

# [PRE-RELEASE] Implementasi **Internet of Things** dengan **Arduino ESP8266/ESP32**, **MQTT**, dan integrasi **Web** menggunakan **JavaScript**, **PHP**, **HTML5**, dan **MySQL**.

Ajang Rahmat, S.Kom.

Tech Stack:



# Pengumuman \_

Mohon Maaf Sebelumnya, ebook ini terbit dengan status **[PRE-RELEASE]**, yang artinya masih ada sebagian dari isi sekitar 25% lagi belum ada di Ebook ini karena sedang di sempurnakan penulis.

Penulis usahakan akan secepatnya rilis 100% dalam bulan ini.

Sambil nunggu silahkan praktekan 75% praktikum yang ada dalam ebook ini, dan rekan-rekan **WAJIB MASUK GRUP TELEGRAM:**

**<https://t.me/+NsCxP9yn9aU1YzQ1>**

Karena Update Mengenai Ebook ini akan di Share Disana.

# Pembahasan

- Persiapan Hardware dan Software
- Pengenalan Internet of Things dan Penerapannya
- Pengembangan Perangkat IoT dengan ESP8266/ESP32
- Integrasi Web dengan JavaScript dan MQTT
- Integrasi Database MySQL dan Pengembangan Lanjutan

# **1.** Persiapan Hardware dan Software

# 1.1. HARDWARE

## 1.1.1. Hardware Yang Harus Disiapkan \_

1. Board ESP8266 / ESP32 gunakan salah satu diantara:
  - NodeMCU Amica / Lolin
  - Wemos D1 Mini
  - ESP32 DO IT DEV KIT
2. Project Board (Breadboard) 400 Lubang
3. Kabel Jumper Tunggal atau Type Male-Male
4. Sensor Cahaya LDR
5. Lampu LED 5MM
6. Push Button
7. Resistor Ukuran 220 ohm, 1K ohm dan 10K ohm

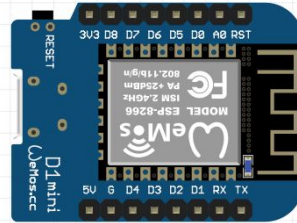
## 1.1.2. Board Microcontroller

Berikut adalah **4 board Microcontroller** yang bisa digunakan sesuai panduan ebook ini.

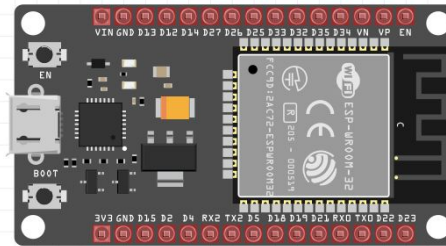
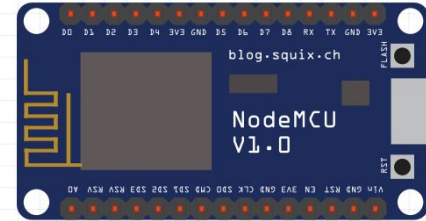
Keempatnya bisa dibeli dengan mudah di Marketplace kesayangan Anda.

Silakan gunakan salah satunya.

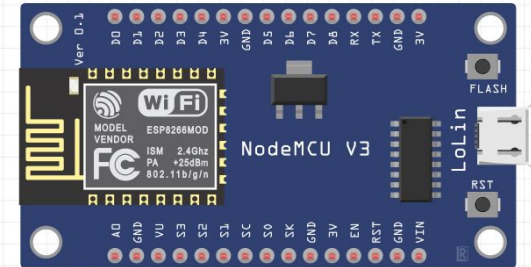
WeMos D1 Mini ESP8266



NodeMCU Amica ESP8266



ESP32 DOIT DEV KIT V1  
ESP32S-PIN30



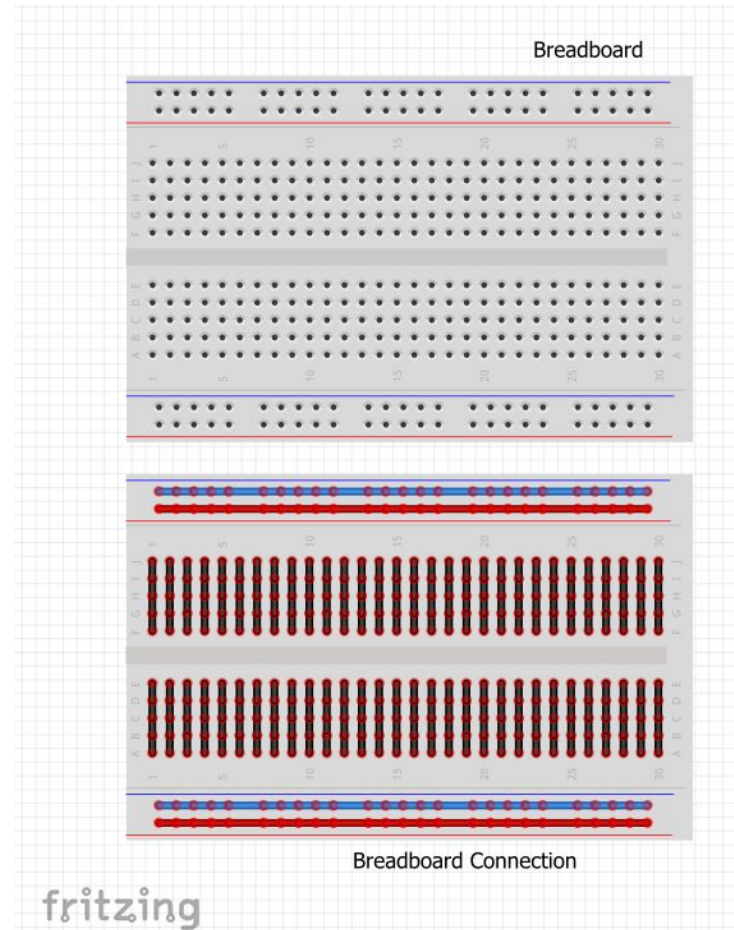
NodeMCU V3

NodeMCU V3

### 1.1.3. Breadboard (Papan Project)

Breadboard menjadi bagian penting dalam belajar Microcontroller, karena dengan mudah proyek yang dibuat bisa di bongkar pasang.

Tapi bagi yang baru, **perlu mengetahui lubang-lubang pada breadboard ada yang terhubung**, bisa dilihat dari gambar **Breadboard Connection**.





## 1.2. SOFTWARE

## 1.2.1. Persiapan Software \_

File Software  
**Sudah Tersedia**  
di Group Telegram

1. Arduino IDE versi 1.8.19:  
<https://downloads.arduino.cc/arduino-1.8.19-windows.exe>
2. Laragon versi Full:  
<https://github.com/leokhoa/laragon/releases/download/6.0.0/laragon-wamp.exe>
3. Visual Studio Code:  
<https://code.visualstudio.com/sha/download?build=stable&os=win32-x64-user>

\*bagi pengguna OS selain Windows, silakan sesuaikan sendiri

## 1.2.2. Tutorial: Persiapan Software \_

1. Tutorial Instalasi Arduino IDE versi 1.8.19:  
<https://www.youtube.com/watch?v=btP7dDq1yOI>
2. Tutorial Instalasi Laragon:  
<https://www.youtube.com/watch?v=T9CP0xSc40g>
3. Tutorial Instalasi Visual Studio Code:  
<https://www.youtube.com/watch?v=EulXNh7JZ7U>

\*bagi yang awam banget, silakan bisa ikuti tutorial tersebut.

### 1.2.3. Menambahkan **Library Board ESP32 & ESP8266** ke **Arduino IDE** \_

\*Pastikan tidak ada antivirus yang run atau di Install, kalo ada matikan dan uninstall dulu.

1. Pada Arduino IDE, buka menu **File** lalu pilih **Preferences**
2. Setelah terbuka Preferences, pada bagian **Additional Board Manager**, masukan:  
[https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\\_esp32\\_index.json](https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json),  
[http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)
3. Lalu Klik **OK**

Selanjutnya silahkan buka menu **Tools**, lalu pilih **Board**, lalu pilih **Board Manager...**

Untuk yang menambahkan **ESP32** silakan cari **esp32**, kemudian **pilih versi 1.0.6**, kemudian **Install**.

Untuk yang menambahkan **ESP8266** silahkan cari **esp8266**, kemudian **pilih versi 2.7.4**, kemudian **Install**.

Proses Instalasi cukup lama silahkan tunggu sambil ngopi.

Kalo sudah tinggal klik **Close** aja.

\*pemilihan versi 1.0.6 untuk ESP32 dan 2.7.4 untuk ESP8266 adalah karena masih versi yang stabil dengan library yang beredar untuk Arduino.

## **1.5. Tutorial: Menambahkan Library Board ESP32 & ESP8266 ke Arduino IDE**

Bagi yang masih bingung dengan panduan sebelumnya, silakan bisa ikuti Video Tutorial berikut:

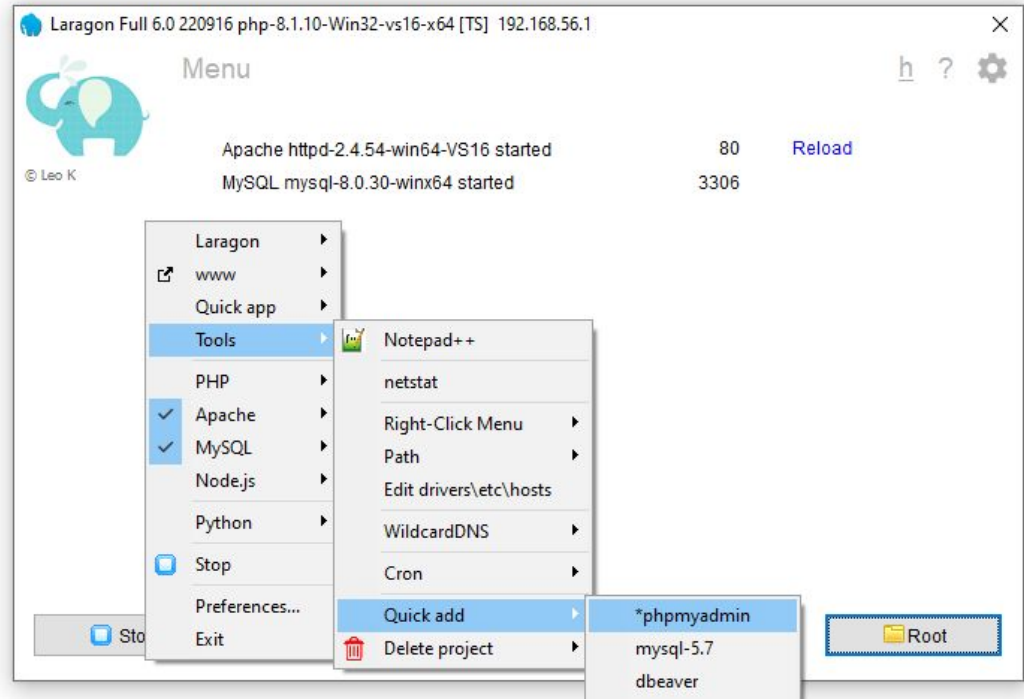
1. Tutorial Menambahkan Library Board ESP32 dan ESP8266 Pada Arduino IDE:

[https://www.youtube.com/watch?v=ucTZPu5FU\\_k](https://www.youtube.com/watch?v=ucTZPu5FU_k)

## 1.6. Menambahkan PHPMyAdmin di Laragon

Awalnya phpmyadmin memang belum terinstall di laragon, tapi bisa dengan mudah ditambahkan dengan cara:

1. **Klik Kanan** lalu pilih **Tools**
2. Pilih **Quick Add**
3. Pilih **\*phpmyadmin**



## 2. Pengenalan Internet of Things dan Penerapannya



## 2.1. Definisi **Internet of Things** \_

Kuncinya ada pada kata Internet, jadi tentu sebuah proyek yang digadang memiliki konsep IoT, Wajib terhubung ke Internet. Tapi tentu dalam proses pengembangan bisa di jaringan lokal terlebih dahulu.

Kemudian ada kata Things disitu, yang merujuk ke benda-benda yang bisa terhubung ke Internet. Dalam hal ini tidak hanya handphone, tablet, laptop, PC aja tetapi benda seperti lampu rumah, kulkas, dan sensor-sensor itu juga bisa dihubungkan ke internet.

Kemudian bisa ditampilkan datanya, bisa dikendalikan benda-bendanya dari jarak jauh dengan syarat terhubung ke Internet.

## 2.2. Sejarah **Internet of Things** \_

IoT pada dasarnya bukanlah teknologi baru, mungkin baru populer setelah booming barengan industri 4.0, yang sering muncul di seminar-seminar, yang sering disebut oleh pejabat-pejabat negeri wakanda. Di Kampus juga sering diadakan kegiatan, bahkan sudah ada matkul nya sendiri sekarang.

Akan tetapi alat yang pertama kali terhubung ke internet adalah **Pemanggang Roti** pada tahun **1989**, oleh **John Romkey** dan **Simon Hackett**.

Dan istilah **IoT** muncul tahun **1999**, dicetuskan oleh **Kevin Ashton**.

## 2.3. Pertumbuhan Perangkat **Internet of Things** \_

Mengutip dari suara.com,

“Di Indonesia, jumlah perangkat IoT diperkirakan akan meningkat ke **678 juta perangkat** pada **2025** dengan hadirnya **5G**”.

Seperti yang dikatakan **Menteri Koinfo Menteri Koinfo Johnny G. Plate** dalam konferensi pers di **ICE BSD, Tangerang**, Selasa (**14/12/2021**).

Untuk saat ini Menteri tersebut memang kena kasus korupsi BTS, tapi kita bisa abaikan saja ya. Hihhi

Yang pasti jumlah perangkat IoT terus bertumbuh.

## 2.4. Potensi Penerapan **Internet of Things** \_

Contoh penerapan IoT pada bidang akuakultur adalah Project Smart Feeder buatan perusahaan eFishery. Dibidang perhotelan ada bobobox, dan hotel-hotel yang berbintang.

Selain itu potensi penerapan IoT juga bisa dibidang pertanian, smart city dan masih banyak lagi.

Untuk saat ini hampir semua bidang, termasuk industrial di pabrik juga sudah menerapkan konsep Internet of Things.

Termasuk beberapa proyek yang sudah pernah saya kerjakan.

### **3. Pengembangan Perangkat IoT dengan ESP8266/ESP32**

## 3.1. UPLOAD PROGRAM

### 3.1.1. Upload Program Arduino ke **ESP32** \_

Bagian ini kita akan mencoba Upload Program ke ESP32, khusus bagi rekan-rekan yang mempunyai Board ESP32.

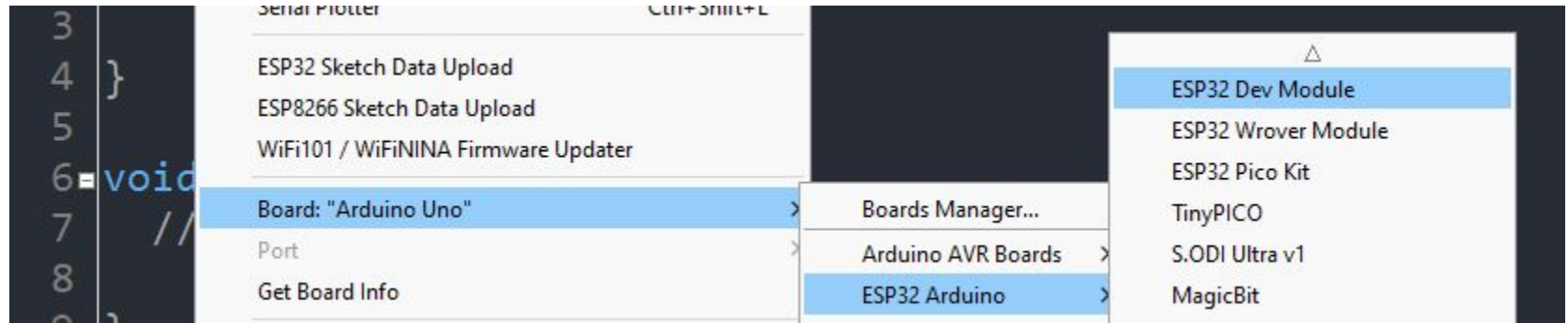
Langkahnya hanya ada 2:

1. **Memilih Board Yang Benar**
2. **Memilih Port Yang Benar**

Seandainya masih gagal, berarti ada kendala di Kabel USB yang digunakan, atau dari Port USB Laptop yang digunakan.

### 3.1.2. Memilih Board Yang Benar (**ESP32**)

Pada Arduino IDE silakan klik menu **Tools**, lalu pilih **Board**, lalu pilih **ESP32 Arduino**, terakhir tinggal pilih **ESP32 Dev Module**.

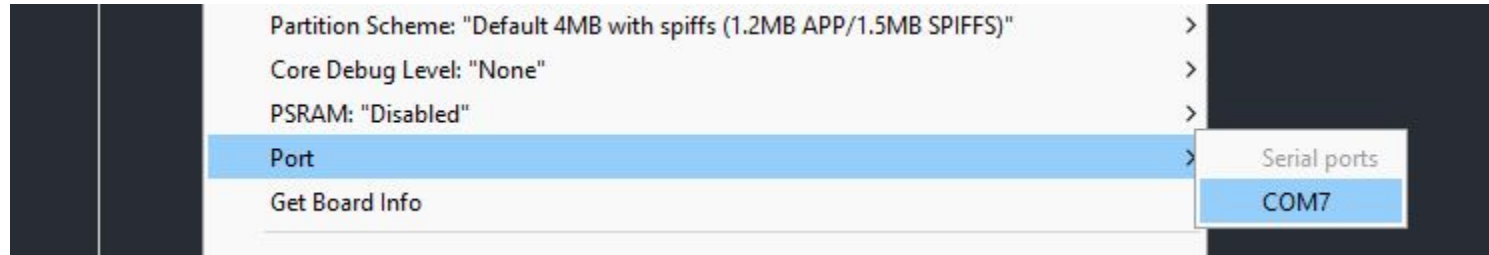




### 3.1.3. Memilih Port Yang Benar (**ESP32**)

Untuk port bisa jadi **punya saya dan punya Anda beda**, jadi paling mudah ya sebelum di pasang USBnya, di cek dulu ke menu **Tools**, lalu pilih **Port**:

Jika pas dipasang USB, **ada port baru muncul**, berarti itu portnya.



### 3.1.4. Upload Program Arduino ke **ESP8266** \_

Untuk ESP8266 baik yang board NodeMCU Amica, Lolin, dan juga Wemos D1 Mini, sama ya dengan ESP32 tahapannya:

- 1. Memilih Board Yang Benar**
- 2. Memilih Port Yang Benar**

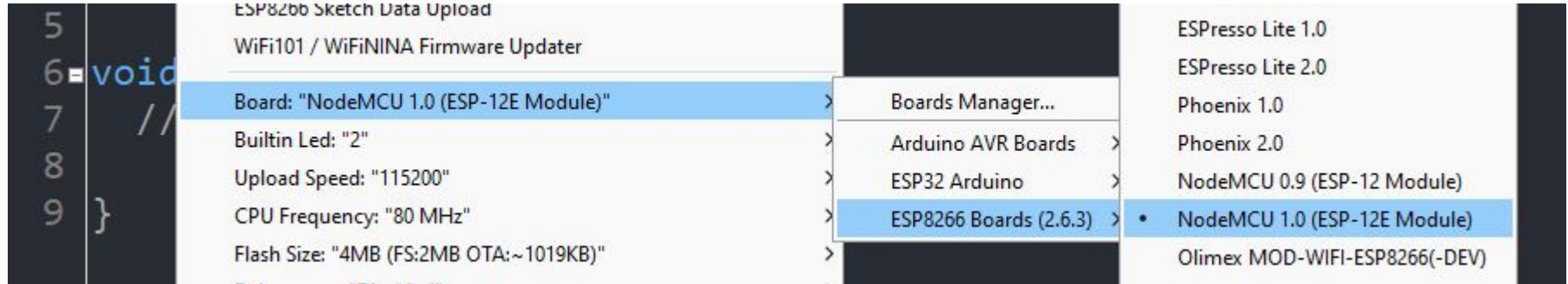
Dan seandainya ada error setelah memilih Board dan Port yang, berarti ada kendala di Kabel USBnya atau juga dari Port USB di laptopnya.

Dan pastikan di compile/verify dulu sebelum upload, untuk memastikan bahwa tidak ada masalah dengan codingnya.

### 3.1.5. Memilih Board Yang Benar (**ESP8266**)

Pada Arduino IDE silakan klik menu **Tools**, lalu pilih **Board**, lalu pilih **ESP8266 Boards**, terakhir tinggal pilih **NodeMCU 1.0**.

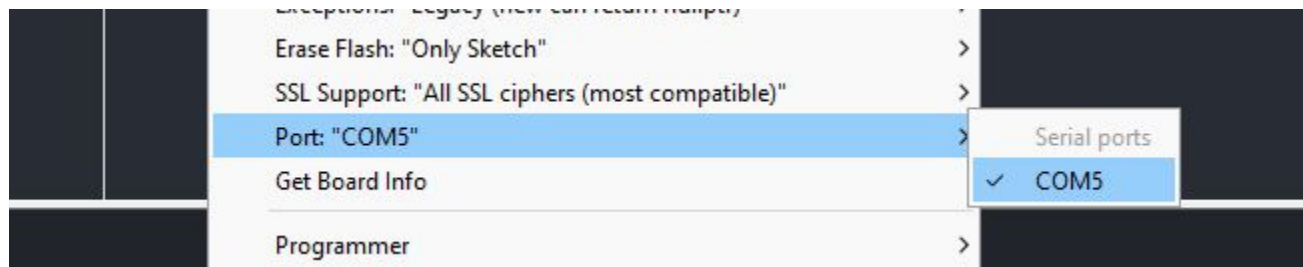
Walaupun board yang digunakan Wemos D1 Mini.



### 3.1.6. Memilih Port Yang Benar (**ESP8266**)

Untuk port bisa jadi **punya saya dan punya Anda beda**, jadi paling mudah ya sebelum di pasang USBnya, di cek dulu ke menu **Tools**, lalu pilih **Port**:

Jika pas dipasang USB, **ada port baru muncul**, berarti itu portnya.



### 3.1.7. Verify dan Upload \_

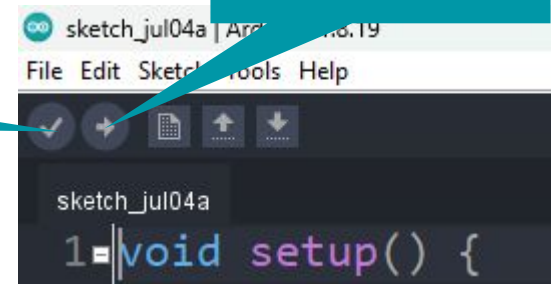
**Verify** adalah proses memastikan coding arduino tidak ada masalah, sebenarnya ketika kita langsung **Upload**, itu juga sebelum **Upload** akan di **Verify** terlebih dahulu.

Tanda berhasil Verify adalah **Done Compiling**, tanda berhasil Upload adalah **Done Uploading**.

```
Done uploading.  
Leaving...  
Hard resetting via RTS pin...
```

Tombol Verify

Tombol Upload



## 3.2. BASIC PROGRAM DIGITAL INPUT DAN OUTPUT \_

# **DIGITAL OUTPUT LED - VERSI BOARD ESP8266**

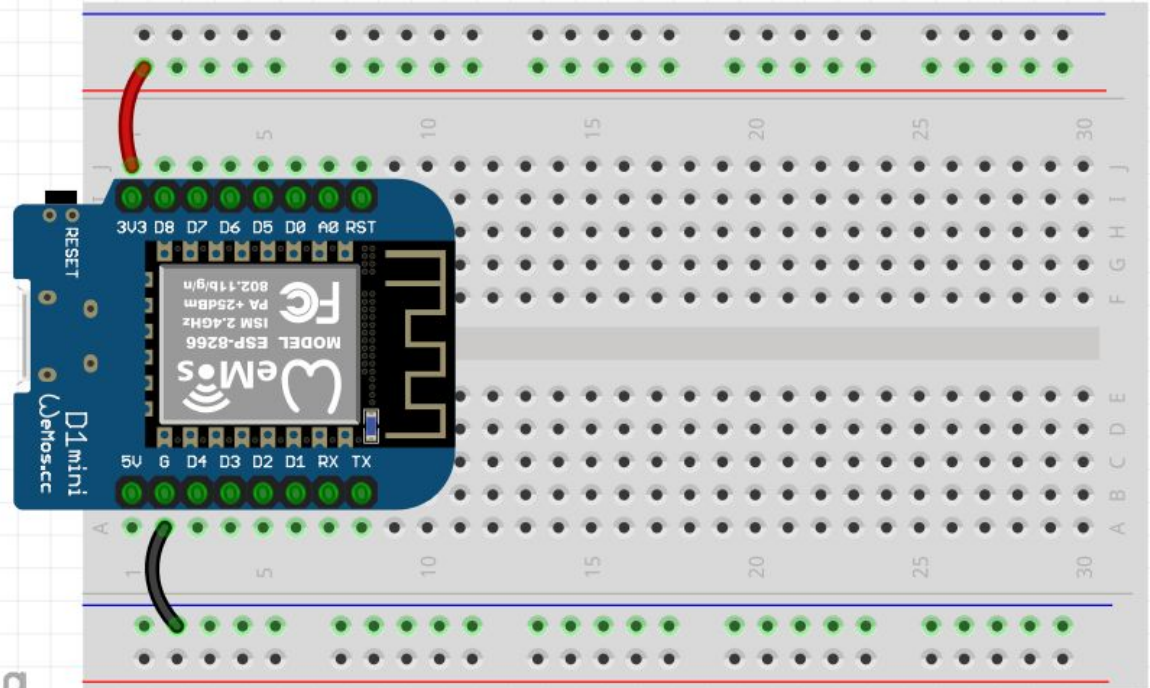
### 3.2.1. Rangkaian Awal versi Wemos D1 Mini \_

Untuk awal silakan pasang **Wemos D1 Mini** ke **Breadboard** seperti posisi gambar disamping.

Lalu kemudian buatlah **terminal 3.3 Volt** dan juga **terminal GND**.

\*Warna Kabel Bebas

fritzing



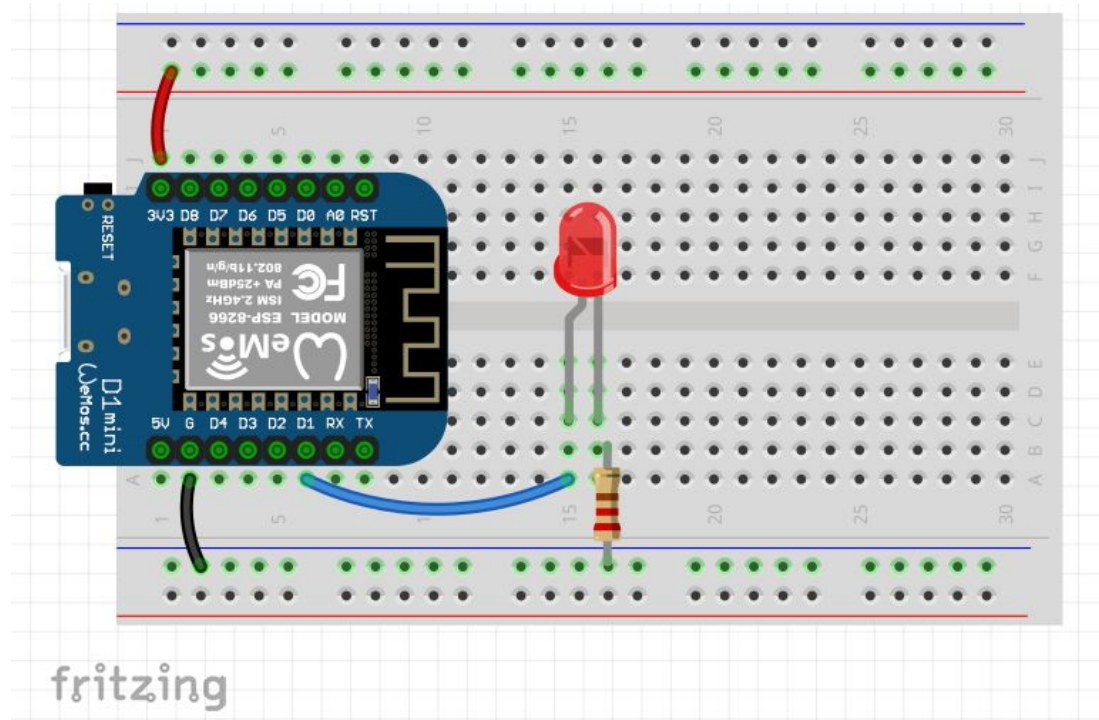


### 3.2.2. Rangkaian Wemos D1 Mini dan LED

Pada LED kaki panjang adalah **Anode (+)** dan kaki pendek adalah **Cathode (-)**.

Rangkaiannya:

1. Anode (kaki panjang LED) ke **Pin D1** Wemos D1 Mini
2. Cathode (kaki pendek LED) ke **Terminal GND** melalui **Resistor 220 Ohm**.

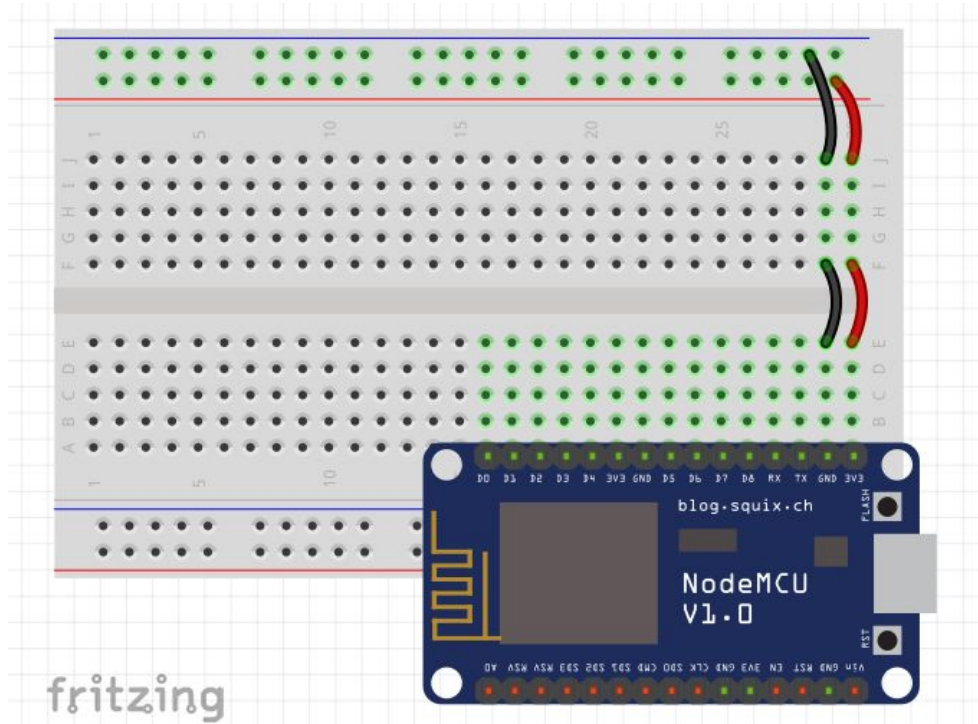


### 3.2.3. Rangkaian Awal versi NodeMCU Amica

Untuk awal silakan pasang **NodeMCU Amica** ke **Breadboard** seperti posisi gambar disamping.

Lalu kemudian buatlah **terminal 3.3 Volt** dan juga **terminal GND**.

\*Warna Kabel Bebas

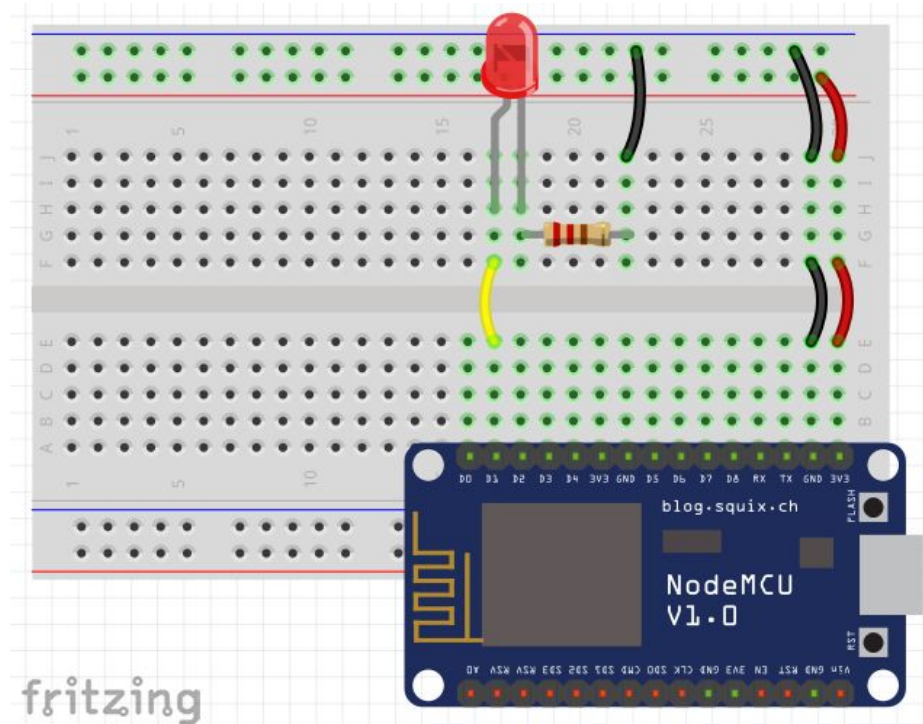


### 3.2.4. Rangkaian NodeMCU Amica dan LED

Pada LED kaki panjang adalah **Anode (+)** dan kaki pendek adalah **Cathode (-)**.

Rangkaiannya:

1. Anode (kaki panjang LED) ke Pin D1 NodeMCU Amica.
2. Cathode (kaki pendek LED) ke Terminal GND melalui Resistor 220 Ohm.







### 3.2.7. Coding Basic ESP8266 LED - Digital Output \_

Untuk yang menggunakan **Board ESP8266**, baik **Wemos D1 Mini**, maupun **NodeMCU Amica** atau **Lolin**.

Silakan gunakan coding disamping, untuk **menyalakan LED selama 3 detik**, kemudian LED akan mati setelahnya.

```
const byte pinLed = D1;

void setup() {
  pinMode(pinLed, OUTPUT);
  digitalWrite(pinLed, HIGH);
  delay(3000);
  digitalWrite(pinLed, LOW);
}

void loop() {}
```

### 3.2.8. Coding ESP8266 Blink LED - Digital Output \_

Berikut kode untuk membuat **LED berkedip** (blink) pada board **ESP8266**.

Kode ini akan **menyalakan LED selama 1 detik, mematikannya selama 1 detik, dan mengulangi** proses tersebut **terus-menerus**.

```
const byte pinLed = D1;

void setup() {
  pinMode(pinLed, OUTPUT);
}

void loop() {
  digitalWrite(pinLed, HIGH);
  delay(1000);
  digitalWrite(pinLed, LOW);
  delay(1000);
}
```

# **DIGITAL OUTPUT LED - VERSI BOARD ESP32**

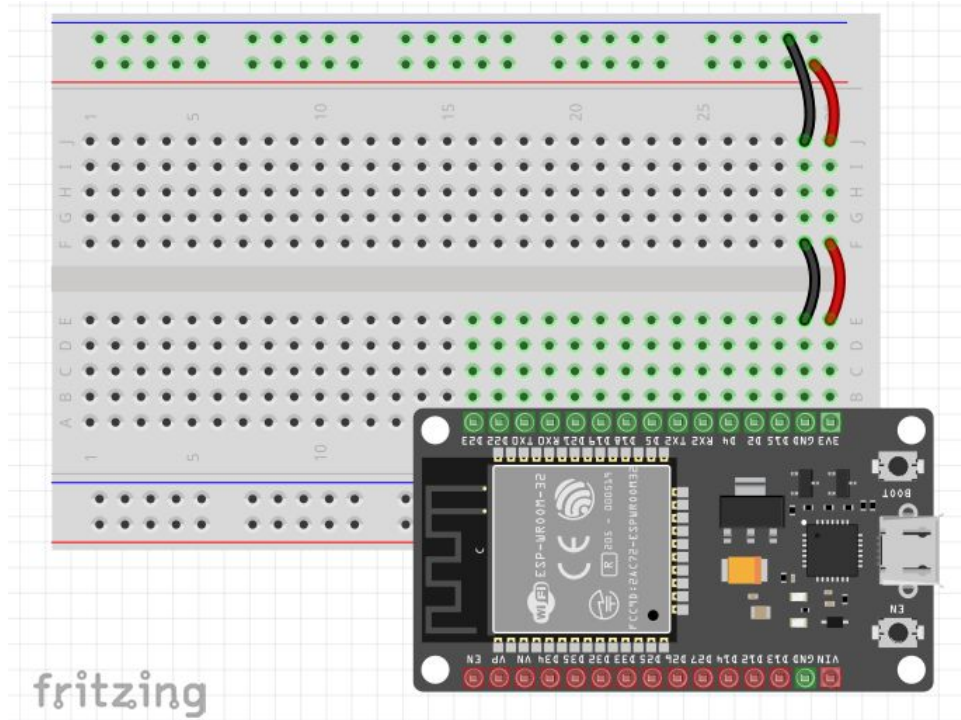


### 3.2.9. Rangkaian Awal versi **ESP32 DEV KIT**

Untuk awal silakan pasang **ESP32 DEV KIT** ke **Breadboard** seperti posisi gambar disamping.

Lalu kemudian buatlah **terminal 3.3 Volt** dan juga **terminal GND**.

\*Warna Kabel Bebas

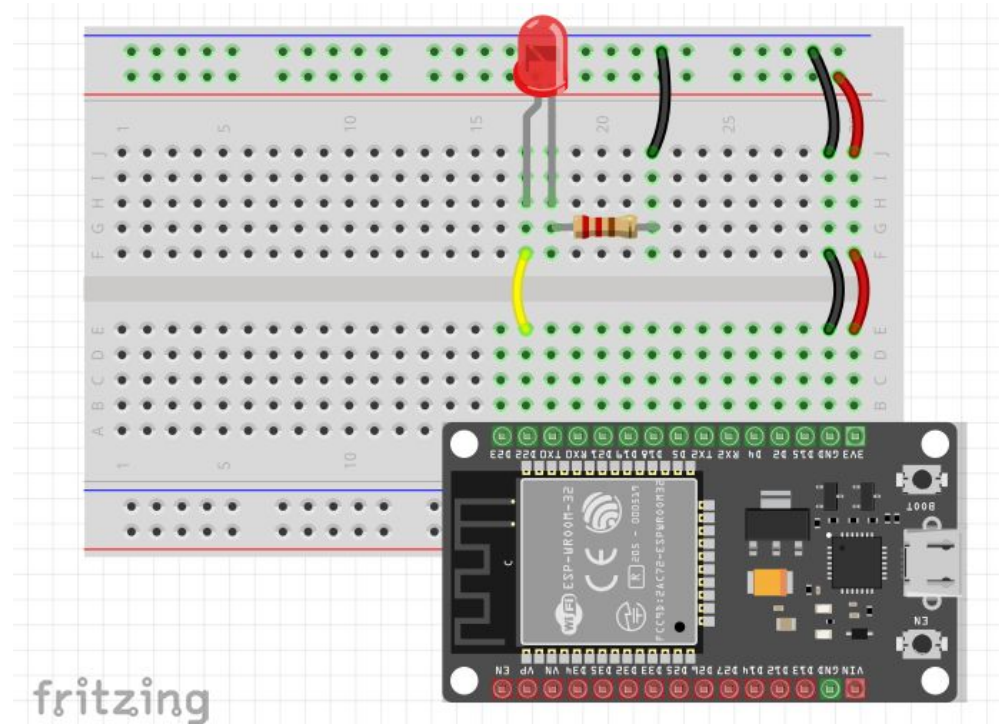


### 3.2.10. Rangkaian **ESP32 DEV KIT** dan **LED** \_

Pada LED **kaki panjang** adalah **Anode (+)** dan **kaki pendek** adalah **Cathode (-)**.

Rangkaiannya:

1. Anode (**kaki panjang LED**) ke **Pin 22** ESP32 DEV KIT.
2. Cathode (**kaki pendek LED**) ke **Terminal GND** melalui **Resistor 220 Ohm**.



### 3.2.11. Coding Basic ESP32 LED - Digital Output \_

Untuk yang menggunakan **Board ESP32**, baik **Board ESP32 DOIT DEV KIT V1** atau Board ESP32 lainnya.

Silakan gunakan coding disamping, untuk **menyalakan LED selama 3 detik**, kemudian LED akan mati setelahnya.

```
const byte pinLed = 22;

void setup() {
  pinMode(pinLed, OUTPUT);
  digitalWrite(pinLed, HIGH);
  delay(3000);
  digitalWrite(pinLed, LOW);
}

void loop() {}
```

## 3.2.12. Coding ESP32 Blink LED - Digital Output \_

Berikut kode untuk membuat **LED berkedip** (blink) pada board **ESP32**.

Kode ini akan **menyalakan LED selama 1 detik, mematikannya selama 1 detik, dan mengulangi** proses tersebut **terus-menerus**.

```
const byte pinLed = 22;

void setup() {
  pinMode(pinLed, OUTPUT);
}

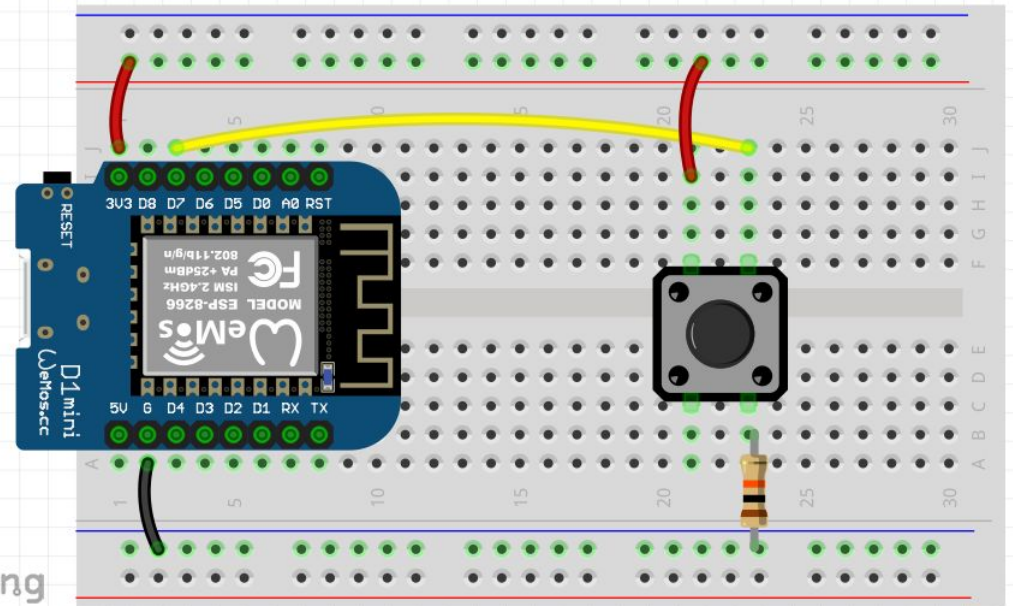
void loop() {
  digitalWrite(pinLed, HIGH);
  delay(1000);
  digitalWrite(pinLed, LOW);
  delay(1000);
}
```

# **DIGITAL INPUT PUSH BUTTON - VERSI BOARD ESP8266**

### 3.2.13. Rangkaian Wemos D1 Mini dan Button

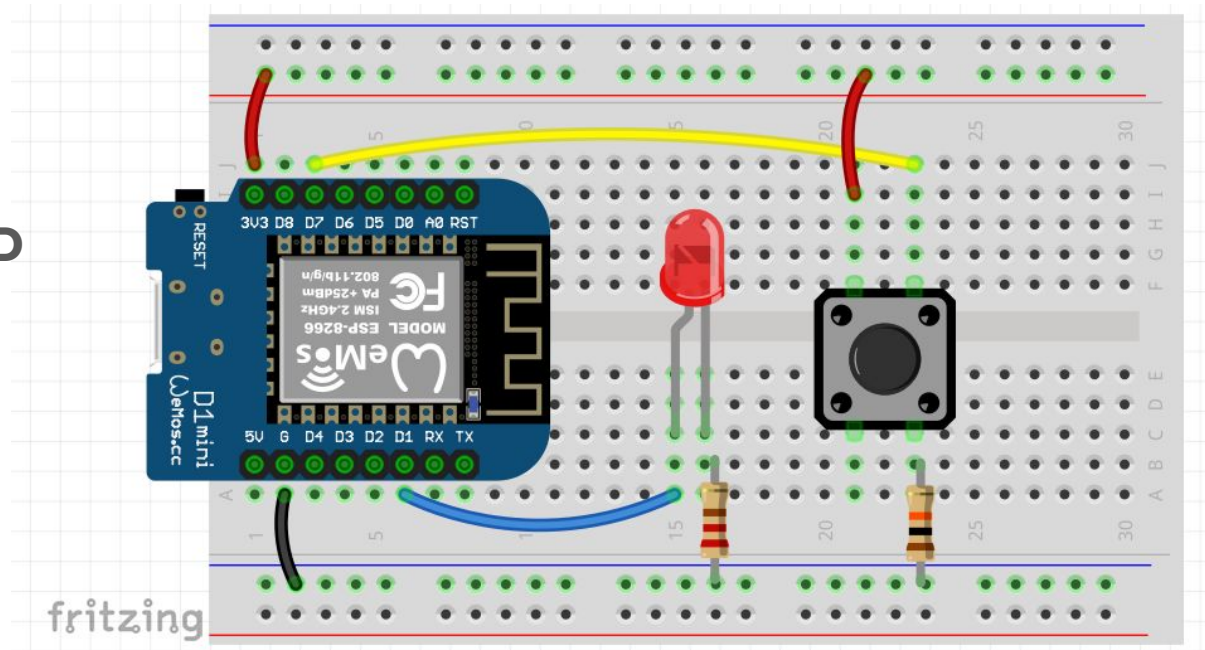
Kita akan membuat rangkaian **Wemos D1 Mini** dan **Push Button** dengan metode **Pull Down**.

Silakan rangkai sesuai gambar disamping, dimana **salah satu kaki Push Button** dipasang ke **3.3 Volt**, seberangnya dipasang ke **GND** melalui **Resistor 10K Ohm**, dan terakhir **signal Output** dipasang ke **Pin D7**.



### 3.2.14. Rangkaian Wemos D1 Mini dan Push Button dan juga LED

Dan disamping adalah gambar rangkaian Wemos D1 Mini dan LED ketika sudah dipasang dengan Push Button juga.

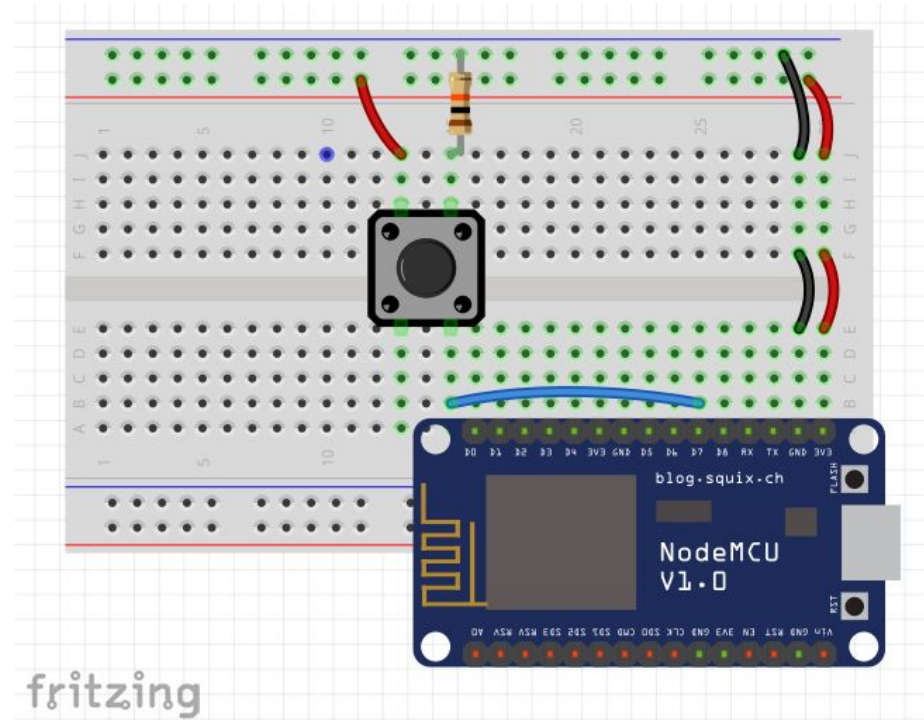




### 3.2.15. Rangkaian NodeMCU Amica dan Button

Kita akan membuat rangkaian **NodeMCU Amica** dan **Push Button** dengan metode **Pull Down**.

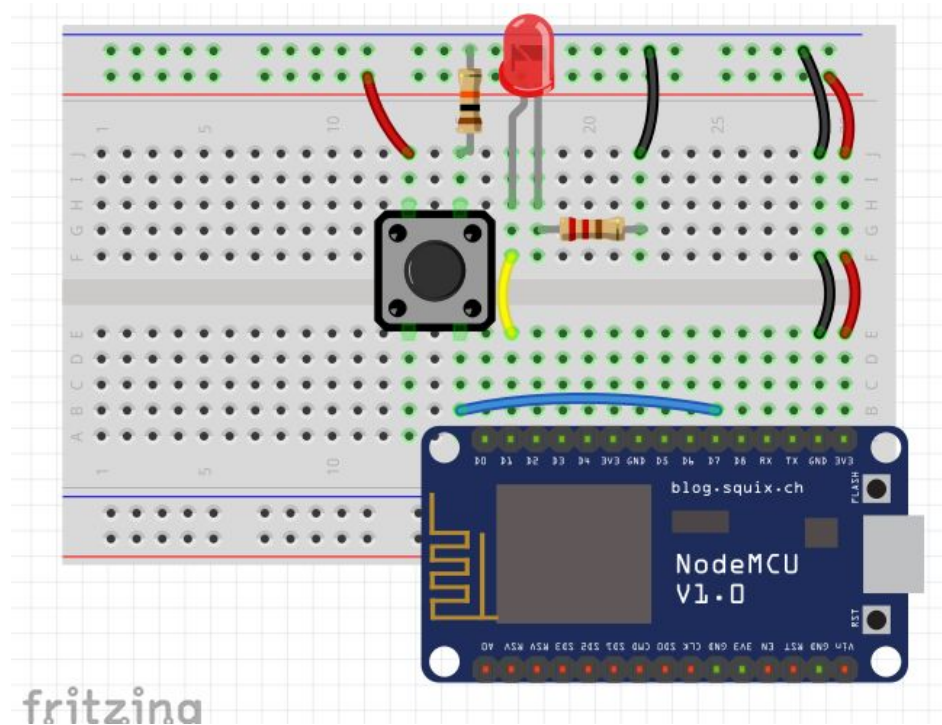
Silakan rangkai sesuai gambar disamping, dimana **salah satu kaki Push Button** dipasang ke **3.3 Volt**, seberangnya dipasang ke **GND** melalui **Resistor 10K Ohm**, dan terakhir **signal Output** dipasang ke **Pin D7**.





### 3.2.16. Rangkaian **NodeMCU Amica** dan Push **Button** dan juga **LED**

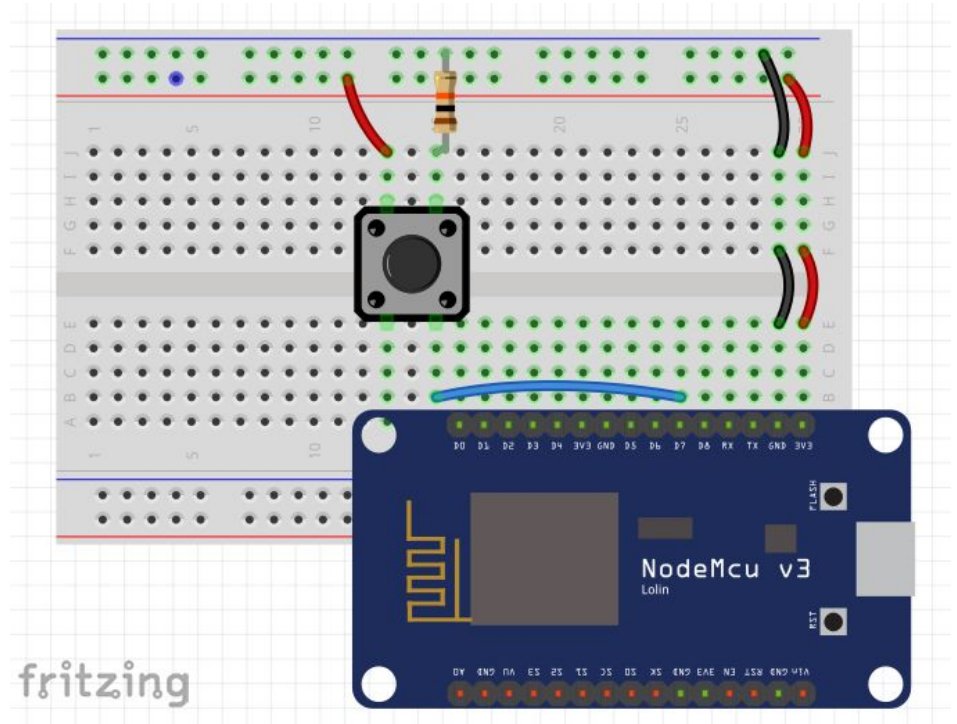
Dan disamping adalah gambar rangkaian **NodeMCU Amica** dan **LED** ketika sudah dipasang dengan **Push Button** juga.



### 3.2.17. Rangkaian NodeMCU Lolin dan Button

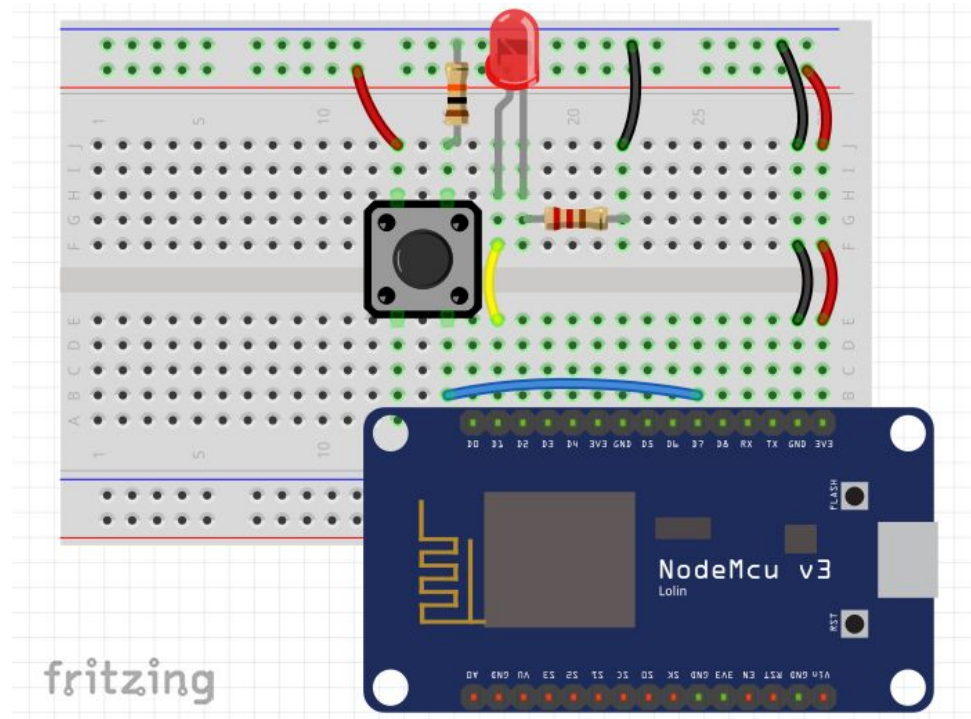
Kita akan membuat rangkaian **NodeMCU Lolin** dan **Push Button** dengan metode **Pull Down**.

Silakan rangkai sesuai gambar disamping, dimana **salah satu kaki Push Button** dipasang ke **3.3 Volt**, seberangnya dipasang ke **GND** melalui **Resistor 10K Ohm**, dan terakhir **signal Output** dipasang ke **Pin D7**.



### 3.2.18. Rangkaian **NodeMCU** **Lolin** dan **Push Button** dan juga **LED**

Dan disamping adalah gambar rangkaian **NodeMCU** **Lolin** dan **LED** ketika sudah dipasang dengan **Push Button** juga.



## 3.2.19. Coding ESP8266 Button - Digital Input \_

Ini adalah coding sederhana untuk melihat output pada **Serial Monitor** berupa tulisan “**Push Button Ditekan!**”, ketika button ditekan.

Silakan atur Baud Rate pada **Serial Monitor** sesuai dengan coding yaitu **115200**.

```
const byte pinButton = D7;

void setup() {
  Serial.begin(115200);
  pinMode(pinButton, INPUT);
}

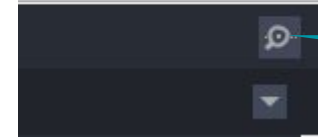
void loop() {
  if (digitalRead(pinButton)) {
    Serial.println("Push Button Ditekan!");
    delay(300);
  }
}
```

Untuk membuat Serial Monitor bisa klik **Icon Kaca Pembesar** yang ada di **Pojok Kanan Atas Arduino IDE**.

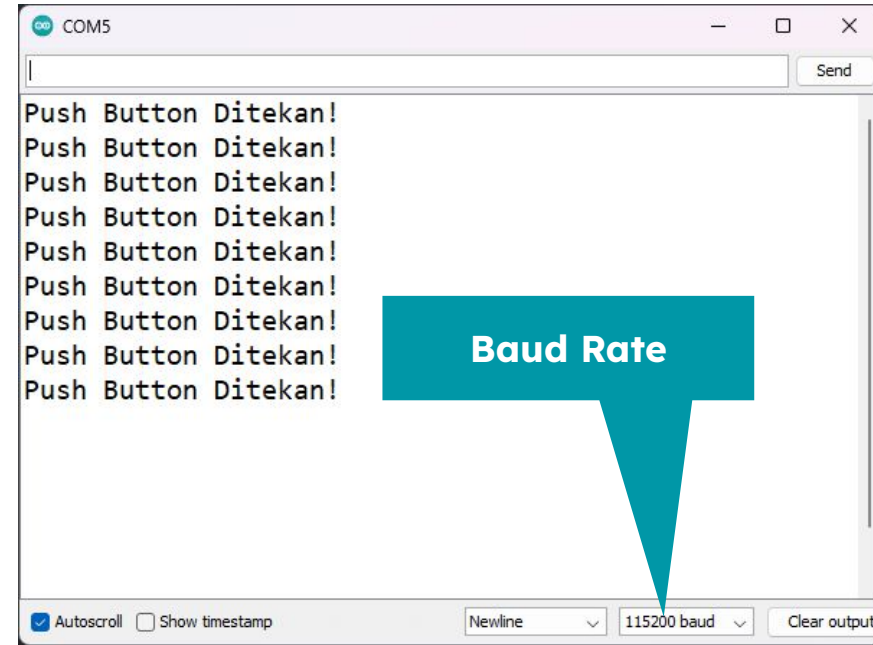
Atau bisa juga dengan klik **menu Tools**, kemudian pilih **Serial Monitor**.

Atau bisa juga dengan menekan kombinasi **CTRL+SHIFT+M** di Keyboard.

Jangan lupa **Baud Rate** ubah dulu ke **115200** ya sesuai dengan coding.



**Serial Monitor**



**Baud Rate**

## 3.2.20. Coding ESP8266 Button ON OFF LED\_

Sebuah coding sederhana untuk **menyalakan LED ketika Push Button ditekan**, dan **mematikan LED ketika Push Button dilepas**.

```
const byte pinButton = D7;
const byte pinLed = D1;

void setup() {
  Serial.begin(115200);
  pinMode(pinButton, INPUT);
  pinMode(pinLed, OUTPUT);
}

void loop() {
  if (digitalRead(pinButton)) {
    Serial.println("Push Button Ditekan!");
    digitalWrite(pinLed, HIGH);
    delay(300);
  } else {
    digitalWrite(pinLed, LOW);
  }
}
```

# Mari Buat Coding Yang Lebih Menantang! \_

Sebagai bahan latihan Anda, dengan rangkaian yang sama silakan buat coding dengan detail berikut:

1. LED Mati ketika Tombol ditekan, dan LED Nyala ketika tombol dilepas.
2. LED Berkedip ketika Tombol ditekan, dan LED Nyala biasa ketika tombol dilepas.

### 3.2.21. Coding ESP8266 Button Switch **ON OFF** **LED**

Led ON ketika ditekan sekali, kemudian OFF ketika ditekan kedua kali, kemudian ON lagi ketika ditekan lagi, OFF lagi ketika ditekan lagi, dan seterusnya.

```
const byte pinButton = D7;
const byte pinLed = D1;
boolean statusLed = false;

void setup() {
  Serial.begin(115200);
  pinMode(pinButton, INPUT);
  pinMode(pinLed, OUTPUT);
}

void loop() {
  if (digitalRead(pinButton) == 1) {
    statusLed = !statusLed;
    Serial.print("Status LED: ");
    Serial.println(statusLed);
    delay(250);
  }
  digitalWrite(pinLed, statusLed);
}
```

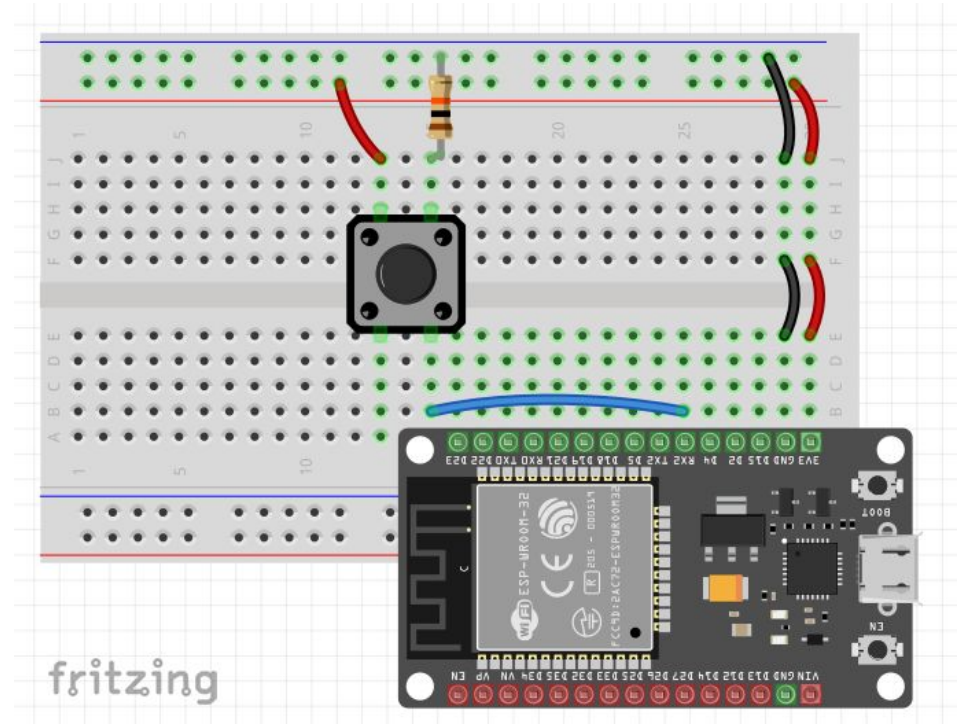


# **DIGITAL INPUT PUSH BUTTON - VERSI BOARD ESP32**

### 3.2.22. Rangkaian **ESP32 DEV KIT** dan **Button**

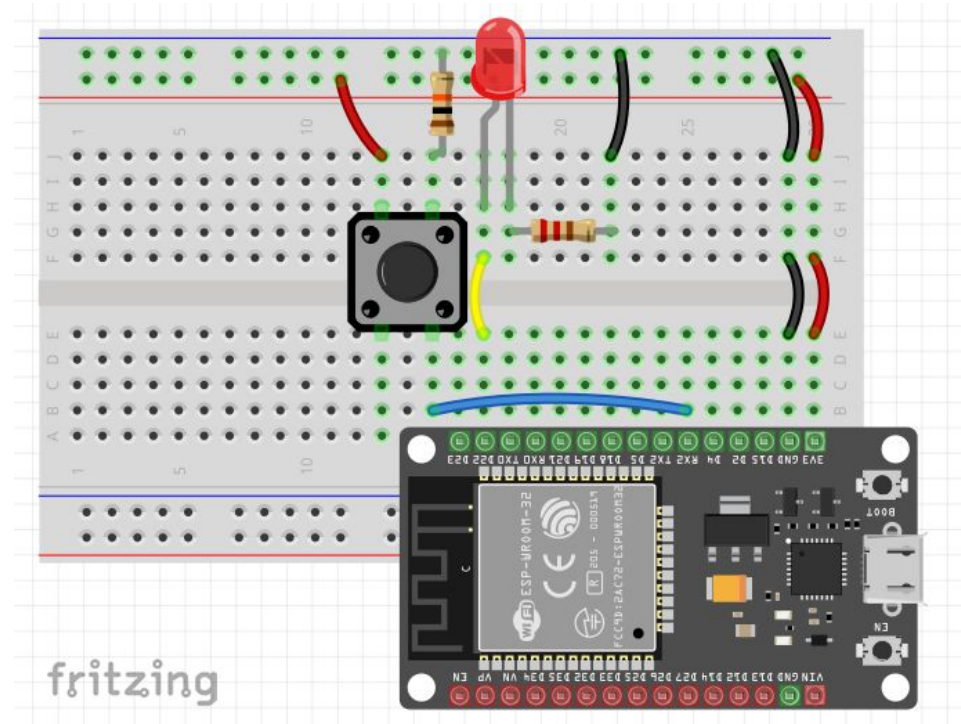
Kita akan membuat rangkaian **ESP32 DEV KIT** dan **Push Button** dengan metode **Pull Down**.

Silakan rangkai sesuai gambar disamping, dimana **salah satu kaki Push Button** dipasang ke **3.3 Volt**, seberangnya dipasang ke **GND** melalui **Resistor 10K Ohm**, dan terakhir **signal Output** dipasang ke **Pin D7**.



### 3.2.23. Rangkaian **ESP32 DEV KIT** dan Push **Button** dan juga **LED**

Dan disamping adalah gambar rangkaian **ESP32 DEV KIT** dan **LED** ketika sudah dipasang dengan **Push Button** juga.



## 3.2.24. Coding ESP8266 Button - Digital Input \_

Ini adalah coding sederhana untuk melihat output pada **Serial Monitor** berupa tulisan “**Push Button Ditekan!**”, ketika button ditekan.

Silakan atur Baud Rate pada **Serial Monitor** sesuai dengan coding yaitu **115200**.

```
const byte pinButton = 16;

void setup() {
  Serial.begin(115200);
  pinMode(pinButton, INPUT);
}

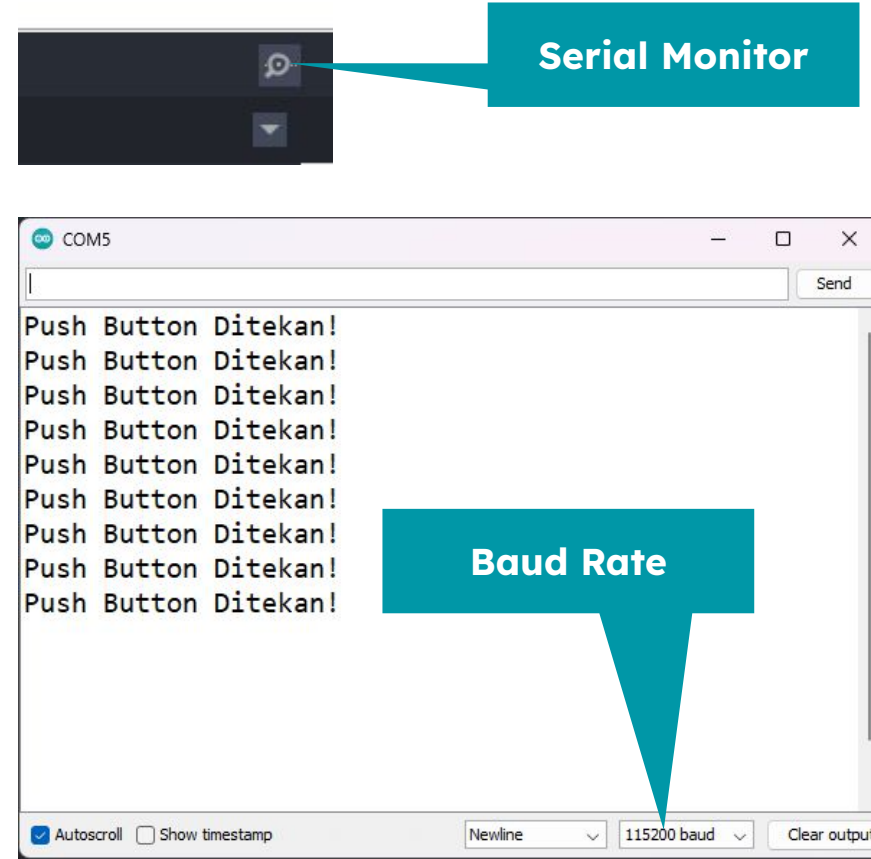
void loop() {
  if (digitalRead(pinButton)) {
    Serial.println("Push Button Ditekan!");
    delay(300);
  }
}
```

Untuk membuat Serial Monitor bisa klik **Icon Kaca Pembesar** yang ada di **Pojok Kanan Atas Arduino IDE**.

Atau bisa juga dengan klik **menu Tools**, kemudian pilih **Serial Monitor**.

Atau bisa juga dengan menekan kombinasi **CTRL+SHIFT+M** di Keyboard.

Jangan lupa **Baud Rate** ubah dulu ke **115200** ya sesuai dengan coding.



### 3.2.25. Coding

## ESP32 Button ON OFF LED\_

Sebuah coding sederhana untuk **menyalakan LED ketika Push Button ditekan, dan mematikan LED ketika Push Button dilepas.**

```
const byte pinButton = 16;
const byte pinLed = 22;

void setup() {
  Serial.begin(115200);
  pinMode(pinButton, INPUT);
  pinMode(pinLed, OUTPUT);
}

void loop() {
  if (digitalRead(pinButton)) {
    Serial.println("Push Button Ditekan!");
    digitalWrite(pinLed, HIGH);
    delay(300);
  } else {
    digitalWrite(pinLed, LOW);
  }
}
```

# Mari Buat Coding Yang Lebih Menantang! \_

Sebagai bahan latihan Anda, dengan rangkaian yang sama silakan buat coding dengan detail berikut:

1. LED Mati ketika Tombol ditekan, dan LED Nyala ketika tombol dilepas.
2. LED Berkedip ketika Tombol ditekan, dan LED Nyala biasa ketika tombol dilepas.

## 3.2.26. Coding ESP32 Button Switch ON OFF LED

Led ON ketika ditekan sekali, kemudian OFF ketika ditekan kedua kali, kemudian ON lagi ketika ditekan lagi, OFF lagi ketika ditekan lagi, dan seterusnya.

```
const byte pinButton = 16;
const byte pinLed = 22;
boolean statusLed = false;

void setup() {
  Serial.begin(115200);
  pinMode(pinButton, INPUT);
  pinMode(pinLed, OUTPUT);
}

void loop() {
  if (digitalRead(pinButton) == 1) {
    statusLed = !statusLed;
    Serial.print("Status LED: ");
    Serial.println(statusLed);
    delay(250);
  }
  digitalWrite(pinLed, statusLed);
}
```



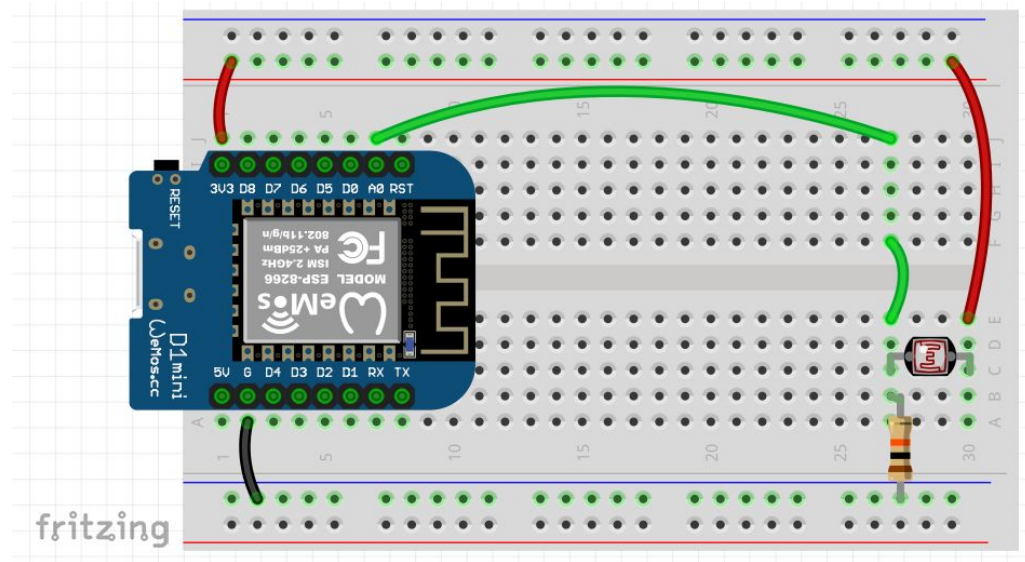
## 3.3. BASIC PROGRAM ANALOG INPUT DAN OUTPUT \_

# **ANALOG INPUT OUTPUT LDR & LED - VERSI BOARD ESP8266**

### 3.3.1. Rangkaian Wemos D1 Mini dan LDR \_

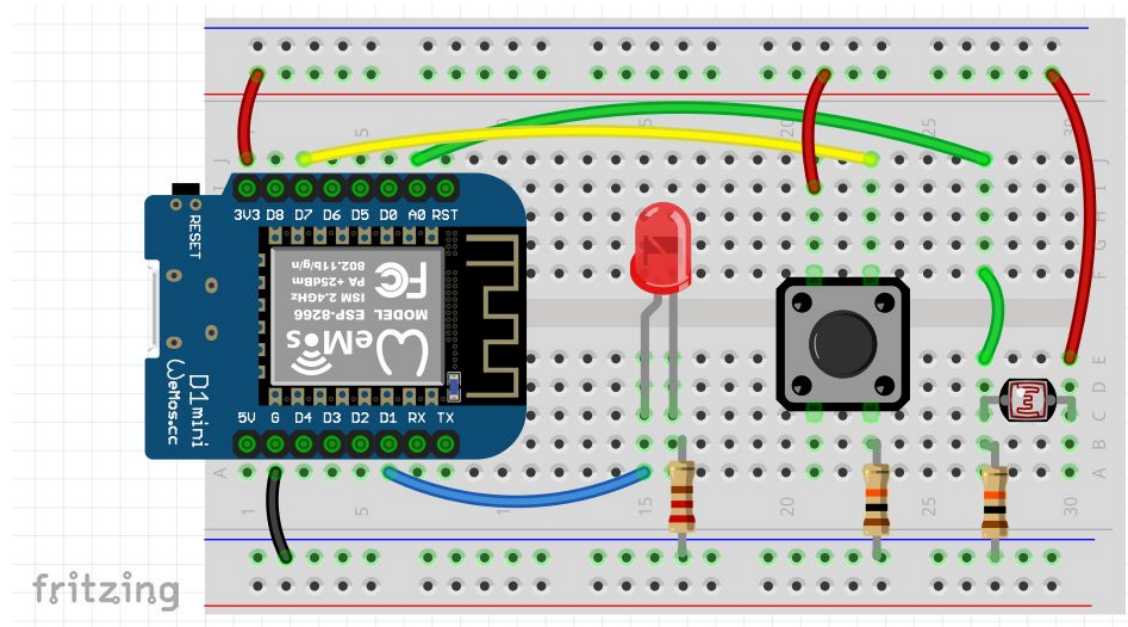
Kita akan membuat rangkaian **Wemos D1 Mini** dan **LDR**.

1. **Kaki Kanan LDR** pasang ke **terminal 3.3 Volt**.
2. **Kaki Kiri LDR** pasang ke **terminal GND** melalui **Resistor 10K ohm**.
3. **Kaki Kiri LDR** pasang ke **Pin A0** Wemos D1 Mini.



### 3.3.2. Rangkaian Wemos D1 Mini dan Push Button dan juga LED dan juga LDR

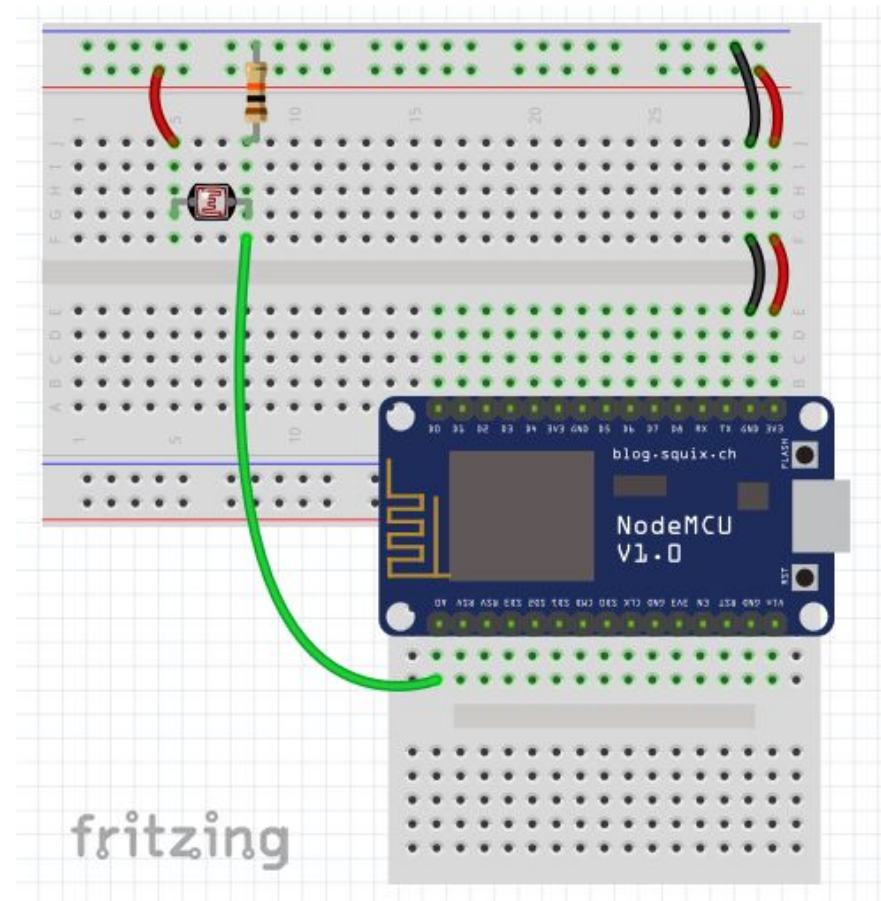
Dan disamping adalah gambar rangkaian Wemos D1 Mini dan LED dan juga Push Button ketika sudah dipasang dengan LDR juga.



### 3.3.3. Rangkaian NodeMCU Amica dan LDR

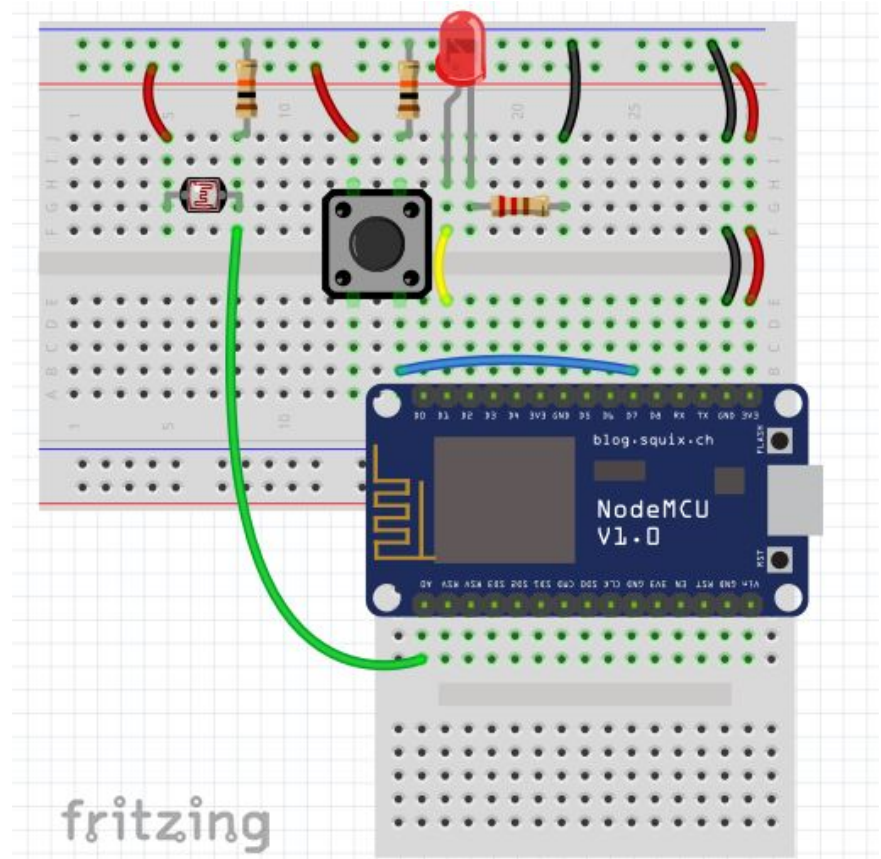
Kita akan membuat rangkaian NodeMCU Amica dan LDR.

1. Kaki Kanan LDR pasang ke terminal 3.3 Volt.
2. Kaki Kiri LDR pasang ke terminal GND melalui Resistor 10K ohm.
3. Kaki Kiri LDR pasang ke Pin A0 NodeMCU Amica.



### 3.3.4. Rangkaian NodeMCU Amica dan Push Button dan juga LED dan juga LDR

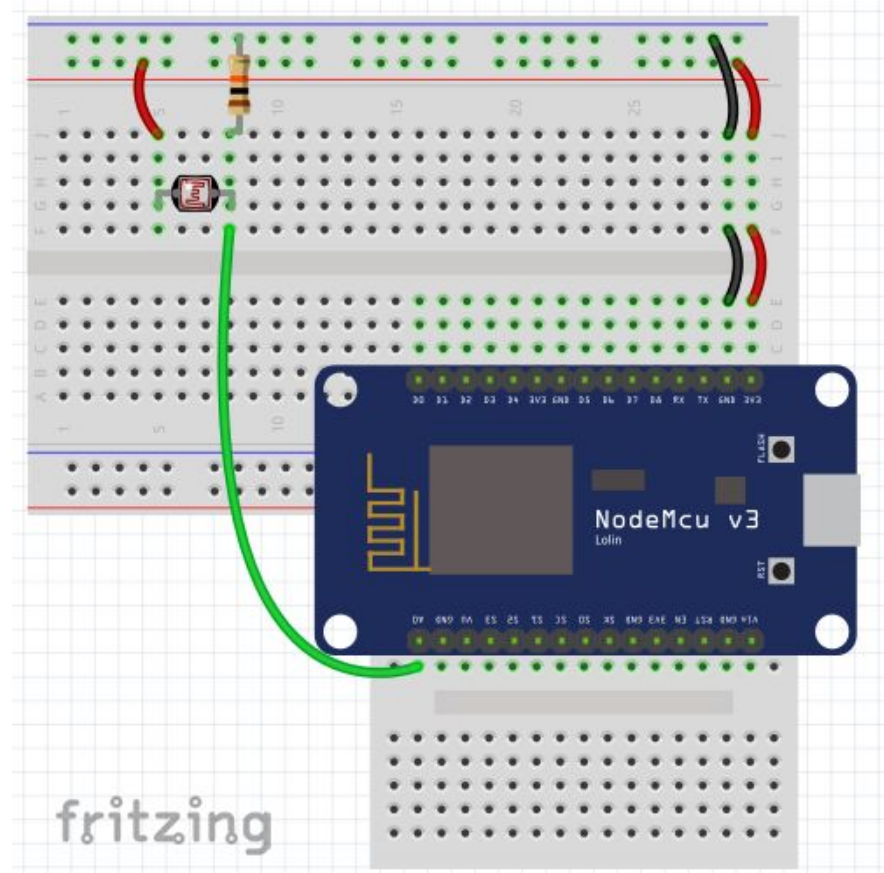
Dan disamping adalah gambar rangkaian NodeMCU Amica dan LED dan juga Push Button ketika sudah dipasang dengan LDR juga.



### 3.3.5. Rangkaian NodeMCU Lolin dan LDR

Kita akan membuat rangkaian  
NodeMCU Lolin dan LDR.

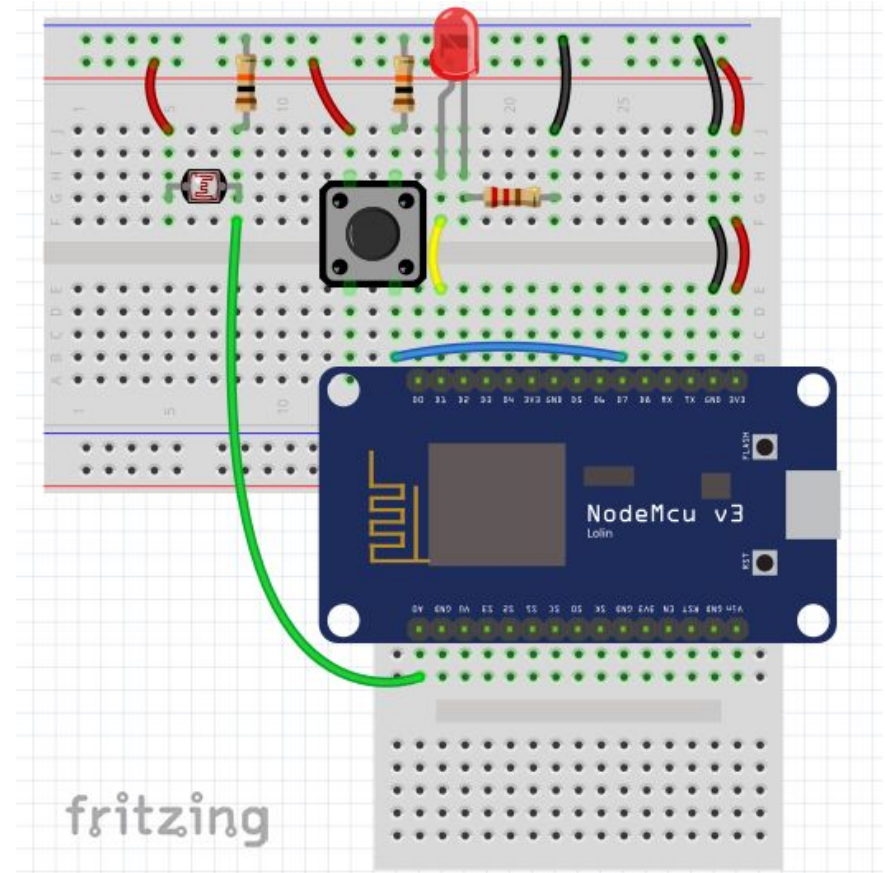
1. **Kaki Kanan LDR** pasang ke **terminal 3.3 Volt**.
2. **Kaki Kiri LDR** pasang ke **terminal GND** melalui **Resistor 10K ohm**.
3. **Kaki Kiri LDR** pasang ke **Pin A0** NodeMCU Lolin.





### 3.3.6. Rangkaian **NodeMCU Lolin** dan Push **Button** dan juga **LED** dan juga **LDR** —

Dan disamping adalah gambar rangkaian **NodeMCU Lolin** dan **LED** dan juga **Push Button** ketika sudah dipasang dengan **LDR** juga.





### 3.3.7. Coding ESP8266 LDR - Analog Input \_

Ini adalah coding sederhana untuk melihat output pada **Serial Monitor** berupa nilai data yang ditampilkan Sensor LDR.

Silakan atur Baud Rate pada **Serial Monitor** sesuai dengan coding yaitu **115200**.

```
const byte pinLdr = A0;
int ldrValue = 0;

void setup() {
  Serial.begin(115200);
}

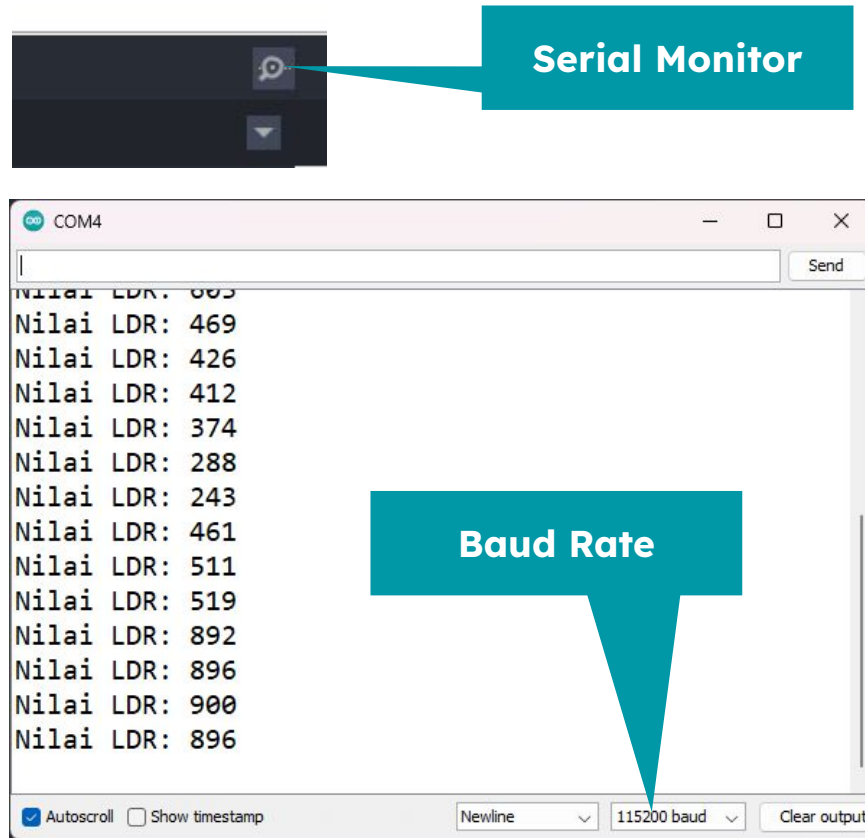
void loop() {
  ldrValue = analogRead(pinLdr);
  Serial.print("Nilai LDR: ");
  Serial.println(ldrValue);
  delay(500);
}
```

Untuk membuat Serial Monitor bisa klik **Icon Kaca Pembesar** yang ada di **Pojok Kanan Atas Arduino IDE**.

Atau bisa juga dengan klik **menu Tools**, kemudian pilih **Serial Monitor**.

Atau bisa juga dengan menekan kombinasi **CTRL+SHIFT+M** di Keyboard.

Jangan lupa **Baud Rate** ubah dulu ke **115200** ya sesuai dengan coding.



### 3.3.7. Coding ESP8266 LED - Analog Output

Lebih tepatnya coding  
**PWM (Pulse Width  
Modulation)**, untuk set  
kecerahan LED mulai dari  
**0 (mati)** sampai **255**  
**(paling terang)**.

```
const int pinLed = D1;
int pwmValue = 0;
int step = 1;

void setup() {
  pinMode(pinLed, OUTPUT);
}

void loop() {
  analogWrite(pinLed, pwmValue);
  delay(10);
  pwmValue += step;
  if (pwmValue == 0 || pwmValue == 255) {
    step = -step;
  }
}
```

### 3.3.8. Coding

## ESP8266 Otomasi LED Sensor LDR

Sebuah coding yang akan membuat LED menyala otomatis berdasarkan perubahan cahaya disekitar.

\*Silahkan sesuaikan nilai **900** dan **100** sesuai pembacaan sensor Anda.

```
const byte pinLed = D1;
const byte pinLdr = A0;
int ldrValue, pwmValue = 0;

void setup() {
  Serial.begin(115200);
}

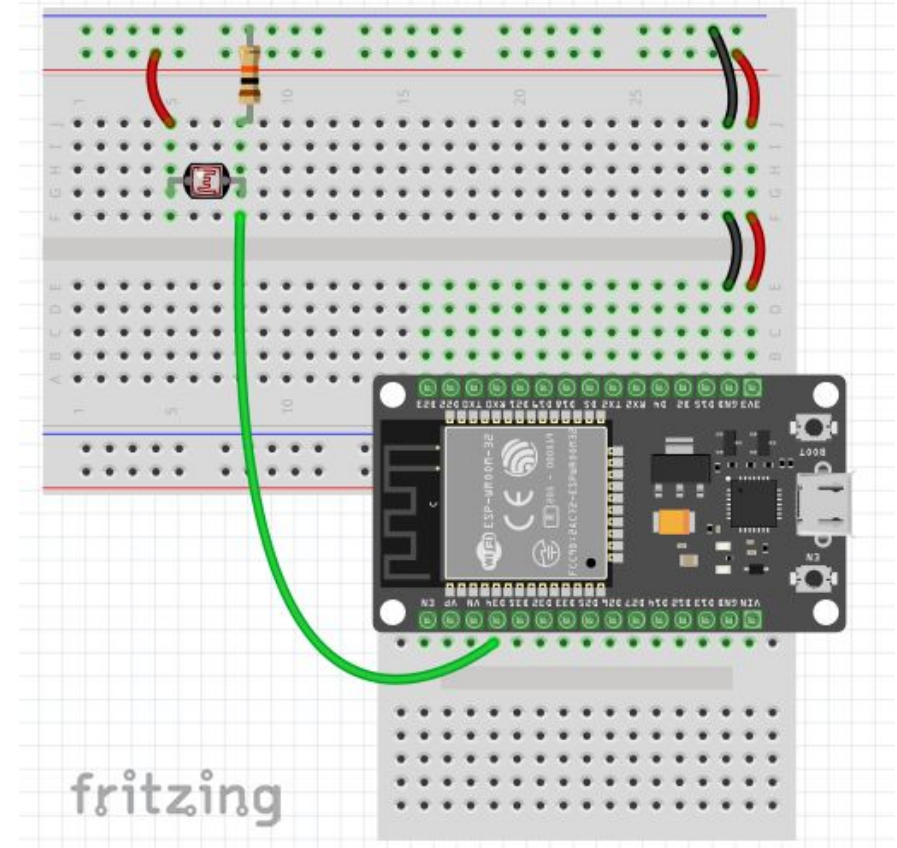
void loop() {
  ldrValue = analogRead(pinLdr);
  pwmValue = map(ldrValue, 900, 100, 0, 255);
  Serial.print("Nilai PWM: ");
  Serial.println(pwmValue);
  analogWrite(pinLed, pwmValue);
  delay(500);
}
```

# **ANALOG INPUT OUTPUT LDR & LED - VERSI BOARD ESP32**

### 3.3.9. Rangkaian **ESP32 DEV KIT** dan **LDR**

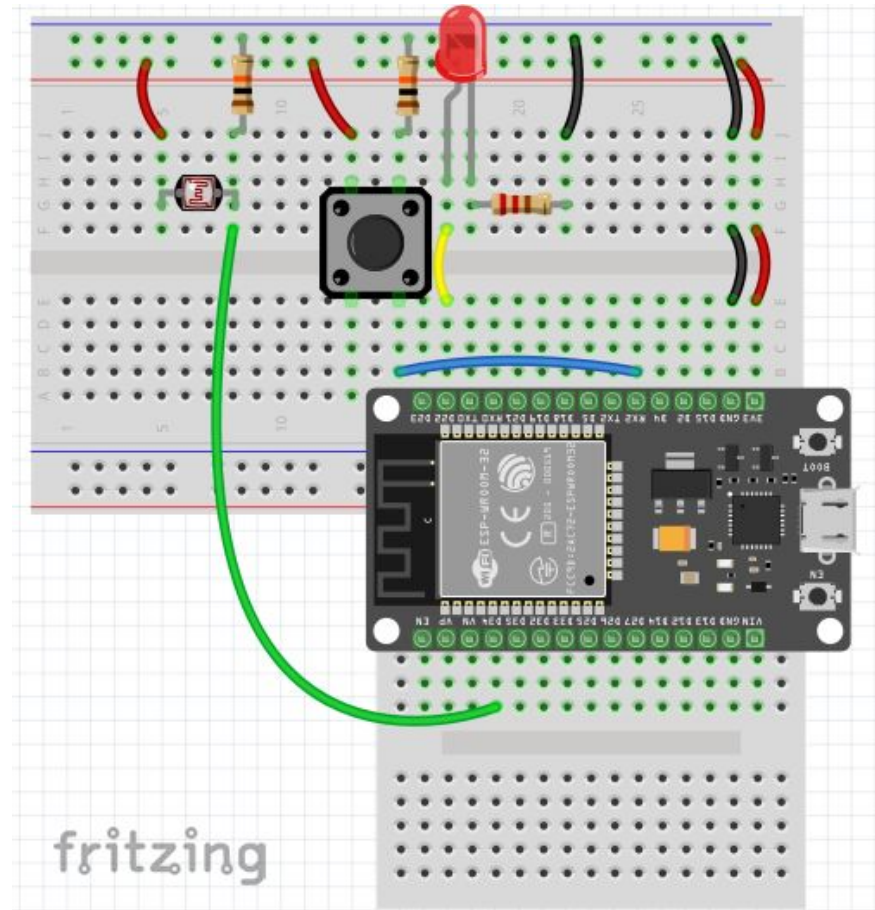
Kita akan membuat rangkaian **ESP32 DEV KIT** dan **LDR**.

1. **Kaki Kanan LDR** pasang ke **terminal 3.3 Volt**.
2. **Kaki Kiri LDR** pasang ke **terminal GND** melalui **Resistor 10K ohm**.
3. **Kaki Kiri LDR** pasang ke **Pin 34** NodeMCU Lolin.



### 3.3.10. Rangkaian **ESP32 DEV KIT** dan Push **Button** dan juga **LED** dan juga **LDR** \_

Dan disamping adalah gambar rangkaian **ESP32 DEV KIT** dan **LED** dan juga **Push Button** ketika sudah dipasang dengan **LDR** juga.



### 3.3.11. Coding ESP32 LDR - Analog Input \_

Ini adalah coding sederhana untuk melihat output pada **Serial Monitor** berupa nilai data yang ditampilkan Sensor LDR.

Silakan atur Baud Rate pada **Serial Monitor** sesuai dengan coding yaitu **115200**.

```
const byte pinLdr = 34;
int ldrValue = 0;

void setup() {
  Serial.begin(115200);
}

void loop() {
  ldrValue = analogRead(pinLdr);
  Serial.print("Nilai LDR: ");
  Serial.println(ldrValue);
  delay(500);
}
```

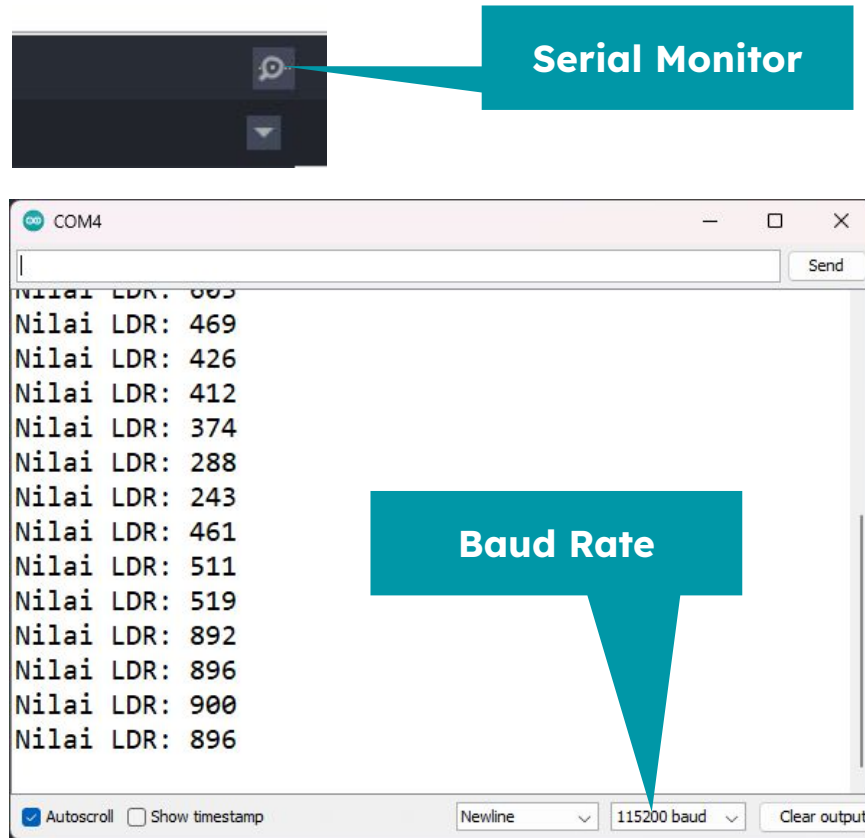


Untuk membuat Serial Monitor bisa klik **Icon Kaca Pembesar** yang ada di **Pojok Kanan Atas Arduino IDE**.

Atau bisa juga dengan klik **menu Tools**, kemudian pilih **Serial Monitor**.

Atau bisa juga dengan menekan kombinasi **CTRL+SHIFT+M** di Keyboard.

Jangan lupa **Baud Rate** ubah dulu ke **115200** ya sesuai dengan coding.



### 3.3.12. Coding ESP32 LED - Analog Output \_

Lebih tepatnya coding  
**PWM (Pulse Width  
Modulation)**, untuk set  
kecerahan LED mulai dari  
**0 (mati)** sampai **255**  
**(paling terang)**.

```
const int pinLed = 34;
int pwmValue = 0;
int step = 1;

void setup() {
  pinMode(pinLed, OUTPUT);
}

void loop() {
  analogWrite(pinLed, pwmValue);
  delay(10);
  pwmValue += step;
  if (pwmValue == 0 || pwmValue == 255) {
    step = -step;
  }
}
```

### 3.3.13. Coding ESP32 Otomasi LED Sensor LDR

Sebuah coding yang akan membuat LED menyala otomatis berdasarkan perubahan cahaya disekitar.

\*Silahkan sesuaikan nilai **900** dan **100** sesuai pembacaan sensor Anda.

```
const byte pinLed = 22;
const byte pinLdr = 34;
int ldrValue, pwmValue = 0;

void setup() {
  Serial.begin(115200);
}

void loop() {
  ldrValue = analogRead(pinLdr);
  pwmValue = map(ldrValue, 900, 100, 0, 255);
  Serial.print("Nilai PWM: ");
  Serial.println(pwmValue);
  analogWrite(pinLed, pwmValue);
  delay(500);
}
```

## 3.4. BASIC PROGRAM WIFI DAN PROTOKOL MQTT \_

### 3.4.1. Menghubungkan **ESP** ke **MQTT Broker** \_

Untuk bisa terhubung ke MQTT Broker, harus **menambahkan dulu 2 Library** ke Arduino IDE yaitu: **MQTT.h** dan **MQTTESP.h**, yang bisa di download di:

1. MQTT.h:  
<https://github.com/256dpi/arduino-mqtt/archive/refs/heads/master.zip>
2. MQTTESP.h:  
<https://github.com/kelasrobot/mqttesp/archive/refs/heads/main.zip>

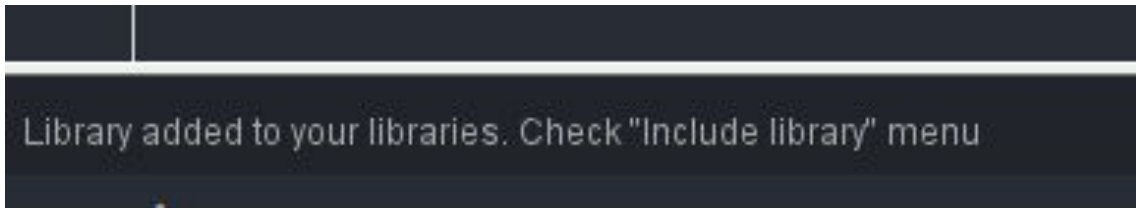
\*File .zip dari kedua Library tersebut tidak perlu di ekstrak ya, biarkan saja dalam kondisi masih bentuk .zip

Untuk menambahkan Library silakan ikuti langkah-langkah berikut:

1. Pada **Arduino IDE** klik menu **Sketch**
2. Pilih **Include Library**
3. Pilih **Add .ZIP Library...**
4. Lalu cari file Librarynya, pilih, kemudian klik **Open**

Lakukan satu persatu untuk kedua Library tersebut.

Jika berhasil akan muncul keterangan seperti gambar berikut:



Silakan isi ssid dengan **nama wifi** yang tersedia, dan juga password dengan **password wifi** yang tersedia.

Untuk broker kita coba menggunakan broker public punya **EMQX**.

Kemudian buka **Serial Monitor**.

```
#include <MQTTESP.h>
const char* ssid = "isi nama wifi";
const char* password = "isi password wifi";
const char* mqtt_server = "broker.emqx.io";
MQTTESP mqtt(ssid, password, mqtt_server);

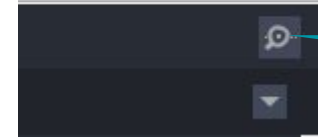
void setup() {
    Serial.begin(115200);
    mqtt.begin();
}

void loop() {}
```

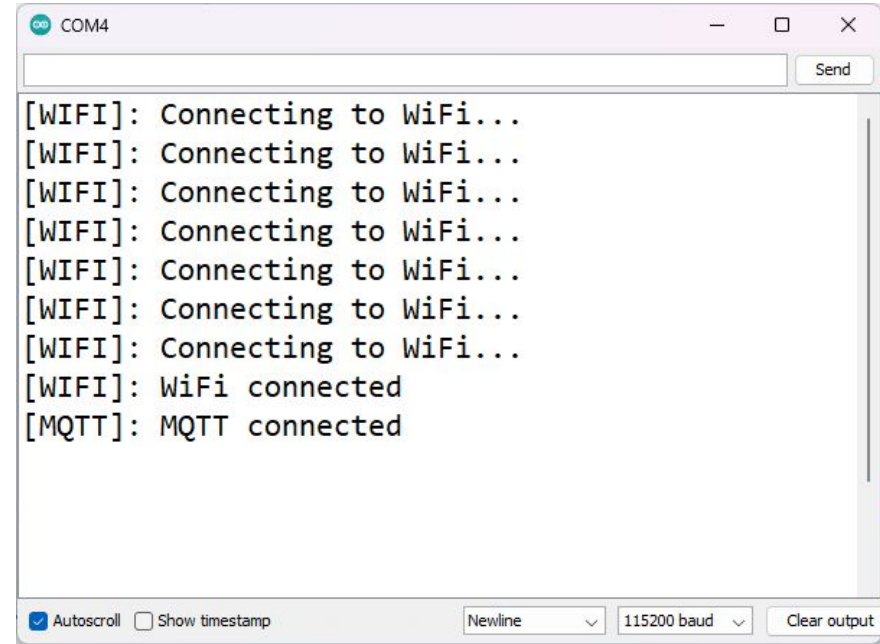
Untuk membuat Serial Monitor bisa klik **Icon Kaca Pembesar** yang ada di **Pojok Kanan Atas Arduino IDE**.

Jika berhasil akan muncul keterangan **WiFi connected** dan **MQTT connected** seperti gambar disamping.

Jika tidak muncul apapun, bisa tekan tombol **RESET / RST** pada Wemos NodeMCU, dan tombol **EN** jika pada ESP32.



**Serial Monitor**





### 3.4.2. Komunikasi 2 Arah **Publish** dan **Subscribe** \_

```
#include <MQTTESP.h>
const char* ssid = "isi nama wifi";
const char* password = "isi password wifi";
const char* mqtt_server = "broker.emqx.io";

MQTTESP mqtt(ssid, password, mqtt_server);

const char* topicSensor = "tes/18921/topic/sensor";
const char* topicLampu = "tes/18921/topic/lampu";

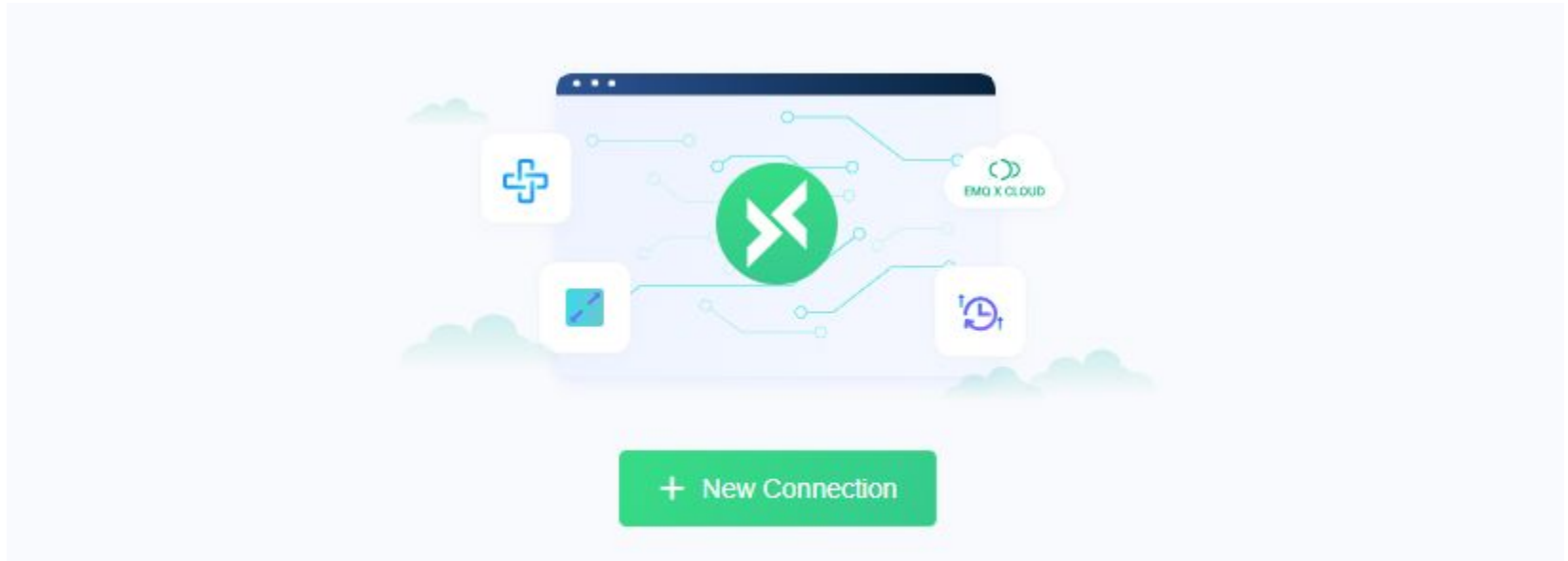
unsigned long timer = 0;
unsigned long counter = 0;
const unsigned long intervalKirim = 1000;
```

```
void setup() {  
    Serial.begin(115200);  
    pinMode(2, OUTPUT);  
    mqtt.begin();  
    mqtt.subscribe(topicLampu);  
}  
  
void loop() {  
    mqtt.loop();  
    if (millis() - timer >= intervalKirim) {  
        timer = millis();  
        mqtt.publish(topicSensor, String(++counter));  
    }  
    String pesan = mqtt.getIncomingMessage();  
    if (pesan != "") {  
        digitalWrite(2, pesan.toInt());  
        mqtt.setIncomingMessage("");  
    }  
}
```

Silakan di  
**Upload**  
dan Buka  
**Serial**  
**Monitor**

Dalam pengujian kita akan menggunakan **MQTT Client** punya **EMQX**, silakan buka: <https://mqttx.app/web-client>

Lalu klik **New Connection**.



Selanjutnya cukup **kasih nama** aja, terus klik **Connect**.  
Bagian lainnya dibiarkan dengan settingan bawaan aja.

< Back      **New**      **Connect** ✓

**General**

\* Name  ⓘ

\* Client ID  ⓘ ⌚

\* Host  ⌵


Lalu klik **New Subscription**.

Kemudian masukan **topic** sensor, pastikan sudah sesuai dan sama dengan di program.

**tes/18921/topic/sensor**

Untuk angka 18921, sebenarnya bisa dibedakan, agar tidak bentrok dengan rekan yang lain.

Lalu klik **Confirm**.

EMQX 

+ New Subscription

New Subscription

\* Topic

tes/18921/topic/sensor

Jika berhasil maka pada **Serial Monitor** akan muncul keterangan **Publish successful**, diikuti Topic dan juga Payload data yang dikirimkan.

Kemudian di MQTT Client EMQX, juga akan muncul **data yang diterima**, berupa angka payload dan juga topiknya.

Jika ingin mencoba pengiriman dengan interval lebih cepat atau lambat silahkan sesuaikan di program saja.

```
[MQTT]: Publish successful
[MQTT]: Topic: tes/18921/topic/sensor
[MQTT]: Payload: 23
```

☐ Autoscroll ☐ Show timestamp Newline 115200 baud Clear output

Plaintext All Received Published

Topic: tes/18921/topic/sensor QoS: 0

22

2024-07-06 21:57:09:562

Topic: tes/18921/topic/sensor QoS: 0

23

2024-07-06 21:57:10:560

Plaintext QoS 0 Retain Meta

Selanjutnya untuk uji coba **publish** dari **MQTT Client EMQX** ke **ESP**, silakan masukan topic **tes/18921/topic/lampu** pada form seperti di gambar bawah ini.

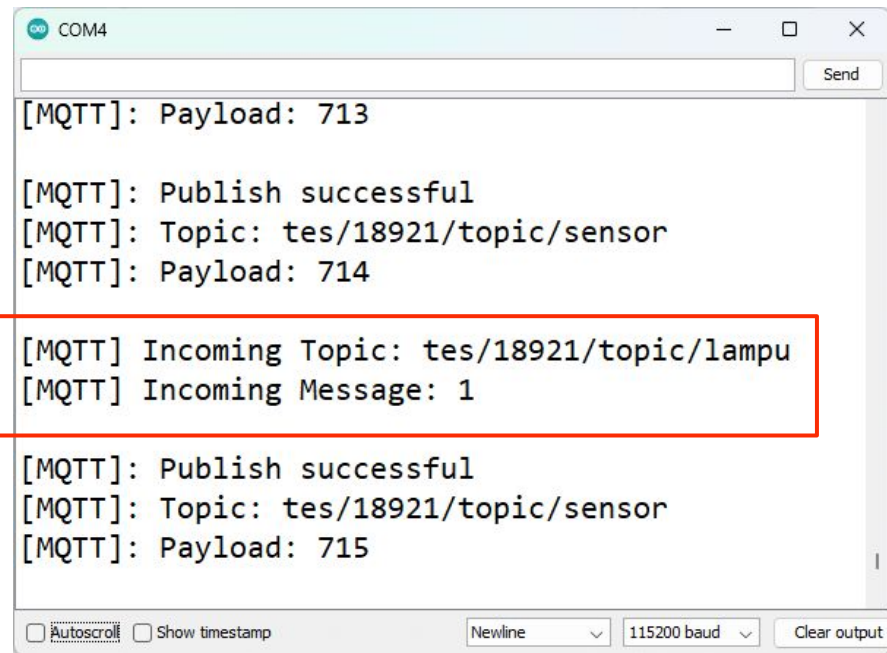
Silakan masukan angka 1 atau 0, kemudian kirimkan.

The image shows a screenshot of the MQTT Client EMQX interface. At the top, there are four buttons: 'Plaintext' (with a dropdown arrow), 'QoS 0' (with a dropdown arrow), 'Retain' (with a radio button), and 'Meta'. Below these is a large text input field for the topic, containing 'tes/18921/topic/lampu'. To the right of the topic field is a small dropdown arrow. Below the topic field is a text input field for the payload, containing the number '1'. To the right of the payload field are three circular buttons with left, center, and right arrows. At the bottom right of the form is a large green circular button with a white paper plane icon, which is the 'Send' button. Red boxes highlight the topic field, the payload field, and the 'Send' button.

Hasilnya pada serial monitor akan muncul **Incoming Topic** dan **Incoming Message**.

Kemudian **lampu bawaan ESP** akan **menyala ketika menerima pesan 0**, dan akan **mati ketika menerima perintah 1**.

Baik sampai sini bapak/ibu sudah bisa **komunikasi 2 arah antara ESP** dengan **MQTT Client versi Web** punya EMQX.



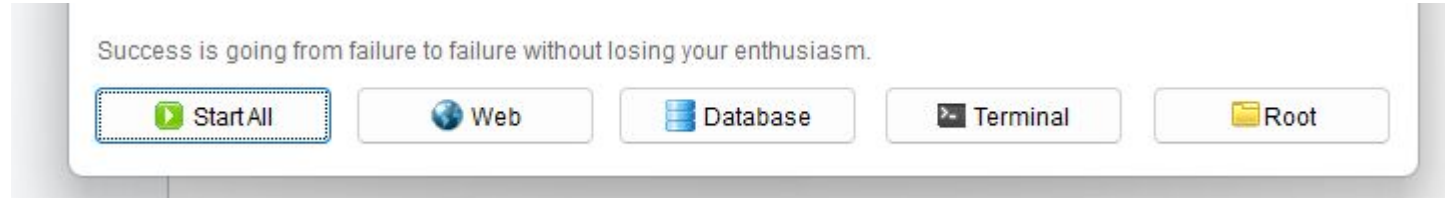
```
[MQTT]: Payload: 713
[MQTT]: Publish successful
[MQTT]: Topic: tes/18921/topic/sensor
[MQTT]: Payload: 714
[MQTT] Incoming Topic: tes/18921/topic/lampu
[MQTT] Incoming Message: 1
[MQTT]: Publish successful
[MQTT]: Topic: tes/18921/topic/sensor
[MQTT]: Payload: 715
```



## 4. Integrasi **Web** dengan **JavaScript** dan **MQTT**

## 4.1. MENJALANKAN LARAGON DAN CLONING PROJECT \_

## 4.1.1. 5 Tombol Sakti Pada Laragon



Bagian bawah Laragon ada 5 tombol sakti, yaitu:

1. **Start All:** Untuk menjalankan Web Server dan MySQL Server
2. **Web:** Untuk membuka halaman utama laragon
3. **Database:** Untuk membuka phpmyadmin
4. **Terminal:** Untuk membuka terminal (CLI) pada Laragon
5. **Root:** Untuk membuka folder project web pada Laragon

## 4.1.2. Menjalankan **Web Server** dan **MySQL** \_

Untuk menjalankan Web Server dan MySQL silakan langsung saja klik **Start All**, maka Apache dan MySQL akan run seperti dibawah ini, Apache Run di Port 80 dan MySQL di Port 3306.

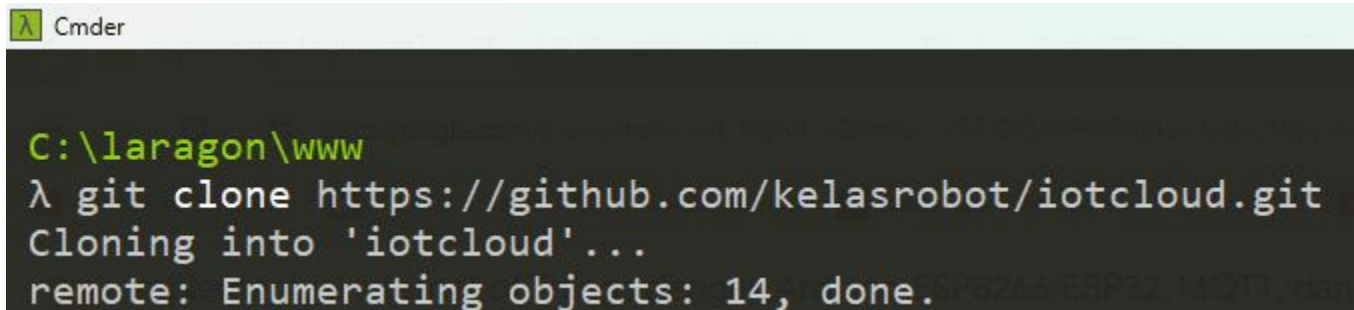


### 4.1.3. Cloning **Project IoTCloud** pada **Github**

Pada laragon buka **Terminal**, kemudian masukan perintah:

**git clone** <https://github.com/kelasrobot/iotcloud.git>

Dan tunggu sampai proses cloning selesai.



```
C:\laragon\www
λ git clone https://github.com/kelasrobot/iotcloud.git
Cloning into 'iotcloud'...
remote: Enumerating objects: 14, done.
```

## 4.1.4. Masuk dan Melihat Isi Folder Project \_

Kemudian **masuk ke folder project** dengan perintah: **cd iotcloud**

Untuk **melihat isi folder project**, bisa gunakan perintah: **ls**

Di dalam folder terdapat file **index.php**, **mqtt-library.js**, **scripts.js** dan folder **layouts**.

```
C:\laragon\www
λ cd iotcloud

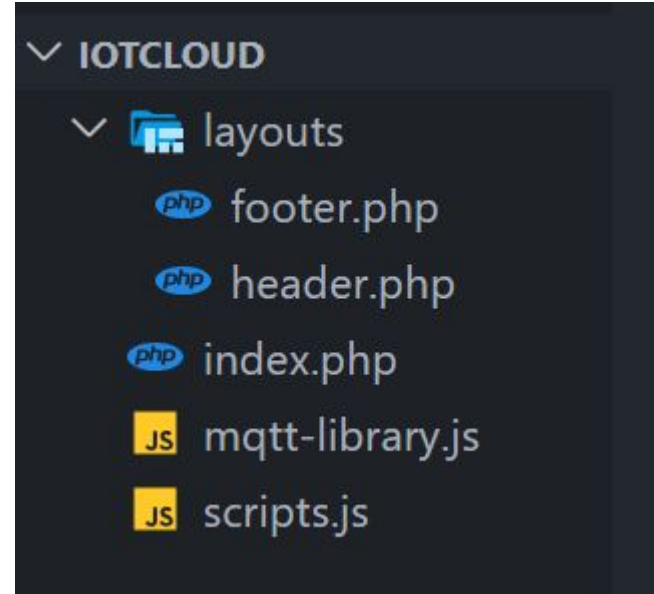
C:\laragon\www\iotcloud(main -> origin)
λ ls
index.php  layouts/  mqtt-library.js  scripts.js
```

## 4.1.5. Membuka **Folder Project** di **VSCode** \_

Untuk membuka folder project di VSCode cara paling mudah adalah dengan memasukkan perintah **code .** pada Terminal.

Tapi bisa juga dengan **klik menu File** pada VSCode, kemudian **pilih Open Folder**.

Cari di data **C**, **laragon**, **www**, dan pilih folder **iotcloud** lalu klik **select folder**.



## **4.2. MEMBUKA WEB IOTCLOUD DAN MELIHAT BAGIAN CONSOLE \_**

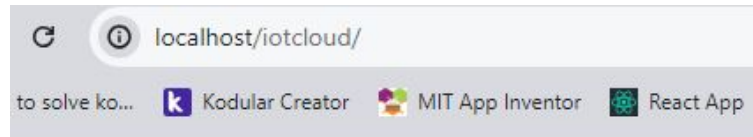


## 4.2.1. Membuka **Web IoTCloud** di **Browser** \_

Untuk membuka web iotcloud di browser, silakan buka pada alamat:

<http://localhost/iotcloud>

Jangan kaget, karena ini hanya project base, jadi hanya menampilkan judul saja.



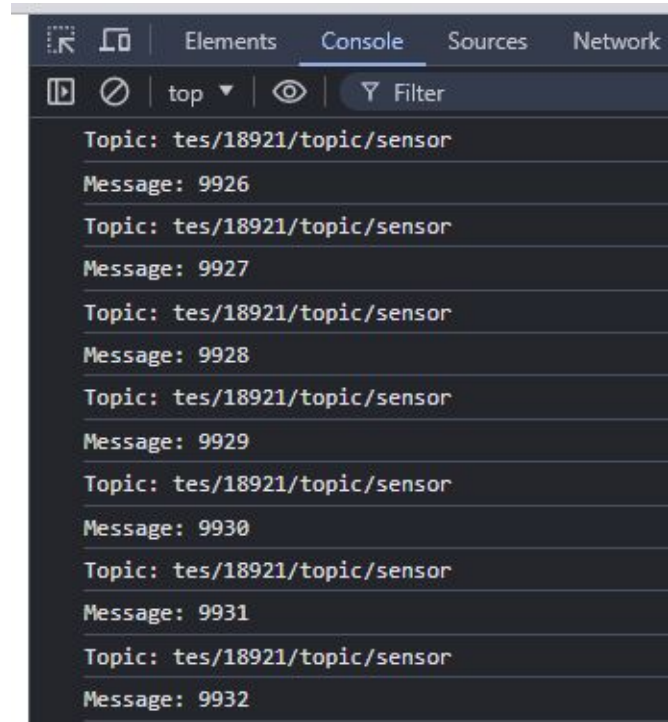
# IoT Cloud Project

## 4.2.2. Melihat **Output** Pada **Bagian Console**

Console ini fitur penting yang ada di browser, karena kita bisa melihat output yang terjadi dibelakang layar oleh Javascript.

Untuk membuka console (Google Chrome), cukup klik kanan kemudian pilih **Inspect**, kemudian pilih **Console**.

Dan tentunya bisa melihat data yang dikirimkan oleh ESP ke Web IoTCloud.



# Pengumuman \_

Mohon Maaf Sebelumnya, ebook ini terbit dengan status **[PRE-RELEASE]**, yang artinya masih ada sebagian dari isi sekitar 25% lagi belum ada di Ebook ini karena sedang di sempurnakan penulis.

Penulis usahakan akan secepatnya rilis 100% dalam bulan ini.

Sambil nunggu silahkan praktekan 75% praktikum yang ada dalam ebook ini, dan rekan-rekan **WAJIB MASUK GRUP TELEGRAM:**

**<https://t.me/+NsCxP9yn9aU1YzQ1>**

Karena Update Mengenai Ebook ini akan di Share Disana.