**LAPORAN PRAKTIKUM INTERNET OF THINGS**

# **Pertemuan 15**

# **Message Broker**

****

**Dosen Pengampu :**

**Noprianto, S.Kom., M.Eng**

**Oleh:**

**Devi Puspitasari 1941720048**

**Fathin Naufaliya 2041723015**

**Fitri Mutiara Devi 1941720016**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

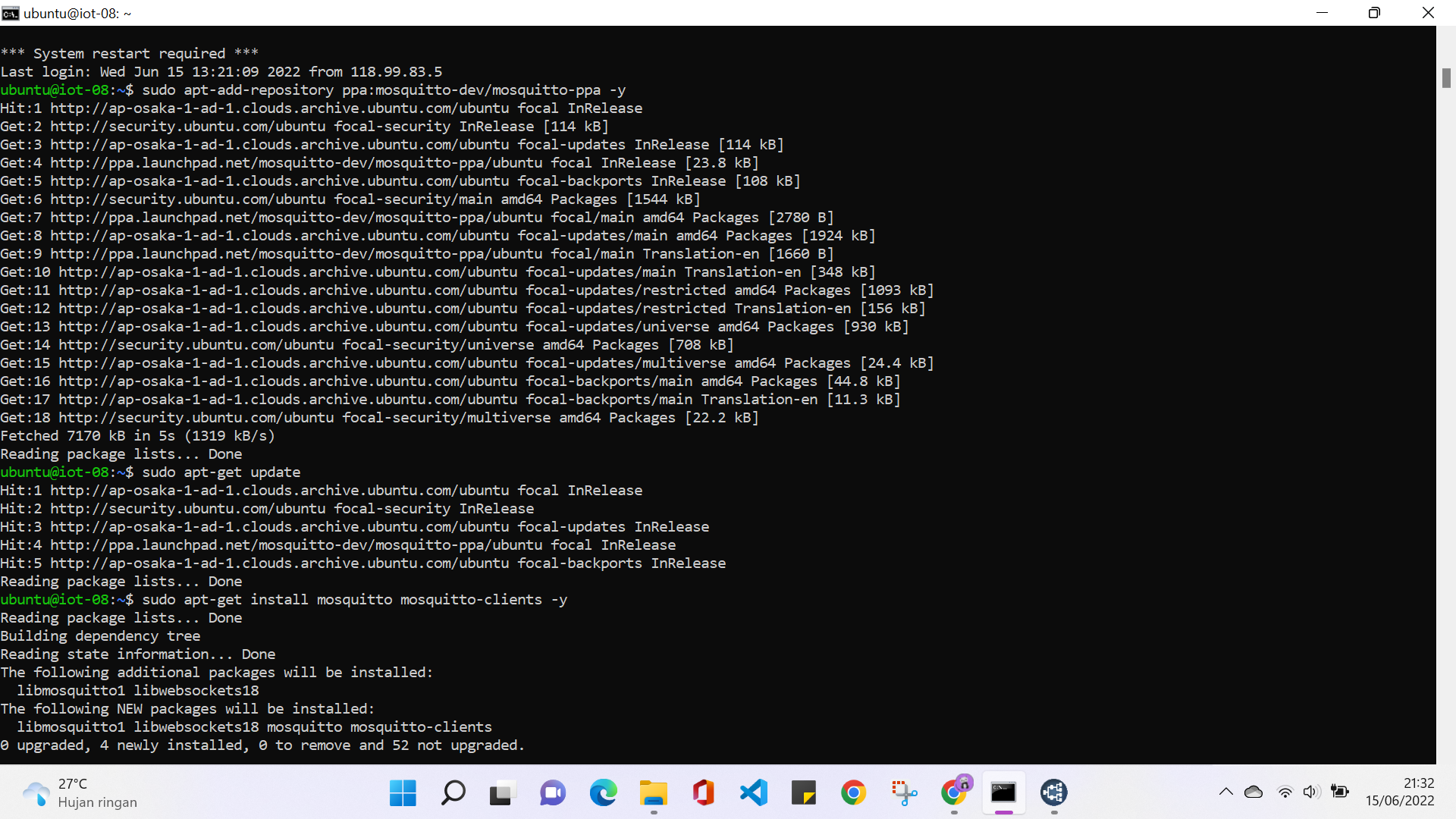
**2022**

#### **Konfigurasi Web Server OCI**

Pada langkah yang kedua akan melakukan konfigurasi pada Oracle Cloud Infrastructure, silakan masuk terlebih dahulu ke instance yang akan Anda konfigurasi kemudian ikuti langkah-langkah di bawah ini

1. Silakan memasang web server terlebih dahulu, web server yang akan digunakan adalah apache. Berikut perintah untuk melakukan instalasi  
   sudo apt update

sudo apt install apache2 -y

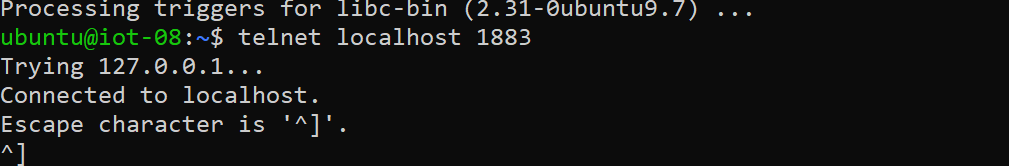
Untuk melakukan pengecekan ketika selesai proses instalasi adalah dengan menggunakan perintah telnet.  
ubuntu@praktikum-iot:~$ telnet localhost 80

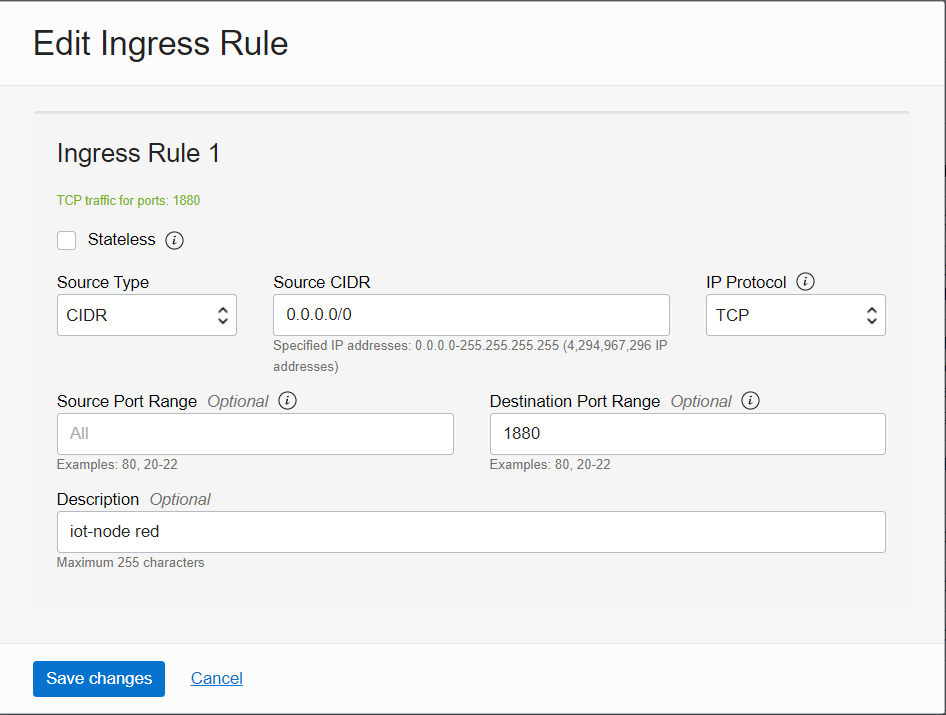
Trying 127.0.0.1…

Connected to localhost.

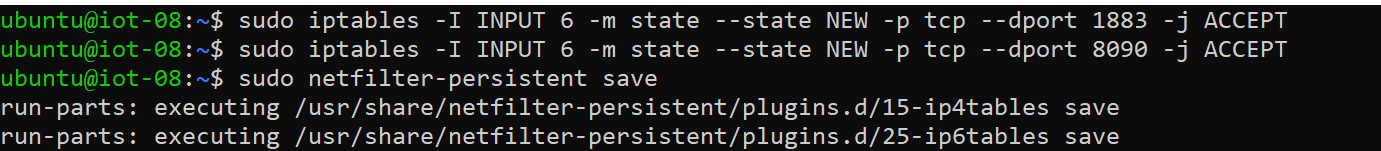
Escape character is '^]'.

^]

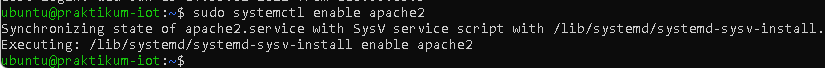
Selain menggunakan telnet, bisa juga menggunakan perintah sudo systemctl status apache2 untuk mengetahui status apakah apache berjalan atau tidak.  
Dari keluaran perintah telnet di atas web server sudah berjalan, akan tetapi ketika ingin mengakses instance kita dari luar masih belum bisa dilakukan. Silakan terlebih dahulu Virtual Cloud Network seperti pada materi yang sebelumnya agar port 80 bisa diakses dari luar.

Jangan lupa jalankan perintah di bawah ini ketika sudah menambahkan atau mendaftarkan port 80.  
sudo iptables -I INPUT 6 -m state --state NEW -p tcp --dport 80 -j ACCEPT

sudo netfilter-persistent save



1. Tampilan ketika sudah berhasil adalah halaman home page, seperti pada gambar berikut Normalnya web server harus dijalankan manual ketika instance tidak berjalan, agar dapat secara otomatis berjalan silahkan ketik perintah di bawah ini  
   sudo systemctl enable apache2



1. Silakan restart instance Anda, untuk membuktikan apakah perintah di atas sesuai yang diharapkan yaitu web server sudah berjalan tanpa menjalankan secara manual kembali.

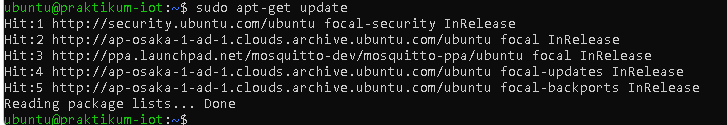
**2. Konfigurasi Message Broker**

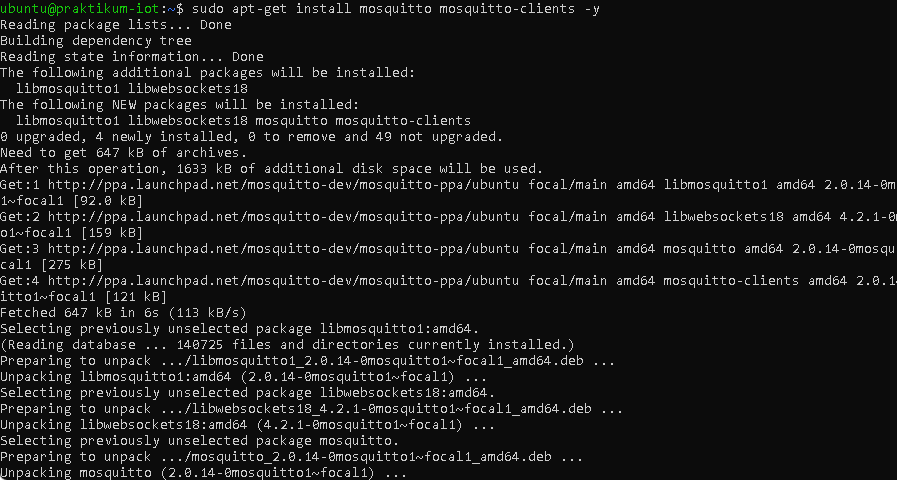
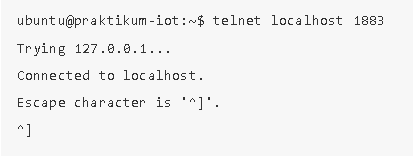
#### **Oracle Cloud Infrastructure (OCI)**

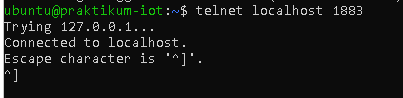
Untuk konfigurasi message broker di OCI sebenarnya sama, hanya langkah-langkah pada menu atau istilah yang berbeda ketika menggunakan Oracle Cloud Infrastructure.

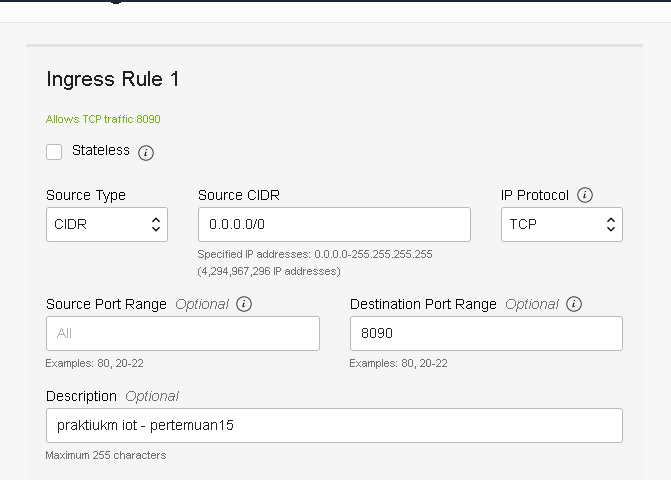
1. Silakan masuk terlebih dahulu ke instance Anda, kemudian jalankan perintah di bawah ini  
   sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa -y

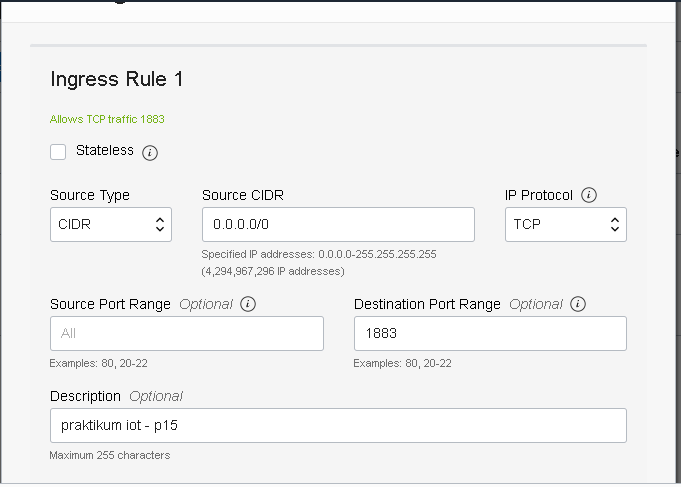
sudo apt-get update

sudo apt-get install mosquitto mosquitto-clients -y

  
Perintah diatas digunakan untuk melakukan instalasi message broker yaitu menggunakan mosquitto dan mosquitto-clients untuk melakukan publish atau subscribe menggunakan terminal. Seperti biasa untuk melakukan pengecekan kita bisa menggunakan perintah telnet apakah sudah berhasil dan berjalan belum message broker.  


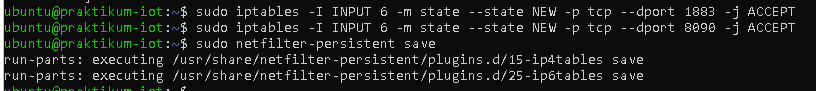
****  
Kita butuh lagi port yang akan digunakan sebagai client untuk terhubung ke message broker menggunakan protokol websocket, misalkan kita akan menggunakan port 8090. Jangan lupa untuk menyimpan perubahan firewall yang telah dikonfigurasi pada VCN menggunakan perintah firewall.



  
sudo iptables -I INPUT 6 -m state --state NEW -p tcp --dport 1883 -j ACCEPT

sudo iptables -I INPUT 6 -m state --state NEW -p tcp --dport 8090 -j ACCEPT

sudo netfilter-persistent save



1. Setelah message broker berhasil dikonfigurasi, selanjutnya kita bisa memanfaatkan atau mencoba untuk publish message data sekaligus menerima pengiriman data. Perintah yang bisa dilakukan adalah sebagai berikut untuk menerima data atau subscribe  
   ubuntu@praktikum-iot:~$ mosquitto\_sub -t "iot/polinema"

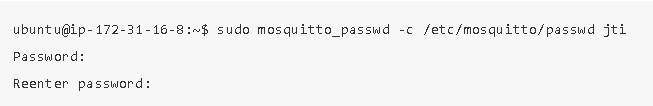
Perintah di atas untuk menerima data dengan topik "iot/polinema", sehingga ketika ada yang mengirimkan data dengan topik tersebut data akan diterima. Kemudian untuk menerima data dapat menggunakan perintah berikut  
ubuntu@praktikum-iot:~$ mosquitto\_pub -t "iot/polinema" -m "iot jaya"

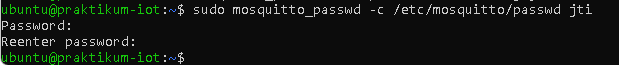
Dengan menggunakan perintah di atas digunakan untuk mengirimkan data "iot jaya" dengan topik "iot/polinema", seharusnya pada perintah subscribe di atas akan menerima data.

#### **Menambahkan Password MQTT**

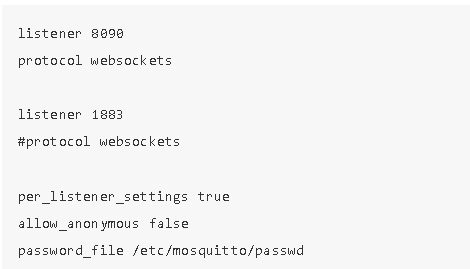
Agar lebih aman terhadap pihak-pihak yang tidak bertanggung jawab, perlu ditambahkan keamanan ketika akan menggunakan message broker yang telah kita install, salah satunya yaitu menambahkan authentifikasi berupa user dan password ketika ingin melakukan subscribe ataupun publish message.

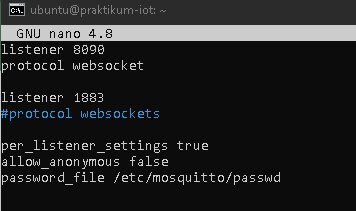
Ikuti langkah-langkah di bawah ini untuk melakukan konfigurasinya

1. Buatlah sebuah user jti dengan perintah di bawah ini. User jti hanya contoh, silakan diganti dengan yang lain jika diperlukan  
   



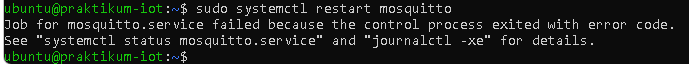
1. Buatlah file konfigurasi yang menginfokan bahwa untuk publish tidak dizinkan tanpa password. Ketik perintah di bawah ini  
   sudo nano /etc/mosquitto/conf.d/jti.conf

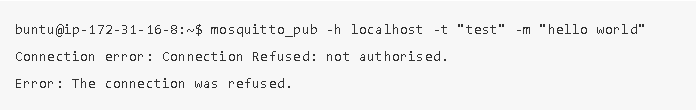
Akan muncul editor nano, selanjutnya tambahkan dua baris perintah berikut  




Tekan CTRL+O untuk menyimpan konfigurasi file dan CTRL+X untuk keluar dari editor nano.

1. Silakan restart mosquitto untuk memberikan perubahan dan coba lakukan publish atau subscribe sebuah message, kira-kira perintahnya adalah sebagai berikut  
   sudo systemctl restart mosquitto



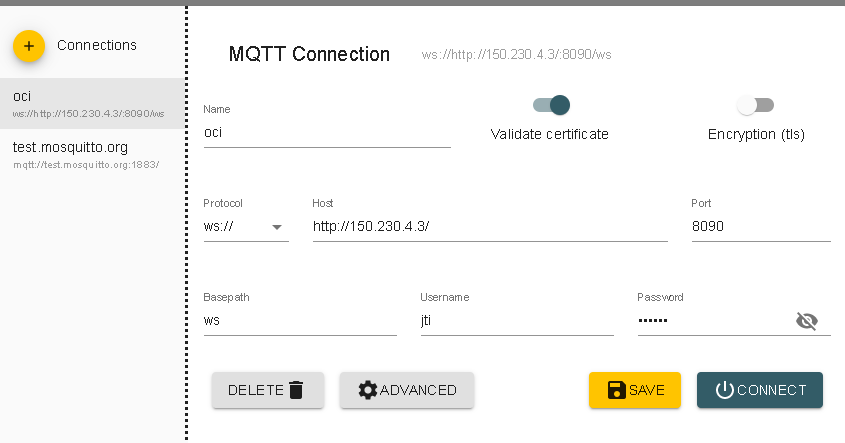
**Verifikasi Hasil Percobaan**Silakan dicoba untuk publish message menggunakan perintah berikut  




Dari keluaran di atas terlihat bahwa terjadi error karena ketika publish message tidak menggunakan user dan password. Kemudian ketika ditambahkan username dan password seperti perintah berikut

ubuntu@ip-172-31-16-8:~$ mosquitto\_pub -h localhost -t "test" -m "hello world" -u jti -P 1234

Untuk memudahkan melakukan pengecekan bisa menggunakan mqtt client menggunakan [http://mqtt-explorer.com](http://mqtt-explorer.com/), silakan download dan installasi. Tampilannya adalah sebagai berikut



#### **Pertanyaan**

1. Apakah fungsi dari baris perintah protocol websockets pada file konfig mosquitto?

**Jawab :**

1. Silahkan ganti menjadi false pada per\_listener\_settings true, restart mosquitto. Apakah yang akan terjadi atau pengaruhnya apa?

**Jawab :**

1. Buatlah user yang lain, kemudian lakukan subscribe dan publish message!

**Jawab :**

### **3. Menghubungkan Smart Device Aplikasi Web**

Setelah menyiapkan web server untuk aplikasi web dan message broker untuk mengirimkan data sensor ataupun aktuator, selanjutnya perlu disiapkan Node-MCU dengan kode program jobsheet sebelumnya dan tentunya akan kita tambahkan halaman dashboard untuk menampilkan data sensor dari Node-MCU. Adapun langkah-langkahnya adalah sebagai berikut

1. Tambahkan kode program pada Node-MCU, dibutuhkan DHT11. Kode programnya adalah sebagai berikut

#include <Arduino.h>

#include <ESP8266WiFi.h>

#include <PubSubClient.h>

#include <SimpleDHT.h>

// hp

// const char \*ssid = "od3ng";

// const char \*password = "0d3n9bro";

// kampus

const char \*ssid = "Smart Parking"; // sesuaikan dengan username wifi

const char \*password = "5m4rT\_P4rk!Ng"; // sesuaikan dengan password wifi

const char \*mqtt\_server = "broker.hivemq.com"; // isikan server broker

WiFiClient espClient;

PubSubClient client(espClient);

SimpleDHT11 dht11(D7);

long now = millis();

long lastMeasure = 0;

String macAddr = "";

void setup\_wifi()

{

delay(10);

Serial.println();

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED)

{

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.print("WiFi connected - ESP IP address: ");

Serial.println(WiFi.localIP());

macAddr = WiFi.macAddress();

Serial.println(macAddr);

}

void reconnect()

{

while (!client.connected())

{

Serial.print("Attempting MQTT connection...");

if (client.connect(macAddr.c\_str()))

{

Serial.println("connected");

}

else

{

Serial.print("failed, rc=");

Serial.print(client.state());

Serial.println(" try again in 5 seconds");

delay(5000);

}

}

}

void setup()

{

Serial.begin(115200);

Serial.println("Mqtt Node-RED");

setup\_wifi();

client.setServer(mqtt\_server, 1883);

}

void loop()

{

if (!client.connected())

{

reconnect();

}

if (!client.loop())

{

client.connect(macAddr.c\_str());

}

now = millis();

if (now - lastMeasure > 5000)

{

lastMeasure = now;

int err = SimpleDHTErrSuccess;

byte temperature = 0;

byte humidity = 0;

if ((err = dht11.read(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess)

{

Serial.print("Pembacaan DHT11 gagal, err=");

Serial.println(err);

delay(1000);

return;

}

static char temperatureTemp[7];

dtostrf(temperature, 4, 2, temperatureTemp);

Serial.println(temperatureTemp);

client.publish("room/suhu", temperatureTemp); // agar lebih unix silakan tambahkan NIM ex: 0001/room/suhu

}

}

Beberapa yang perlu disesuaikan adalah terkait dengan ssid dan password, kemudian port dan server untuk message broker, silahkan menggunakan message broker yang telah dikonfigurasi sebelumnya. Untuk port, yang perlu disesuaikan adalah client.setServer(mqtt\_server, 1883);

1. Edit file atau buat file index.html yang terdapat di /var/www/html/index.html menggunakan file seperti di bawah ini  
   Isi dari file php tersebut adalah sebagai berikut  
   <!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>JTI IoT</title>

<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.2/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/paho-mqtt/1.0.1/mqttws31.min.js" type="text/javascript"></script>

<script type="text/javascript">

var MQTTbroker = '\*\*\*\*';//servernya disesuaikan

var MQTTport = \*\*\*\*; //sesuaikan port websockets messsage broker,

var MQTTsubTopic = '\*\*\*\*'; //topiknya perlu disesuaikan

var chart; // global variuable for chart

var dataTopics = new Array();

//mqtt broker

var client = new Paho.MQTT.Client(MQTTbroker, MQTTport, "jti\_" + parseInt(Math.random() \* 100, 10));

client.onMessageArrived = onMessageArrived;

client.onConnectionLost = onConnectionLost;

//mqtt connecton options including the mqtt broker subscriptions

var options = {

userName: "\*\*\*\*",//silakan disikan username dan password yang didaftarkan mqtt

password: "\*\*\*\*",

timeout: 3,

useSSL: false,

onSuccess: function() {

console.log("mqtt connected");

// Connection succeeded; subscribe to our topics

client.subscribe(MQTTsubTopic, {

qos: 1

});

},

onFailure: function(message) {

console.log("Connection failed, ERROR: " + message.errorMessage);

//window.setTimeout(location.reload(),20000); //wait 20seconds before trying to connect again.

}

};

//can be used to reconnect on connection lost

function onConnectionLost(responseObject) {

console.log("connection lost: " + responseObject.errorMessage);

//window.setTimeout(location.reload(),20000); //wait 20seconds before trying to connect again.

};

//what is done when a message arrives from the broker

function onMessageArrived(message) {

console.log(message.destinationName, '', message.payloadString);

//check if it is a new topic, if not add it to the array

if (dataTopics.indexOf(message.destinationName) < 0) {

dataTopics.push(message.destinationName); //add new topic to array

var y = dataTopics.indexOf(message.destinationName); //get the index no

//create new data series for the chart

var newseries = {

id: y,

name: message.destinationName,

data: []

};

chart.addSeries(newseries); //add the series

};

var y = dataTopics.indexOf(message.destinationName); //get the index no of the topic from the array

var myEpoch = new Date().getTime(); //get current epoch time

var thenum = message.payloadString.replace(/^\D+/g, ''); //remove any text spaces from the message

var plotMqtt = [myEpoch, Number(thenum)]; //create the array

if (isNumber(thenum)) { //check if it is a real number and not text

console.log('is a propper number, will send to chart.')

plot(plotMqtt, y); //send it to the plot function

};

};

//check if a real number

function isNumber(n) {

return !isNaN(parseFloat(n)) && isFinite(n);

};

//function that is called once the document has loaded

function init() {

//i find i have to set this to false if i have trouble with timezones.

Highcharts.setOptions({

global: {

useUTC: false

}

});

// Connect to MQTT broker

client.connect(options);

};

//this adds the plots to the chart

function plot(point, chartno) {

console.log(point);

var series = chart.series[0],

shift = series.data.length > 20; // shift if the series is

// longer than 20

// add the point

chart.series[chartno].addPoint(point, true, shift);

};

//settings for the chart

$(document).ready(function() {

chart = new Highcharts.Chart({

chart: {

renderTo: 'container',

defaultSeriesType: 'spline'

},

title: {

text: 'Dashboard IoT JTI - Suhu Live Websockets'

},

subtitle: {

text: 'broker: ' + MQTTbroker + ' | port: ' + MQTTport + ' | topic : ' + MQTTsubTopic

},

xAxis: {

type: 'datetime',

tickPixelInterval: 150,

maxZoom: 20 \* 1000

},

yAxis: {

minPadding: 0.2,

maxPadding: 0.2,

title: {

text: 'Value',

margin: 80

}

},

series: []

});

});

</script>

<script src="http://code.highcharts.com/stock/highstock.js"></script>

<script src="http://code.highcharts.com/stock/modules/exporting.js"></script>

</head>

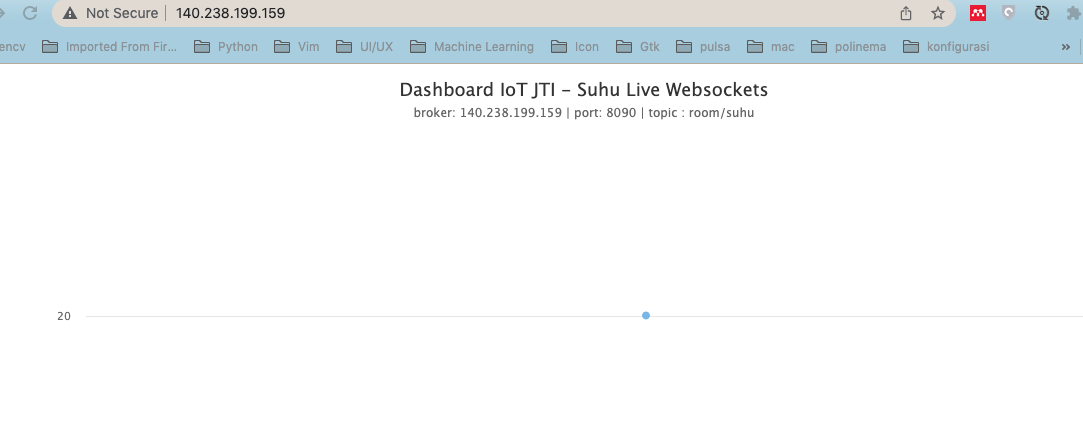
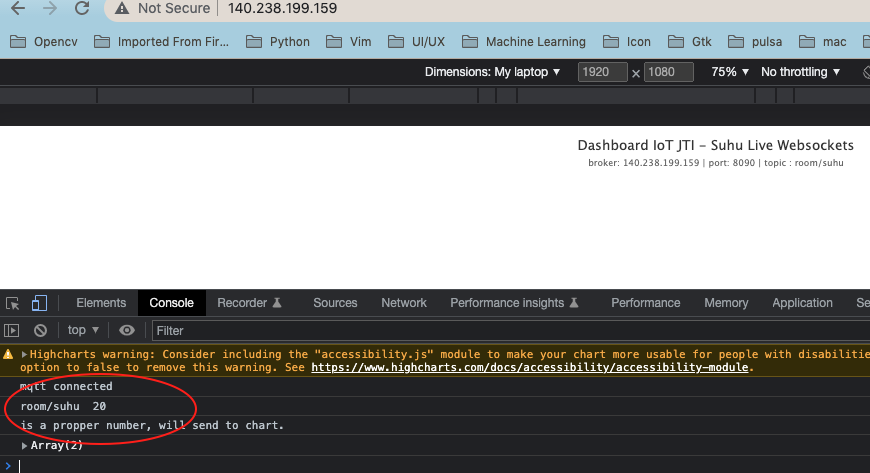
<body onload="init();">

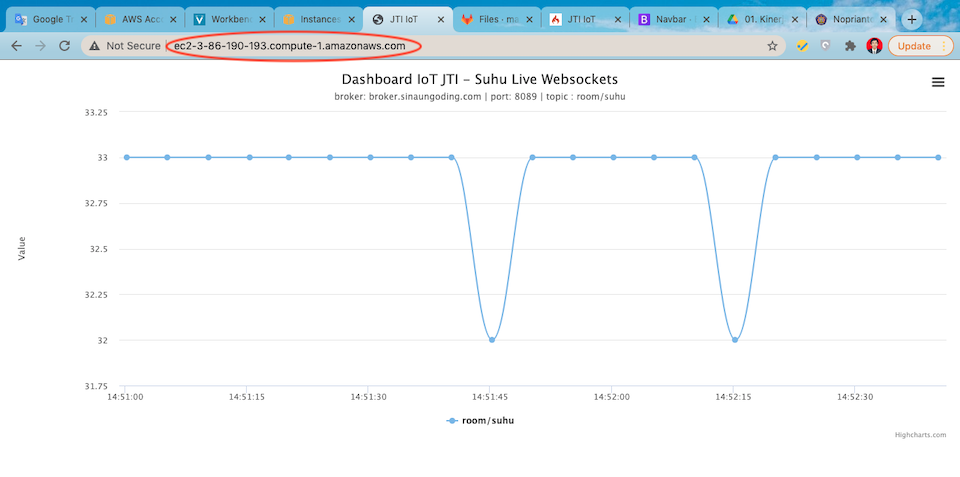
<!--Start the javascript ball rolling and connect to the mqtt broker-->

<div id="container" style="height: 500px; min-width: 500px"></div><!-- this the placeholder for the chart-->

</body>

</html>

Code di atas digunakan untuk menampilkan suhu yang dikirimkan oleh Node-MCU, karena aplikasi web berjalan di atas browser sehingga protokol yang digunakan tidak menggunakan http biasanya untuk komunikasi antara aplikasi dengan server broker. Yang diguankan adalah protokol websocket, komunikasi tersebut di-hanlde oleh javascript dengan bantuan library pahomqtt.  
Perlu disesuikan port, server, dan topik dari kode di atas agar bisa menampilkan data, silakan disesuaikan dengan konfigurasi yang telah Anda lakukan sebelumnya. Untuk melakukan pengecekan silakan buka browser Anda kemudian ketika alamat server Anda dana amati log dengan cara inspect element. Berikut ini adalah tampilan ketika web socket sudah terhubung ke message broker.  Tampilan di atas ketika tidak ada data  Sedangkan tampilan di atas adalah dengan cara inspect element untuk mengetahui logging pada kode program untuk mengetahui web socket terhubung ke message broker.  
Ketika tidak terhubung dipastikan kembali konfigurasi pada NodeMCU dan halaman html terkait dengan host message broker, port, user, dan password. Ketika tidak terhubung menggunakan user password coba pada bagian per\_listener\_settings true, silakan ubah menjadi false.

1. Upload kode tersebut ke server, bisa menggunakan WinSCP atau ketika sudah menggunakan repository berarti perlu push ke repo selanjutnya yang di server perlu dilakukan pull. Selanjutnya buka browser Anda dan ketik alamat aplikasi web yang ada di instance EC2, seharusnya menampilkan halaman seperti berikut  
   

#### **Pertanyaan**

1. Berapa lama sekali pengiriman data sensor DHT dari Node-MCU ke server broker? Silakan diubah menjadi lebih cepat!

**Jawab :**

1. Fungsi baris perintah dtostrf(temperature, 4, 2, temperatureTemp); digunakan untuk apa? Jelaskan!

**Jawab :**

1. Modifikasi bentuk chart pada halaman index.html dalam bentuk yang lain, misalkan chart bar ataupun gauge!

**Jawab :**

## 

## 

## 

## **Tugas**

Buatlah sebuah tampilan website yang fungsi utama adalah untuk menampilkan sensor cahaya, suhu, dan kelembaban. Kemudian Dapat juga menghidupkan masing-masing LED RGB, website tersebut harus telah terupload di OCI(lampirkan alamatnya pada laporan praktikum) dan videokan hasilnya.