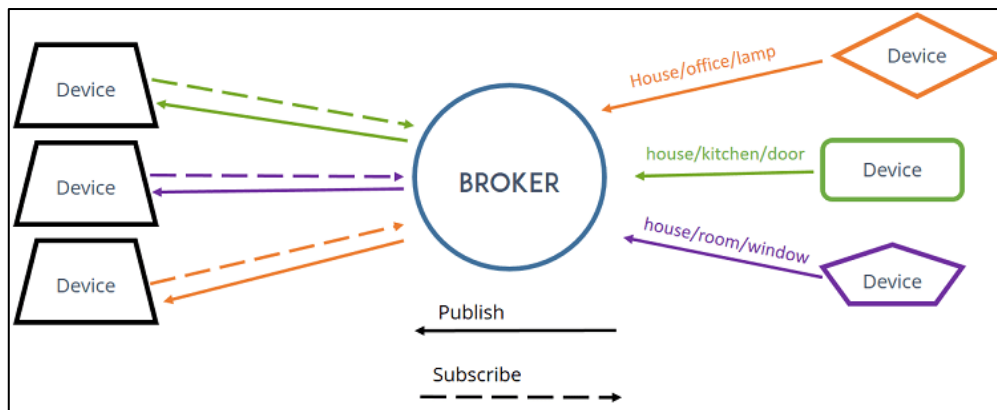


MODUL 6

Membuat Website Sederhana dengan Protokol MQTT

Percobaan 1

Menginstal Mosquitto Broker



Gambar 1. Skema Fungsi Broker Pada Protkol MQTT

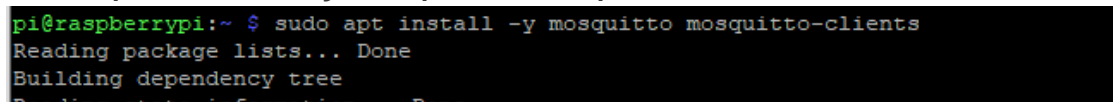
1. Buka Terminal



2. Install mosquitto broker terlebih dahulu,

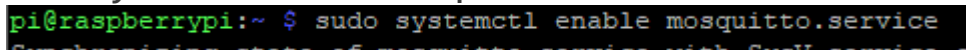
```
sudo apt update
```

```
sudo apt install -y mosquitto mosquitto-clients
```



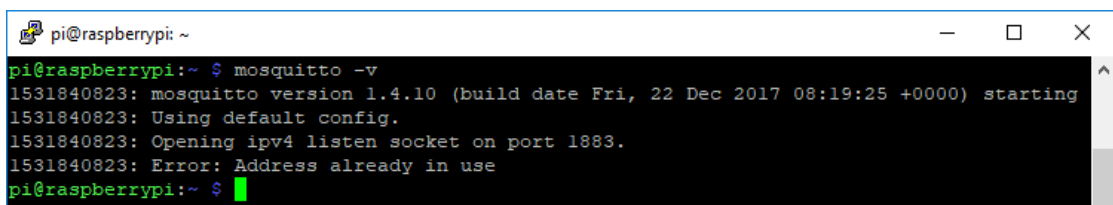
3. Kemudian aktivasi servis dari mosquitto broker

```
sudo systemctl enable mosquitto.service
```



4. Kemudian untuk mengecek mosquitto sudah terinstall

```
mosquitto -v
```

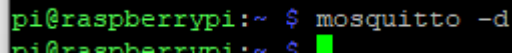


Percobaan 2

Membuat webserver Flask

1. Nyalakan Mosquitto

```
mosquitto -d
```



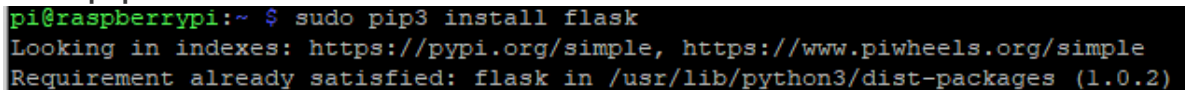
2. Kemudian install Flask

```
sudo apt update
```

```
sudo apt upgrade
```

```
sudo apt-get install python-pip python-flask
```

```
sudo pip3 install flask
```

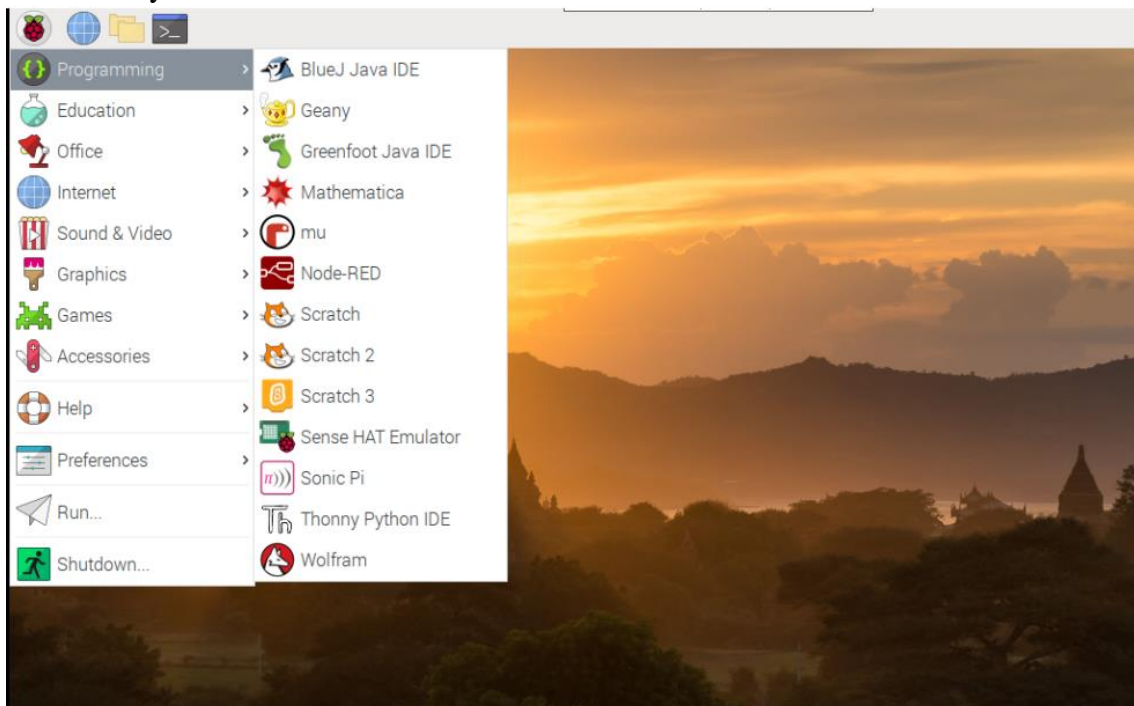


3. Install Paho MQTT

```
sudo pip3 install paho-mqtt
```



4. Buka Geany



5. Isikan kode program python berikut:

```
# Created by Rui Santos
# Complete project details: https://randomnerdtutorials.com
#

import paho.mqtt.client as mqtt
from flask import Flask, render_template, request
app = Flask(__name__)

mqttc=mqtt.Client()
mqttc.connect("localhost",1883,60)
mqttc.loop_start()

# Create a dictionary called pins to store the pin number, name, and
pin state:
pins = {
    4 : {'name' : 'GPIO 4', 'board' : 'esp8266', 'topic' :
'esp8266/4', 'state' : 'False'},
    5 : {'name' : 'GPIO 5', 'board' : 'esp8266', 'topic' :
'esp8266/5', 'state' : 'False'}
}

# Put the pin dictionary into the template data dictionary:
templateData = {
    'pins' : pins
}

@app.route("/")
def main():
    # Pass the template data into the template main.html and return
it to the user
    return render_template('main.html', **templateData)

# The function below is executed when someone requests a URL with
the pin number and action in it:
@app.route("/<board>/<changePin>/<action>")

def action(board, changePin, action):
    # Convert the pin from the URL into an integer:
    changePin = int(changePin)
    # Get the device name for the pin being changed:
    devicePin = pins[changePin]['name']
    # If the action part of the URL is "on," execute the code
indented below:
    if action == "1" and board == 'esp8266':
        mqttc.publish(pins[changePin]['topic'], "1")
        pins[changePin]['state'] = 'True'

    if action == "0" and board == 'esp8266':
        mqttc.publish(pins[changePin]['topic'], "0")
```

```

    pins[changePin]['state'] = 'False'

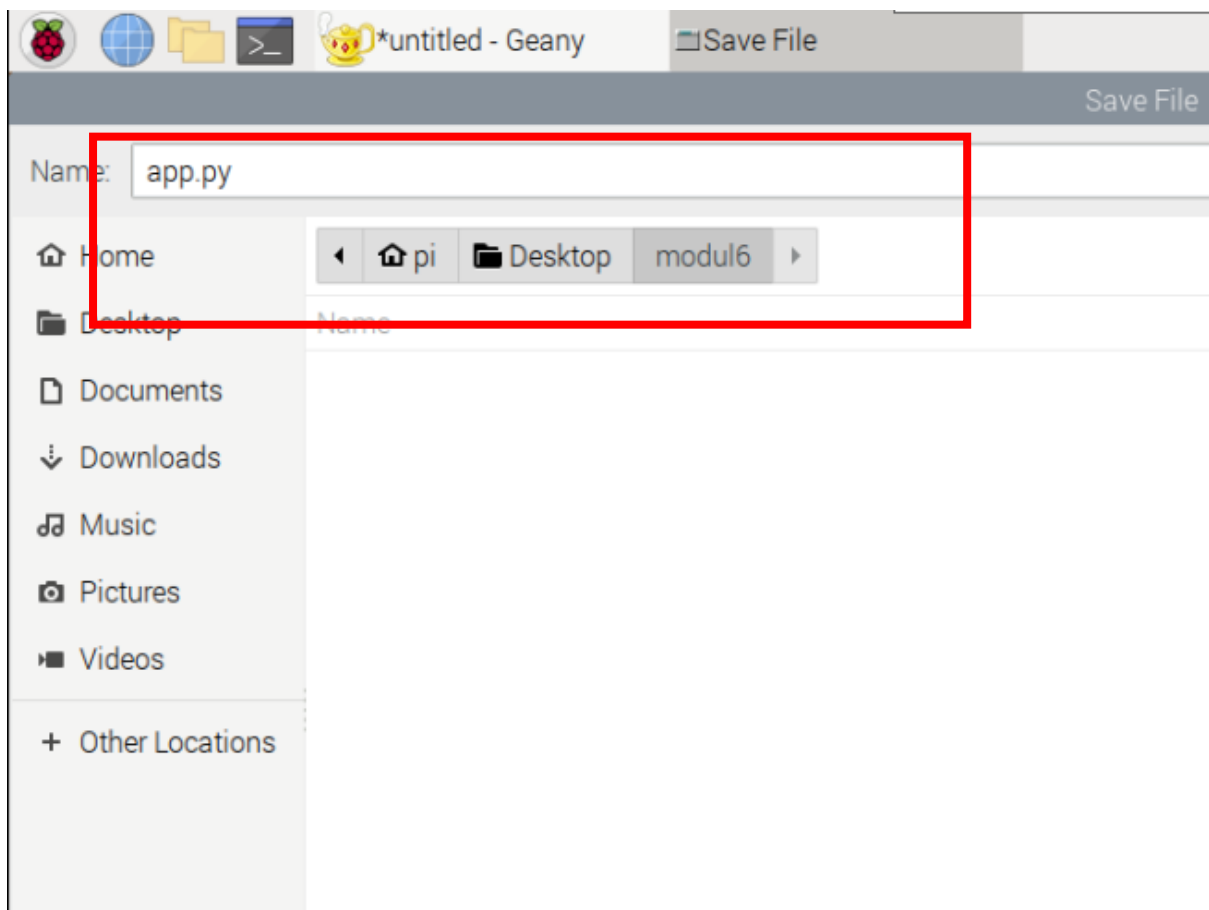
    # Along with the pin dictionary, put the message into the
    template data dictionary:
    templateData = {
        'pins' : pins
    }

    return render_template('main.html', **templateData)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=8000, debug=True)

```

6. Kemudian buat folder bernama modul6 dan simpan file dengan nama app.py

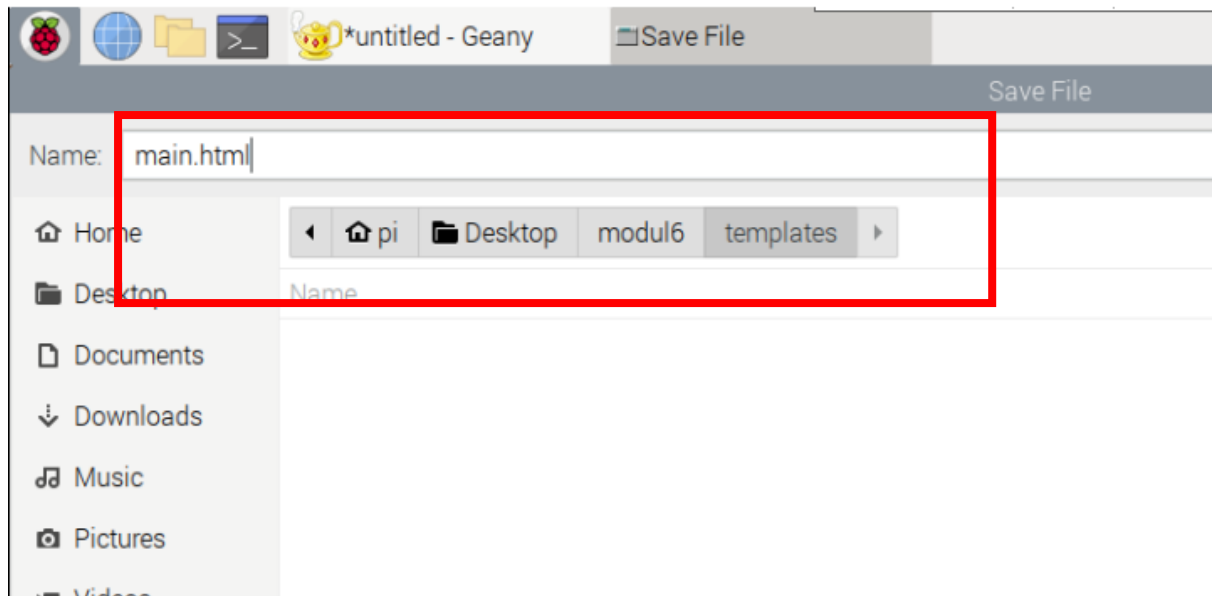


7. Buka kembali tab baru di geany dan isikan kode program berikut

```
<!DOCTYPE html>
<head>
  <title>RPi Web Server</title>
  <!-- Latest compiled and minified CSS -->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.
min.css" integrity="sha384-
1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7"
crossorigin="anonymous">
  <!-- Optional theme -->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap-
theme.min.css" integrity="sha384-
fLW2N01lMqjakBkx3l/M9EahuwpsfSfeNvV63J5ezn3uZzapT0u7EYsXMjQV+0En5r"
crossorigin="anonymous">
  <!-- Latest compiled and minified JavaScript -->
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.mi
n.js" integrity="sha384-
0mSbJDEHialfmuBBQP6A4Qrprq5OVfw37PRR3j5ELqxs1yVq0tnepnHVP9aJ7xS"
crossorigin="anonymous"></script>
  <meta name="viewport" content="width=device-width, initial-
scale=1">
</head>

<body>
  <h1>RPi Web Server - ESP8266 MQTT</h1>
  {% for pin in pins %}
  <h2>{{ pins[pin].name }}
  {% if pins[pin].state == 'True' %}
    is currently <strong>on</strong></h2><div class="row"><div
class="col-md-2">
    <a href="/esp8266/{{pin}}/0" class="btn btn-block btn-lg btn-
default" role="button">Turn off</a></div></div>
    {% else %}
    is currently <strong>off</strong></h2><div class="row"><div
class="col-md-2">
    <a href="/esp8266/{{pin}}/1" class="btn btn-block btn-lg btn-
primary" role="button">Turn on</a></div></div>
    {% endif %}
  {% endfor %}
</body>
</html>
```

8. Kemudian buat folder lagi bernama templates pada folder modul6 kemudian simpan file dengan nama main.html



9. Kemudian jalankan file **app.py**, maka server akan berjalan seperti ini

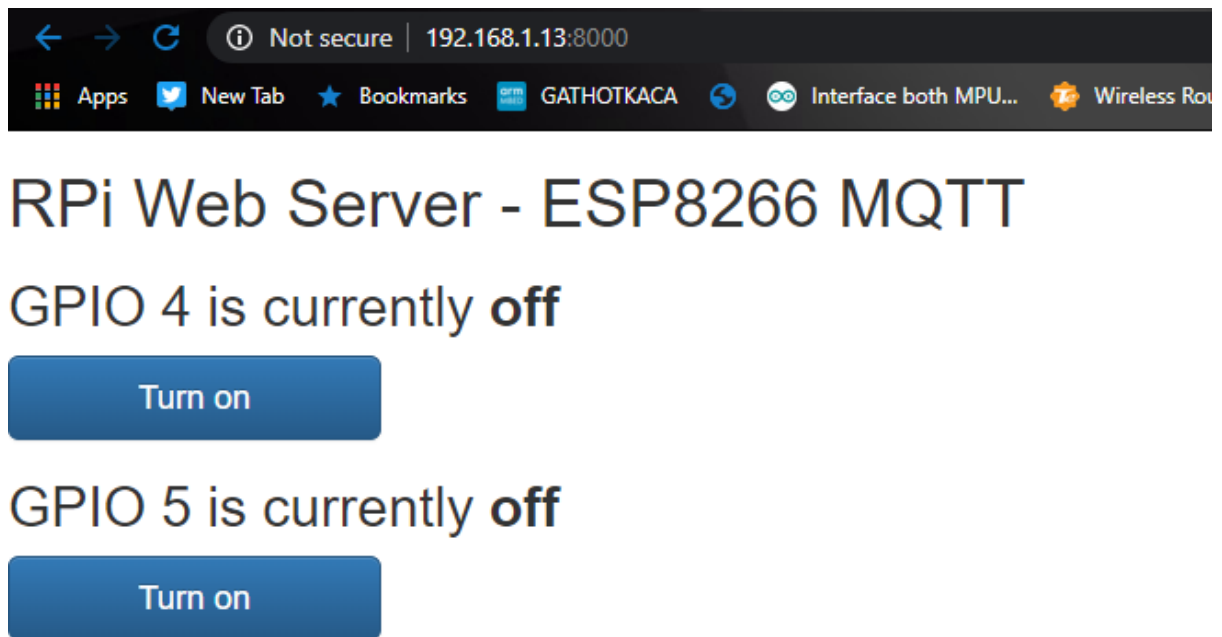
A screenshot of a terminal window titled 'geany_run_script_RSINM0.sh'. The terminal displays the output of running a Flask application. The output includes a warning about SSH, Flask app information, and HTTP request logs. The logs show a successful GET request for '/' and a 404 error for '/favicon.ico'.

```
geany_run_script_RSINM0.sh
File Edit Tabs Help

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://0.0.0.0:8000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 432-744-008
192.168.1.8 - - [29/Jun/2020 10:25:32] "GET / HTTP/1.1" 200 -
192.168.1.8 - - [29/Jun/2020 10:25:33] "GET /favicon.ico HTTP/1.1" 404 -
```

10. Buka pada browser komputer anda dengan **ip_raspberry:8000** maka akan muncul tampilan seperti ini. (IP raspberry disini contohnya 192.168.1.13, kemudian port 8000)

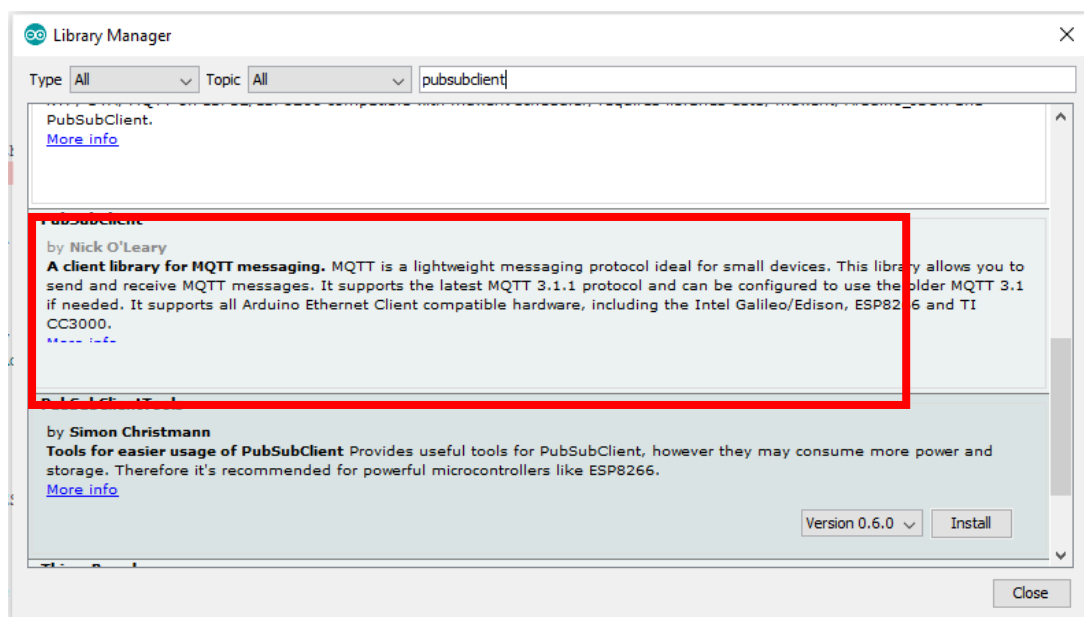


Percobaan 3

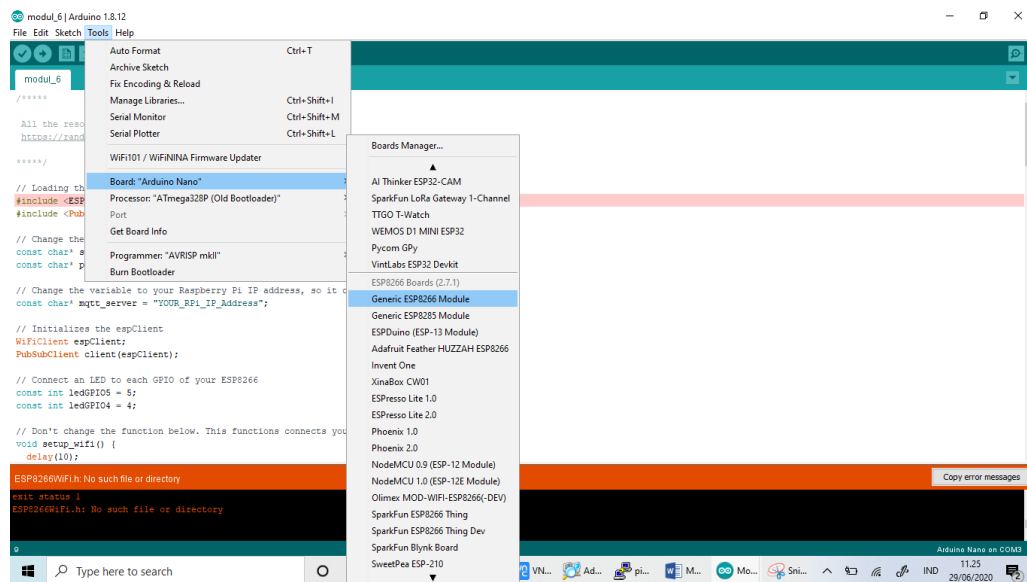
Membuat Node Menggunakan ESP8266

1. Download Library Pubsubclient dilink berikut
<https://github.com/knolleary/pubsubclient/archive/master.zip>

atau install melalui Arduino IDE, pilih milik nick O'Leary



2. Pilih Board ESP8266 dan asukkan program berikut ke Arduino IDE dan Flash pada ESP8266



JANGAN LUPA MENGGANTI SSID, PASSWORD, DAN IP PADA PROGRAM ARDUINO SESUAI DENGAN MILIK ANDA.

```
/*  
All the resources for this project:  
https://randomnerdtutorials.com/  
*/  
  
// Loading the ESP8266WiFi library and the PubSubClient library  
#include <ESP8266WiFi.h>  
#include <PubSubClient.h>  
  
// Change the credentials below, so your ESP8266 connects to your  
router  
const char* ssid = "YOUR_SSID";  
const char* password = "YOUR_PASSWORD";  
  
// Change the variable to your Raspberry Pi IP address, so it connects  
to your MQTT broker  
const char* mqtt_server = "YOUR_RPi_IP_Address";  
  
// Initializes the espClient  
WiFiClient espClient;  
PubSubClient client(espClient);  
  
// Connect an LED to each GPIO of your ESP8266  
const int ledGPIO5 = 5;  
const int ledGPIO4 = 4;  
  
// Don't change the function below. This functions connects your  
ESP8266 to your router
```



```

void setup_wifi() {
    delay(10);
    // We start by connecting to a WiFi network
    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());
}

// This functions is executed when some device publishes a message to a
// topic that your ESP8266 is subscribed to
// Change the function below to add logic to your program, so when a
// device publishes a message to a topic that
// your ESP8266 is subscribed you can actually do something
void callback(String topic, byte* message, unsigned int length) {
    Serial.print("Message arrived on topic: ");
    Serial.print(topic);
    Serial.print(". Message: ");
    String messageTemp;

    for (int i = 0; i < length; i++) {
        Serial.print((char)message[i]);
        messageTemp += (char)message[i];
    }
    Serial.println();

    // Feel free to add more if statements to control more GPIOs with
    MQTT

    // If a message is received on the topic home/office/esp1/gpio2, you
    // check if the message is either 1 or 0. Turns the ESP GPIO according to
    // the message
    if(topic=="esp8266/4"){
        Serial.print("Changing GPIO 4 to ");
        if(messageTemp == "1"){
            digitalWrite(ledGPIO4, HIGH);
            Serial.print("On");
        }
        else if(messageTemp == "0"){
            digitalWrite(ledGPIO4, LOW);
            Serial.print("Off");
        }
    }
    if(topic=="esp8266/5"){

```

```

    Serial.print("Changing GPIO 5 to ");
    if(messageTemp == "1"){
        digitalWrite(ledGPIO5, HIGH);
        Serial.print("On");
    }
    else if(messageTemp == "0"){
        digitalWrite(ledGPIO5, LOW);
        Serial.print("Off");
    }
}
Serial.println();
}

// This functions reconnects your ESP8266 to your MQTT broker
// Change the function below if you want to subscribe to more topics
with your ESP8266
void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        // Attempt to connect
        /*
        YOU NEED TO CHANGE THIS NEXT LINE, IF YOU'RE HAVING PROBLEMS WITH
MQTT MULTIPLE CONNECTIONS
        To change the ESP device ID, you will have to give a unique name
to the ESP8266.
        Here's how it looks like now:
        if (client.connect("ESP8266Client")) {
        If you want more devices connected to the MQTT broker, you can do
it like this:
        if (client.connect("ESPOffice")) {
        Then, for the other ESP:
        if (client.connect("ESPGarage")) {
        That should solve your MQTT multiple connections problem

        THE SECTION IN loop() function should match your device name
        */
        if (client.connect("ESP8266Client")) {
            Serial.println("connected");
            // Subscribe or resubscribe to a topic
            // You can subscribe to more topics (to control more LEDs in this
example)
            client.subscribe("esp8266/4");
            client.subscribe("esp8266/5");
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds");
            // Wait 5 seconds before retrying
            delay(5000);
        }
    }
}

```

```

    }
}

// The setup function sets your ESP GPIOs to Outputs, starts the serial
communication at a baud rate of 115200
// Sets your mqtt broker and sets the callback function
// The callback function is what receives messages and actually
controls the LEDs
void setup() {
    pinMode(ledGPIO4, OUTPUT);
    pinMode(ledGPIO5, OUTPUT);

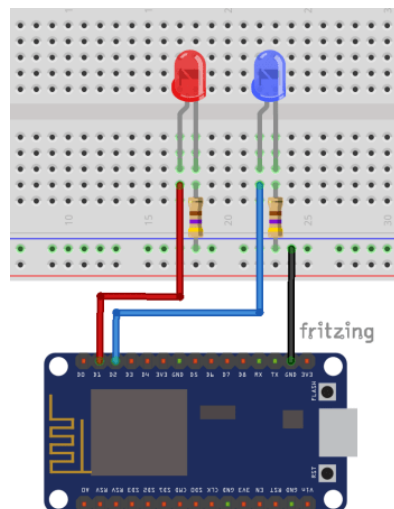
    Serial.begin(115200);
    setup_wifi();
    client.setServer(mqtt_server, 1883);
    client.setCallback(callback);
}

// For this project, you don't need to change anything in the loop
function.
// Basically it ensures that you ESP is connected to your broker
void loop() {
    if (!client.connected()) {
        reconnect();
    }
    if(!client.loop())
        /*
        YOU NEED TO CHANGE THIS NEXT LINE, IF YOU'RE HAVING PROBLEMS WITH
MQTT MULTIPLE CONNECTIONS
        To change the ESP device ID, you will have to give a unique name
to the ESP8266.
        Here's how it looks like now:
        client.connect("ESP8266Client");
        If you want more devices connected to the MQTT broker, you can do
it like this:
        client.connect("ESPOffice");
        Then, for the other ESP:
        client.connect("ESPGarage");
        That should solve your MQTT multiple connections problem

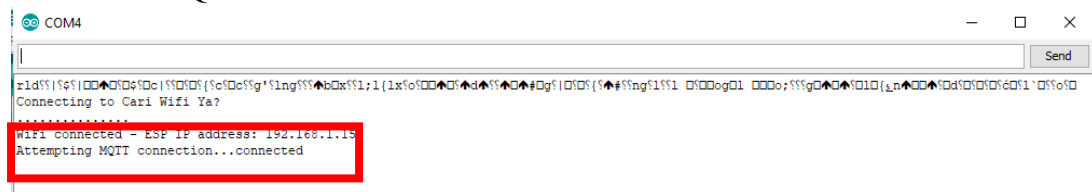
        THE SECTION IN reconnect() function should match your device name
        */
        client.connect("ESP8266Client");
    }
}

```

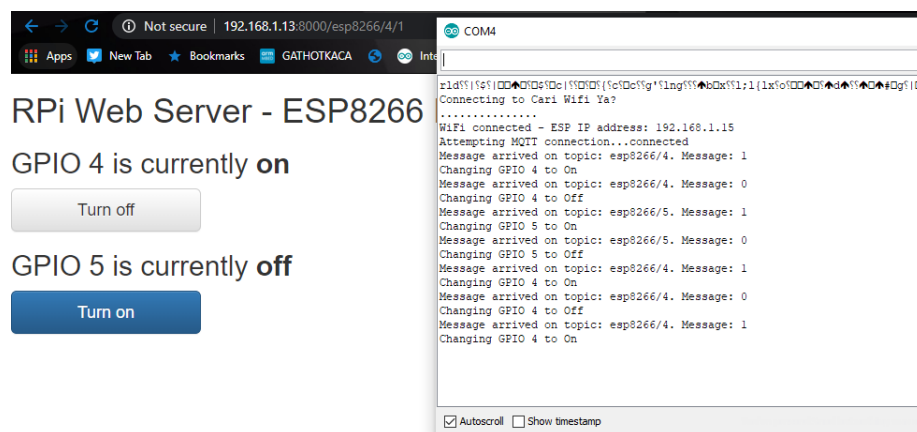
Kemudian rangkai LED dan ESP8266 seperti berikut: Gunakan GPIO4 dan GPIO5



3. Buka Serial monitor dengan baudrate 115200. Jika ESP8266 sudah berhasil ke SSID wifi maka akan muncul MQTT connection connected.



4. Jika berhasil terhubung ke webserver yang anda buat, maka ketika anda menekan tombol pada web lampu LED akan menyala dan mati sesuai dengan tombol yang anda tekan. Dan untuk mengetahui pesan sudah masuk atau belum, anda dapat melihat pesannya di serial monitor ESP8266.



TUGAS

Tuliskan dalam laporan anda:

1. Sebutkan jenis-jenis protokol komunikasi yang sering digunakan dalam IoT (minimal 3)!
2. Jelaskan kelebihan dan kekurangan protokol komunikasi tersebut!

Sumber:

1. (<https://randomnerdtutorials.com/raspberry-pi-publishing-mqtt-messages-to-esp8266/>)
2. (<https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/>)