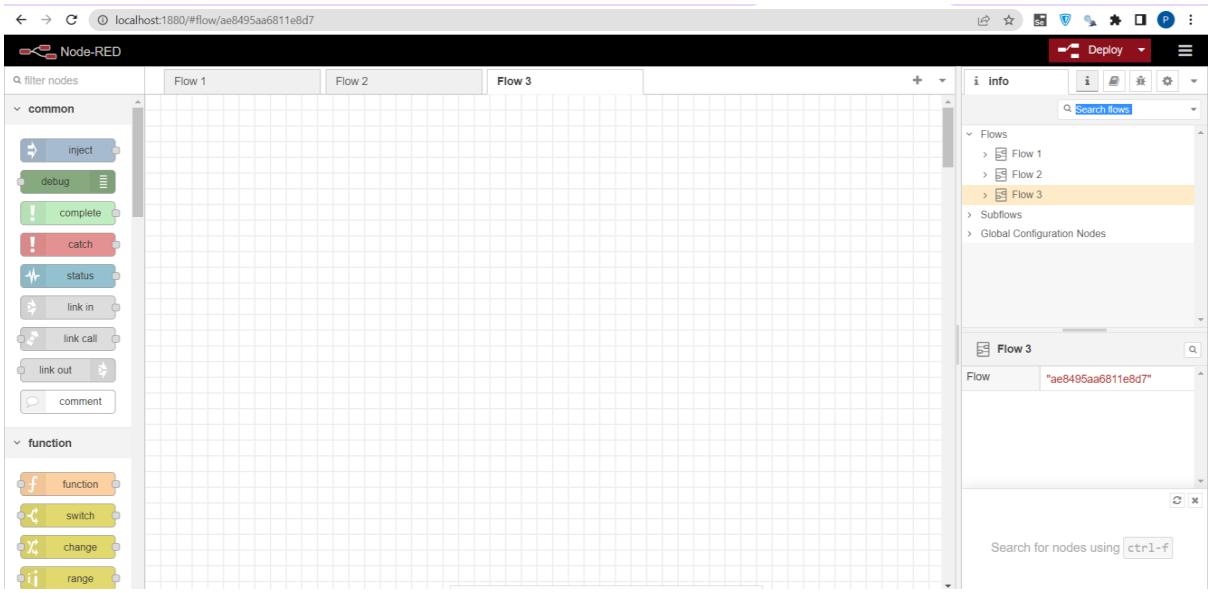
Then open the browser and start node red at the port 1880 using the link -

localhost:1880/

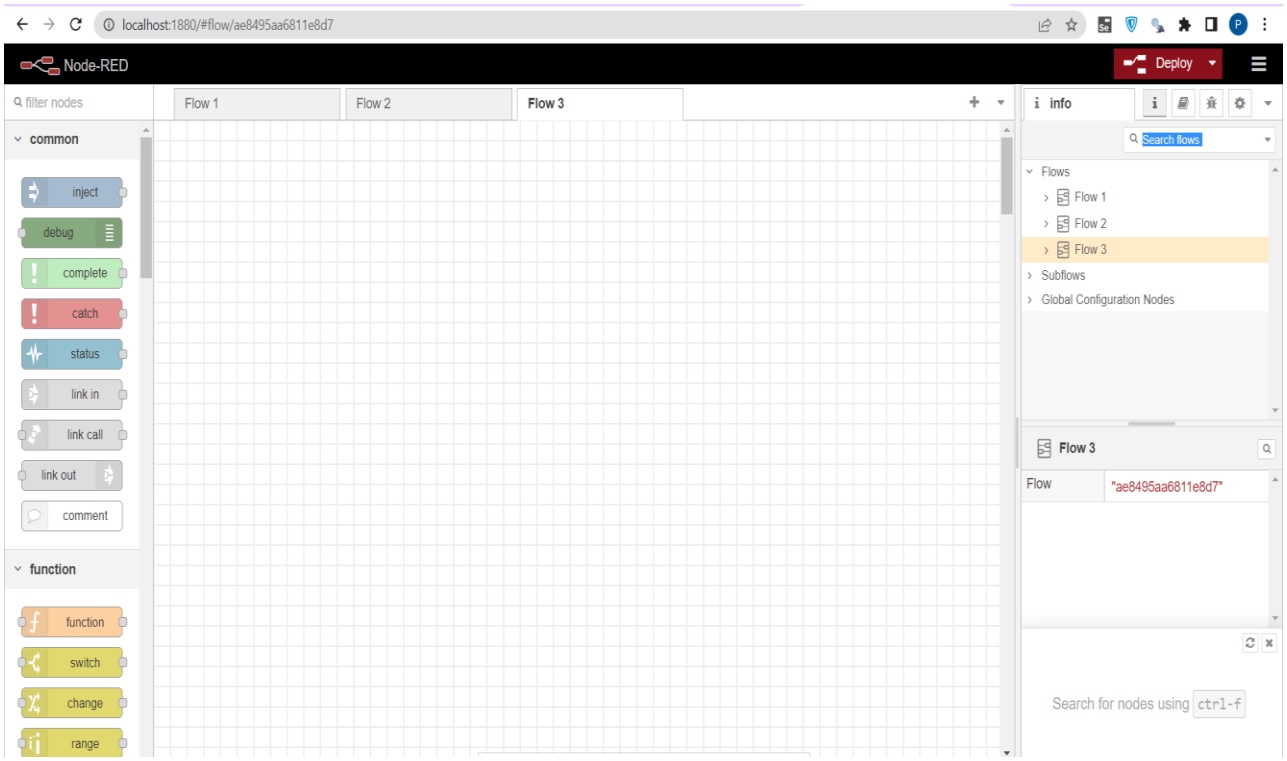
An Interface as below will be displayed



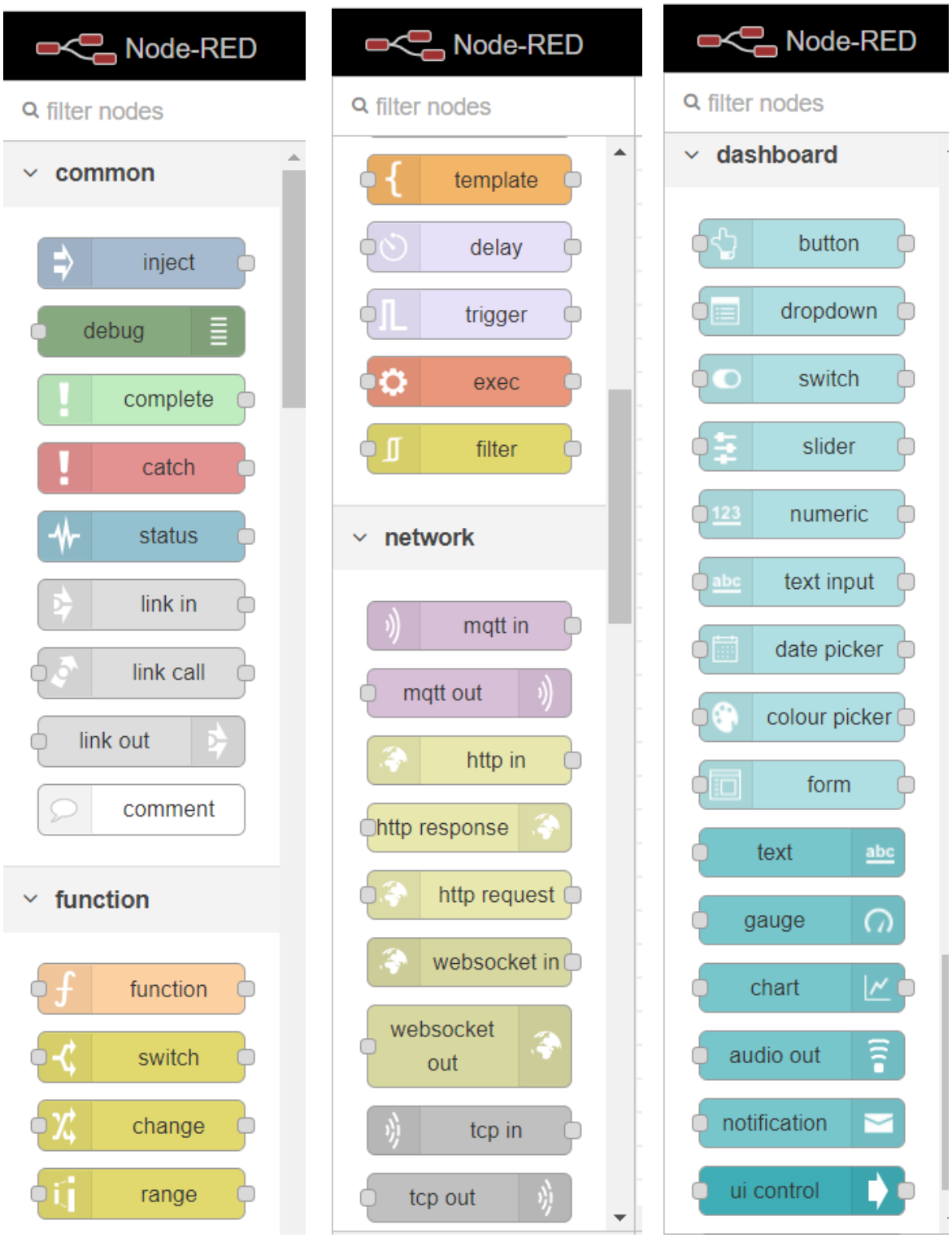
Once fully loaded, You will see a screen as below -



Node-RED IDE contains a palette of nodes that can be easily dragged and dropped onto the canvas (white portion in the center).



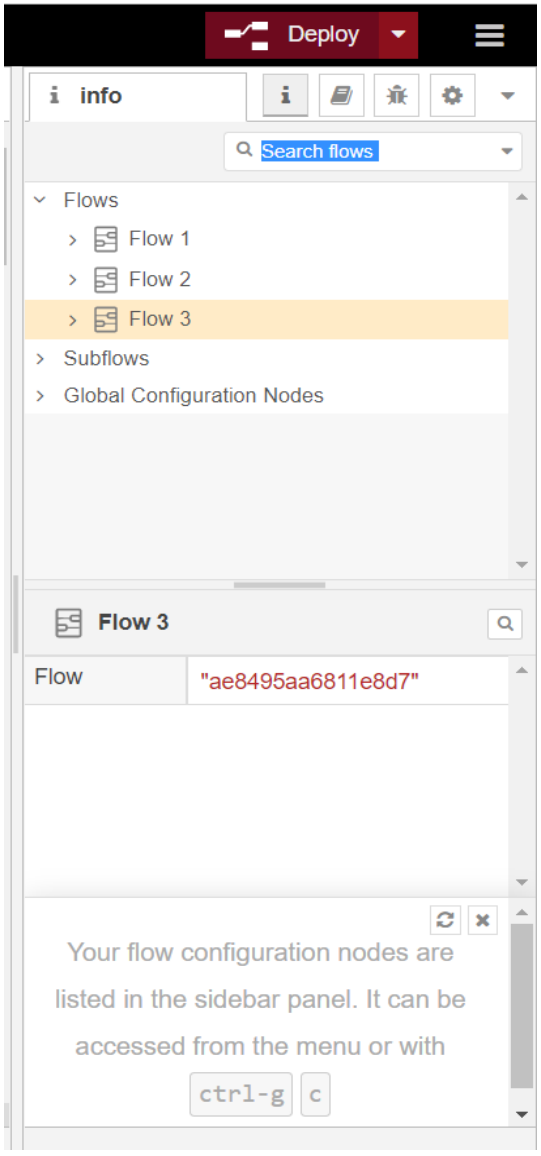
Left side of the IDE contains a lot of palette nodes each assigned with its own set of actions / tasks to be carried out. Some of them are common nodes, some are application specific like - network, function, Dashboard. It contains required nodes to carry out the relevant operations associated with the application.

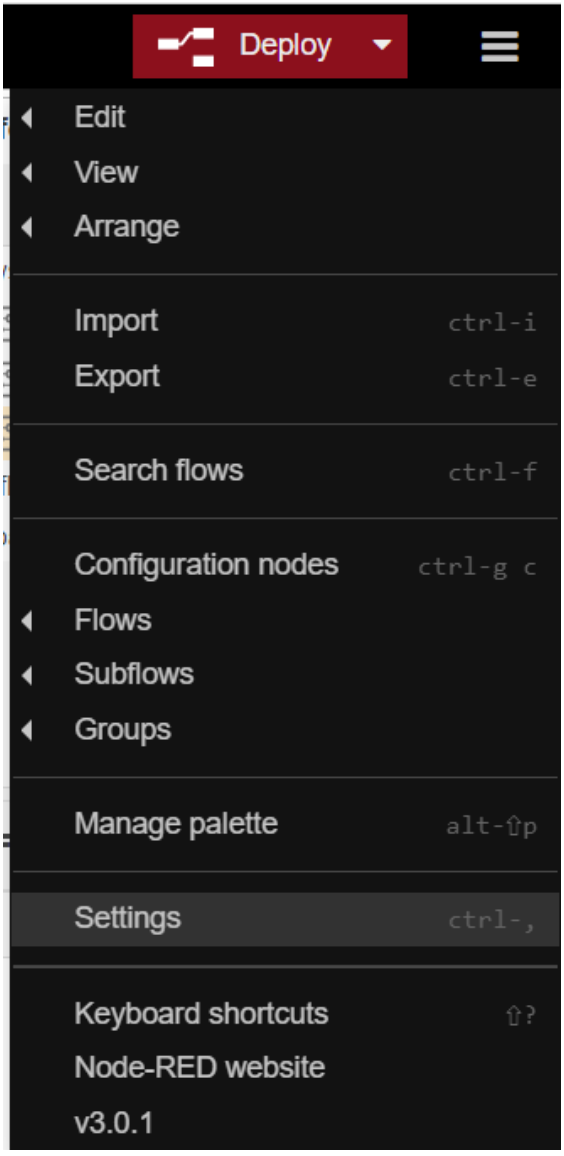


Right Side of the IDE contains a Menu list of basic functions to easily handle and manage the palette.

Along with the menu there are five different options one is info (gives info about Flows on the project), Notes, Debug window (to check for debug messages output by the function), Settings and Dashboard option to alter / update the look while dealing with dashboard applications.

And a most important Deploy Button which is used to deploy any changes performed.



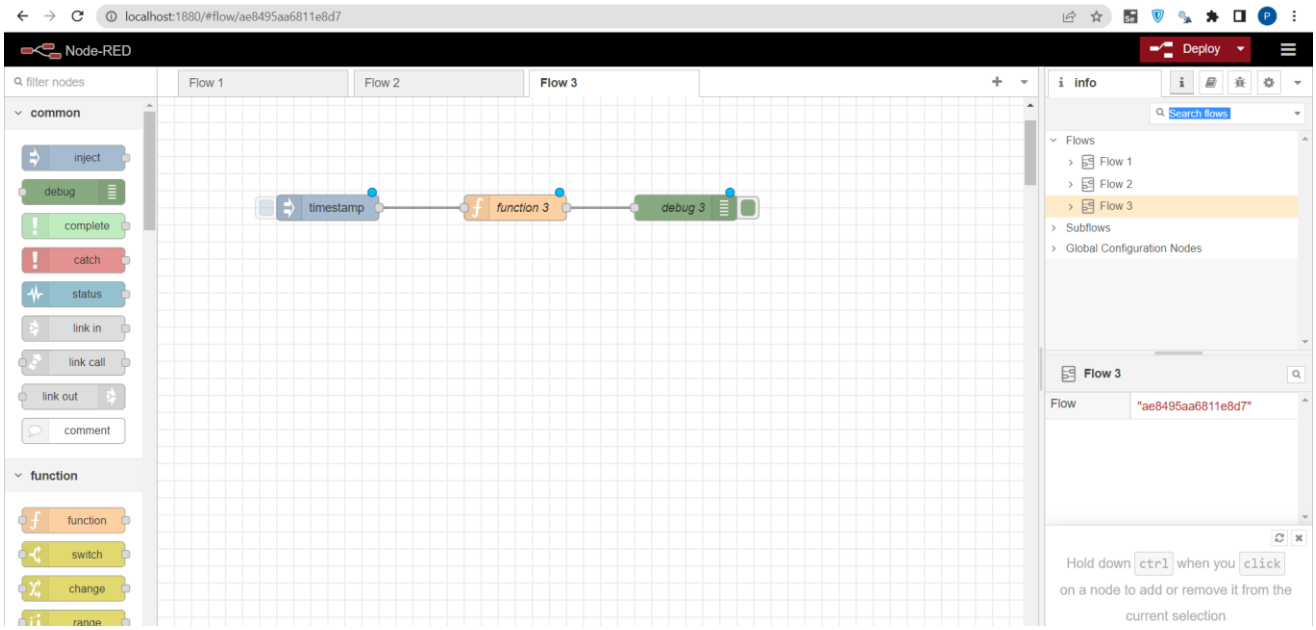


A Node is the basic building block of a flow. Nodes are triggered by either receiving a message from the previous node in a flow, or by waiting for some external event, such as an incoming HTTP request, a timer or GPIO hardware change. They process that message, or event, and then may send a message to the next nodes in the flow.

A Flow is represented as a tab within the editor workspace and is the main way to organise nodes.

The term “flow” is also used to informally describe a single set of connected nodes. So a flow (tab) can contain multiple flows (sets of connected nodes).

Following snapshot has 3 nodes namely - inject, function and debug each one of them connected with flow

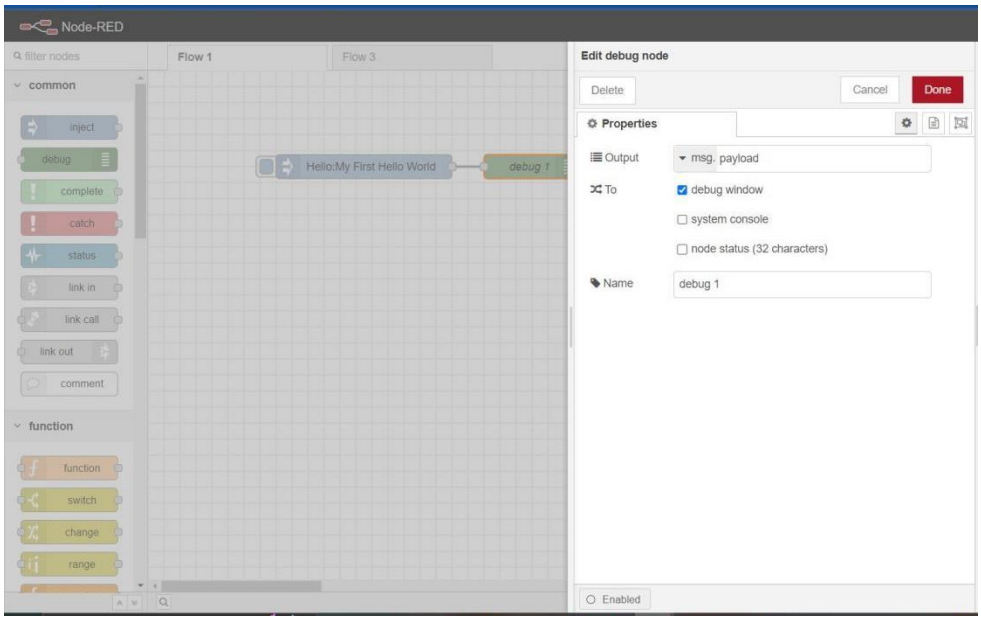


Practical 4

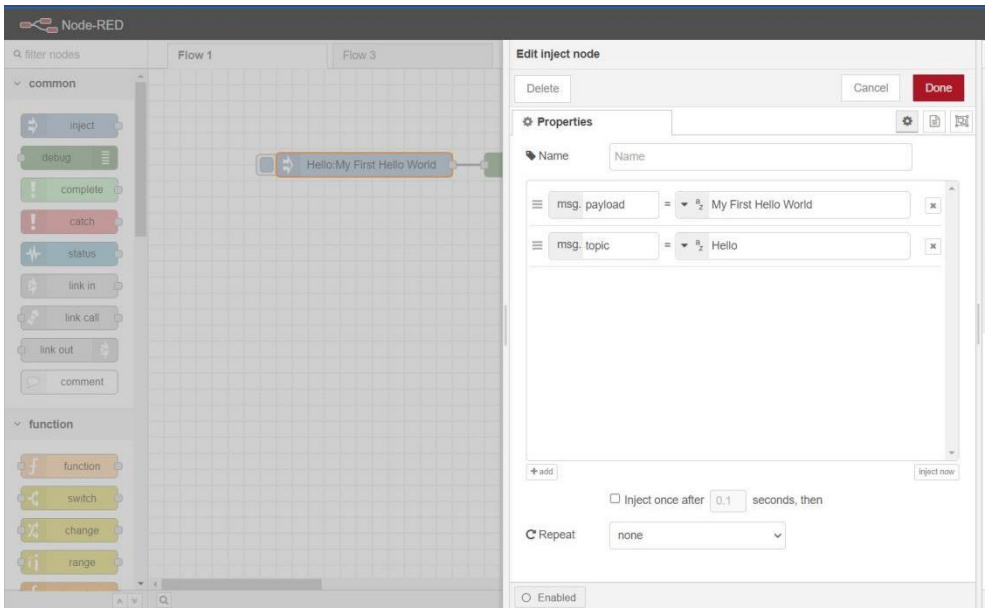
Introducing the inject, function, debug and switch nodes

Create the design as below using the inject and debug node from pallet and connect them

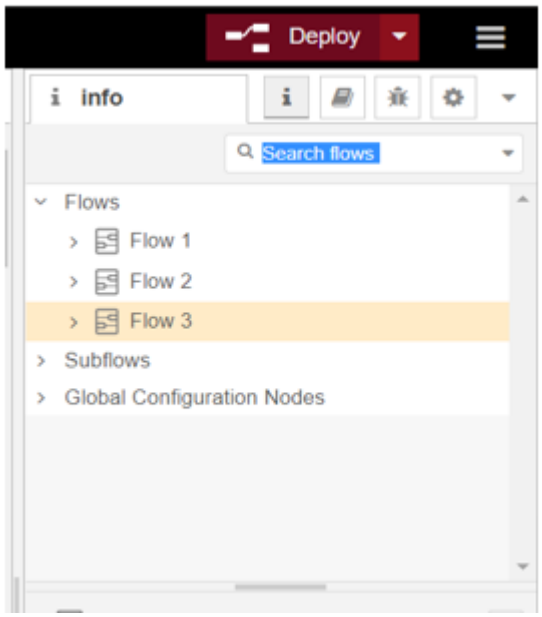
Double click debug node and add the below details into it.



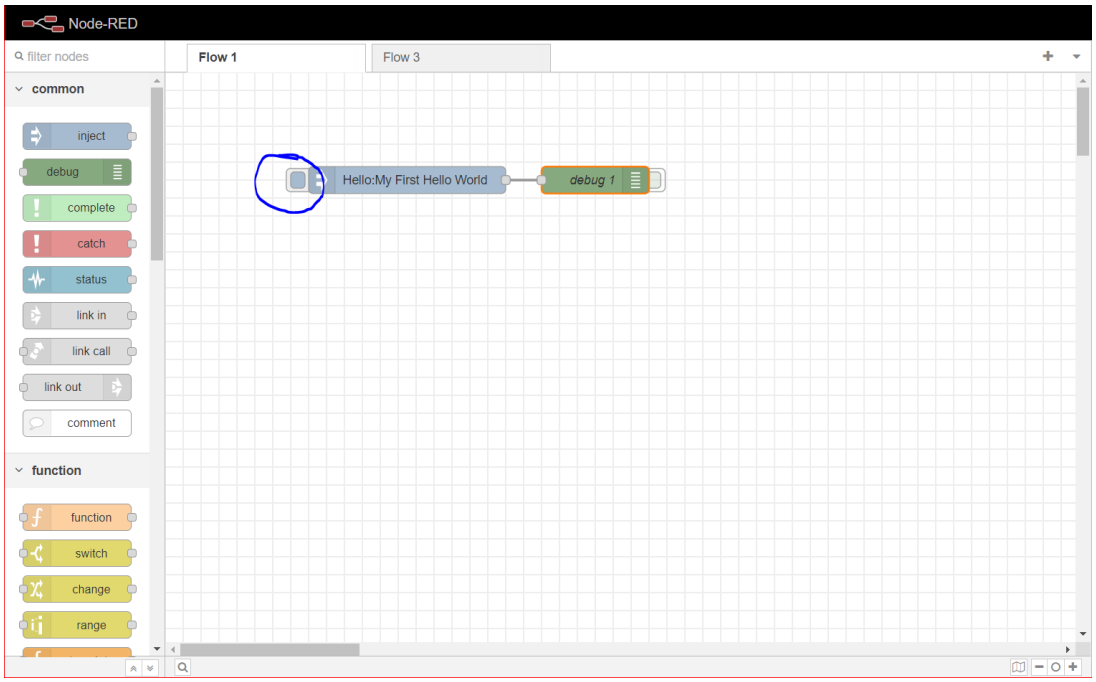
Double click on the Inject node and add the below details -



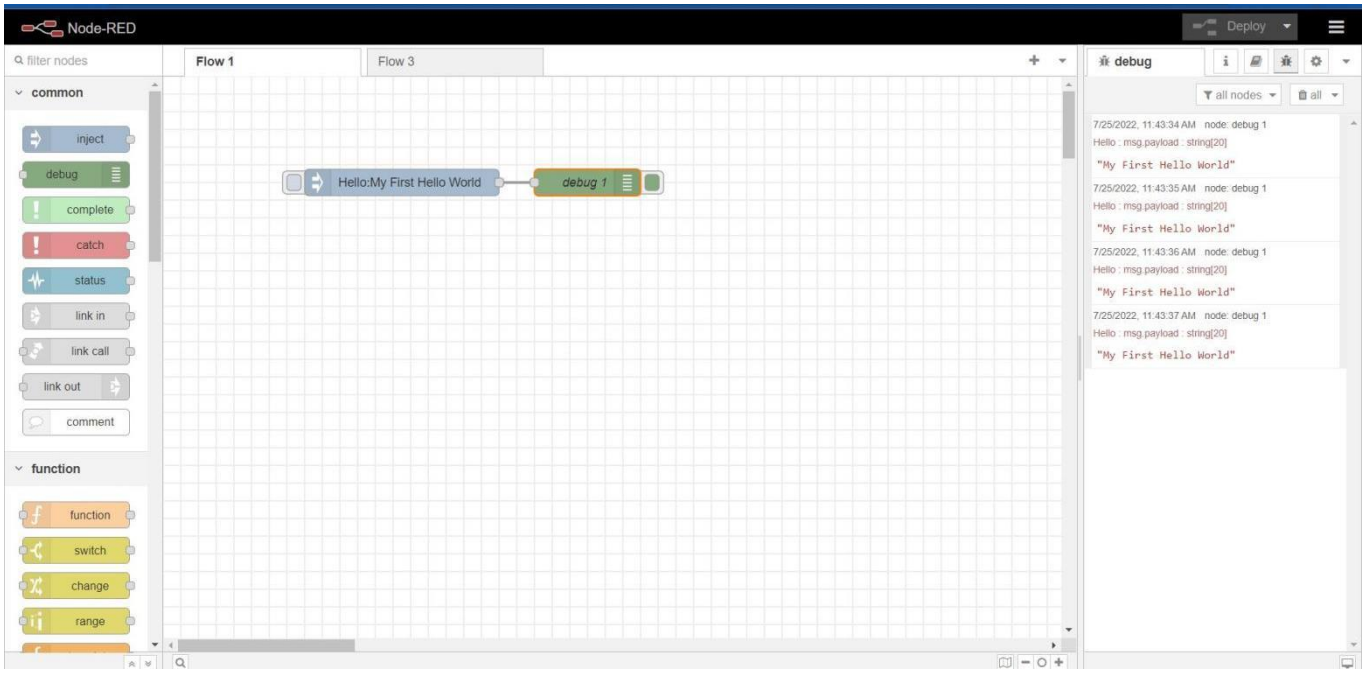
Once the details are added click done and then click on Deploy to deploy the changes done



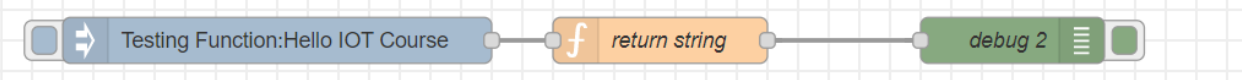
Click on the highlighted part of the inject node, to inject a value.



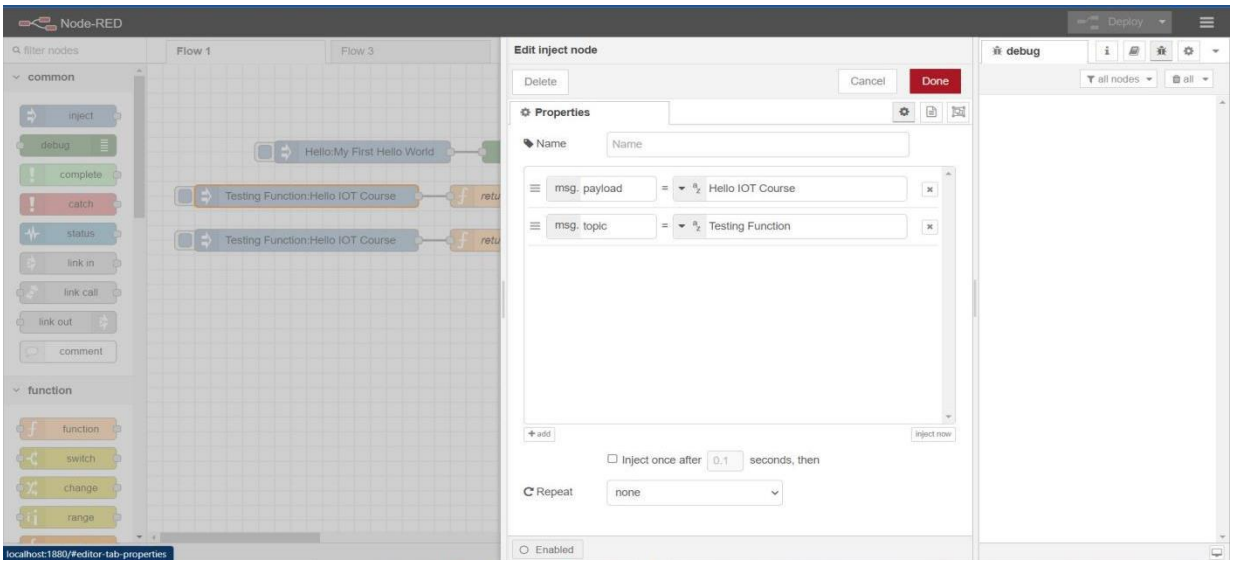
Once done to check if the supplied message is reaching the right place. Open up the Debug window and click on inject button. It will showcase the passed message as below -



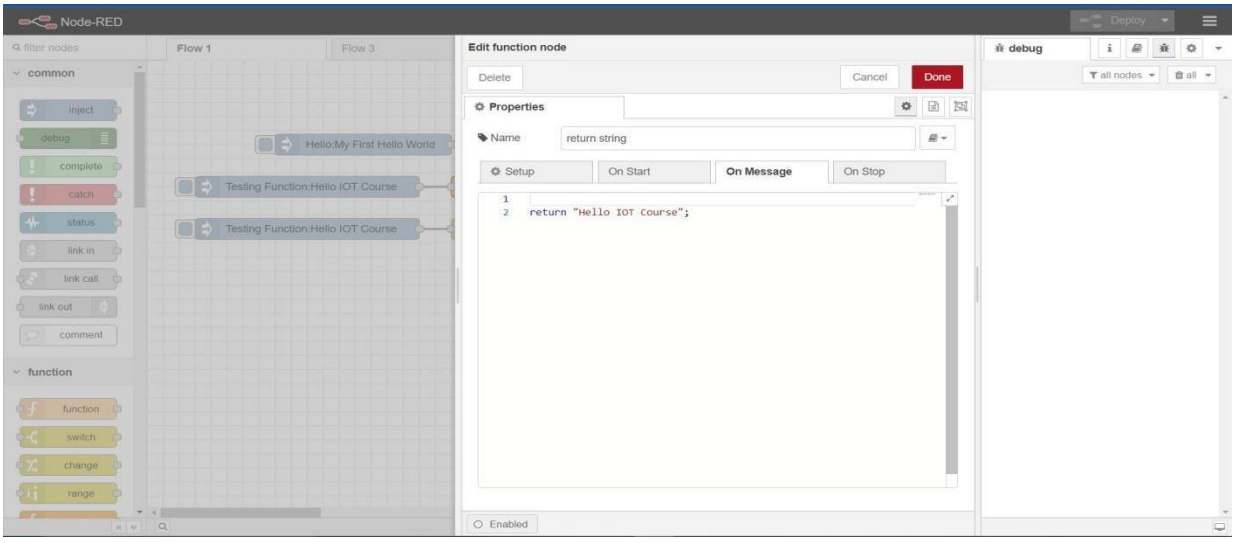
Create the design as below using the inject, function and debug node from pallet and connect them



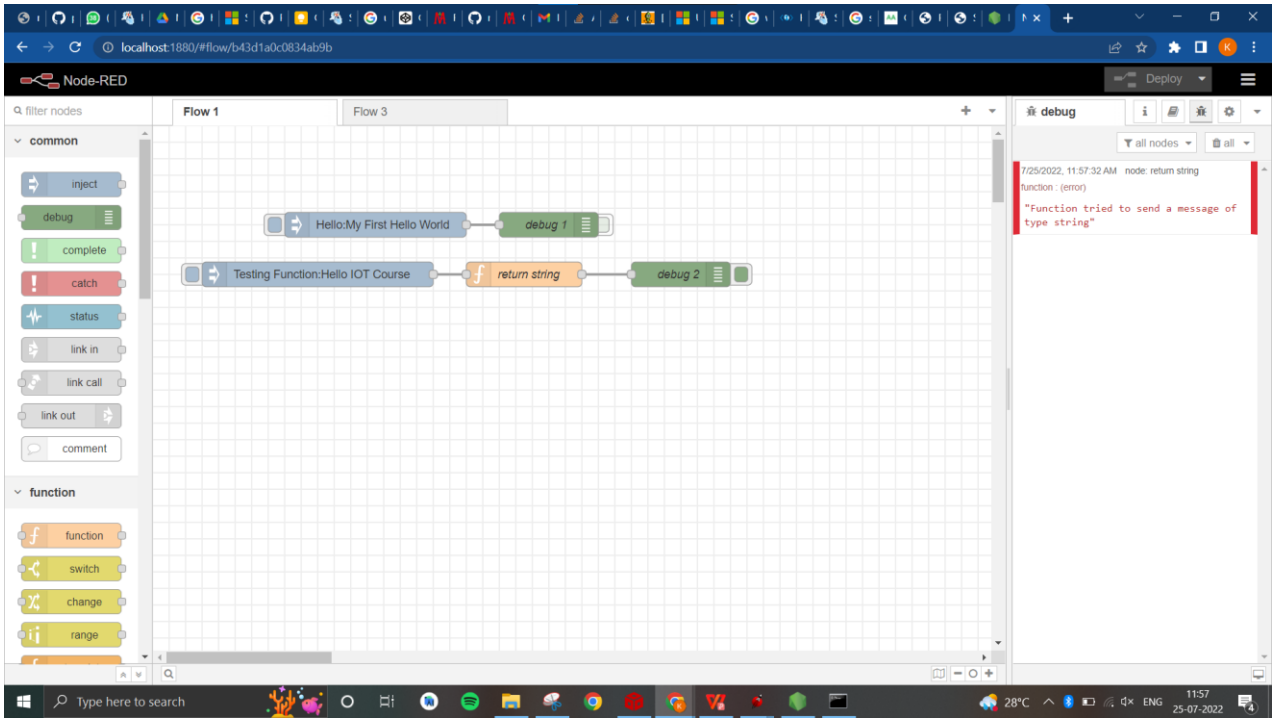
Double click on inject node and add the properties as below.



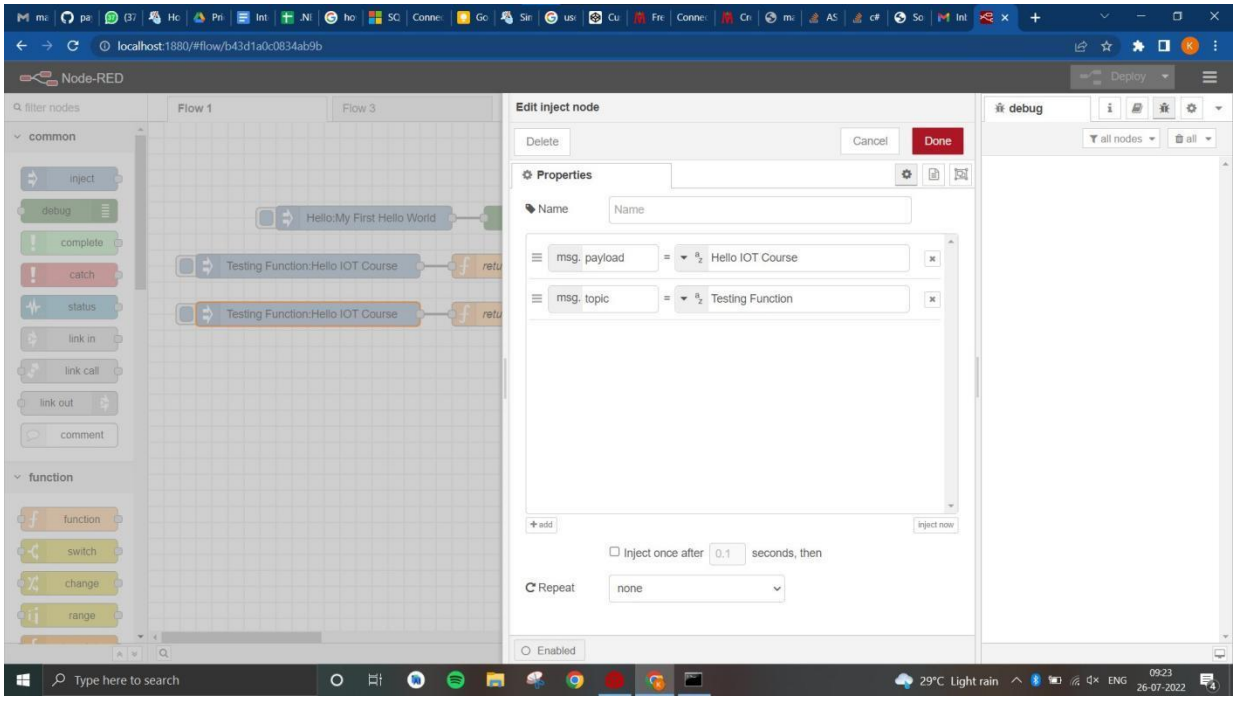
Add the function definition. Here the function returns string “Hello IOT Course”.



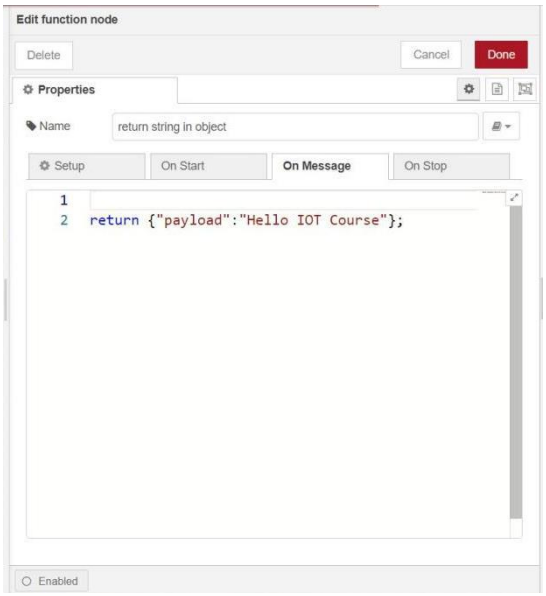
Open the debug window and click on the inject button. This throws the below error -



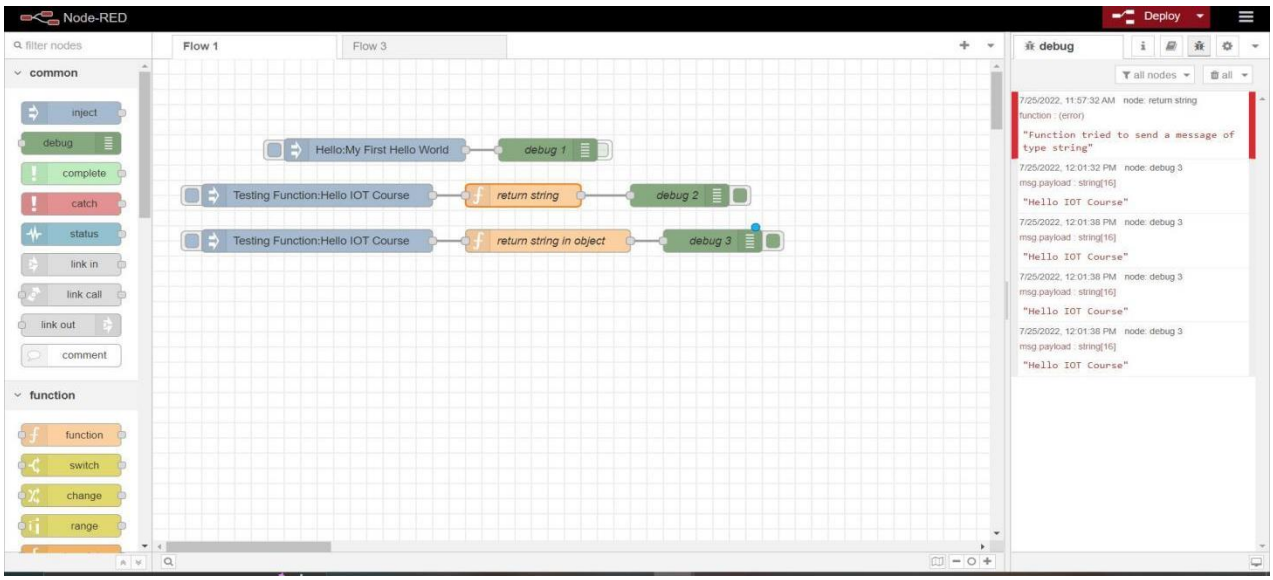
Corrected function -



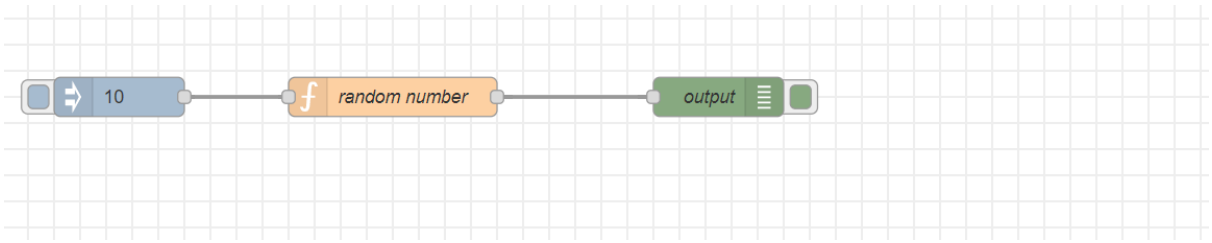
Modify the function definition and return the string variable

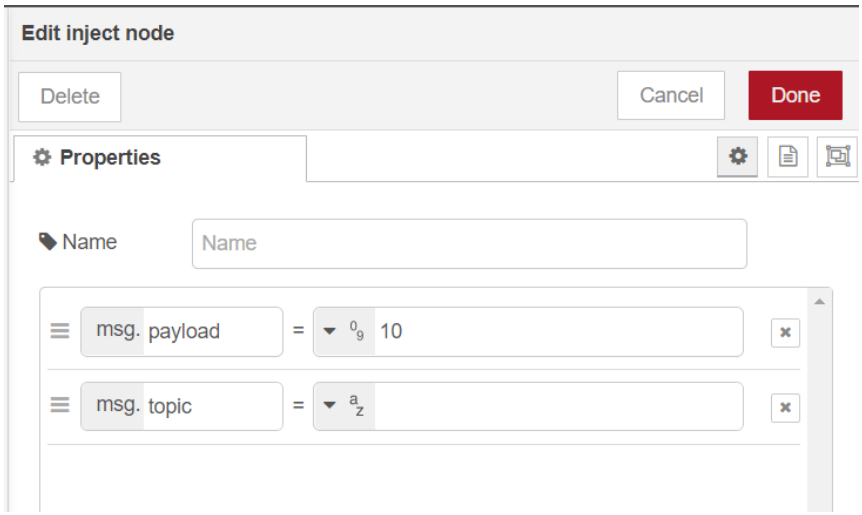


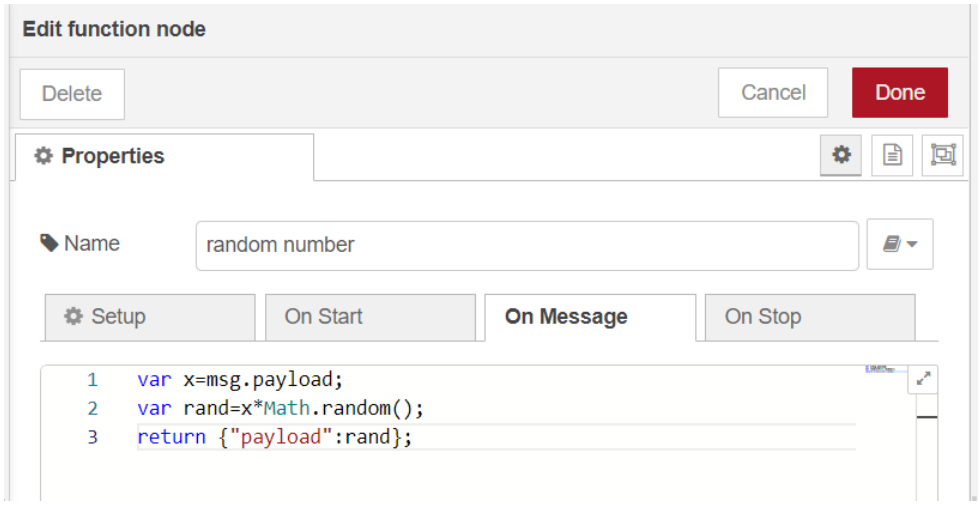
Now when called the inject function, correct output is returned.

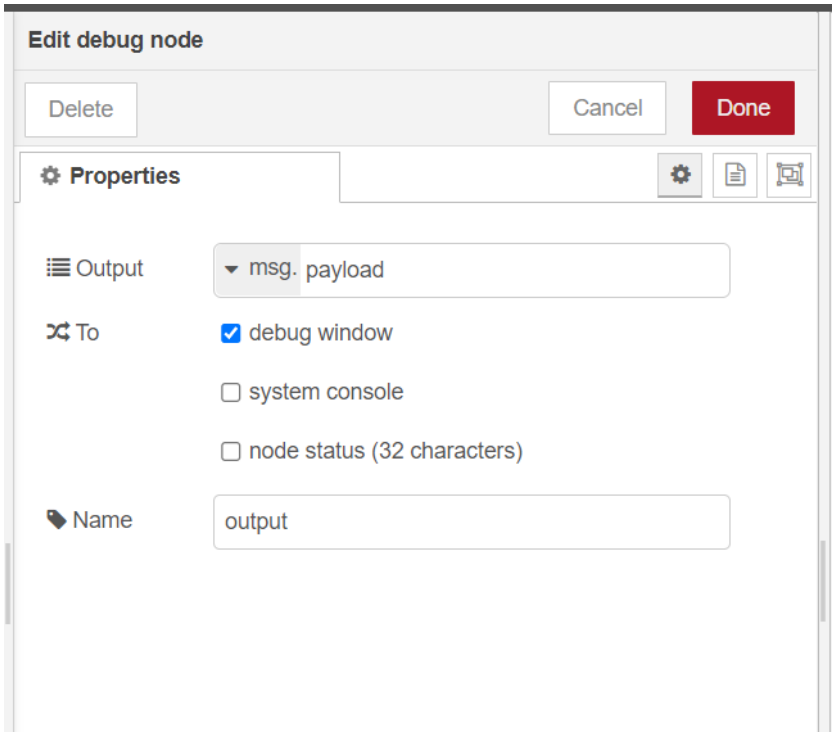


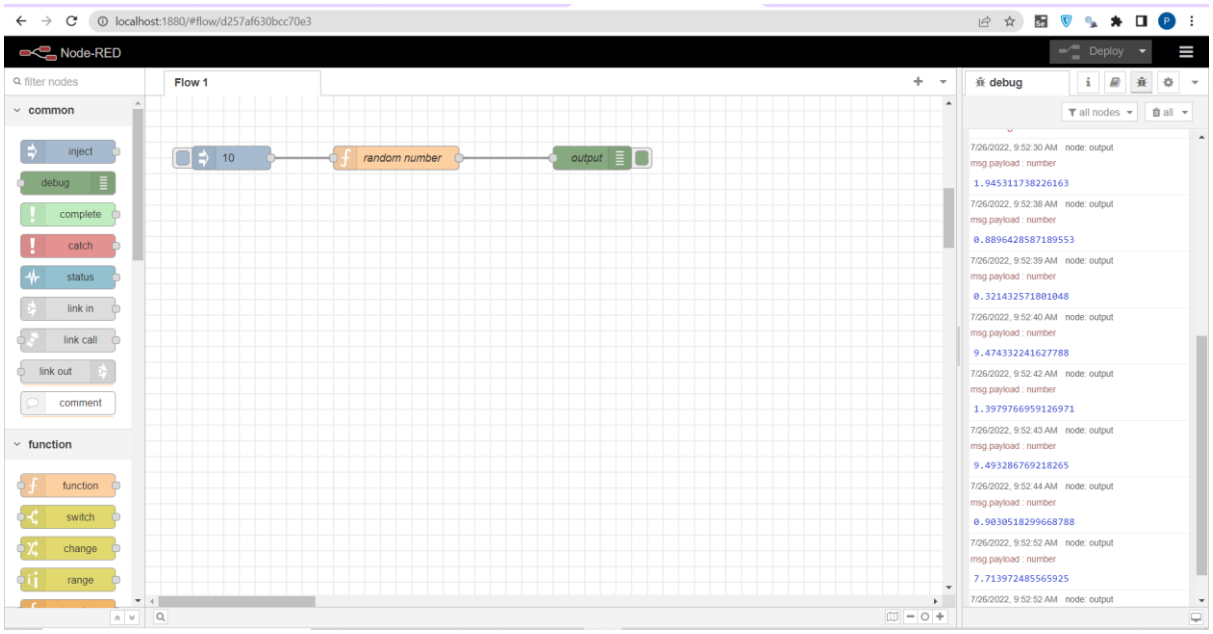
Create the design as below using the inject, function and debug node from pallet and connect them



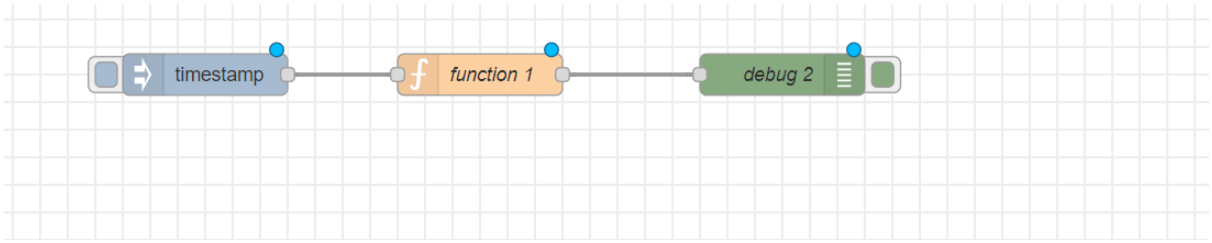


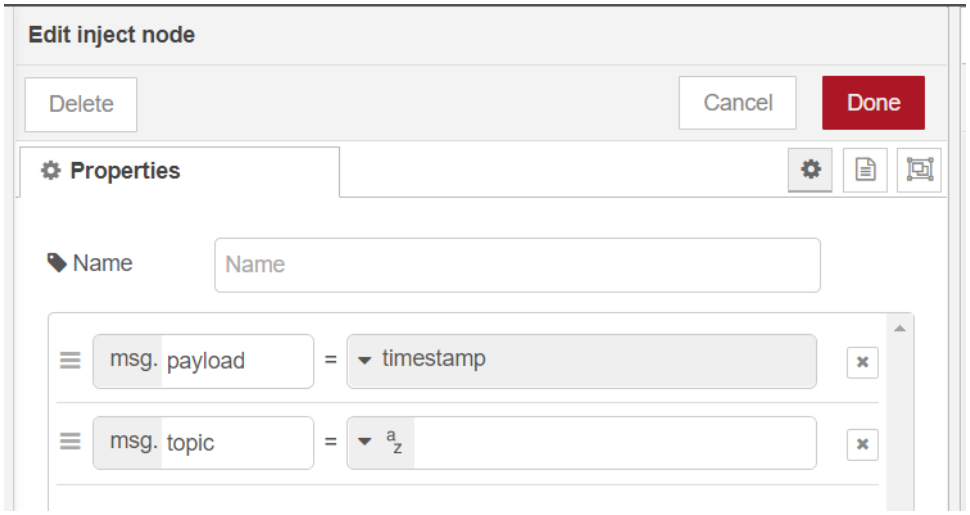


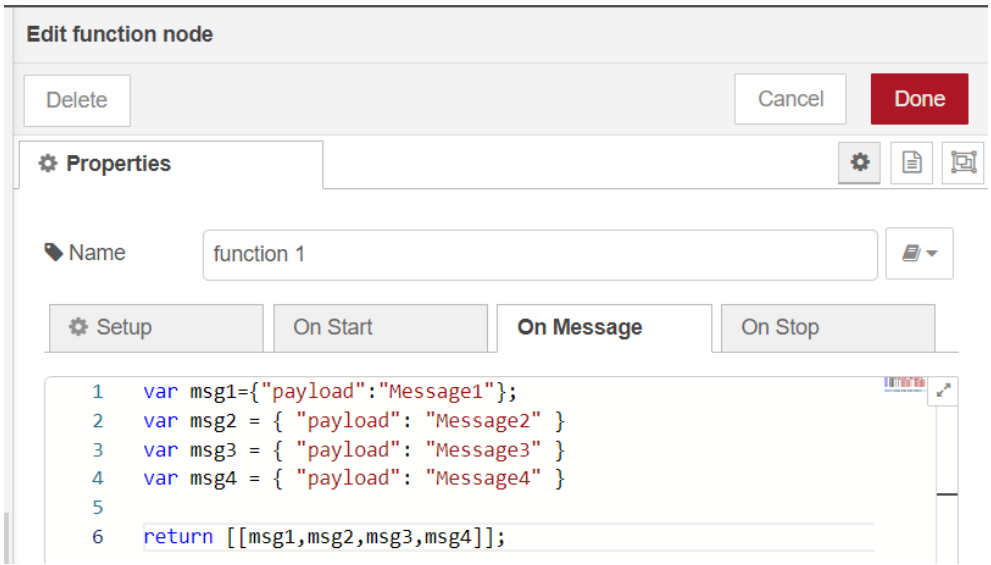

Edit debug node
Delete
Cancel
Done
Properties
Output
msg. payload
To
debug window
system console
node status (32 characters)
Name
output

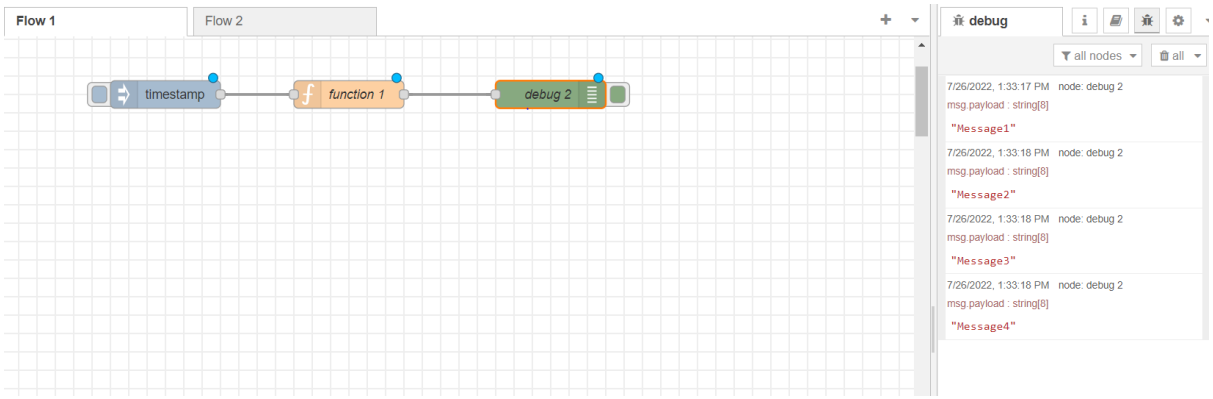

localhost:1880/#flow/d257af630bcc70e3
Node-RED
Deploy
filter nodes
Flow 1
common
inject
debug
complete
catch
status
link in
link call
link out
comment
function
function
switch
change
range
10
random number
output
debug
all nodes
all
7/26/2022, 9:52:30 AM node: output
msg.payload : number
1.945311738226163
7/26/2022, 9:52:38 AM node: output
msg.payload : number
0.8896428587189553
7/26/2022, 9:52:39 AM node: output
msg.payload : number
0.321432571801048
7/26/2022, 9:52:40 AM node: output
msg.payload : number
9.474332241627788
7/26/2022, 9:52:42 AM node: output
msg.payload : number
1.3979766959126971
7/26/2022, 9:52:43 AM node: output
msg.payload : number
9.493286769218265
7/26/2022, 9:52:44 AM node: output
msg.payload : number
0.9030518299668788
7/26/2022, 9:52:52 AM node: output
msg.payload : number
7.713972485565925
7/26/2022, 9:52:52 AM node: output

Function2

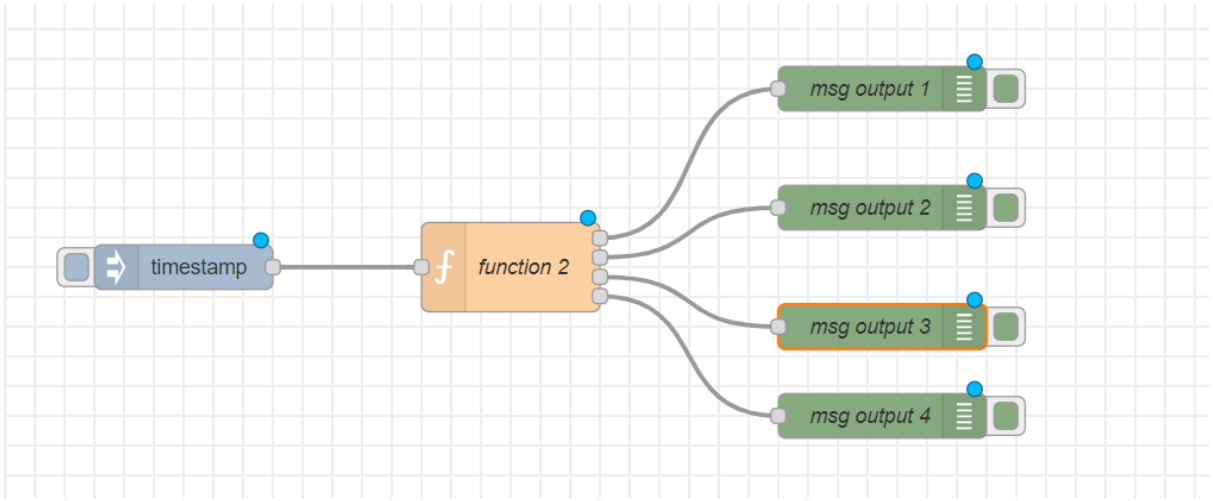

timestamp
function 1
debug 2

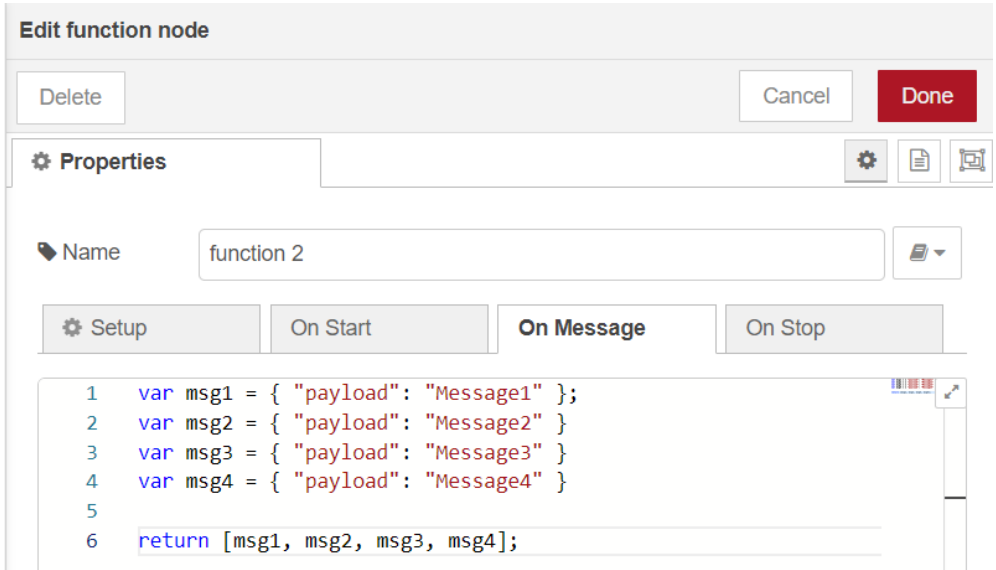

Edit inject node
Delete
Cancel
Done
Properties
Name
Name
msg. payload
=
timestamp
msg. topic
=

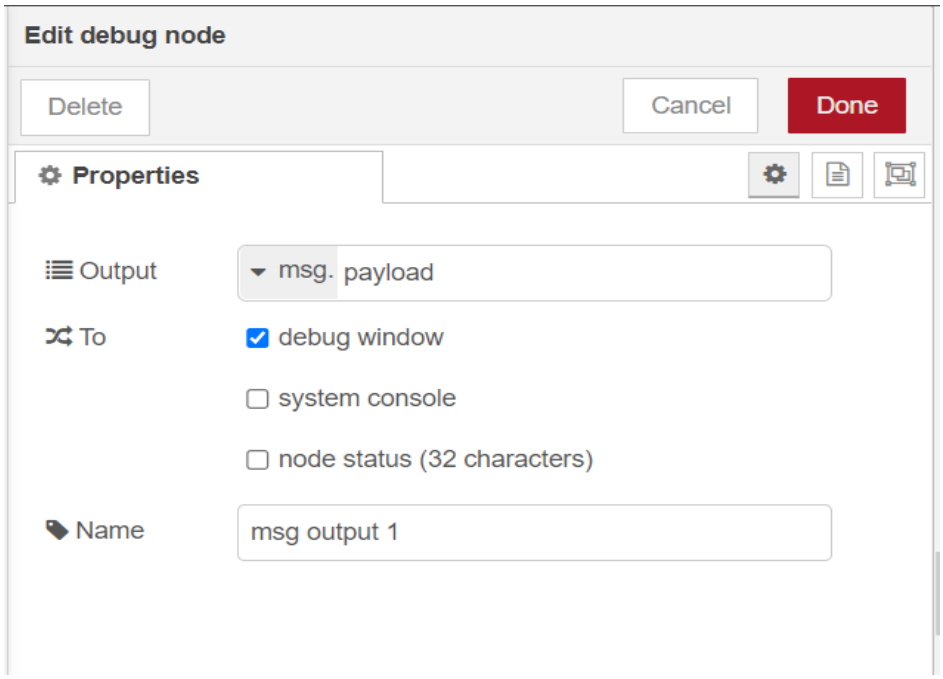

Edit function node
Delete
Cancel
Done
Properties
Name
function 1
Setup
On Start
On Message
On Stop
1 var msg1={"payload":"Message1"};
2 var msg2 = { "payload": "Message2" }
3 var msg3 = { "payload": "Message3" }
4 var msg4 = { "payload": "Message4" }
5
6 return [[msg1,msg2,msg3,msg4]];

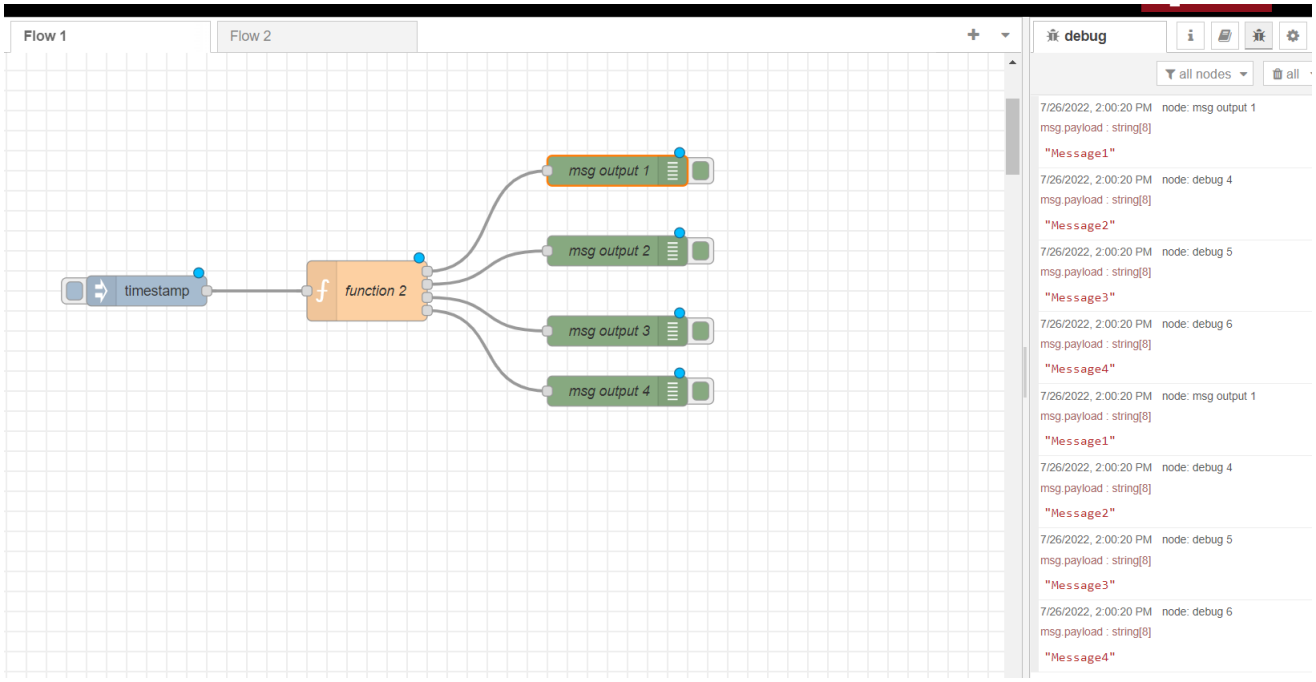

Flow 1
Flow 2
timestamp
function 1
debug 2
debug
all nodes
all
7/26/2022, 1:33:17 PM node: debug 2
msg.payload : string[8]
"Message1"
7/26/2022, 1:33:18 PM node: debug 2
msg.payload : string[8]
"Message2"
7/26/2022, 1:33:18 PM node: debug 2
msg.payload : string[8]
"Message3"
7/26/2022, 1:33:18 PM node: debug 2
msg.payload : string[8]
"Message4"

Function 3


timestamp
function 2
msg output 1
msg output 2
msg output 3
msg output 4


Edit function node
Delete
Cancel
Done
Properties
Name
function 2
Setup
On Start
On Message
On Stop
1 var msg1 = { "payload": "Message1" };
2 var msg2 = { "payload": "Message2" }
3 var msg3 = { "payload": "Message3" }
4 var msg4 = { "payload": "Message4" }
5
6 return [msg1, msg2, msg3, msg4];

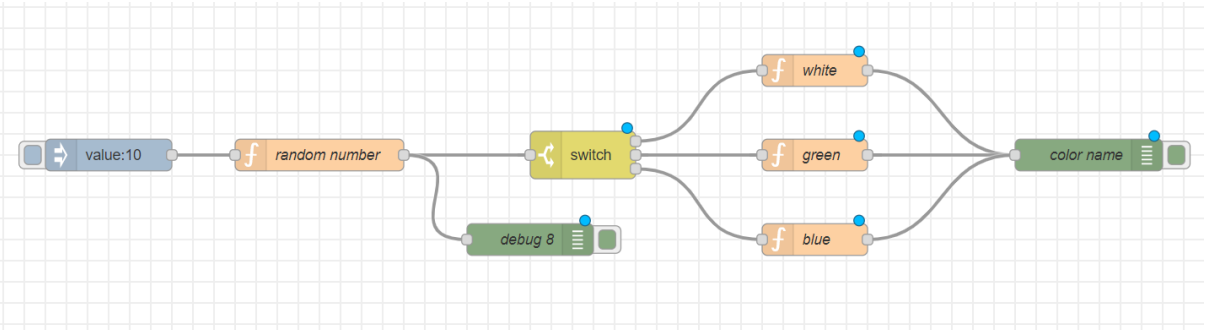

Edit debug node
Delete
Cancel
Done
Properties
Output
msg. payload
To
debug window
system console
node status (32 characters)
Name
msg output 1


Flow 1
Flow 2
timestamp
function 2
msg output 1
msg output 2
msg output 3
msg output 4
debug
all nodes
all
7/26/2022, 2:00:20 PM node: msg output 1
msg.payload : string[8]
"Message1"
7/26/2022, 2:00:20 PM node: debug 4
msg.payload : string[8]
"Message2"
7/26/2022, 2:00:20 PM node: debug 5
msg.payload : string[8]
"Message3"
7/26/2022, 2:00:20 PM node: debug 6
msg.payload : string[8]
"Message4"
7/26/2022, 2:00:20 PM node: msg output 1
msg.payload : string[8]
"Message1"
7/26/2022, 2:00:20 PM node: debug 4
msg.payload : string[8]
"Message2"
7/26/2022, 2:00:20 PM node: debug 5
msg.payload : string[8]
"Message3"
7/26/2022, 2:00:20 PM node: debug 6
msg.payload : string[8]
"Message4"

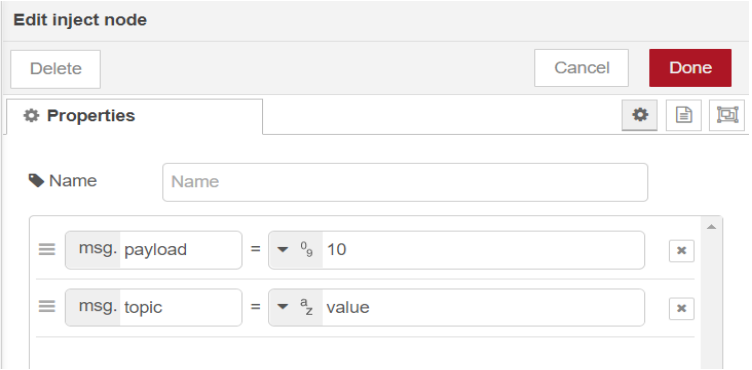
Practical 5

Random number generator with selection of color. Introduce the HTTP node.

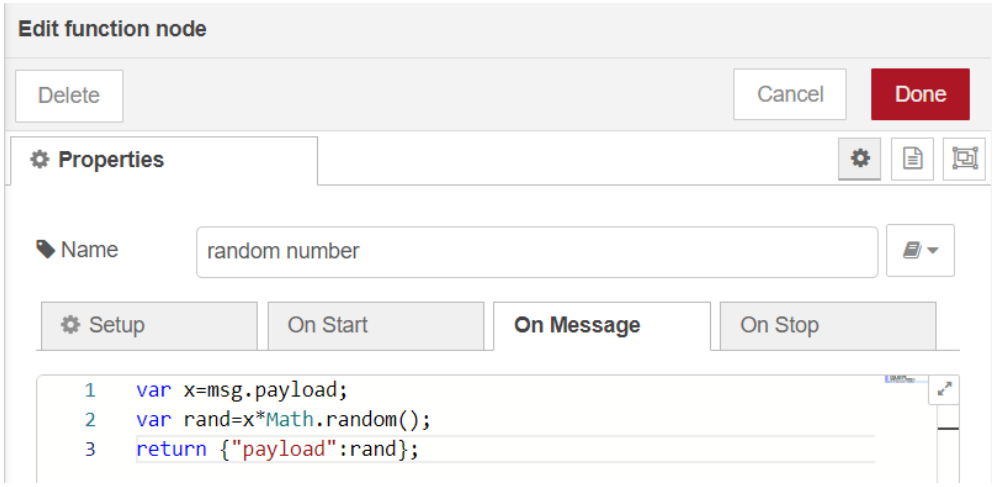
Arrange the nodes as below and connect them.



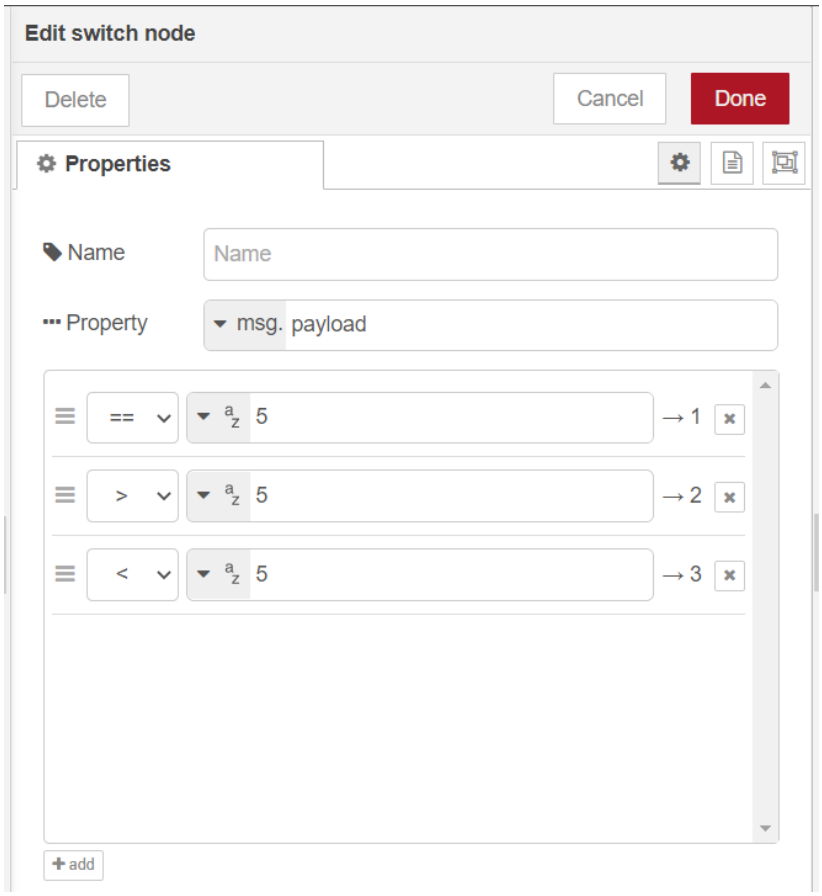
Add the properties as below -



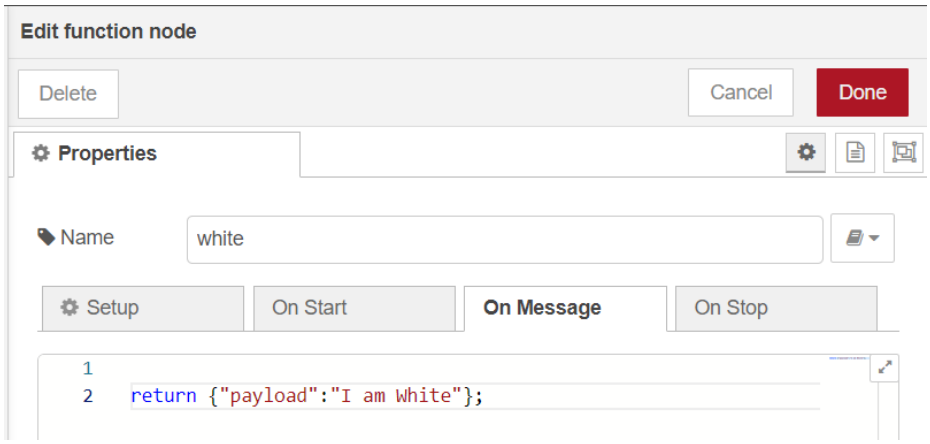
This function generates random numbers and assigns it to the payload variable

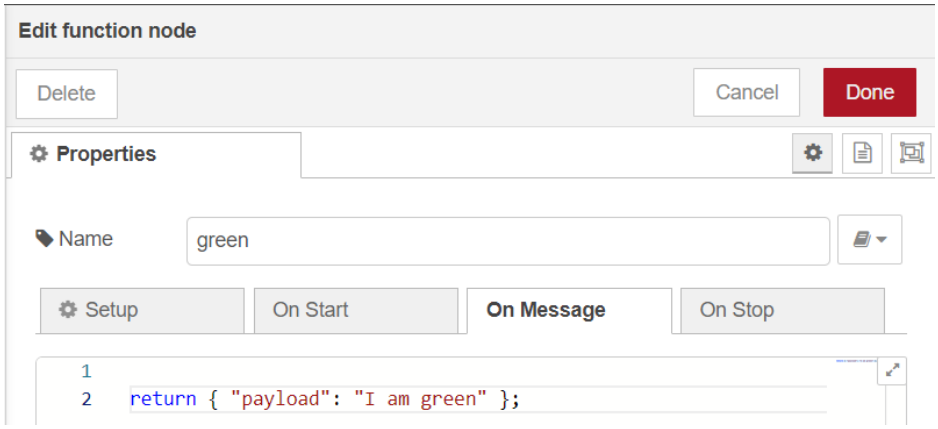


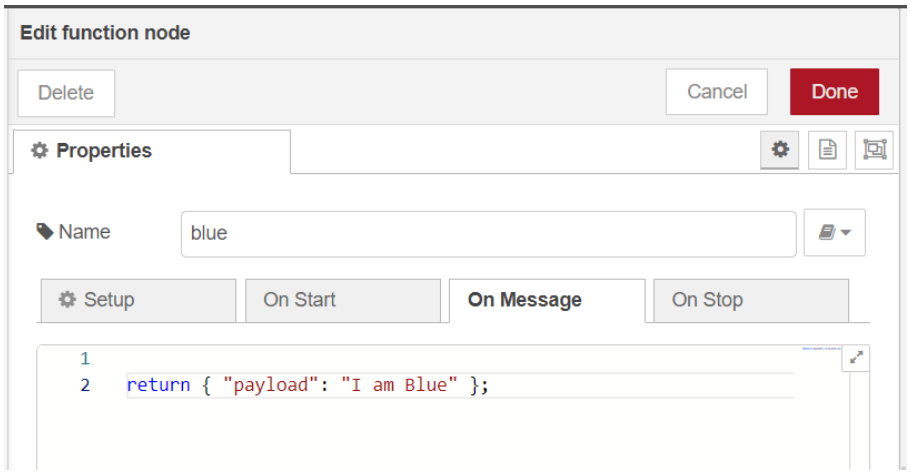
Create three cases as below



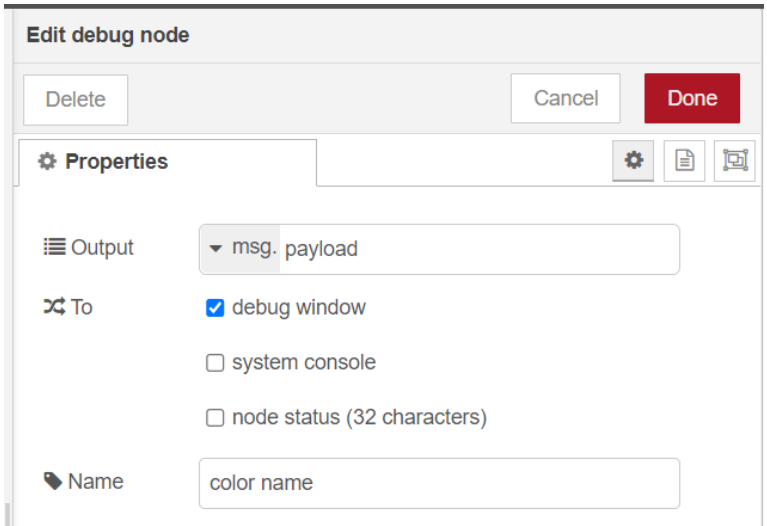
Assign a function to every case of the switch. Each return one of the color.



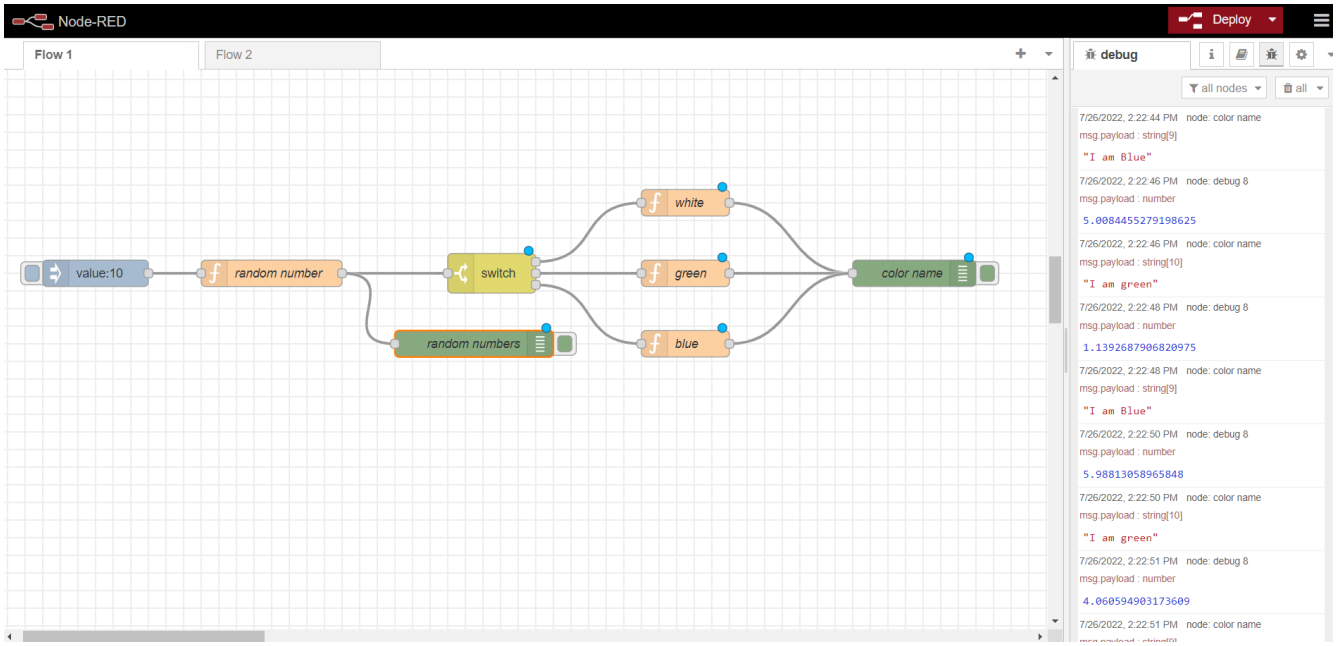




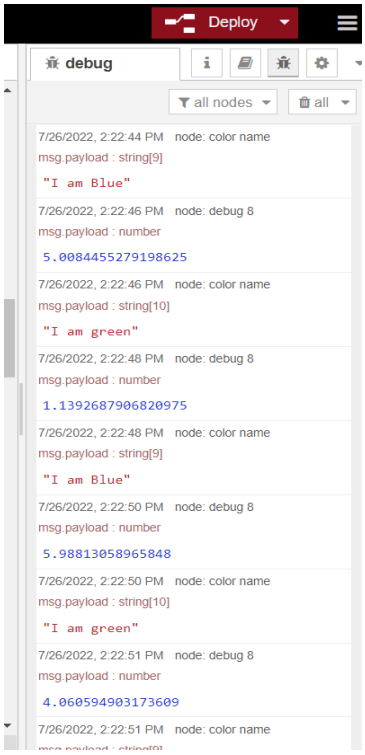
Create the debug node as follow and assign name to you.



Now the setup is ready, click on the inject node. This will trigger the random number function generating a random number and sending it to the switch node. The switch node based upon the number generated executes either of the three cases and send the output which is displayed on the debug window.



Random number and sting message will be displayed on the debug window as below -

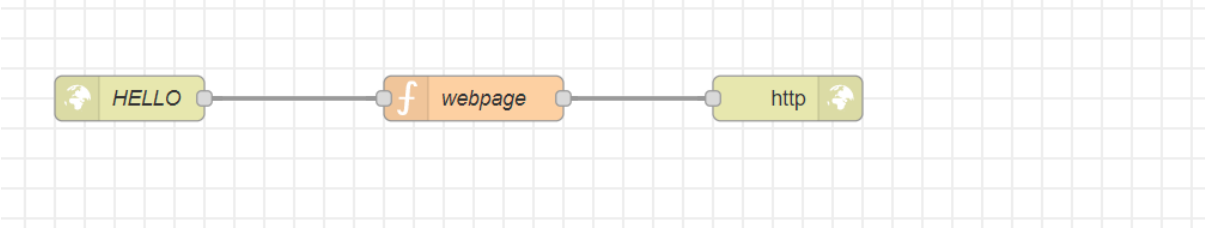


- Introducing Http node

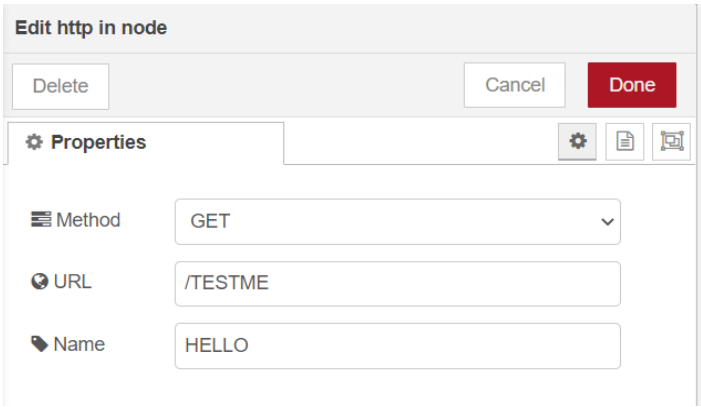
Http nodes are used in creating an HTTP endpoint that responds to GET requests with some static content, such as an HTML page or CSS stylesheet.

Use the HTTP In node to listen for requests, a Template node to include the static content, and an HTTP Response node to reply to the request.

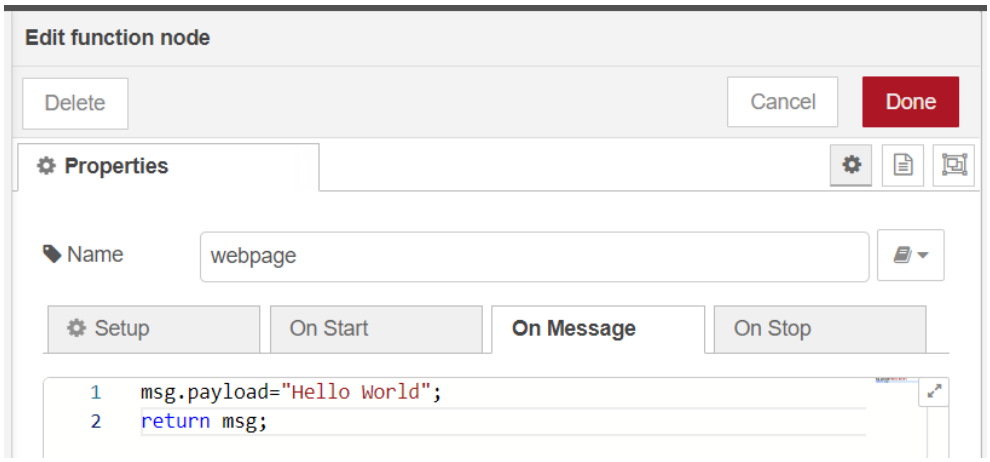
Use the http in and http response node with template and connect them as below



Add the URL in the http in node. This url will be used to open the web page.

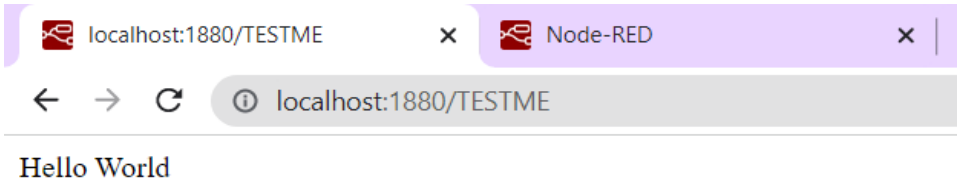


Add function code which returns a variable.

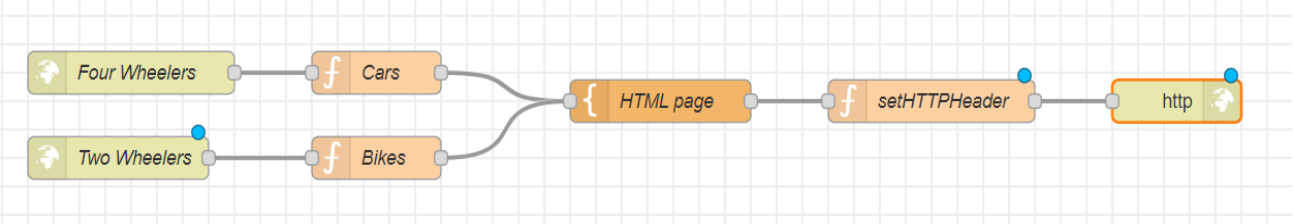


Once done, deploy these changes and open the link.

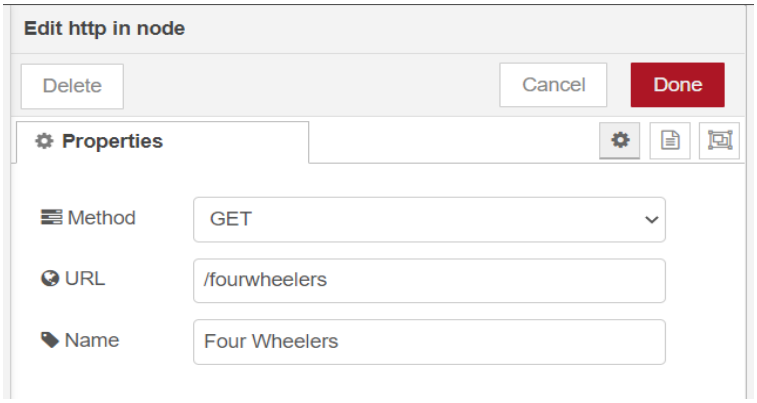
Link will be in the format localhost:1880/TESTME

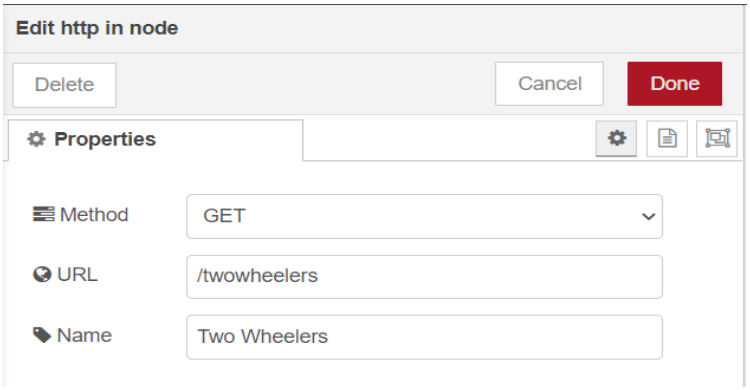


Arrange the nodes from the network as below. Use Http in, function, Template, http response nodes.

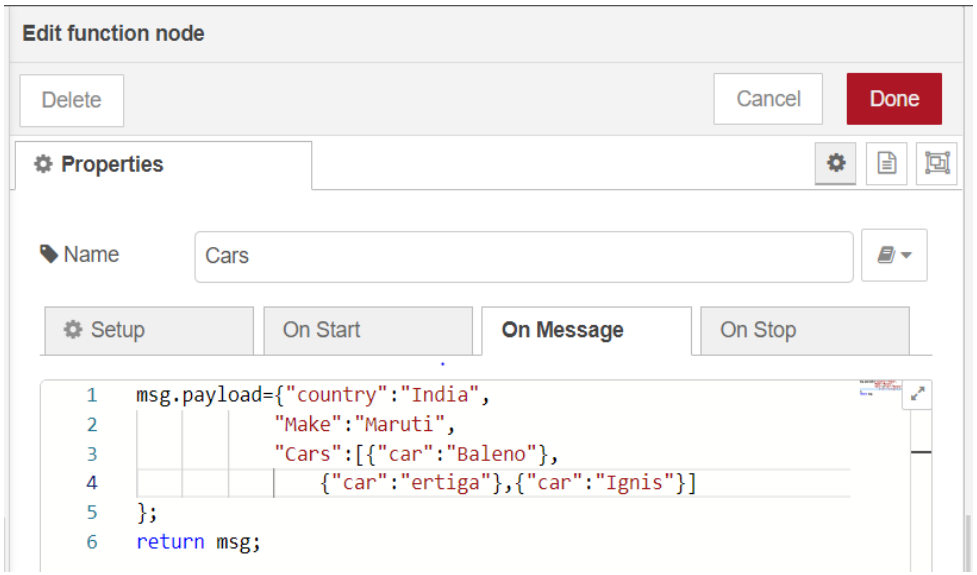


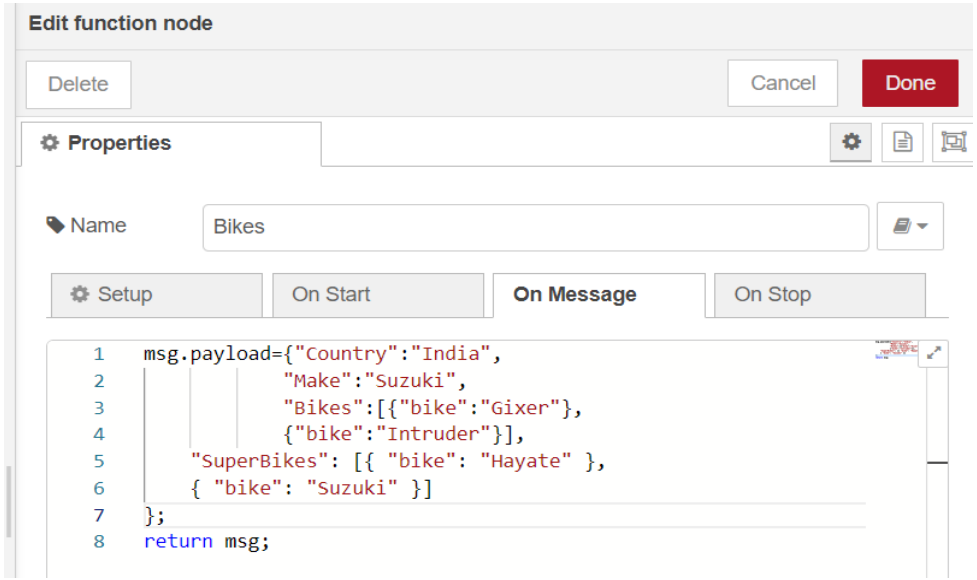
Edit the nodes as below, assign URL for both the http in nodes.



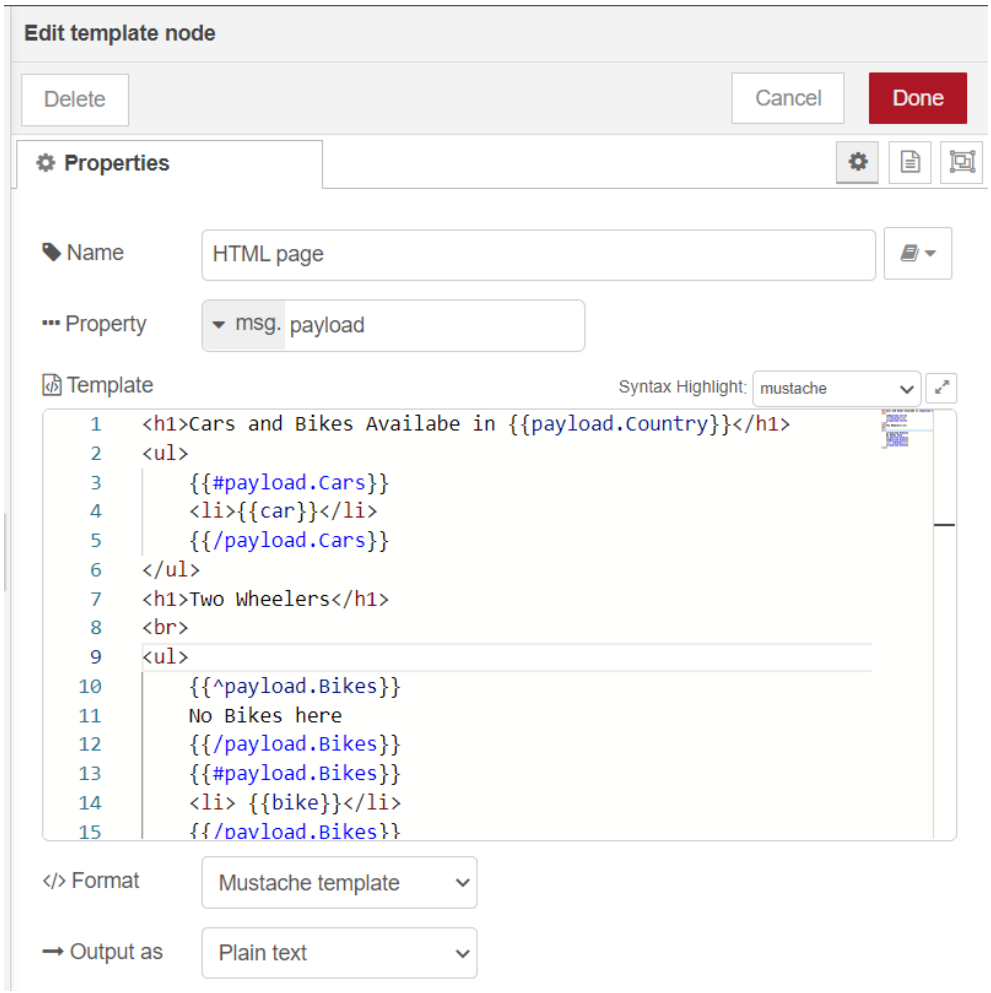


Add the dictionary for cars and bikes. Dictionary contains the country, make and a list of cars, same for bikes

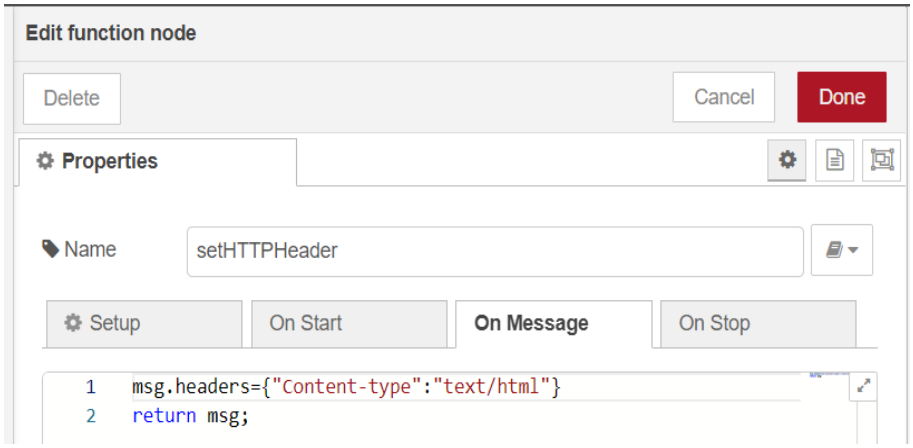

Edit function node
Delete
Cancel
Done
Properties
Name
Cars
Setup
On Start
On Message
On Stop
msg.payload={"country":"India",
"Make":"Maruti",
"Cars":[{"car":"Baleno"},
{"car":"ertiga"},{"car":"Ignis"}]
};
return msg;


Edit function node
Delete
Cancel
Done
Properties
Name
Bikes
Setup
On Start
On Message
On Stop
msg.payload={"Country":"India",
"Make":"Suzuki",
"Bikes":[{"bike":"Gixer"},
{"bike":"Intruder"}],
"SuperBikes": [{ "bike": "Hayate" },
{ "bike": "Suzuki" }]
};
return msg;

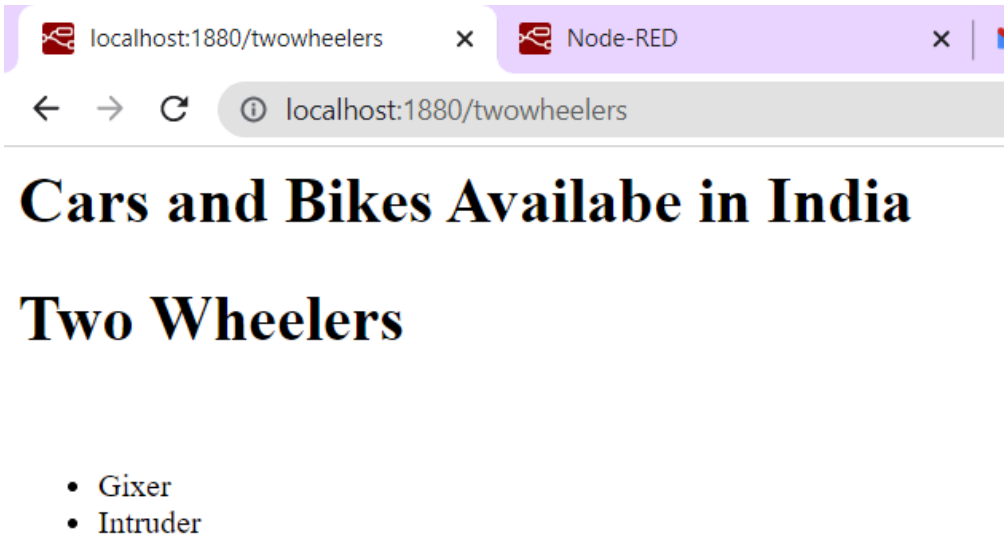
Add the below code in the template. Code renders a basic HTML page that takes up values from the dictionary created above.

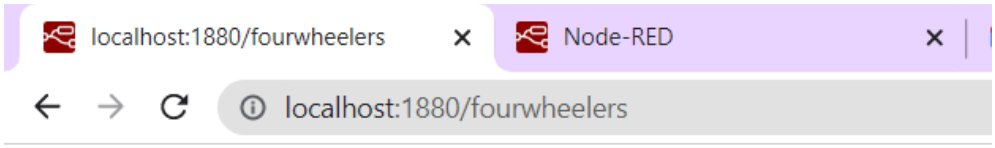


Set the header as content type for html.



Open the given url with node red localhost port. The output rendered will be as follows -





Cars and Bikes Availabe in

- Baleno
- ertiga
- Ignis

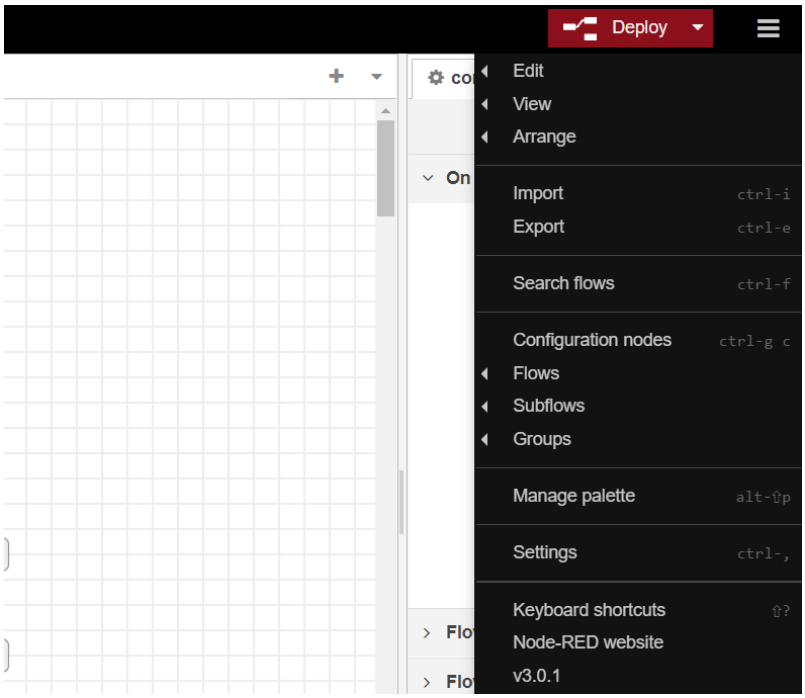
Two Wheelers

No Bikes here

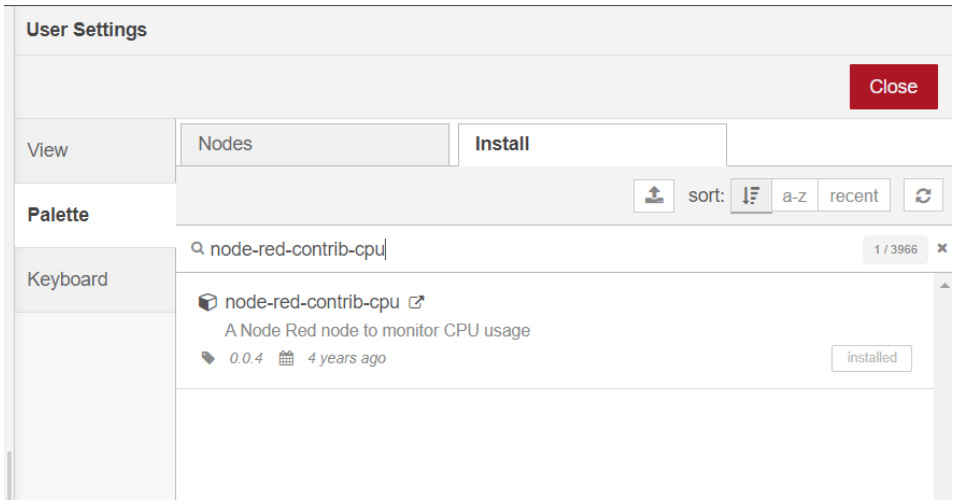
Practical 6

CPU utilization flow

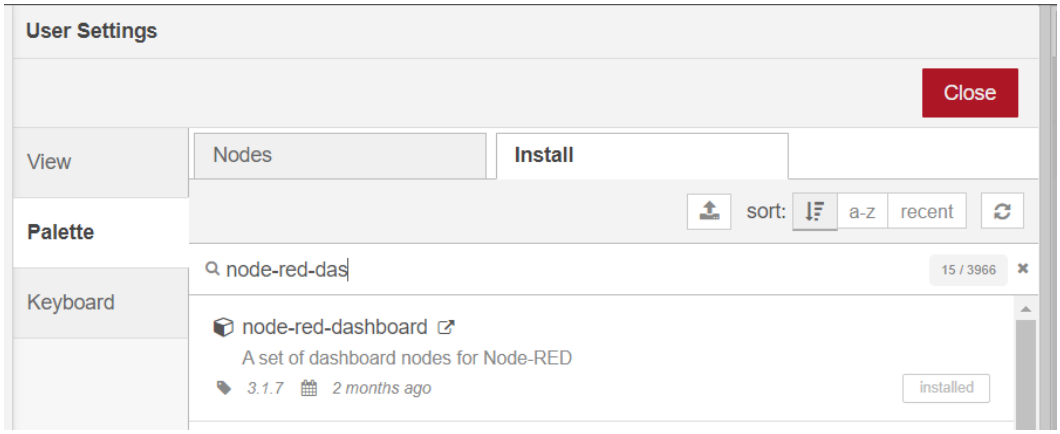
Go to the manage palette option from file menu



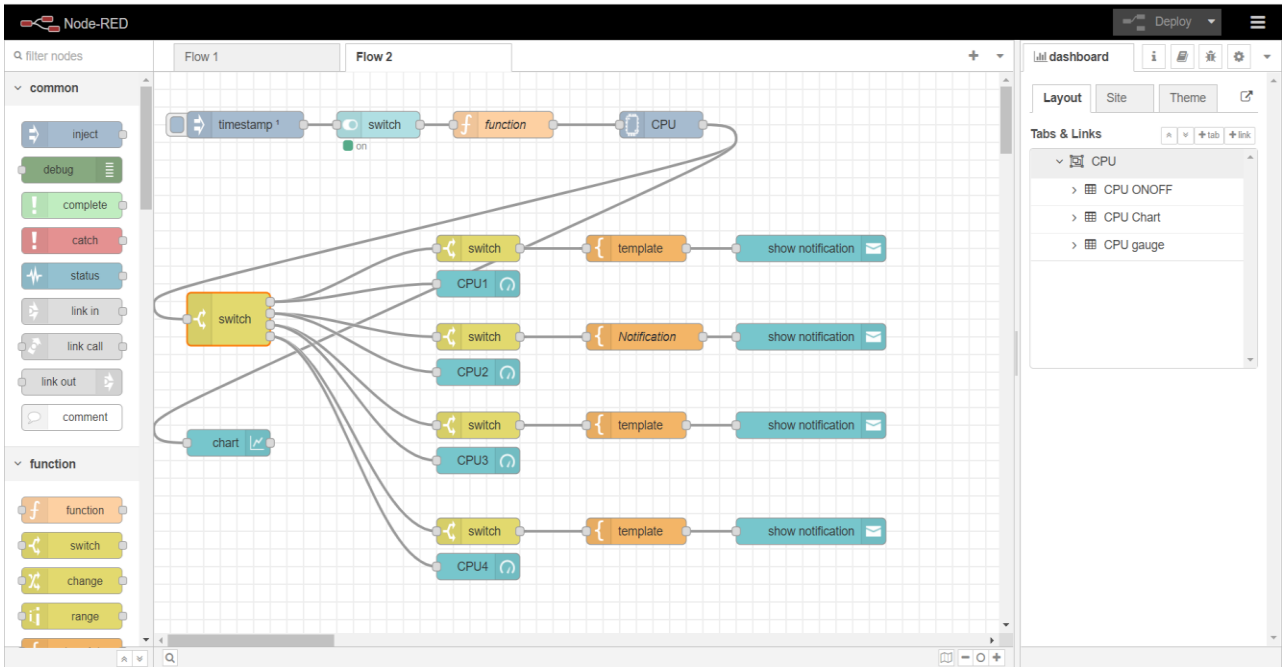
Under install tab search for **node-red-contrib-cpu** and click install.



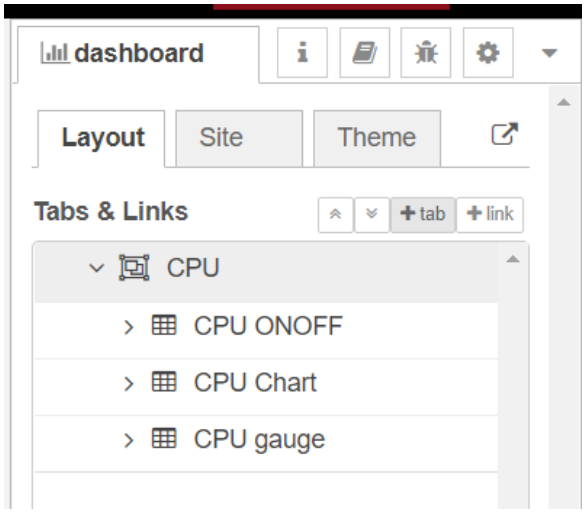
Next install **node-red-dashboard.** These two Palettes will install set of new nodes that would be required for generation of dashboard and other network specific application.



Create the Flow as below.

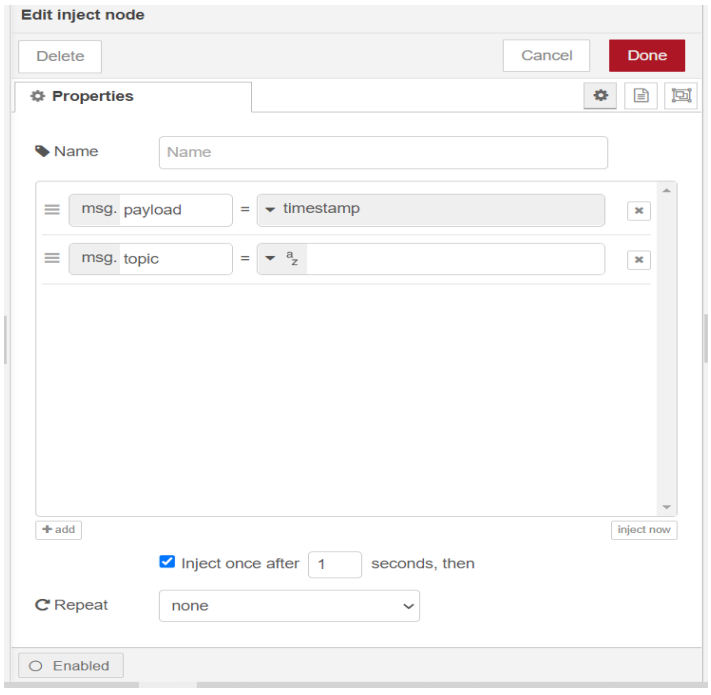


Create below groups from dashboard by clicking on '+tab' option

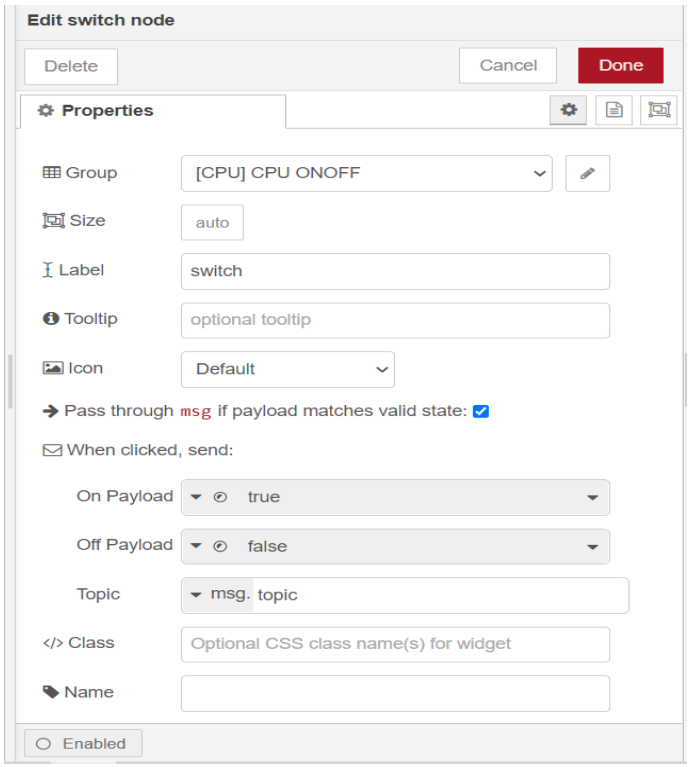


Edit the inject node properties as below -

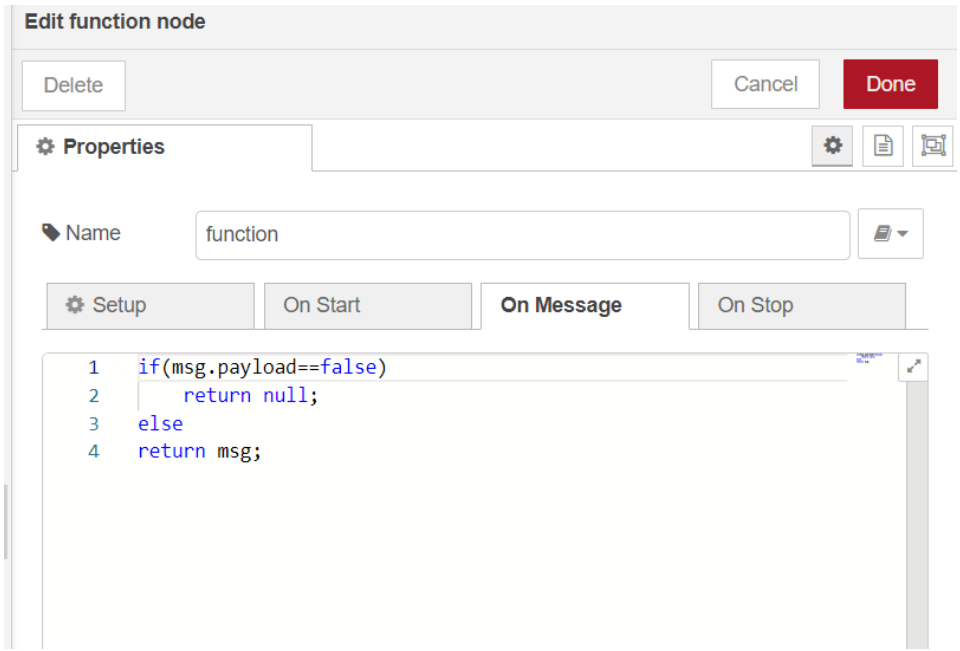
Select the inject node option to auto refresh / inject the data.



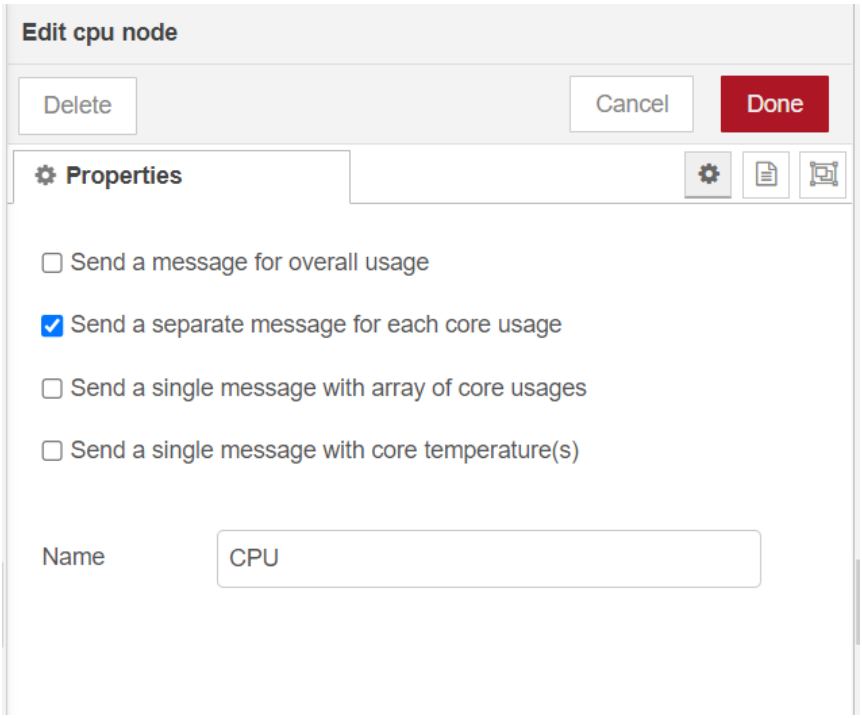
Next is the switch node, modify the properties as below -



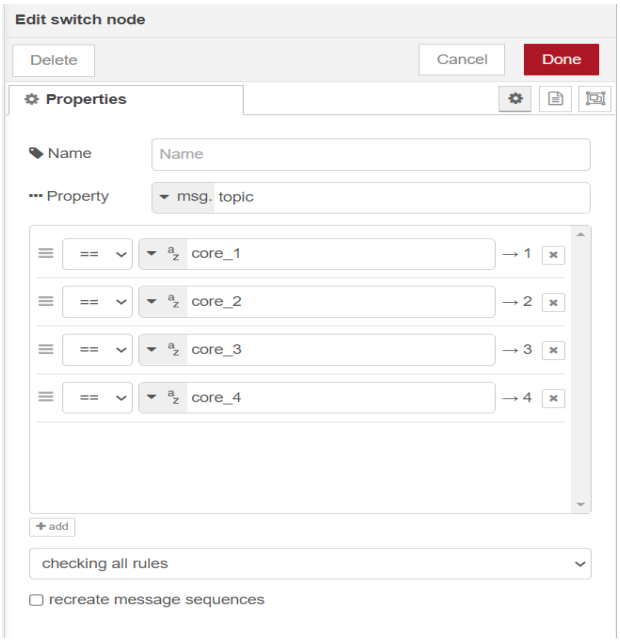
Function to control the flow according to the content in payload.



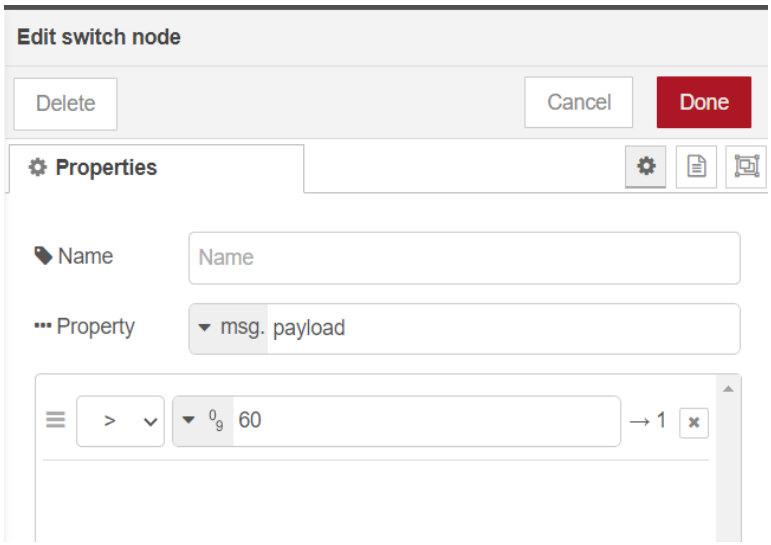
Next add the CPU node, this will act as the main CPU node of the machine which will be further divided into different cores.



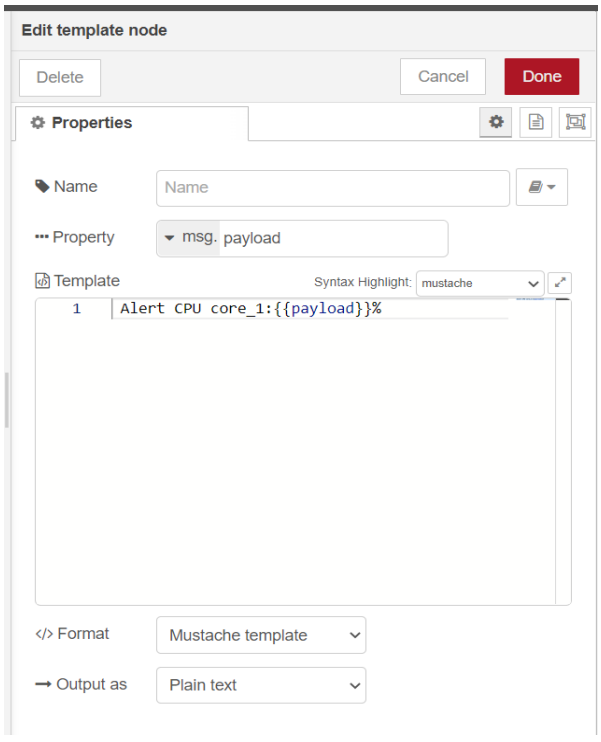
Next add a switch from network palette, this will act as main switch and divide the CPU flows into four components.



Add 4 switch nodes for every case and provide the below threshold condition for each of them.

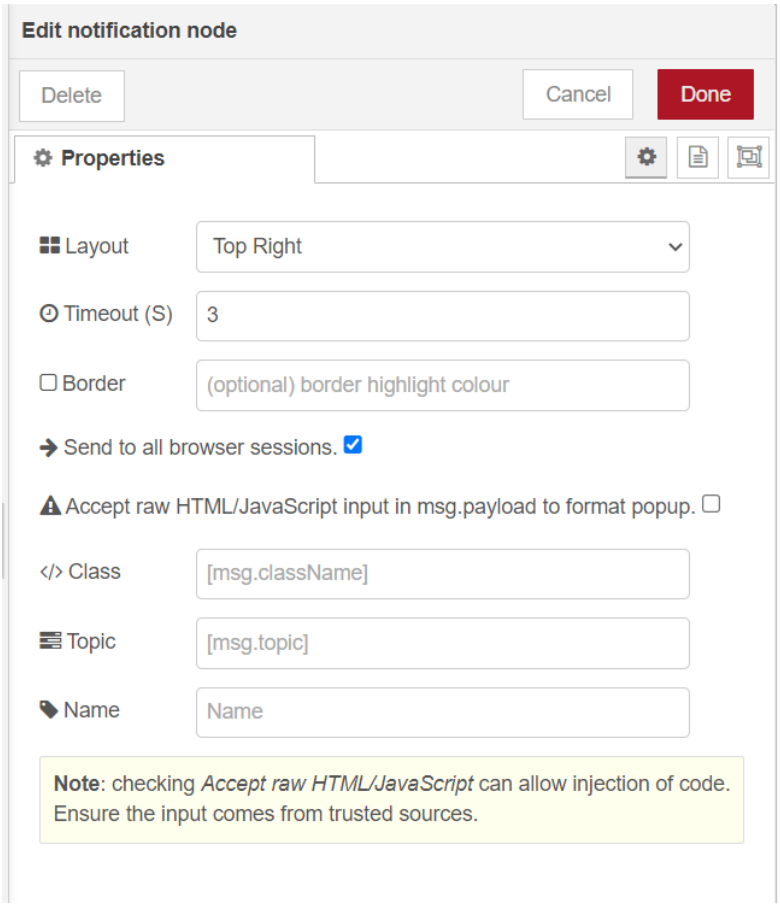


Template node for every case works by displaying an alert message of current utilization. It can be popped up using below code. Same code is to be added for every template node. Just change the core number.

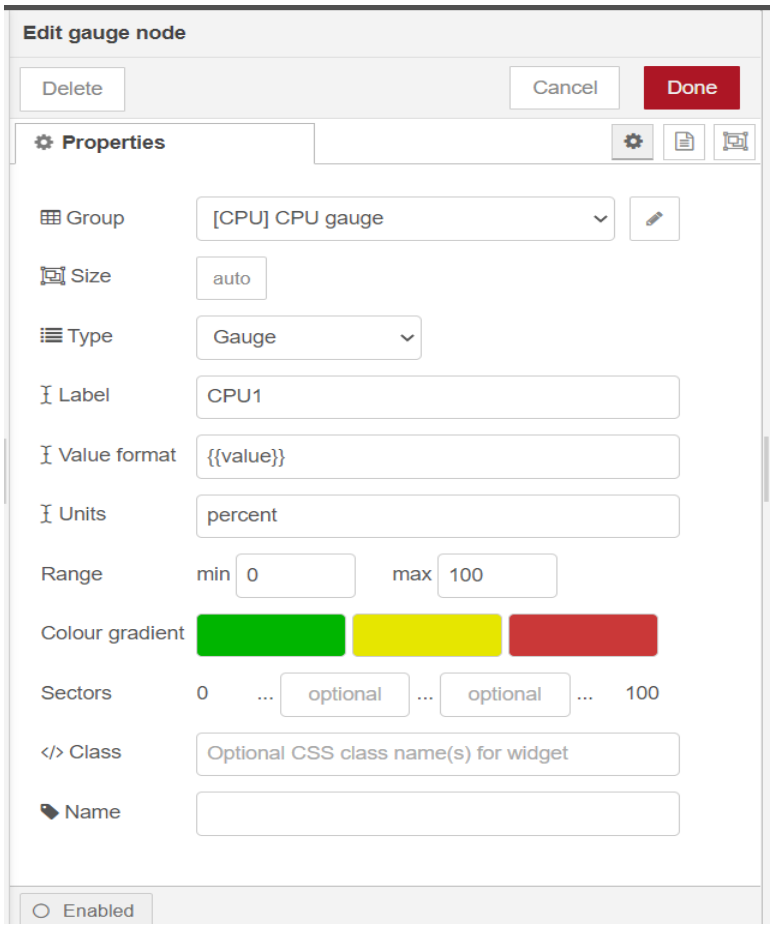


Each case even has a Show Notification function, the alert message configured in template will be displayed using this.

We can specify where this data appears on the screen by choosing the appropriate Layout option as below.

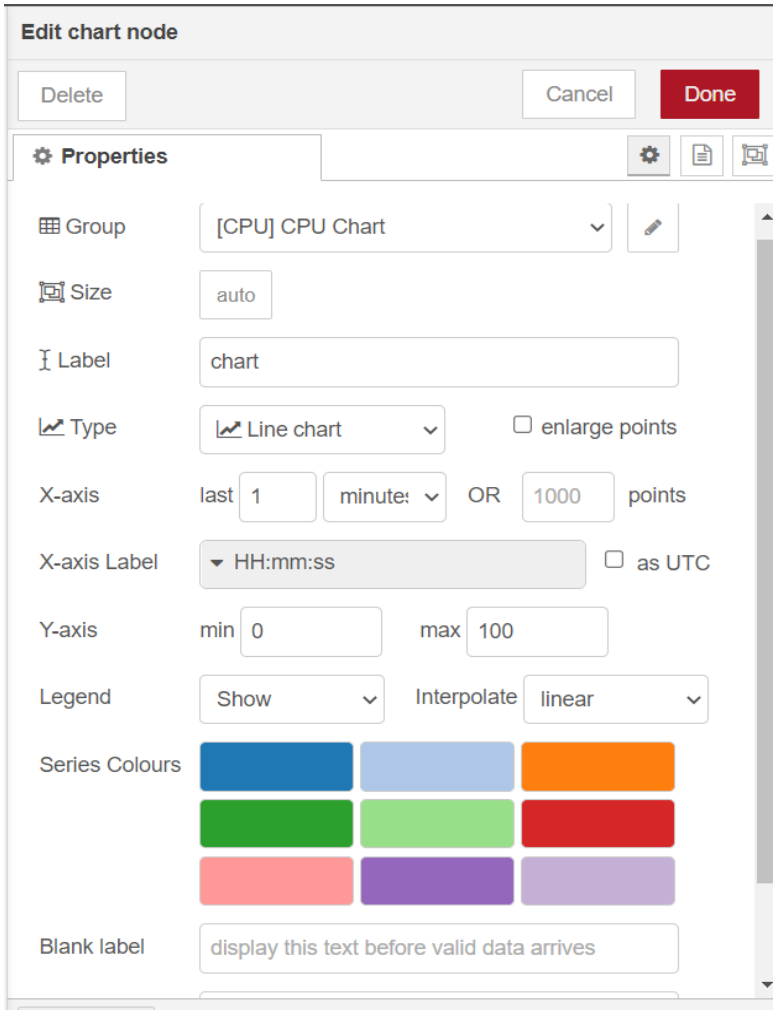


Next is add the gauge, this will act as virtual display for the CPU Utilization. Select the appropriate group and add the details as below



Next is to add properties to chart, which will display overall performance by creating graph for every core.

Select the appropriate group, define x and y measures and limits.



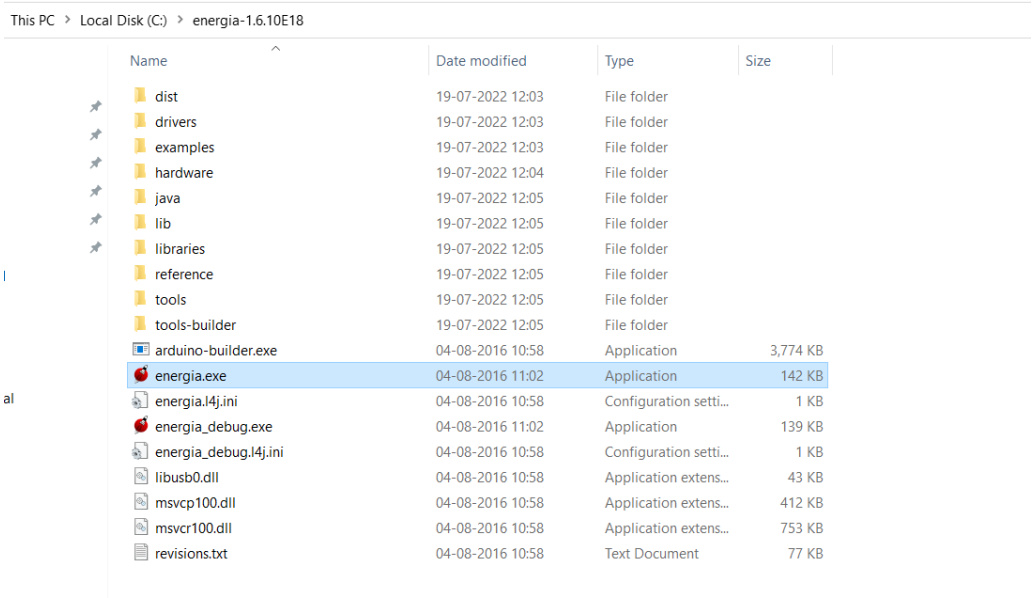
Enable the legend by selecting show for legend as by default it's not selected.

Once all changes are done click on the deploy. Now go to the dashboard and click on the open button, this will display the below dashboard.

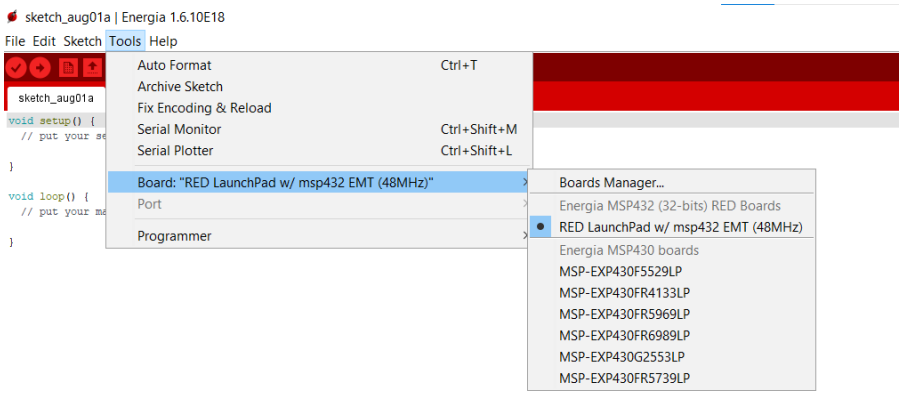
Practical 7

WiFi Setup

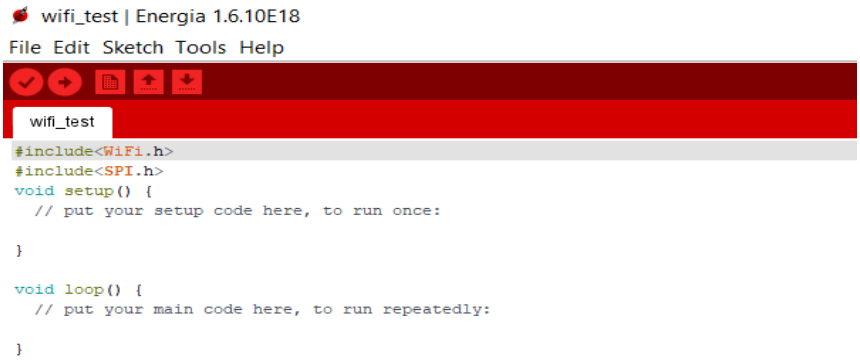
Go to C:\energia-1.6.10E18 and Launch energia.exe



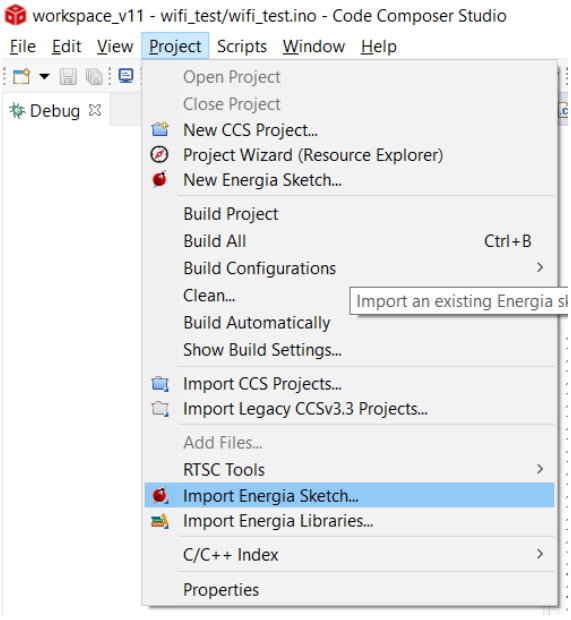
Under Tools select RED LaunchPad w/msp432 EMT (48 MHz)



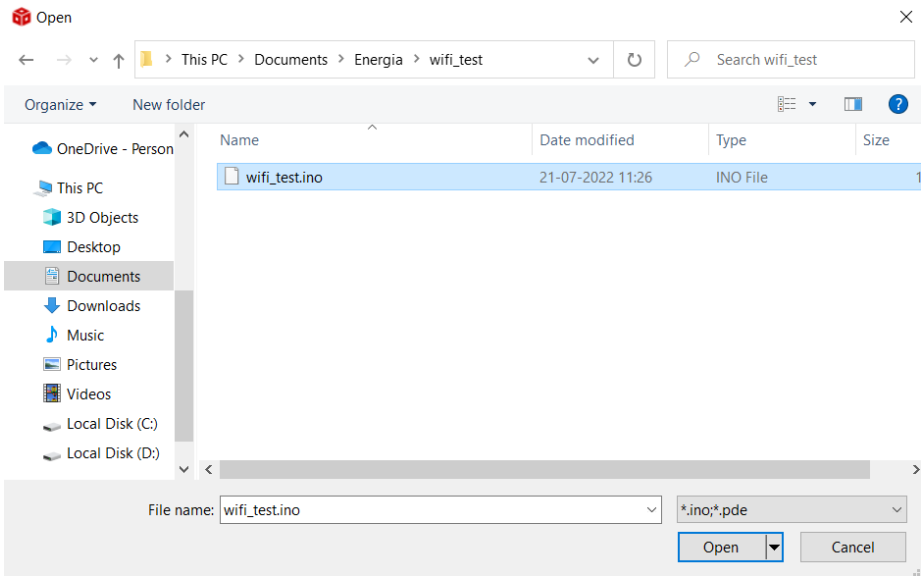
Add two API -> WiFi and SPI as below and save the file as wifi



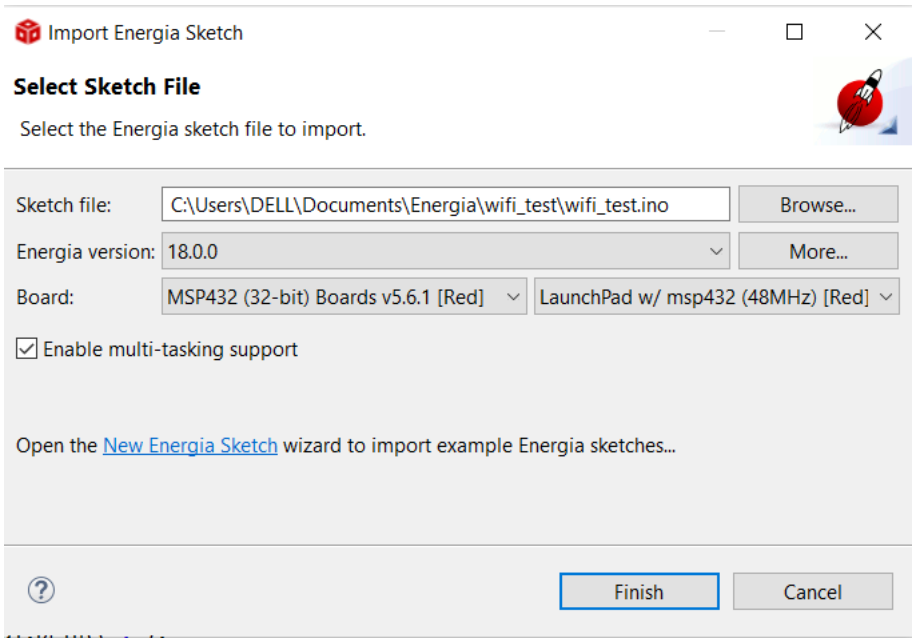
Now go to Code Composer Studio, and under Project Menu Import the energia sketch option



Browse and select the energia sketch saved as **wifi_test.ino**



Check if remaining setting are as follows and click finish



Add the below code into the file.

Wifi_test.ino:

```
#include <WiFi.h>
#include <SPI.h>
//#define SSID "octanauts"
//#define PASSWORD "aditya30"
IPAddress shieldIP,subnetMask,gatewayIP;
uint8_t rssi;
uint8_t networkId;
byte macAddr[6]; //mac address is 6 bytes
byte encryptionType;
char ssid[]="S";//my cell phone
void setup() {
 // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.print("Connecting to WiFi..");
  while(WiFi.begin(ssid)!=WL_CONNECTED)  //for cell phone  with password:-
while(WiFi.begin(ssid,password)!=WL_CONNECTED)
  {
     Serial.print(".");
     delay(1);
  }
  Serial.println("");
  Serial.print("WiFi connected , Fetching WiFi sheild's IP Address:");
  while(WiFi.localIP()==INADDR_NONE ) {
     Serial.print(".");
     delay(1);
  }
  shieldIP = WiFi.localIP();
  Serial.println(shieldIP);
```

```
Serial.print("Access point name :");
Serial.println(ssid);
Serial.print("Signal strength: ");
rssi=WiFi.RSSI();
Serial.println(rssi);

uint8_t networkId = WiFi.scanNetworks();
Serial.print("Number of access points in range:");
Serial.println(networkId);
for(int i=1;i<=networkId;i++){
   Serial.print("Name of Access Points and encryption type:");
   Serial.print(WiFi.SSID(i));
   Serial.print(",");
   encryptionType = WiFi.encryptionType(i);
   //
   Serial.println(encryptionType,DEC);
}

subnetMask = WiFi.subnetMask();
Serial.print("Subnet Mask:");
Serial.println(subnetMask);

gatewayIP = WiFi.gatewayIP();
Serial.print("Gateway IP address:");
Serial.println(gatewayIP);



WiFi.macAddress(macAddr);
Serial.print("Mac Address of shield:");
```

```
  for(int i =0;i<6;i++){
     Serial.print(macAddr[i],HEX);
     if(i<=4)
        Serial.print(':');
     else
        Serial.println();
  }
  }

void loop() {
 // put your main code here, to run repeatedly
}
```

Once Done, Compile the code to check for any errors. If the Build is successful, connect the board.

wifi_test.ino

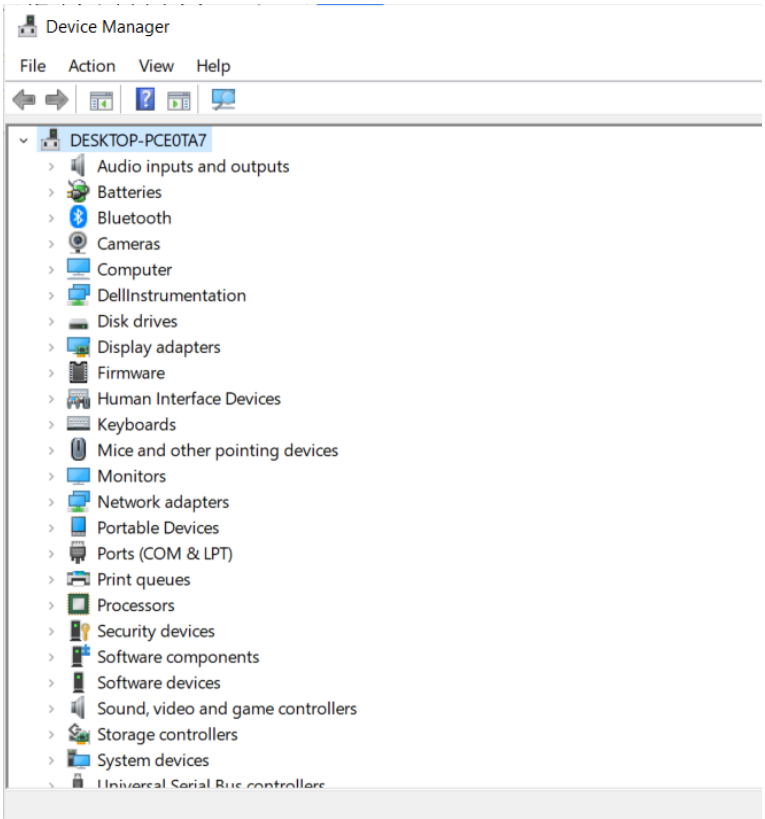
```
#include <WiFi.h>
#include <SPI.h>

//#define SSID "octanauts"
//#define PASSWORD "aditya30"
IPAddress shieldIP,subnetMask,gatewayIP;
uint8_t rssi;
uint8_t networkId;
byte macAddr[6]; //mac address is 6 bytes
byte encryptionType;

char ssid[]="S";//my cell phone


void setup() {
  // put your setup code here, to run once:
   Serial.begin(115200);
   Serial.print("Connecting to WiFi..");
   while(WiFi.begin(ssid)!=WL_CONNECTED)  //for cell phone  with password:- while(Wi

   {
       Serial.print(".");
       delay(1);
   }
```

Console

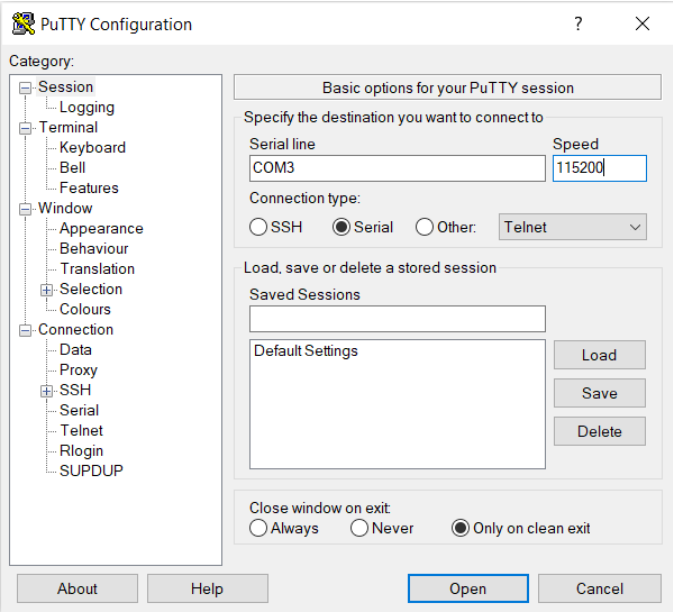
CDT Build Console [wifi_test]

```
**** Build of configuration Debug for project wifi_test ****

"C:\\ti\\ccs1120\\ccs\\utils\\bin\\gmake" -k -j 4 all -O

gmake[1]: 'wifi_test.out' is up to date.

**** Build Finished ****
```

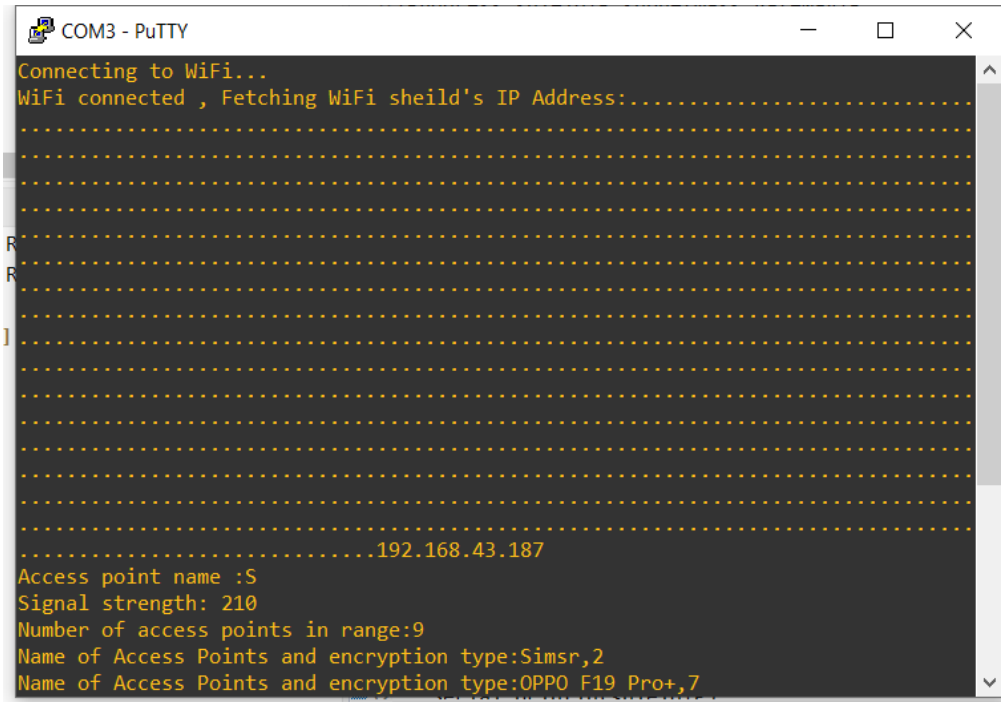
After Connecting the board check the port on which it is connected using the device manager

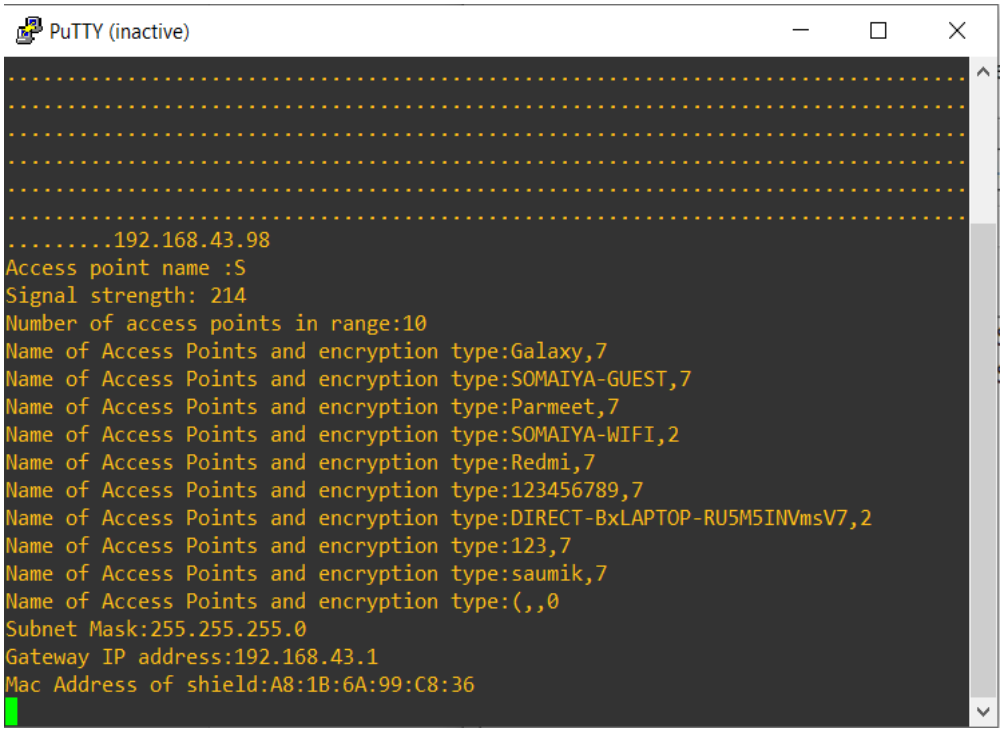


If the board is connected onto port - **COM3**. Open the same using putty along with below configuration -



Now burn the code on chip using Debug option and then click on the resume option for the program to start running. This will start the code execution and produce the below output -





Putty Terminal is used to check the output generated. It gives away the Connected wifi's information as well as the nearby connections detected by the wifi chip on the board.

Practical 8

Analog To Digital Converter

An ADC (Analog to Digital Converter) is an electronic integrated circuit which converts analog signals to digital signals.

IOT projects normally would connect different sensors to obtain information regarding the physical world and do some processing based on that information.

When we use analog sensors and communicate with a microcontroller, it is not possible for the microcontroller to directly understand these analog signals because microcontrollers only understand digital signals which are formed by 1's and 0's.

Include the following APIs in the energia sketch file.

All the APIs included serve some purpose, listed as below

WiFi - Enables network connection (local and Internet) using the board's WiFi shield.

SLFS - SimpleLink WiFi Filesystem I/O API

WiFiClient - Creates a client that can connect to a specified internet IP address and port as defined in client.

WiFiServer – To create a server that listens for incoming connections on the specified port.

WiFiUdp – To send and receive UDP messages

SPI - This library allows you to communicate with SPI devices, with the Arduino as the controller device.

Code:

```
#include <ti/devices/msp432p4xx/driverlib/driverlib.h>
#include<WiFi.h>
#include<SLFS.h>
#include<WiFiClient.h>
#include<WiFiServer.h>
#include<WiFiUdp.h>
#include<SPI.h>
IPAddress shieldIP,subnetMask,gatewayIP;
uint8_t rssi;
uint8_t networkId;
byte macAddr[6]; //mac address is 6 bytes
byte encryptionType;
uint16_t cResult;

char ssid[]="S";//my cell phone

void setup() {
  // put your setup code here, to run once:
    Serial.begin(115200);
    Serial.print("Connecting to WiFi..");
    while(WiFi.begin(ssid)!=WL_CONNECTED)  //for cell phone  with password:-
while(WiFi.begin(ssid,password)!=WL_CONNECTED)

    {
      Serial.print(".");
      delay(1);
    }
    Serial.println("");
    Serial.print("WiFi connected , Fetching WiFi sheild's IP Address:");
    while(WiFi.localIP()==INADDR_NONE ) {
      Serial.print(".");
      delay(1);
    }
    shieldIP = WiFi.localIP();
    Serial.println(shieldIP);

    Serial.print("Access point name :");
    Serial.println(ssid);

    Serial.print("Signal strength: ");
    rssi=WiFi.RSSI();
    Serial.println(rssi);
    uint8_t networkId = WiFi.scanNetworks();
    Serial.print("Number of access points in range:");
    Serial.println(networkId);
    for(int i=1;i<=networkId;i++){
      Serial.print("Name of Access Points and encryption type:");
      Serial.print(WiFi.SSID(i));
      Serial.print(",");
      encryptionType = WiFi.encryptionType(i);
```

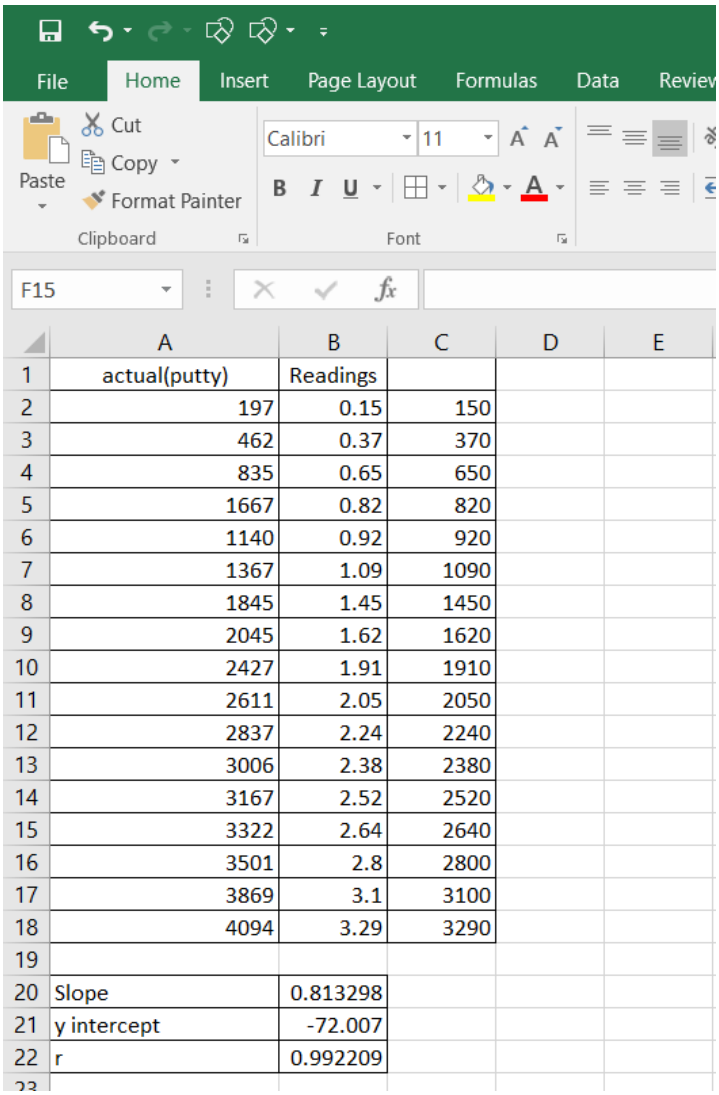
```
        Serial.println(encryptionType,DEC);
    }

    subnetMask = WiFi.subnetMask();
    Serial.print("Subnet Mask:");
    Serial.println(subnetMask);

    gatewayIP = WiFi.gatewayIP();
    Serial.print("Gateway IP address:");
    Serial.println(gatewayIP);

    WiFi.macAddress(macAddr);
    Serial.print("Mac Address of shield:");
    for(int i =0;i<6;i++){
        Serial.print(macAddr[i],HEX);
        if(i<=4)
            Serial.print(':');
        else
            Serial.println();
    }
    GPIO_setAsOutputPin(GPIO_PORT_P2,GPIO_PIN0|GPIO_PIN1|GPIO_PIN2);

GPIO_setAsPeripheralModuleFunctionInputPin(GPIO_PORT_P4,GPIO_PIN7,GPIO_TERTIARY_MODULE_FUNCTION);
        ADC14_initModule(ADC_CLOCKSOURCE_MCLK,ADC_PREDIVIDER_1,ADC_PREDIVIDER_1,ADC_NOROUTE);
        ADC14_configureSingleSampleMode(ADC_MEM6, true); /* out put is stored here in ADC_MEM6 */

ADC14_configureConversionMemory(ADC_MEM6,ADC_VREFPOS_AVCC_VREFNEG_VSS,ADC_INPUT_A6,false);
//2nd parameter is an macros that tells the reference voltage is 3.3V
        ADC14_setSampleHoldTime(ADC_PULSE_WIDTH_32,ADC_PULSE_WIDTH_4); //creating pulse with 64 clock
cycles, again here 2 parameters are min req is of 2; 2nd one stands useless as ADC_INPUT_6 will come in first 16 i.e
the first parameter
        ADC14_setResolution(ADC_12BIT);
        ADC14_enableSampleTimer(ADC_AUTOMATIC_ITERATION);// tells to keep iterating
        ADC14_enableModule();//to start the module
        ADC14_enableConversion();//start ADC conversion
        ADC14_toggleConversionTrigger(); // trigger the conversion continuously

}

void loop() {
  // put your main code here, to run repeatedly:
  int Result,regressedData1;
  float regressedData;
  while(!ADC14_isBusy());
            Result=ADC14_getResult(ADC_MEM6);
            P2OUT=Result>>8;
            Serial.println(Result);
            delay(500);
            ADC14_toggleConversionTrigger();
}
```

PuTTY (inactive)

```
Connecting to WiFi....
WiFi connected , Fetching WiFi sheild's IP Address:.............................
.192.168.43.232
Access point name :S
Signal strength: 201
Number of access points in range:8
Name of Access Points and encryption type:govind,7
Name of Access Points and encryption type:RAJ,2
Name of Access Points and encryption type:Office TV,7
Name of Access Points and encryption type:abcdef,2
Name of Access Points and encryption type:shanmukh,7
Name of Access Points and encryption type:Redmi,7
Name of Access Points and encryption type:206STUD60 9701,2
Name of Access Points and encryption type:(,,0
Subnet Mask:255.255.255.0
Gateway IP address:192.168.43.1
Mac Address of shield:60:64:5:A6:24:DA
Connecting to WiFi....
WiFi connected , Fetching WiFi sheild's IP Address:.............................
.....................................................192.168.43.232
Access point name :S
Signal strength: 197
Number of access points in range:7
Name of Access Points and encryption type:vivoY73,7
Name of Access Points and encryption type:govind,7
Name of Access Points and encryption type:shanmukh,7
Name of Access Points and encryption type:Redmi,7
Name of Access Points and encryption type:abcdef,2
Name of Access Points and encryption type:206STUD60 9701,2
Name of Access Points and encryption type:(,,0
Subnet Mask:255.255.255.0
Gateway IP address:192.168.43.1
Mac Address of shield:60:64:5:A6:24:DA
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
```

Programs first executes wifi code that connects to provided network and scans for nearby networks and fetches its name and encryption type. Next when it executes ADC code, it send the converted voltage of the potentiometer as output. Initially 0, can be increased or decreased by rotating the potentiometer

Connect Potentiometer. Increase voltage and mark the readings



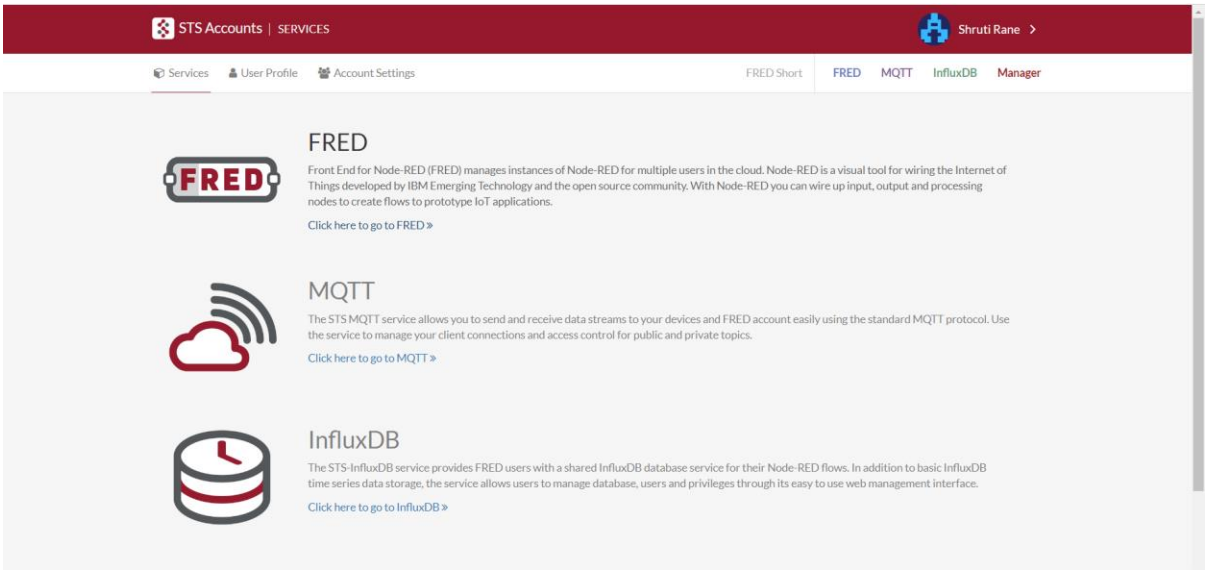
	A	B	C	D	E
1	actual(putty)	Readings			
2	197	0.15	150		
3	462	0.37	370		
4	835	0.65	650		
5	1667	0.82	820		
6	1140	0.92	920		
7	1367	1.09	1090		
8	1845	1.45	1450		
9	2045	1.62	1620		
10	2427	1.91	1910		
11	2611	2.05	2050		
12	2837	2.24	2240		
13	3006	2.38	2380		
14	3167	2.52	2520		
15	3322	2.64	2640		
16	3501	2.8	2800		
17	3869	3.1	3100		
18	4094	3.29	3290		
19					
20	Slope	0.813298			
21	y intercept	-72.007			
22	r	0.992209			

Extract out the slope, y-intercept and error values based upon the readings taken

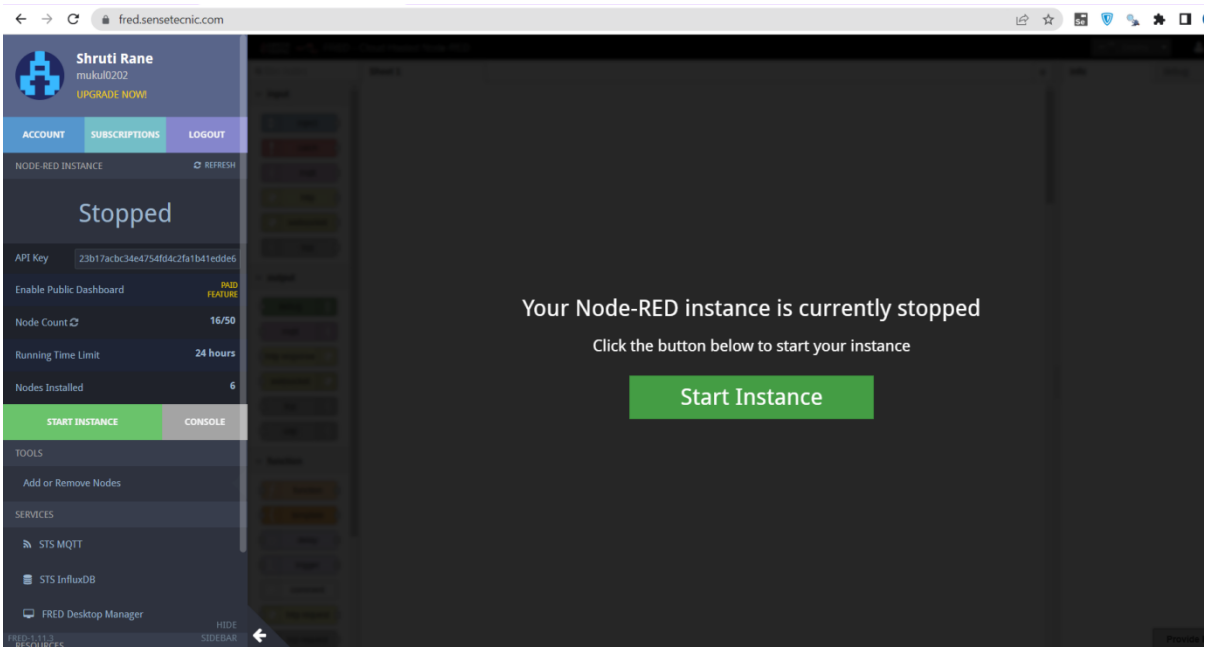
Practical 9

Sending Raw data to the cloud

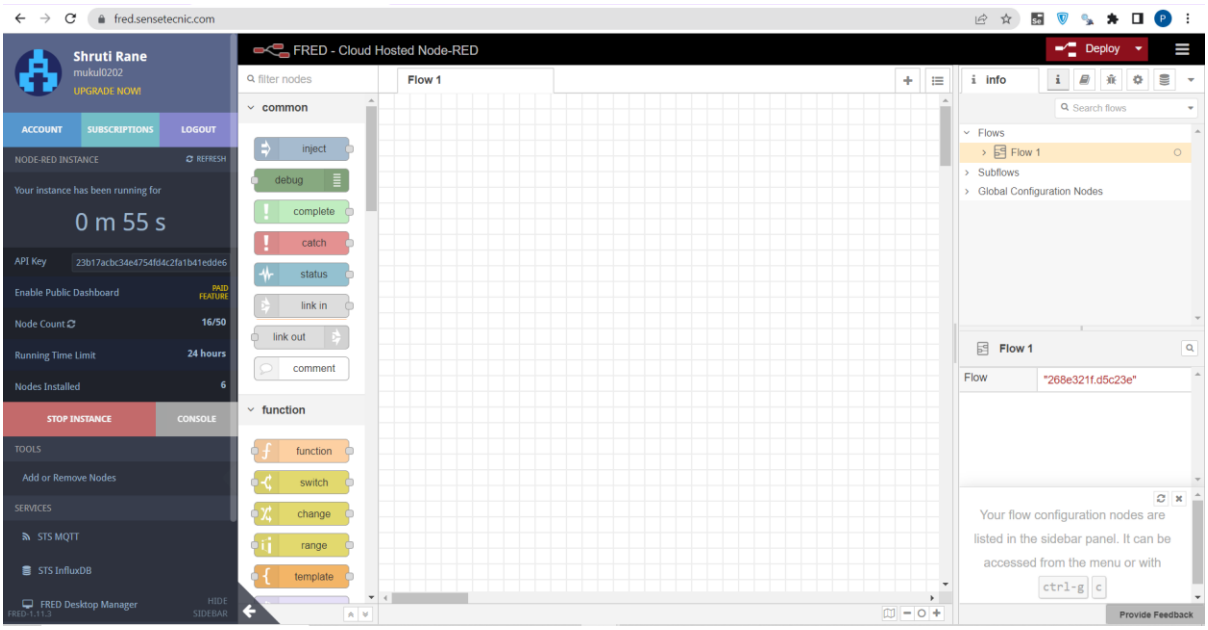
Create Account on FRED



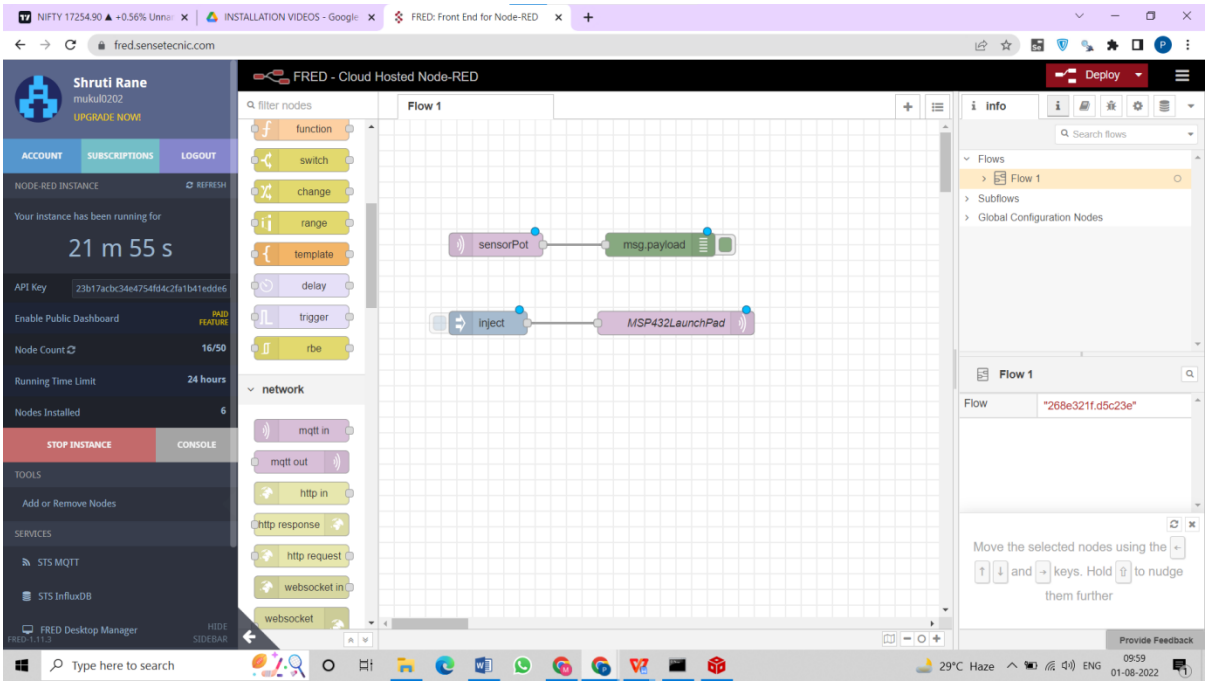
Once created, start the node-red instance



It has the same canvas as that in local node-red



Create the flow as below. Add mqttIn and mqttOut node

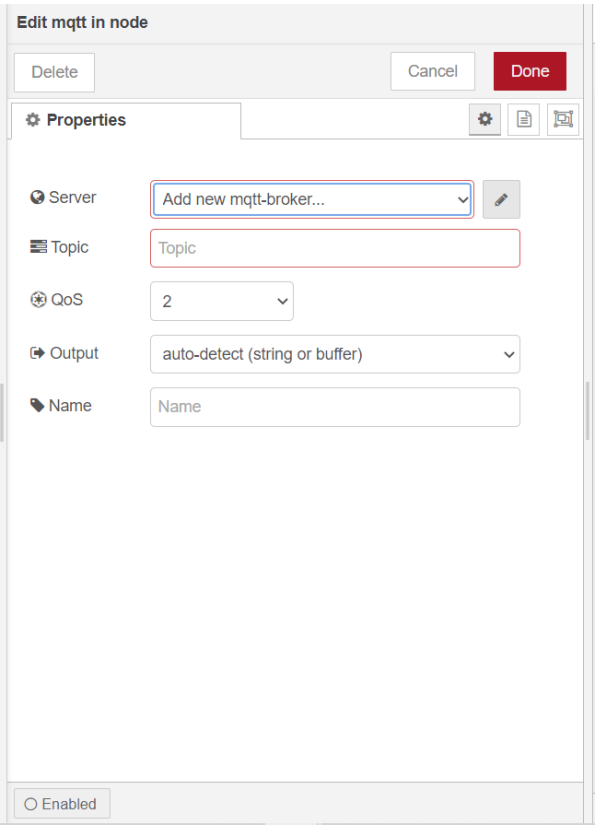


Node-Red provides both an MQTT subscribe (input) and publish (output) node.

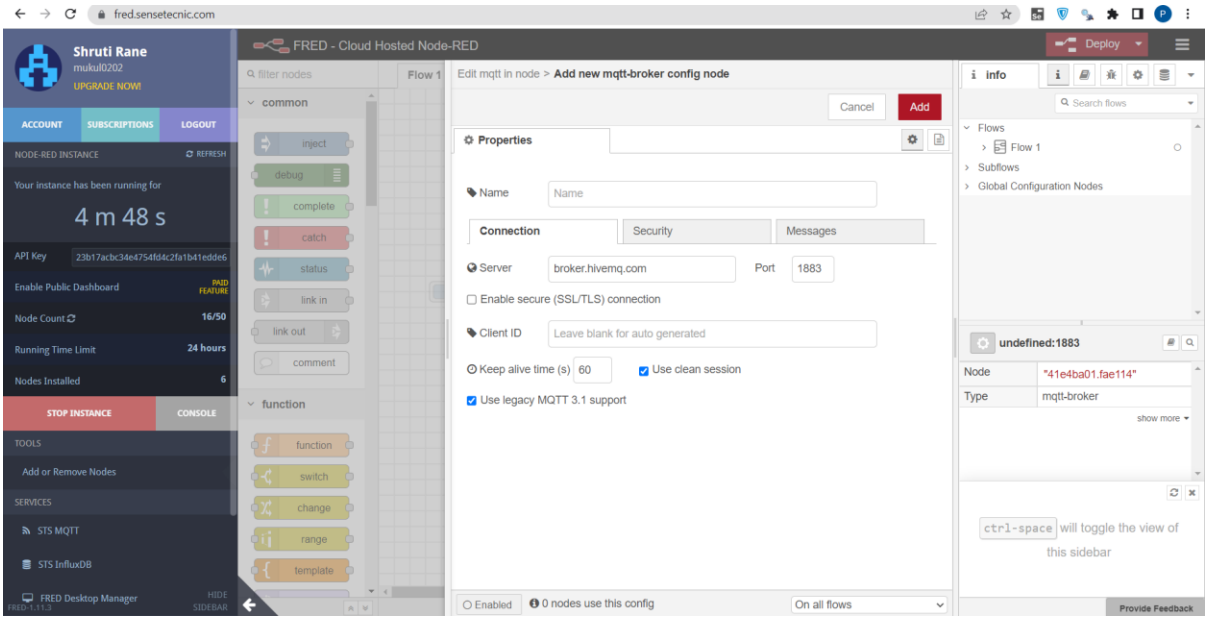
The configuration for these nodes are almost Identical as the main part of the configuration concerns the actual client connection.

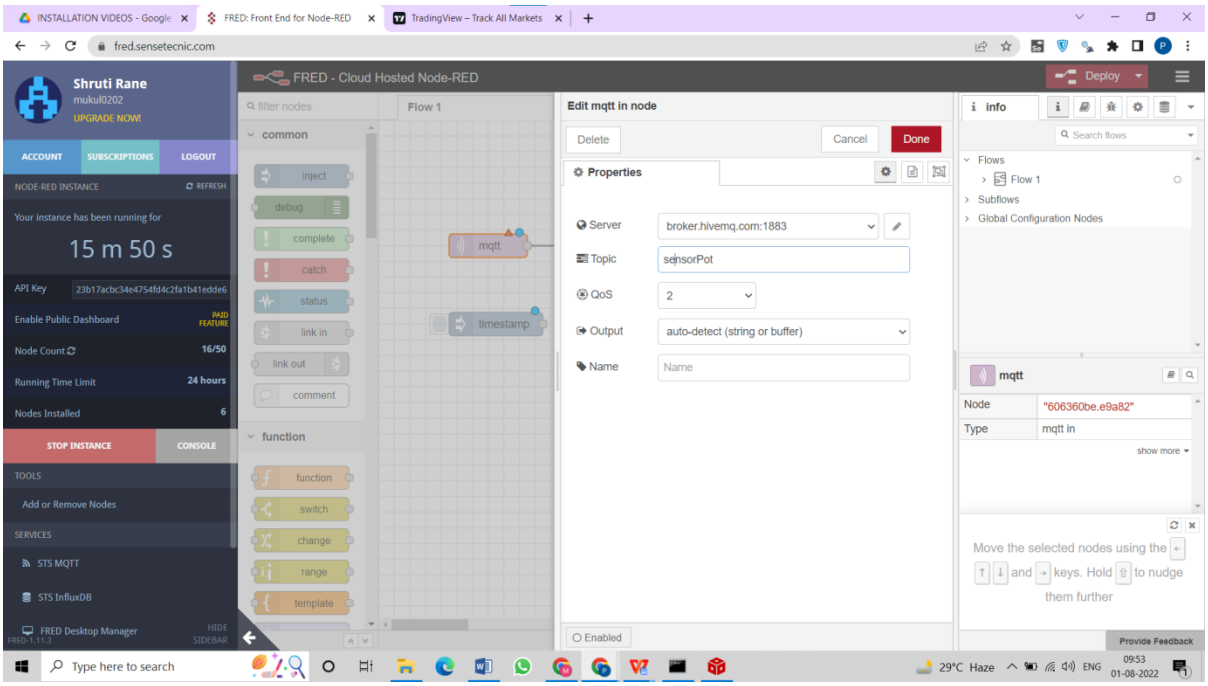
Edit the mqqttin node and add the below properties.

Click on the edit option to create new broker

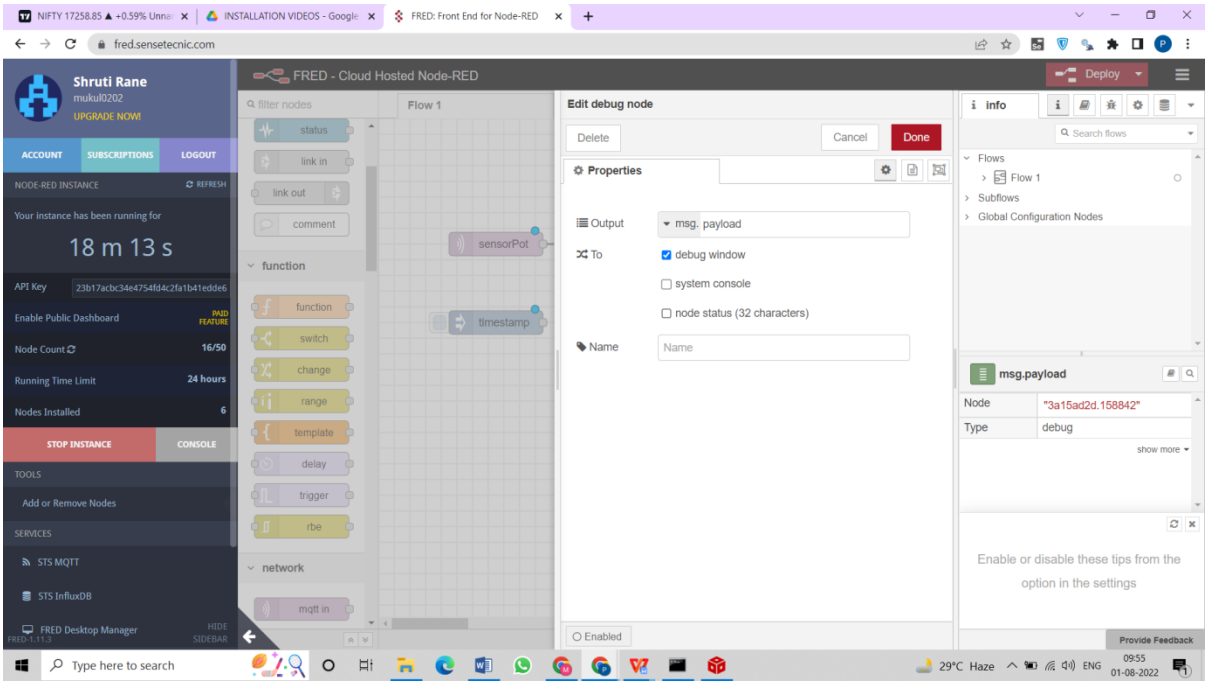


And add the below details

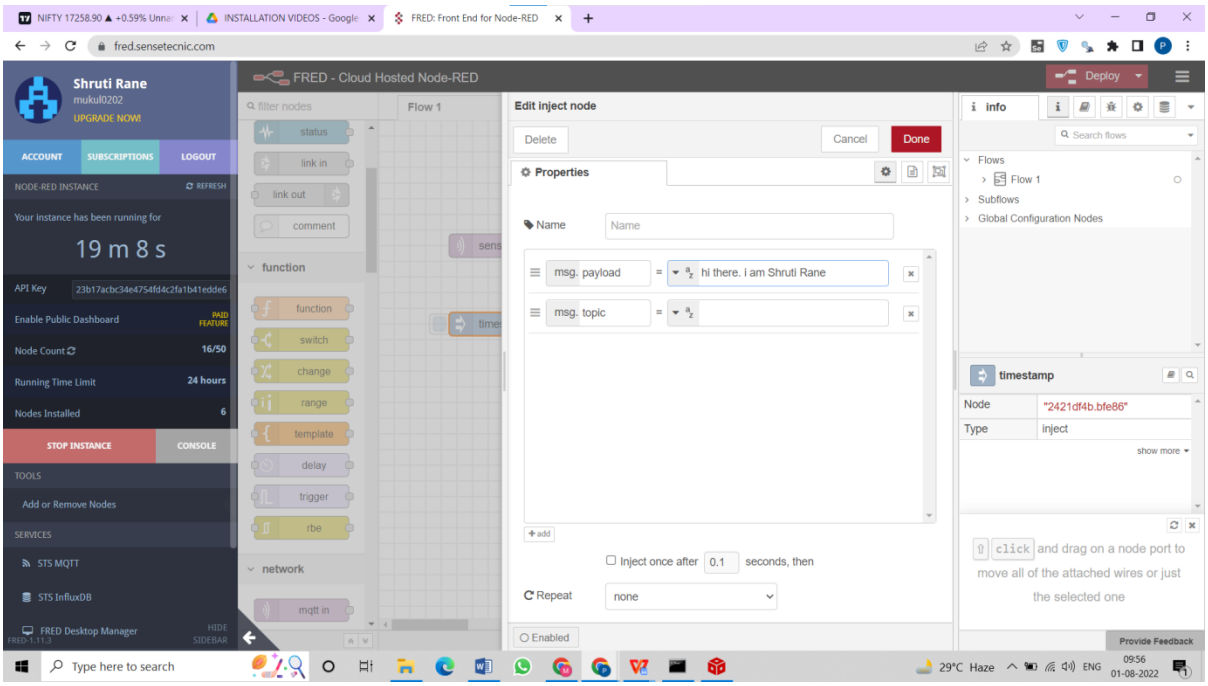

fred.sensetecnic.com
Shruti Rane
mukul0202
UPGRADE NOW!
ACCOUNT
SUBSCRIPTIONS
LOGOUT
NODE-RED INSTANCE
REFRESH
Your instance has been running for
4 m 48 s
API Key
23b17acbc34e4754fd4c2fa1b41edde6
Enable Public Dashboard
PAID FEATURE
Node Count
16/50
Running Time Limit
24 hours
Nodes Installed
6
STOP INSTANCE
CONSOLE
TOOLS
Add or Remove Nodes
SERVICES
STS MQTT
STS InfluxDB
FRED Desktop Manager
HIDE SIDEBAR
FRED-1.11.3
FRED - Cloud Hosted Node-RED
Deploy
filter nodes
Flow 1
common
inject
debug
complete
catch
status
link in
link out
comment
function
function
switch
change
range
template
Edit mqtt in node > Add new mqtt-broker config node
Cancel
Add
Properties
Name
Name
Connection
Security
Messages
Server
broker.hivemq.com
Port
1883
Enable secure (SSL/TLS) connection
Client ID
Leave blank for auto generated
Keep alive time (s)
60
Use clean session
Use legacy MQTT 3.1 support
Enabled
0 nodes use this config
On all flows
info
Search flows
Flows
Flow 1
Subflows
Global Configuration Nodes
undefined:1883
Node
"41e4ba01.fae114"
Type
mqtt-broker
show more
ctrl-space will toggle the view of this sidebar
Provide Feedback


INSTALLATION VIDEOS - Google
FRED: Front End for Node-RED
TradingView – Track All Markets
fred.sensetecnic.com
Shruti Rane
mukul0202
UPGRADE NOW!
ACCOUNT
SUBSCRIPTIONS
LOGOUT
NODE-RED INSTANCE
REFRESH
Your instance has been running for
15 m 50 s
API Key
23b17acbc34e4754fd4c2fa1b41edde6
Enable Public Dashboard
PAID FEATURE
Node Count
16/50
Running Time Limit
24 hours
Nodes Installed
6
STOP INSTANCE
CONSOLE
TOOLS
Add or Remove Nodes
SERVICES
STS MQTT
STS InfluxDB
FRED Desktop Manager
HIDE SIDEBAR
FRED-1.11.3
FRED - Cloud Hosted Node-RED
Deploy
filter nodes
Flow 1
common
inject
debug
complete
catch
status
link in
link out
comment
function
function
switch
change
range
template
mqtt
timestamp
Edit mqtt in node
Delete
Cancel
Done
Properties
Server
broker.hivemq.com:1883
Topic
sensorPot
QoS
2
Output
auto-detect (string or buffer)
Name
Name
Enabled
info
Search flows
Flows
Flow 1
Subflows
Global Configuration Nodes
mqtt
Node
"606360be.e9a82"
Type
mqtt in
show more
Move the selected nodes using the ← ↑ ↓ and → keys. Hold ⇧ to nudge them further
Provide Feedback
Type here to search
29°C Haze
ENG
09:53
01-08-2022

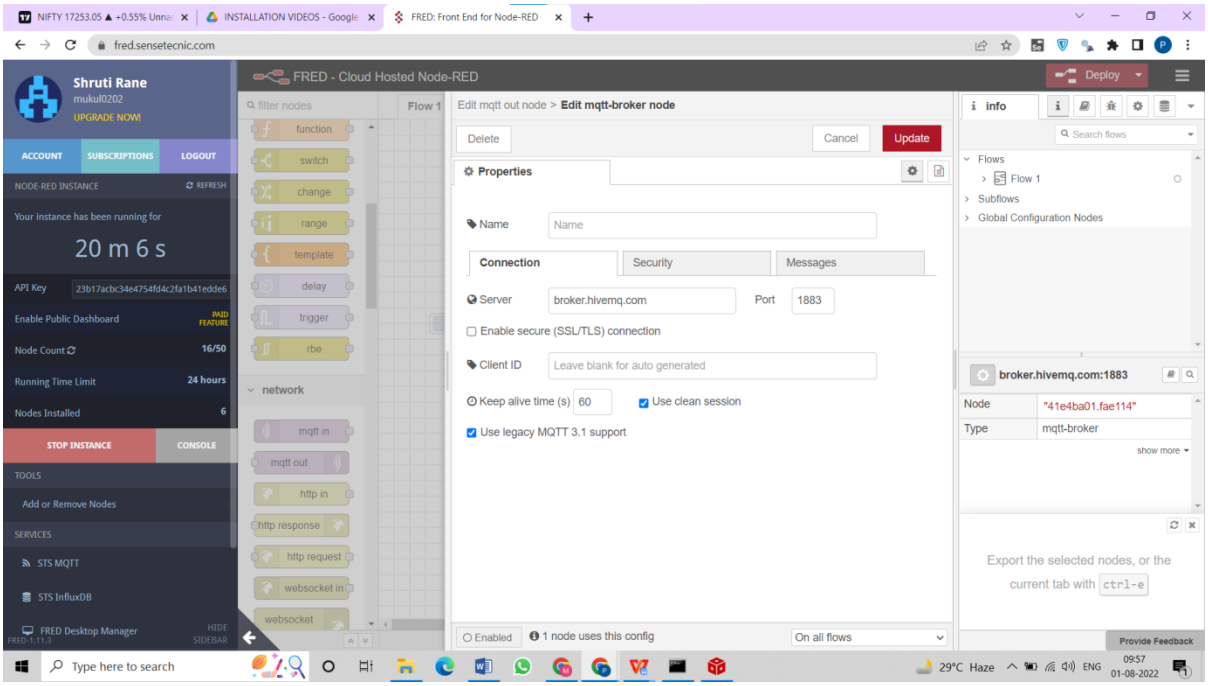
Debug node will display the value stored in the payload property.

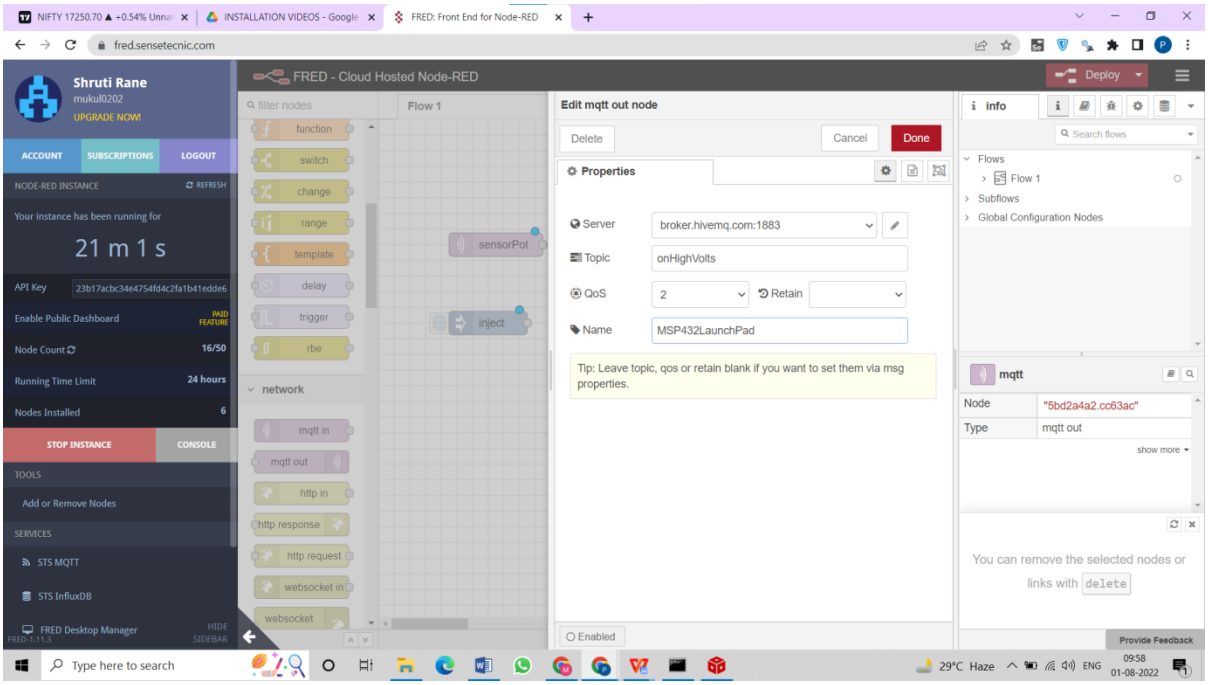


Store the payload property value here in the inject node.

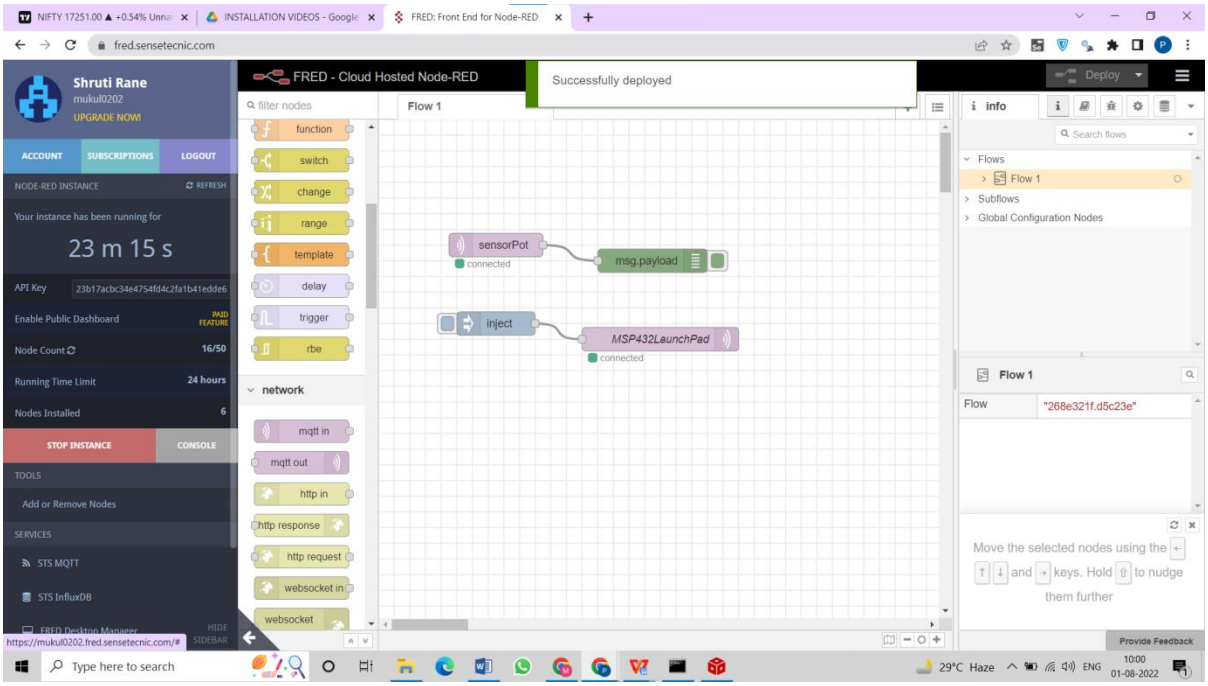


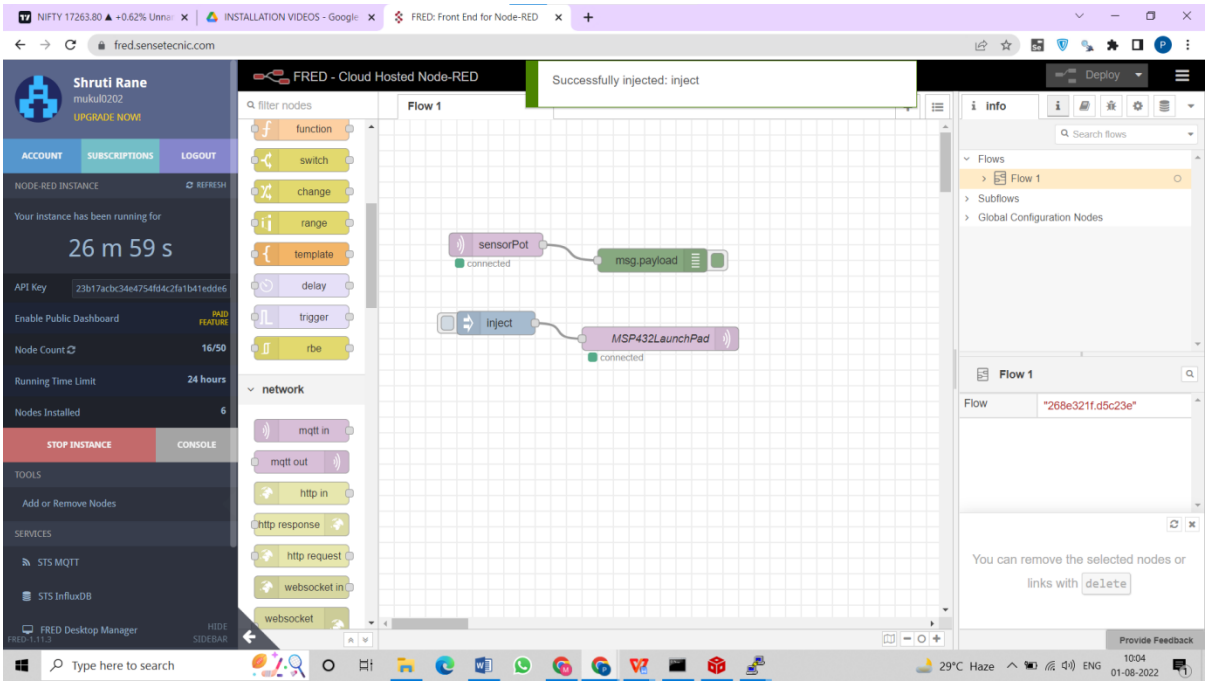
Edit the mqtt out node. add broker node as done for mqtt in as below



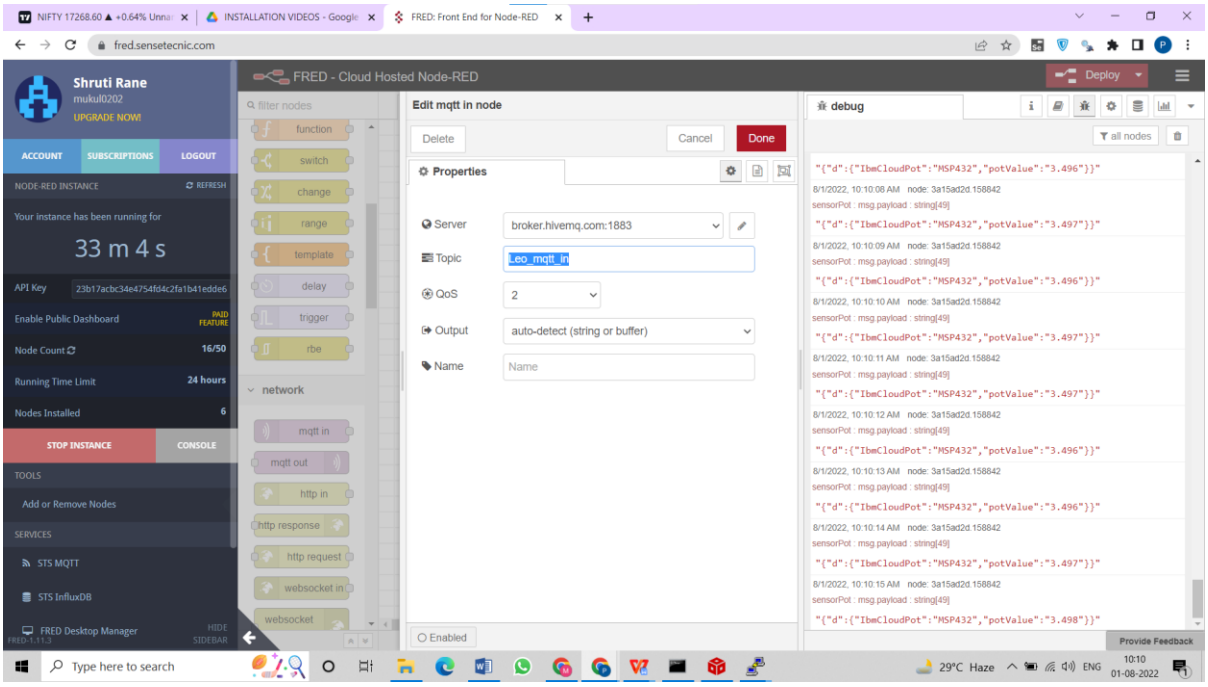


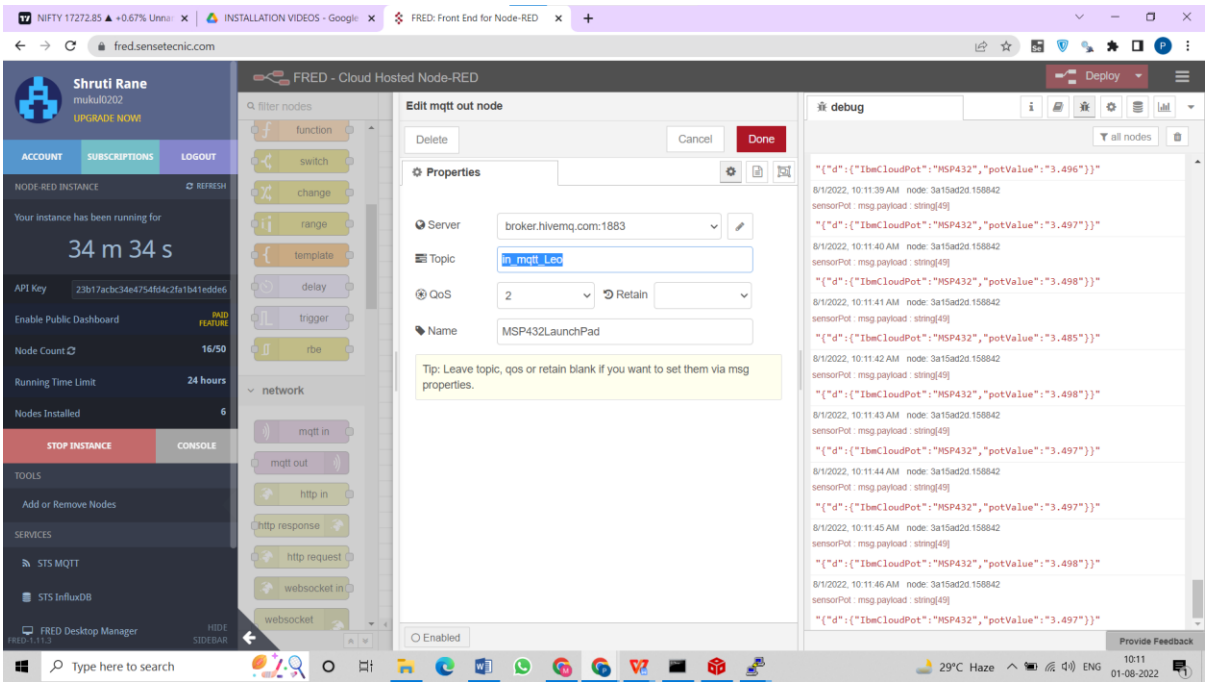
Once done, deploy the changes



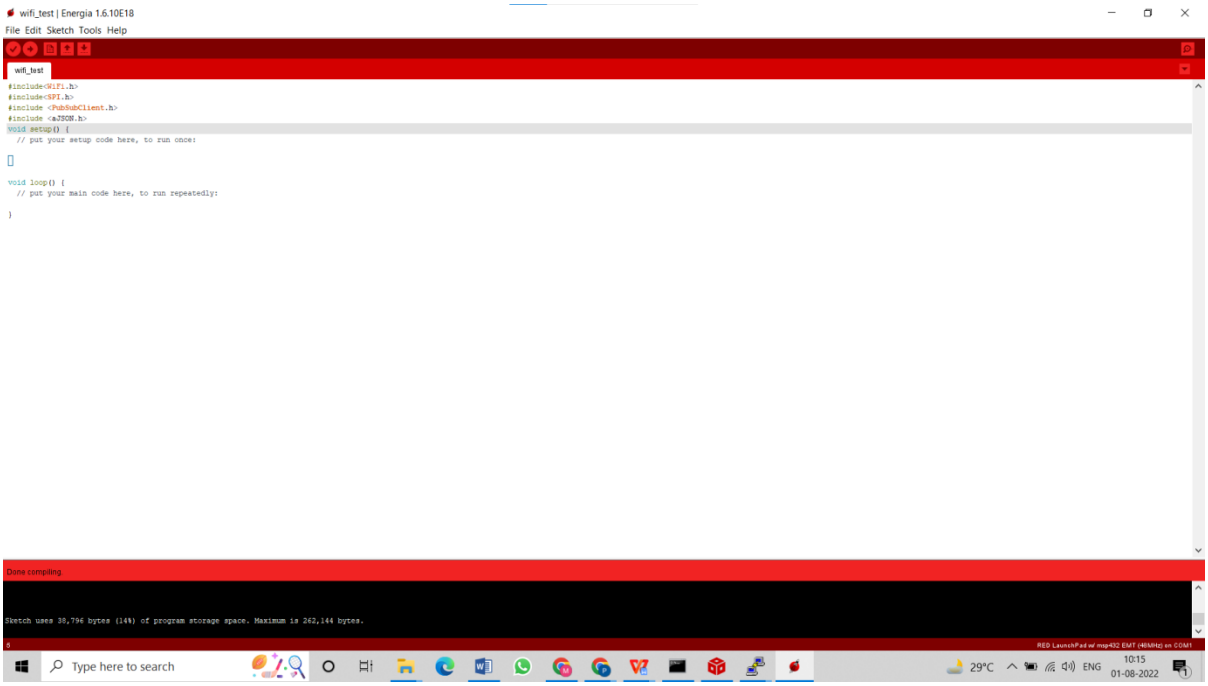


Add the below topics for mqtt in and out nodes.

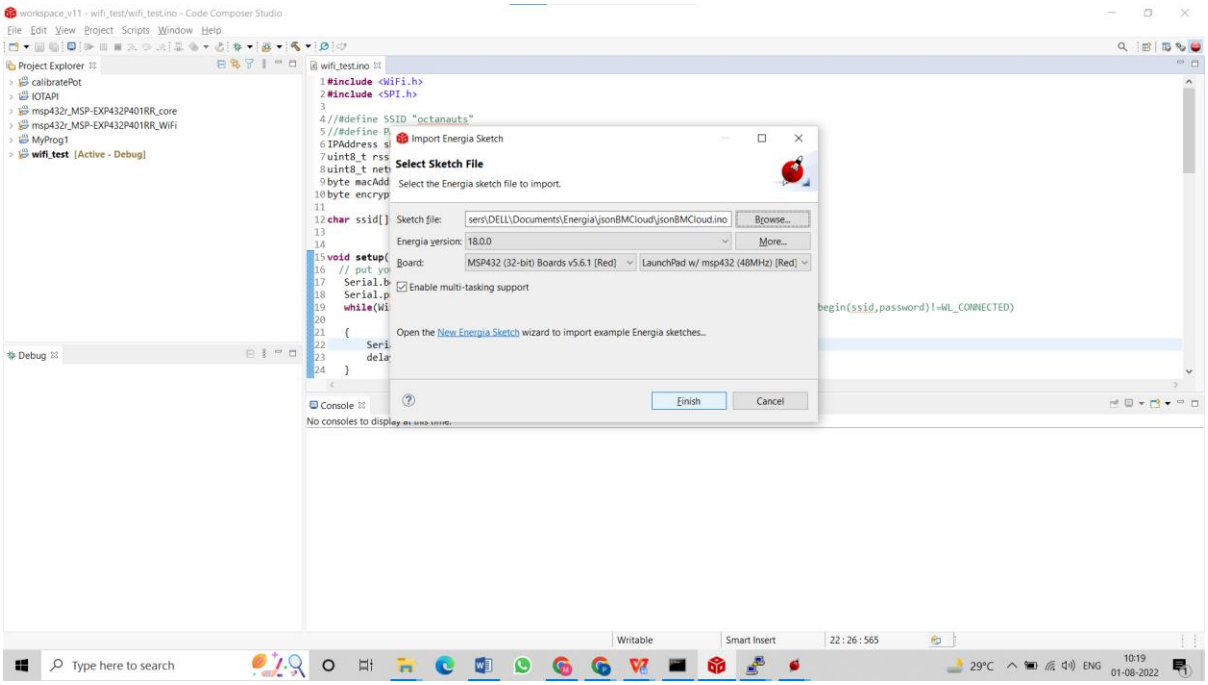




Create energia sketch with below APIs



Import the created sketch in Code Composer Studio



Code:

```
#include <ti/devices/msp432p4xx/driverlib/driverlib.h>
#include<WiFi.h>
#include<SPI.h>
#include<PubSubClient.h>
#include<aJSON.h>
IPAddress shieldIP,subnetMask,gatewayIP;
uint8_t rssi;
uint8_t networkId;
byte macAddr[6]; //mac address is 6 bytes
byte encryptionType;
uint16_t cResult;
char ssid[]="S";//my cell phone
WiFiClient wclient;
//byte server[]
char server[]="broker.hivemq.com";
PubSubClient client(server,1883,callback,wclient);
char* jsonPayload;
void setup() {
  Serial.begin(115200);
      Serial.print("Connecting to WiFi..");
      while(WiFi.begin(ssid)!=WL_CONNECTED)  //for cell phone  with password:-
while(WiFi.begin(ssid,password)!=WL_CONNECTED)
      {
        Serial.print(".");
        delay(1);
      }
      Serial.println("");
      Serial.print("WiFi connected , Fetching WiFi sheild's IP Address:");
      while(WiFi.localIP()==INADDR_NONE ) {
        Serial.print(".");
        delay(1);
```

```
}
shieldIP = WiFi.localIP();
Serial.println(shieldIP);
Serial.print("Access point name :");
Serial.println(ssid);
Serial.print("Signal strength: ");
rssi=WiFi.RSSI();
Serial.println(rssi);
uint8_t networkId = WiFi.scanNetworks();
Serial.print("Number of access points in range:");
Serial.println(networkId);
for(int i=1;i<=networkId;i++){
   Serial.print("Name of Access Points and encryption type:");
   Serial.print(WiFi.SSID(i));
   Serial.print(",");
   encryptionType = WiFi.encryptionType(i);
   Serial.println(encryptionType,DEC);
}
subnetMask = WiFi.subnetMask();
Serial.print("Subnet Mask:");
Serial.println(subnetMask);
gatewayIP = WiFi.gatewayIP();
Serial.print("Gateway IP address:");
Serial.println(gatewayIP);
WiFi.macAddress(macAddr);
Serial.print("Mac Address of shield:");
for(int i =0;i<6;i++){
   Serial.print(macAddr[i],HEX);
   if(i<=4)
      Serial.print(':');
```

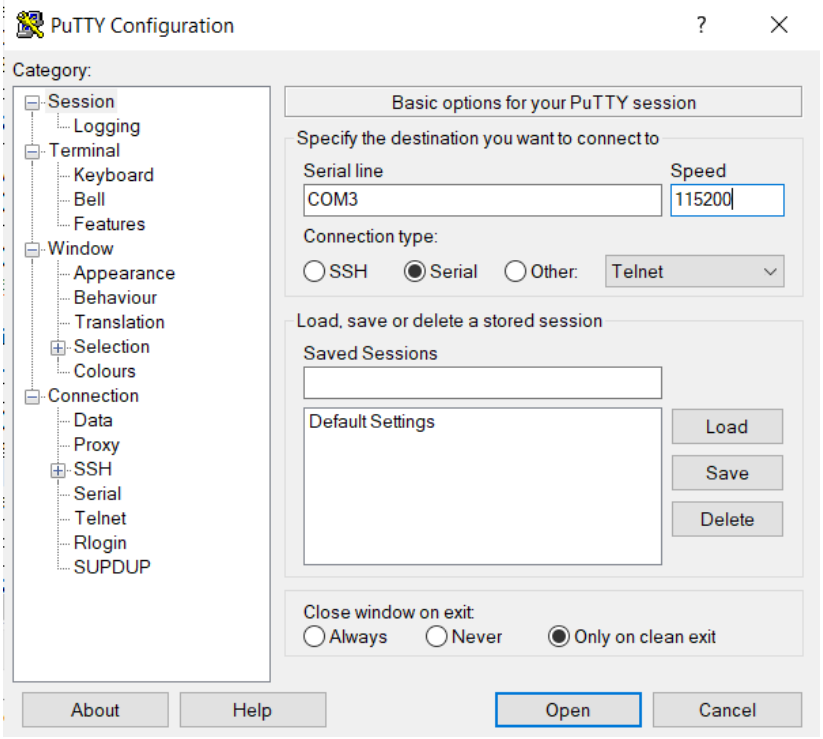
```
        else
            Serial.println();
        }
        GPIO_setAsOutputPin(GPIO_PORT_P2,GPIO_PIN0|GPIO_PIN1|GPIO_PIN2);

GPIO_setAsPeripheralModuleFunctionInputPin(GPIO_PORT_P4,GPIO_PIN7,GPIO_TERTIARY_MODULE_FUNCTION);
        ADC14_initModule(ADC_CLOCKSOURCE_MCLK,ADC_PREDIVIDER_1,ADC_PREDIVIDER_1,ADC_NOROUTE);
        ADC14_configureSingleSampleMode(ADC_MEM6, true); /* out put is stored here in ADC_MEM6 */

ADC14_configureConversionMemory(ADC_MEM6,ADC_VREFPOS_AVCC_VREFNEG_VSS,ADC_INPUT_A6,false);
//2nd parameter is an macros that tells the reference voltage is 3.3V
        ADC14_setSampleHoldTime(ADC_PULSE_WIDTH_32,ADC_PULSE_WIDTH_4); //creating pulse with 64 clock
cycles, again here 2 parameters are min req is of 2; 2nd one stands useless as ADC_INPUT_6 will come in first 16 i.e
the first parameter
        ADC14_setResolution(ADC_12BIT);
        ADC14_enableSampleTimer(ADC_AUTOMATIC_ITERATION);// tells to keep iterating
        ADC14_enableModule();//to start the module
        ADC14_enableConversion();//start ADC conversion
        ADC14_toggleConversionTrigger(); // trigger the conversion continuously
    }
    void loop() {
     // put your main code here, to run repeatedly:
      if (!client.connected())
      {
         Serial.println("Disconnected:Reconnecting..");
         if(!client.connect("sssssh"))
         {
            Serial.println("connection failed");
         }
         else{
            Serial.println("connection success");
            if(client.subscribe("in_mqtt_Leo")){
```

```
            Serial.println("Subscription successful");
        }
    }
}
int result,regressedData1;
float regressedData;
while(!ADC14_isBusy());
result=ADC14_getResult(ADC_MEM6);
P2OUT=result>>8;
Serial.println(result);
ADC14_toggleConversionTrigger();
regressedData=((result*0.813298)+(-72.007))/1000;
Serial.println(regressedData);
regressedData1=(result*0.813298)+(-72.007);
//slope+yintercept/1000

String str=(String)regressedData1;
int str_len=str.length()+2;
char char_array[str_len];
str.toCharArray(char_array,str_len);
char pot_reading[str_len];

if(regressedData<1){
   pot_reading[0]='0';
   pot_reading[1]='.';
   for(int i=2;i<=str_len;i++){
      pot_reading[i]=char_array[i-2];
   }
}
else{
```

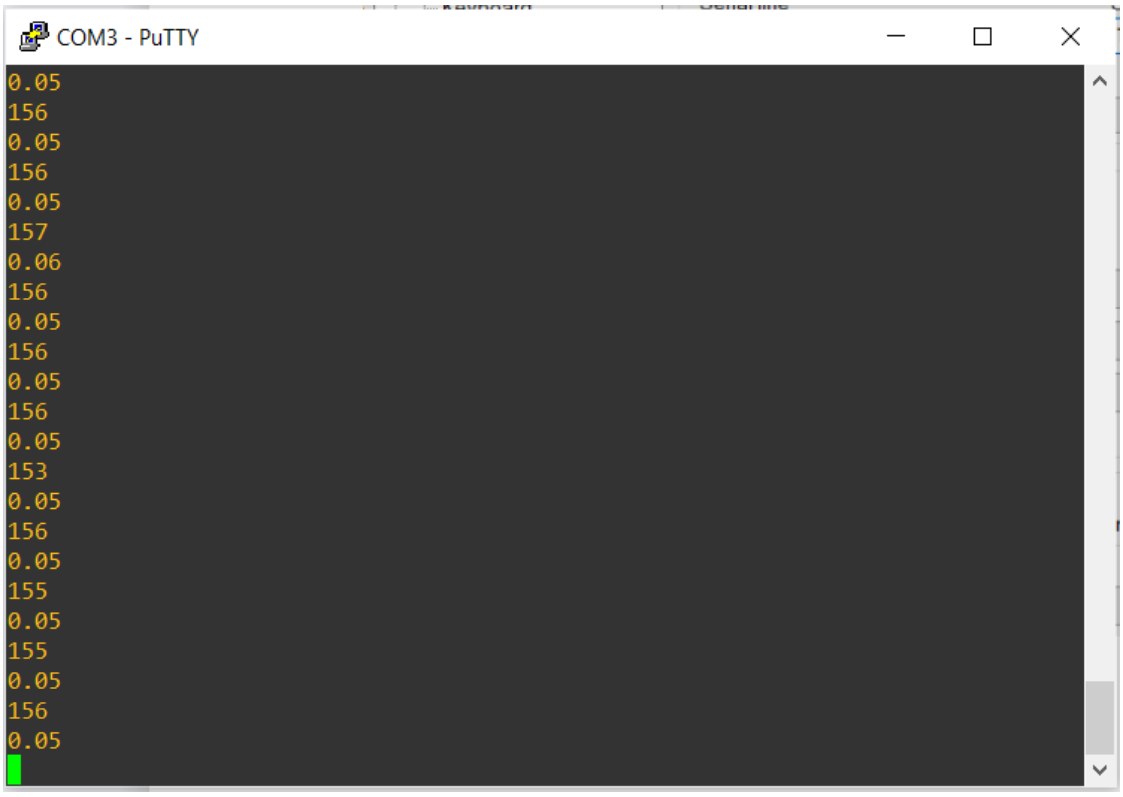
```
        pot_reading[0]=char_array[0];
        pot_reading[1]='.';
        for(int i=2;i<=str_len;i++){
            pot_reading[i]=char_array[i-1];
        }

    }
    aJsonObject *root=aJson.createObject();
    aJsonObject *d=aJson.createObject();
    aJson.addItemToObject(root,"d",d);
    aJson.addStringToObject(d,"IbmCloudPot","MSP432");
    aJson.addStringToObject(d,"potValue",pot_reading);
    jsonPayload=aJson.print(root)+'\0';
    aJson.deleteItem(d);
    aJson.deleteItem(root);
    //publish data to MQTT broker
    client.publish("Leo_mqtt_in",jsonPayload);
    client.poll();
    delay(1000);
}
void callback(char* inTopic,byte* payload,unsigned int length){
    Serial.println("Message arrived on topic:");
    Serial.println(inTopic);
    Serial.println(". Message: ");
    String arrivedMessage;
    for(int i=0;i<length;i++){
        Serial.print((char)(payload[i]));
        arrivedMessage+=(char)payload[i];
    }
    Serial.println();
```

```
}
```

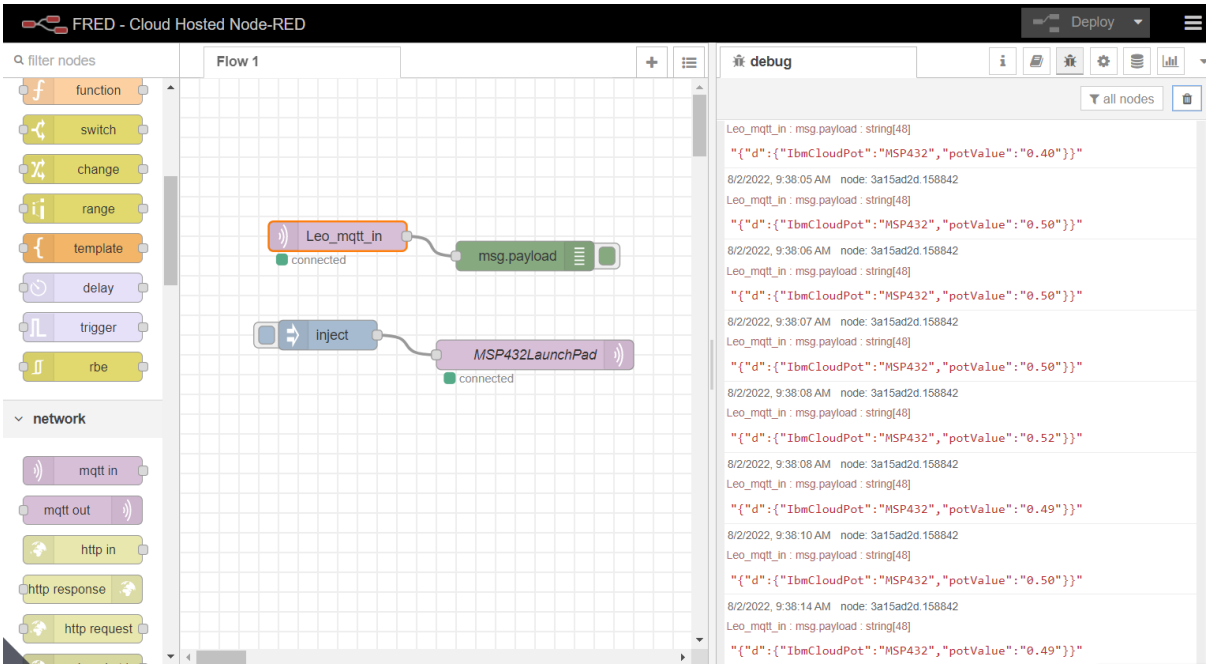
Connect to putty -



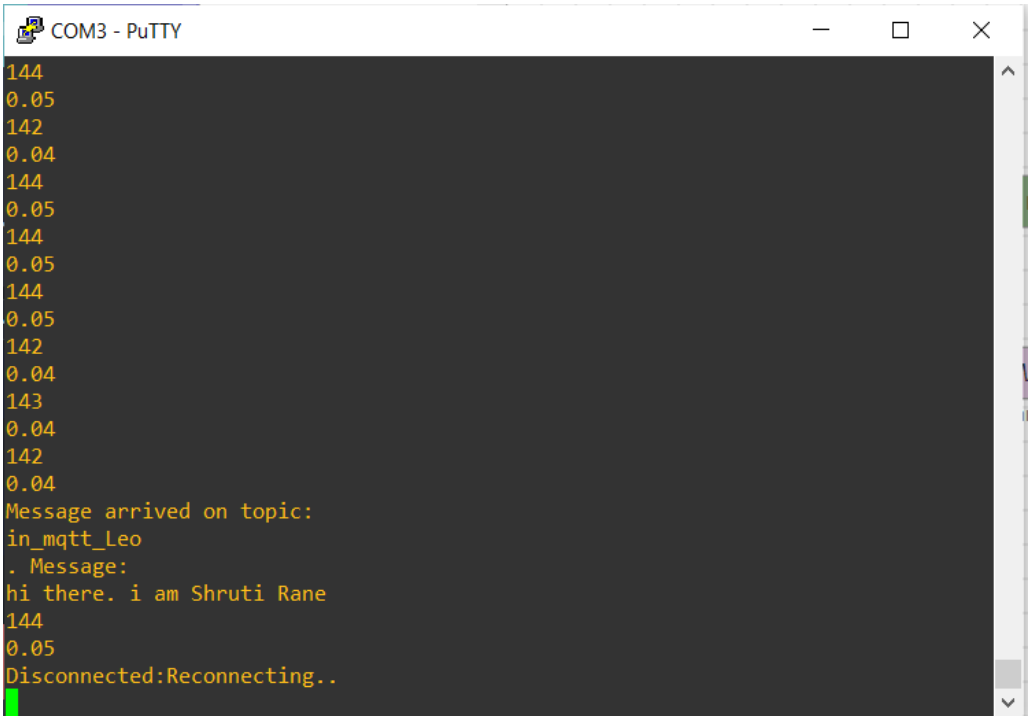
And execute the code. It will generate below output -



Go to Fred cloud. Same Values will be displayed here



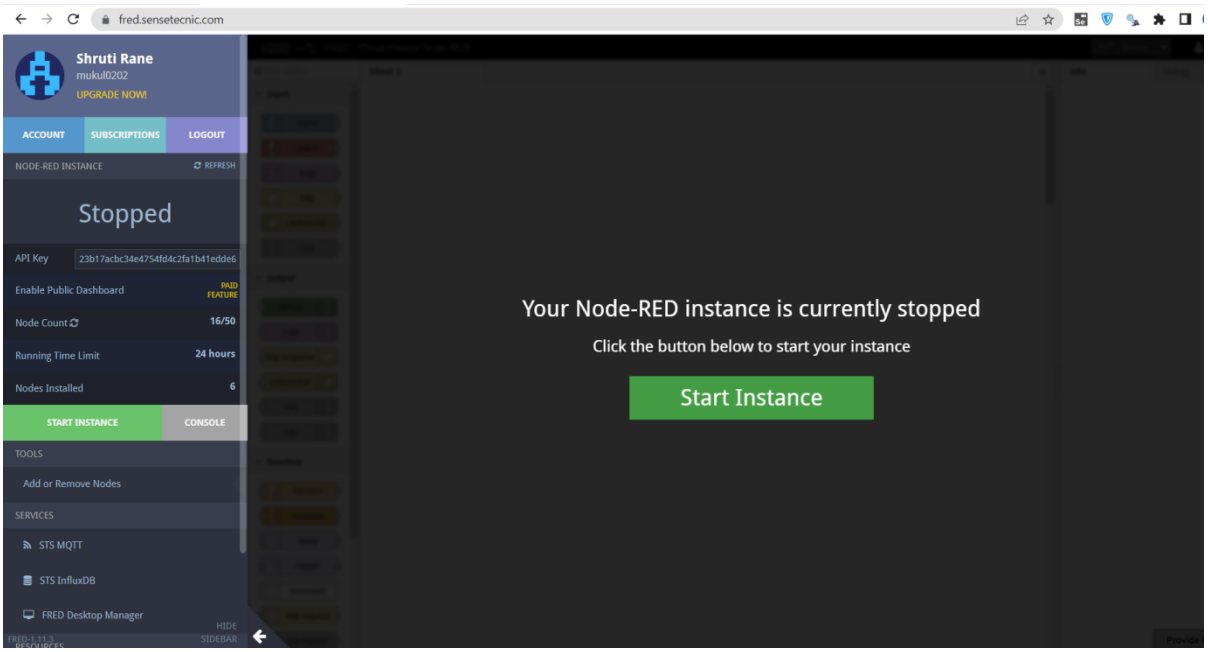
Click on the inject node. The message in the inject node will be displayed in putty as output from the board.

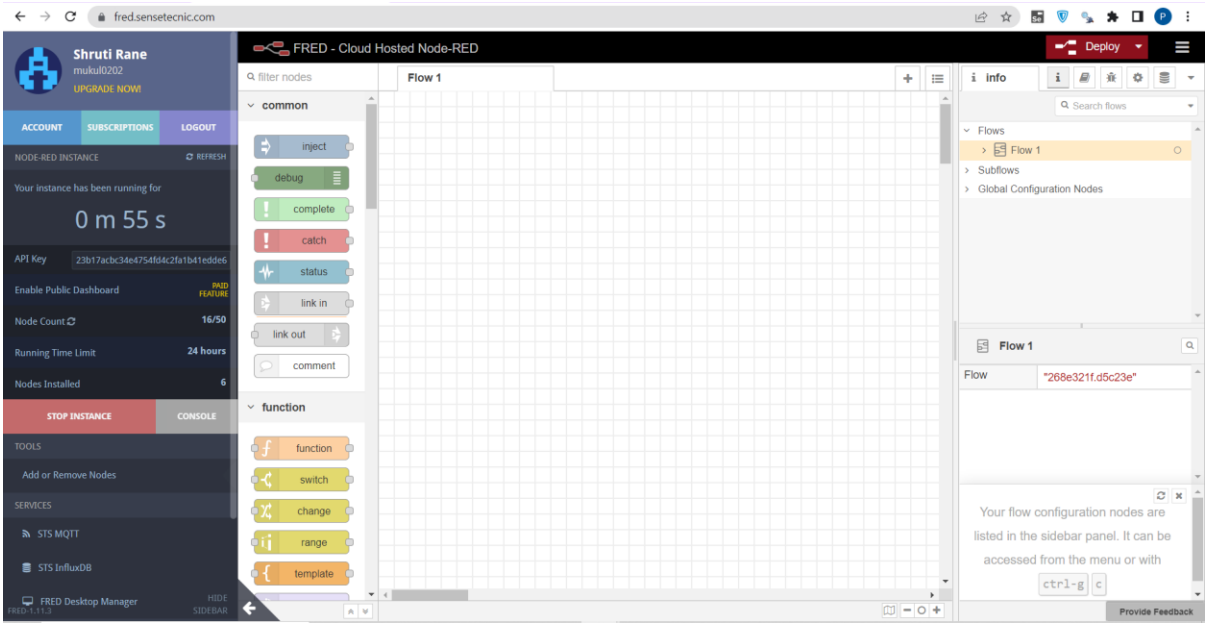


Practical 10

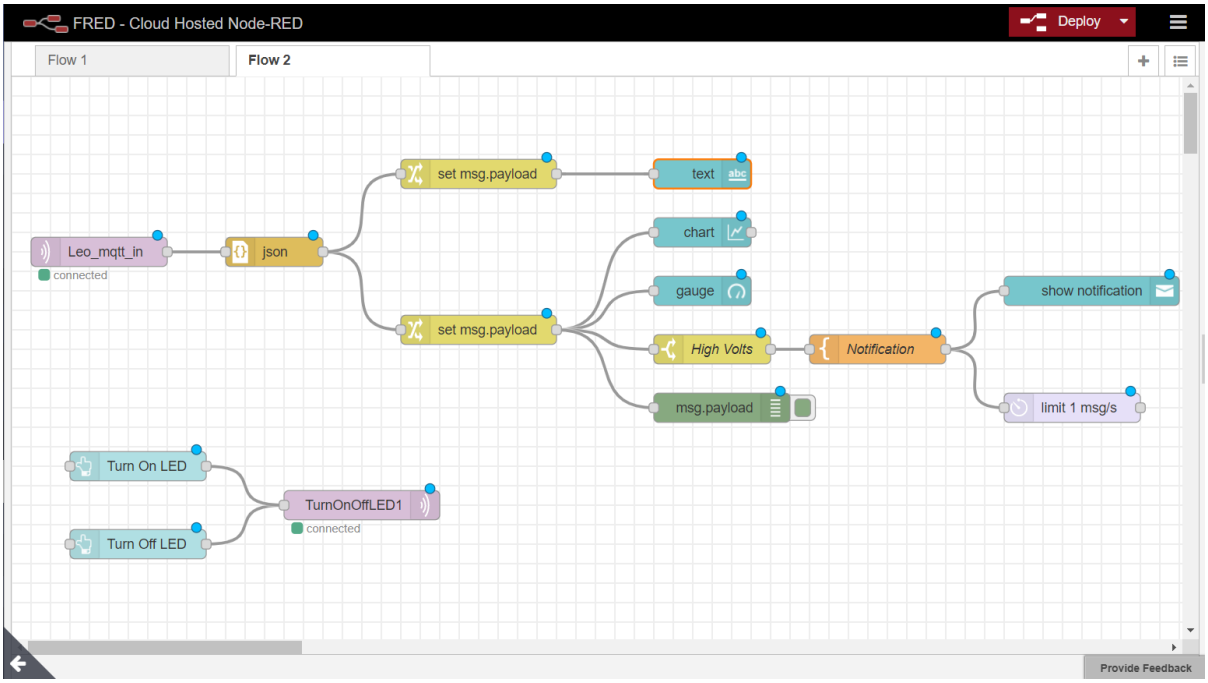
Connecting MSP432 to the cloud

Start the cloud node- red instance

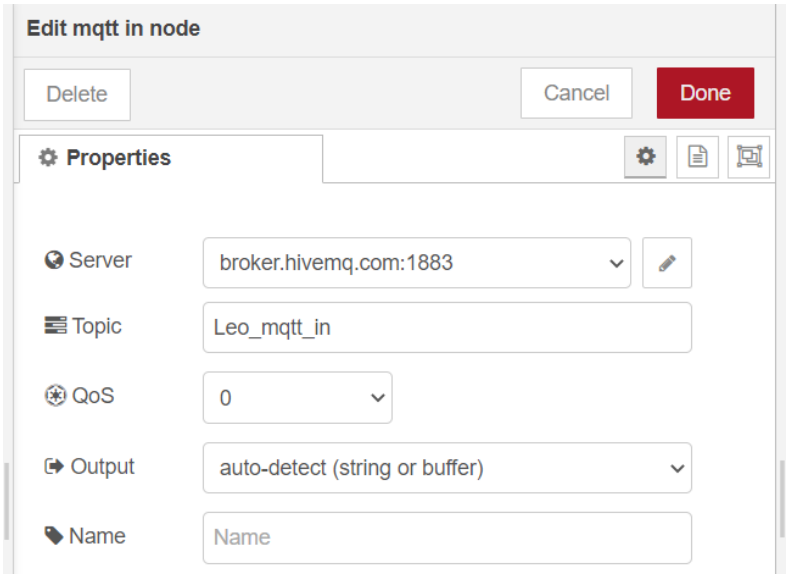




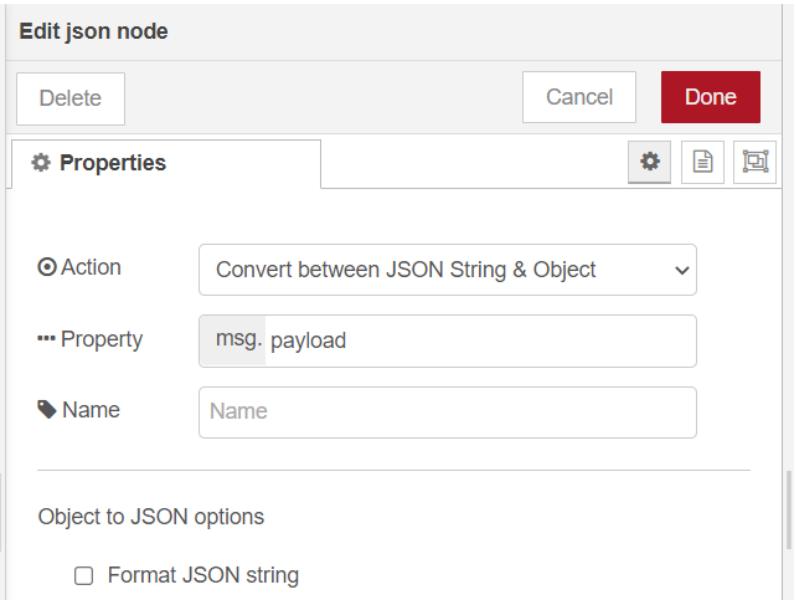
Create the flow as below -



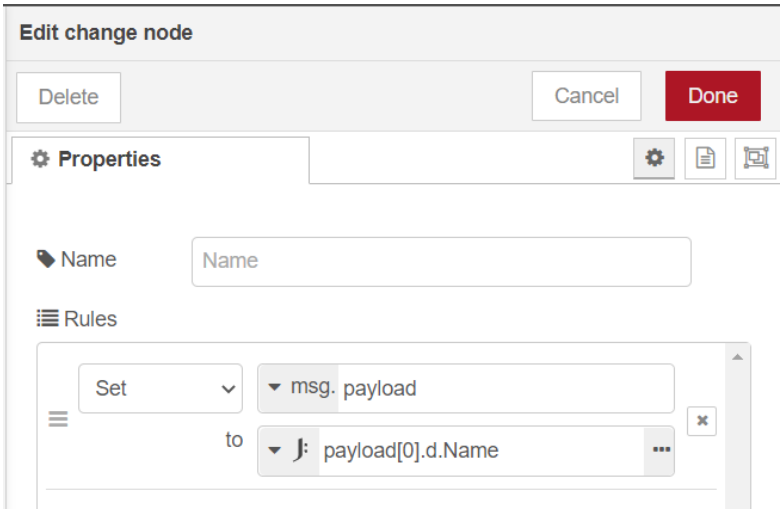
Add the mqttin node properties as below to connect over the topic “Leo_mqtt_in”



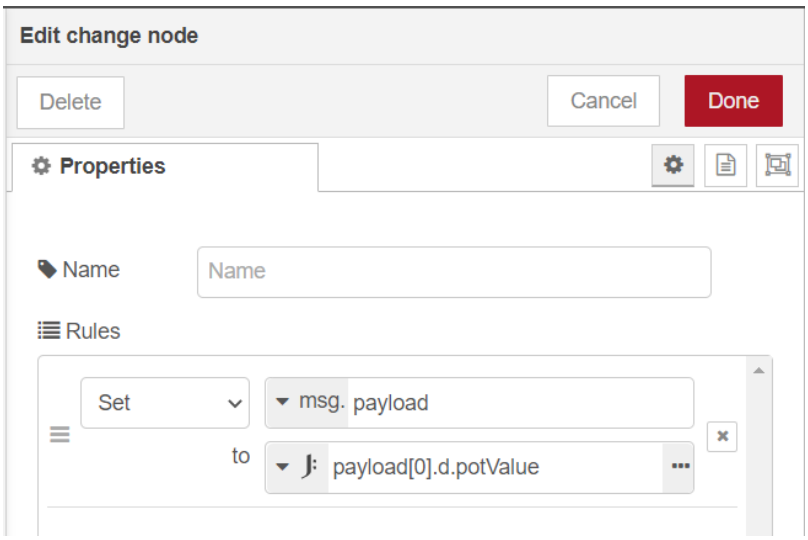
Next in the flow is json node. This will be used for conversion between string and objects, add the msg payload property as below



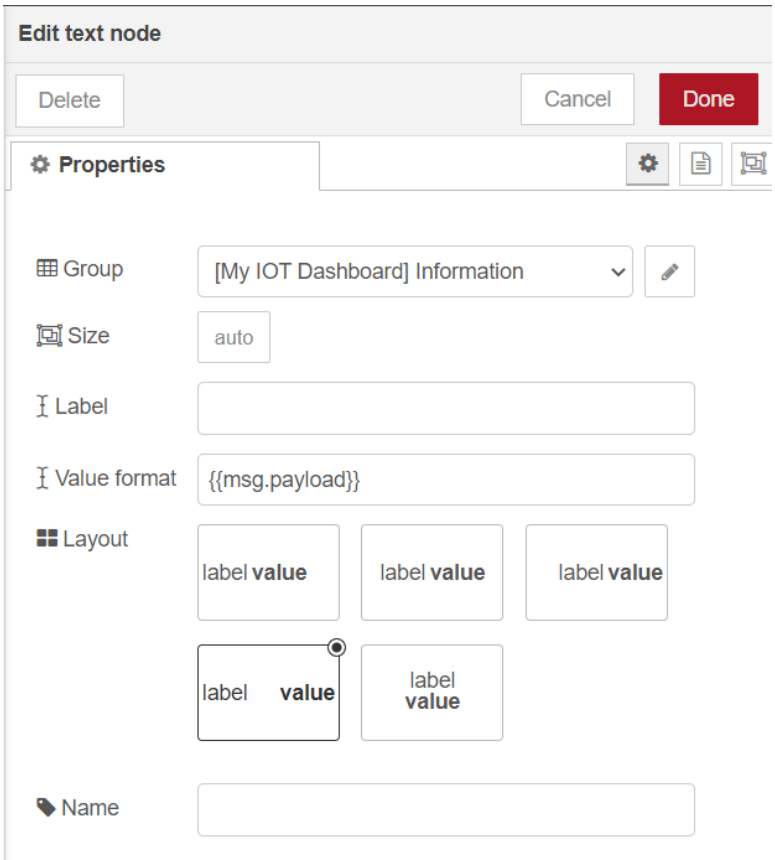
To pass just the name parameter edit the change node properties as follow -



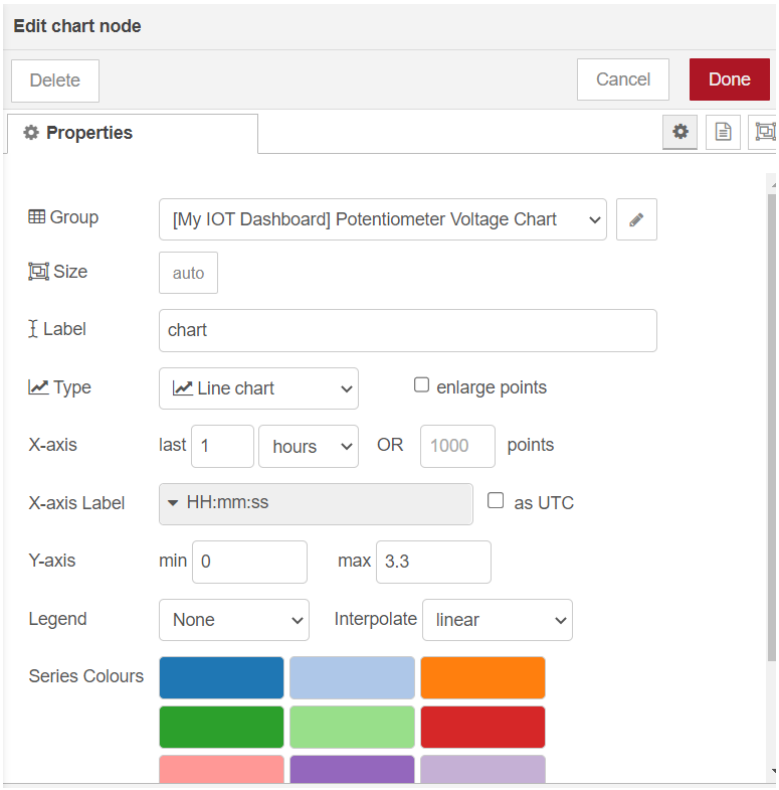
To pass the values edit the another change node as follows -



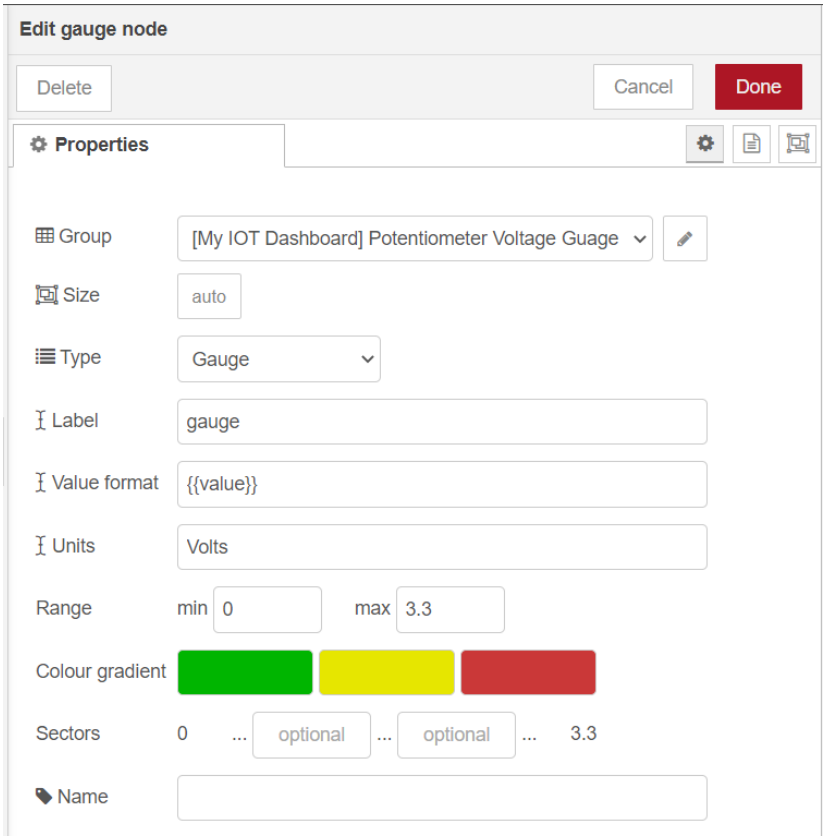
Text node will display the value of name parameter, further we can align the layout using its properties



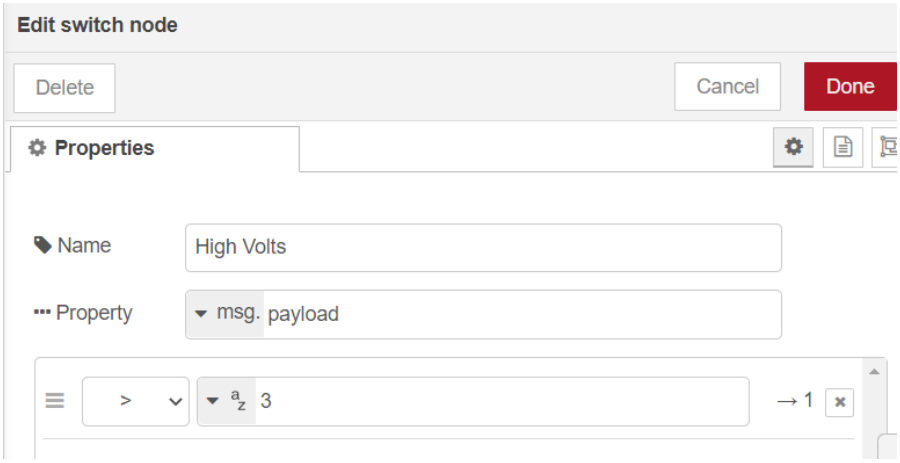
Add below chart properties to the chart node.



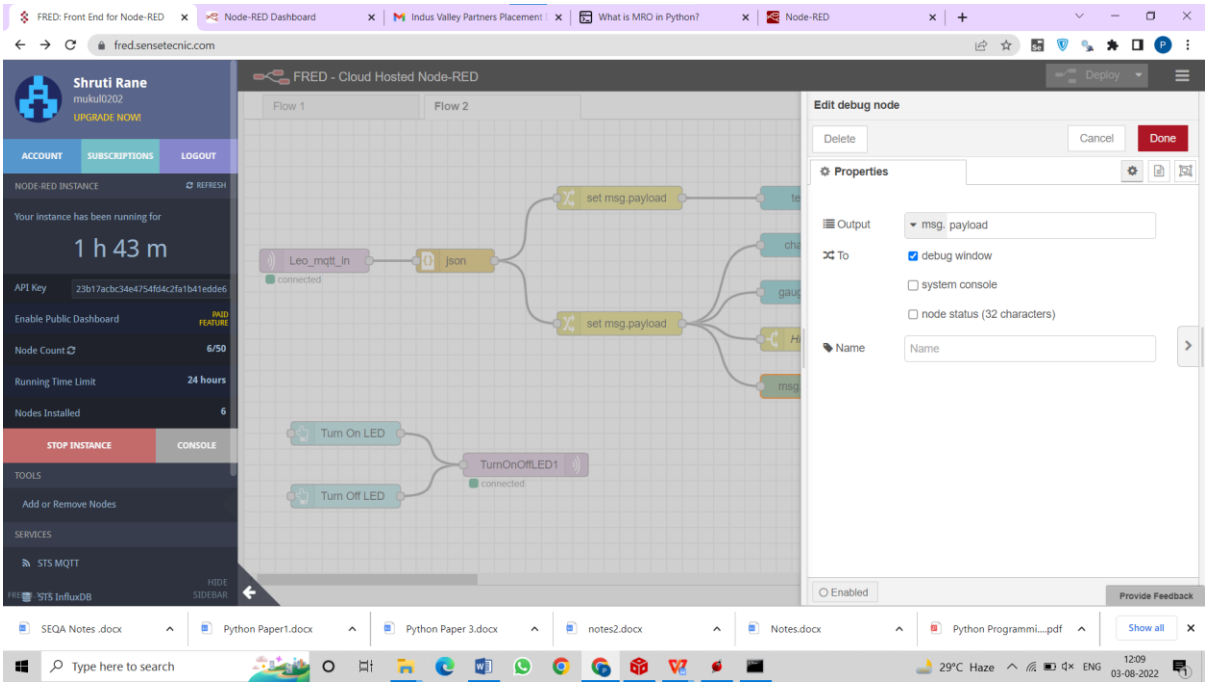
Add the gauge properties as below.



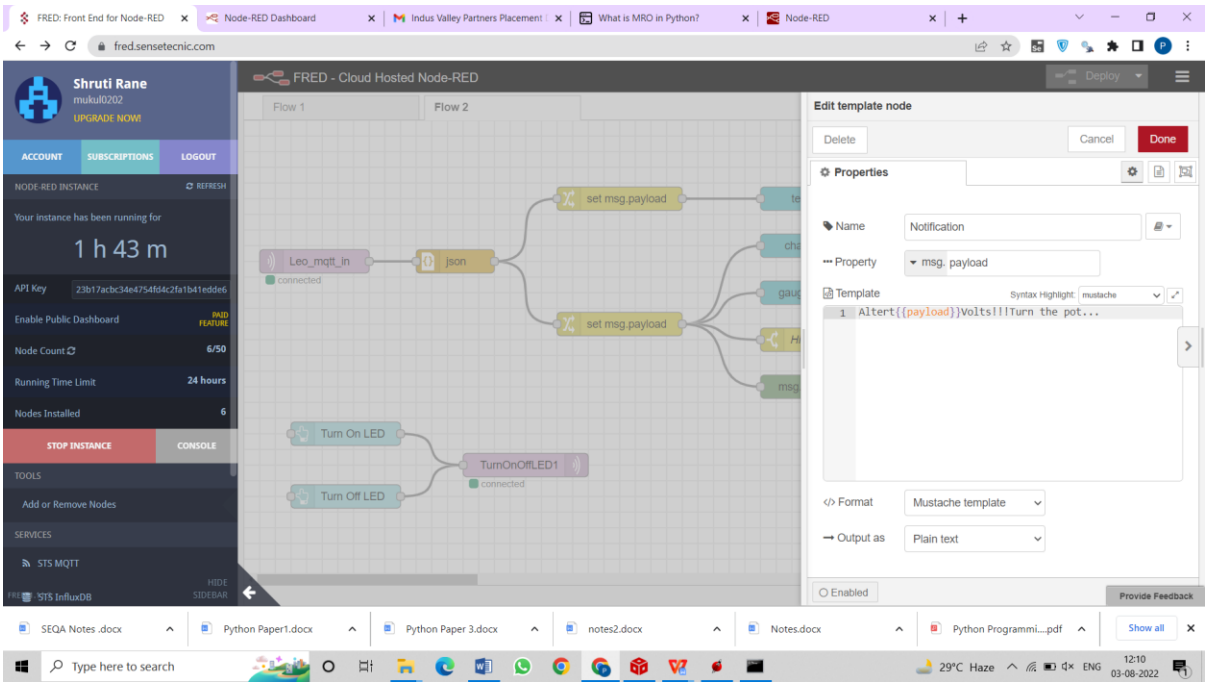
set the threshold of switch node as below



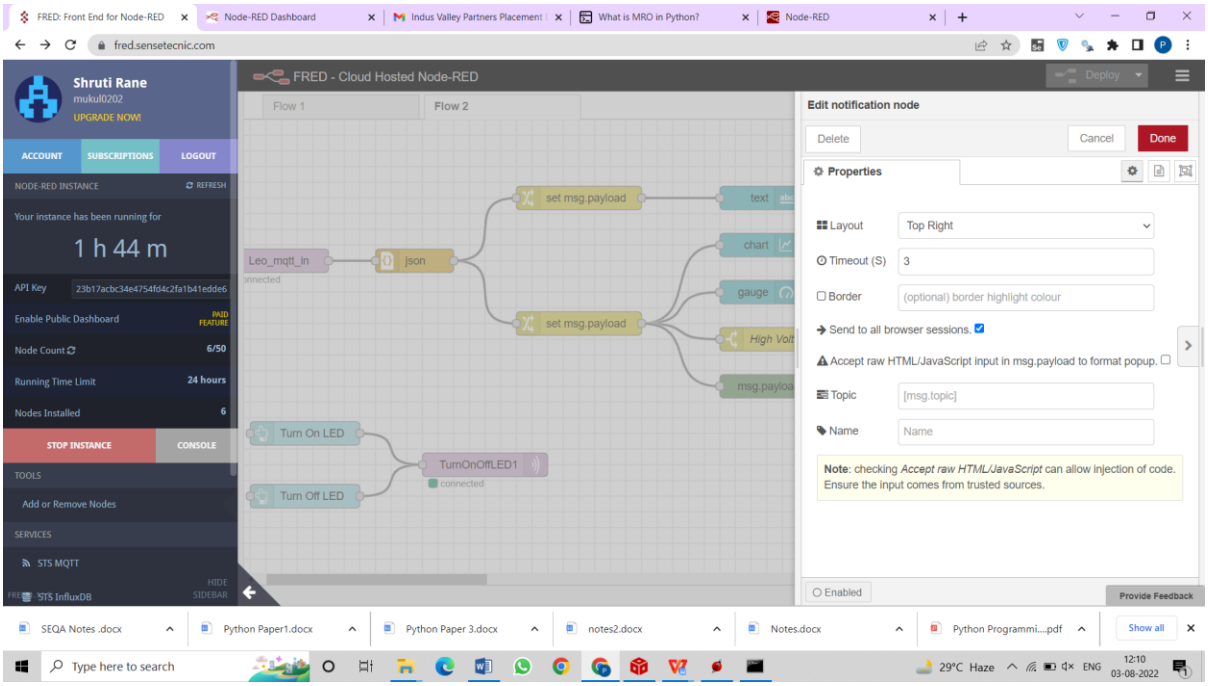
Debug node will display the payload property's value



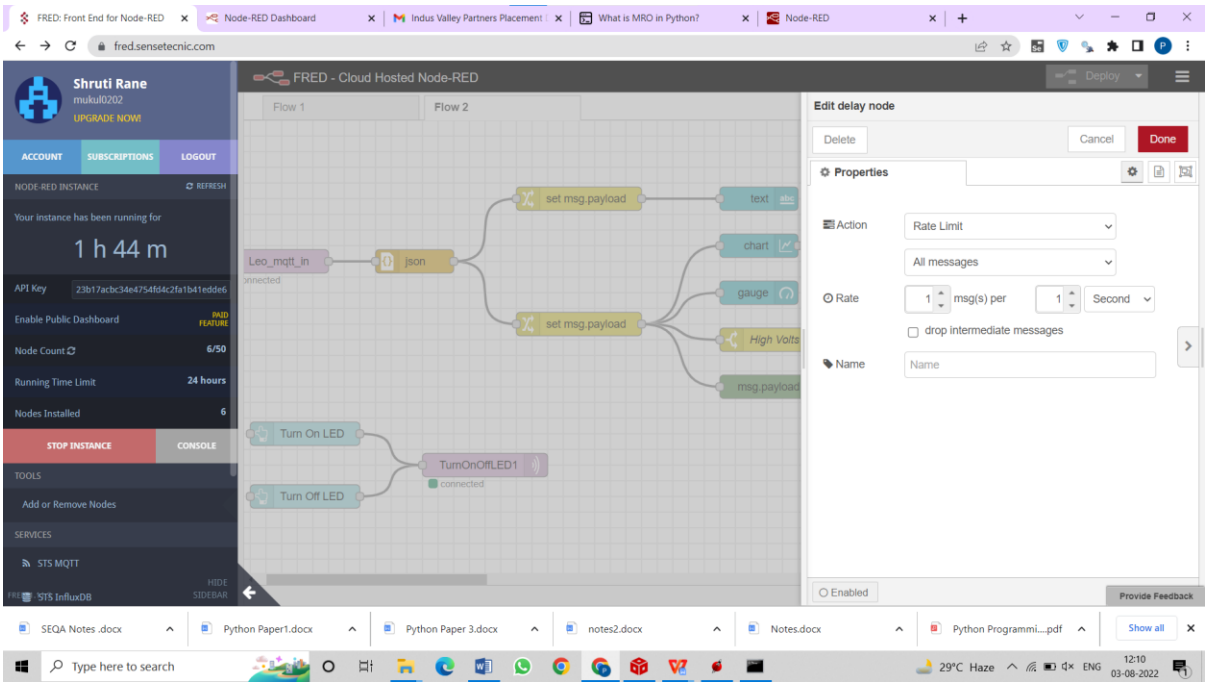
Set the below alert threshold value



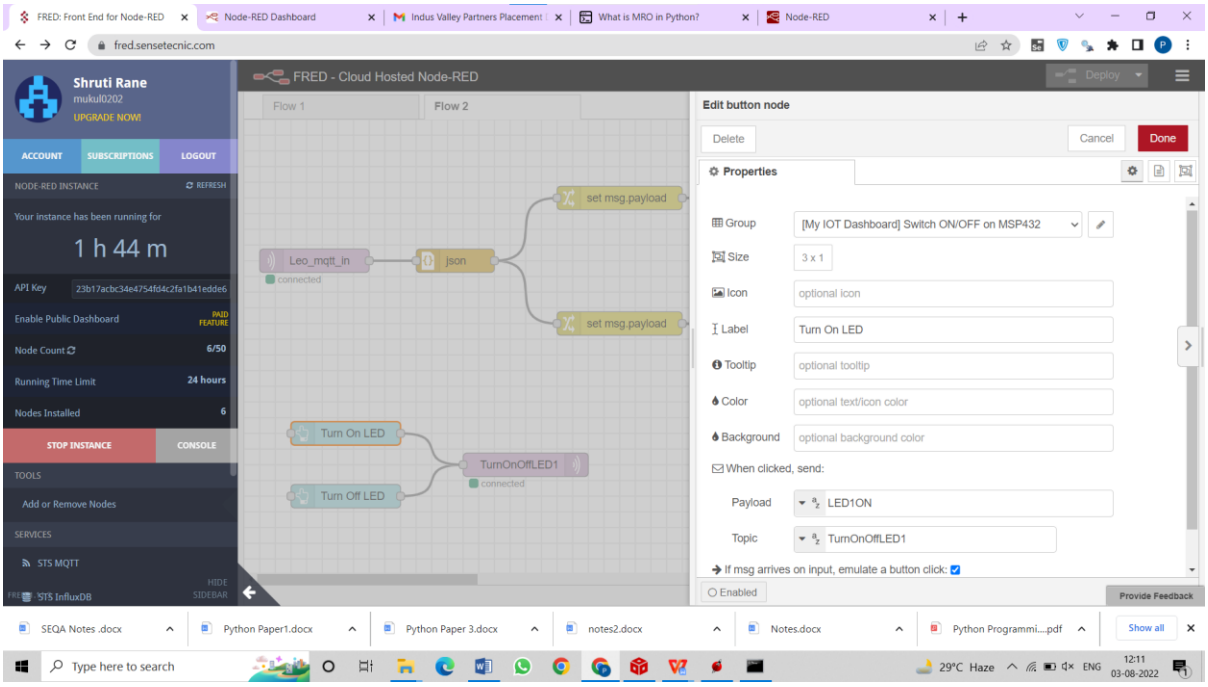
Align where the notification will display on the dashboard

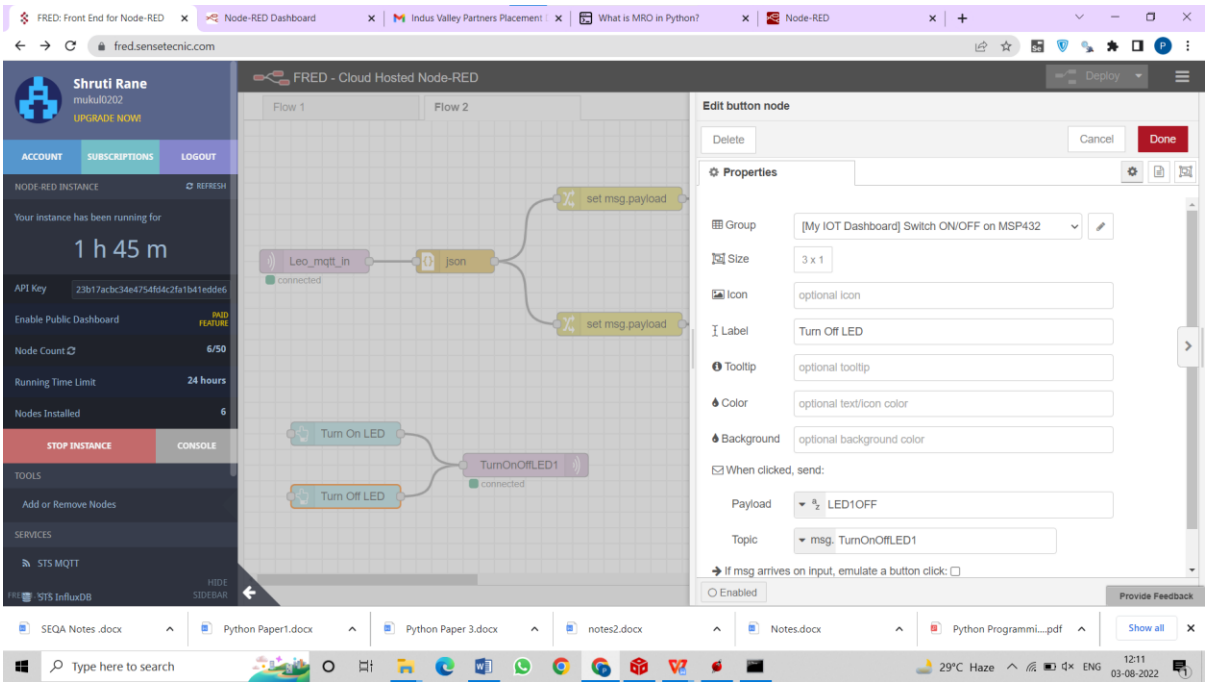


Add delay node properties as below

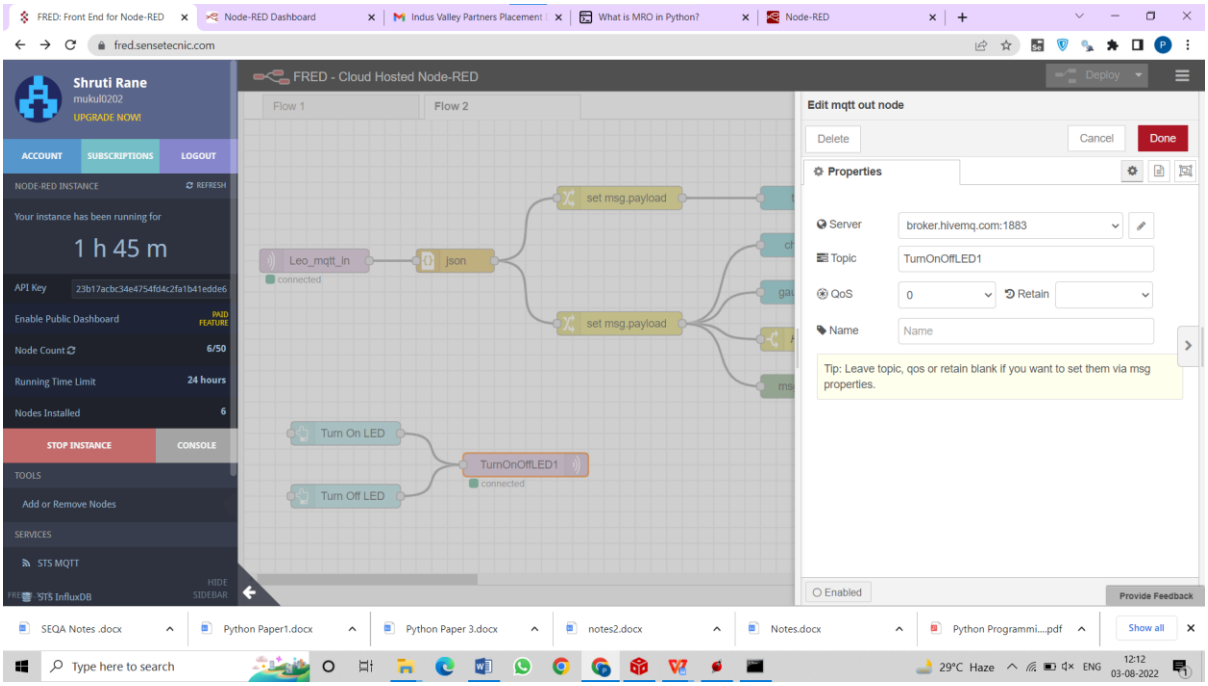


Edit where and how the switch nodes display on the dashboard.

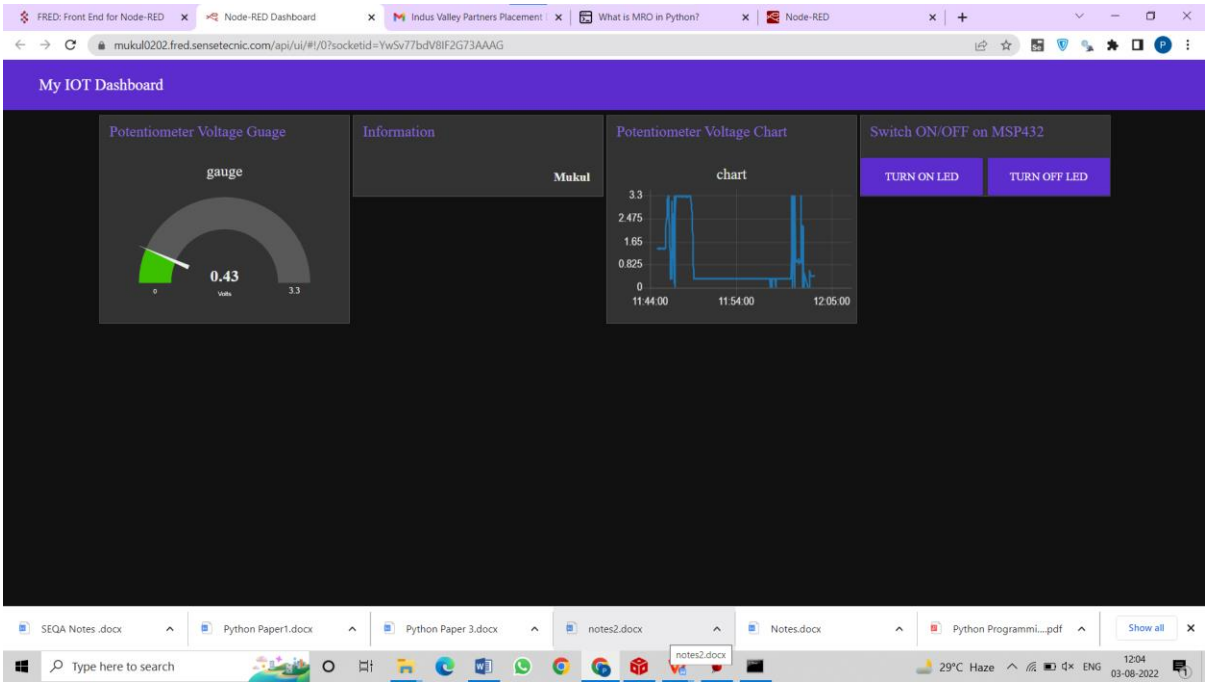




Edit the broker node and add the topic details

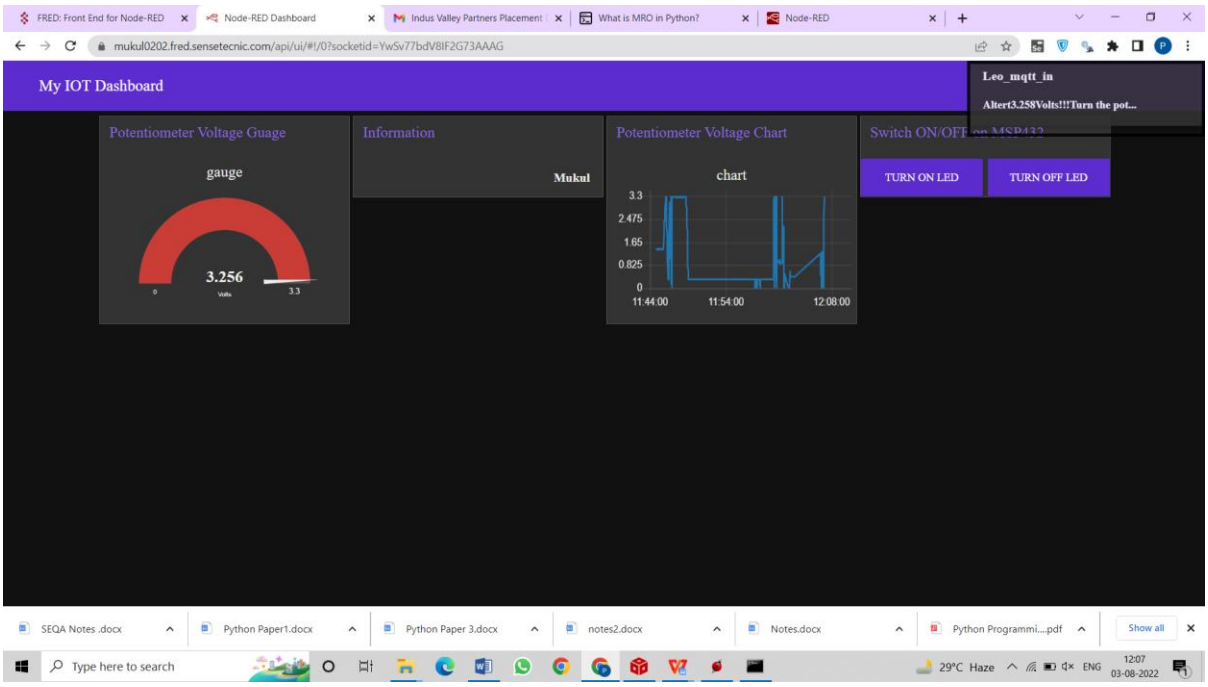


Once these configurations are done, deploy the changes and open the dashboard. It will generate the below output



As we rotate and increase / decrease the voltage the changes would be displayed on the dashboard, and an alert message will be displayed at the top right corner as and when the set threshold is met.

Two button, Turn led on and off when pressed, turns on/off the LED on the board.



This is how we can remotely control our devices or any object with the help of cloud.