



ELETRÔNICA **ÔMEGA**

**E-BOOK INTERNET DAS COISAS
PARA INICIANTES**

Sumário

1 INTRODUÇÃO	2
2 CONCEITOS IMPORTANTES	3
3 COMPOSIÇÃO DO NODEMCU.....	12
4 BAIXANDO A IDE DO ARDUINO.....	16
5 CONFIGURANDO O NODEMCU	19
6 CONHECENDO O BLYNK.....	24
7 MONITORAMENTO DE TEMPERATURA	27
8 ALERTA DE CHUVA.....	45
9 CONTROLANDO RELÉS.....	55
10 LIGANDO LÂPADA E VENTILADOR	67
11 CONTROLANDO LEDs COM IR.....	78
12 ALARME	90
13 CONTROLANDO DISPLAY LCD	96
14 CONTROLE DE ACESSO COM RFID.....	101
15 IRRIGAÇÃO AUTOMÁTICA	113
16 FINALIZAÇÃO.....	121



Introdução

Esse E-book foi desenvolvido pela [Arduino Ômega](#) e destina-se aos makers iniciantes que querem aprender sobre Internet das Coisas (IoT) de forma prática e divertida! Aqui você vai aprender como construir projetos utilizando os materiais disponíveis no [Kit Internet das Coisas](#). Será apresentado qual a função dos principais componentes, conceitos de eletrônica e elétrica, possíveis projetos de Internet das Coisas/Automação Residencial utilizando sensores e componentes, ao finalizar este livro você será capaz de construir os seus próprios projetos!.

Kit Internet das Coisas (IoT)

Este é o Kit ideal para o maker que está em busca de projetos de [Internet das Coisas \(IoT\)](#) e [automação residencial](#). Com este kit você poderá fazer projetos para apagar e acender lâmpadas por Wi-Fi, automatizar a irrigação do seu jardim, controlar relés pela internet, monitorar sua casa com os dados de temperatura, umidade de solo, pressão atmosférica, chuva, entre outros.



Você pode adquirir esse Kit
[Clicando Aqui !!](#)

Antes de darmos início em nossos projetos, é importante conhecer alguns **Conceitos** que serão utilizados no decorrer do E-book.

Internet of Things (IoT)

Internet of Things, traduzido para a língua portuguesa torna-se Internet das Coisas, descreve objetos físicos incorporados a **sensores**, **software** e **outras tecnologias** com o objetivo de **conectar e trocar dados com outros dispositivos e sistemas pela internet**. Esses dispositivos variam de objetos domésticos comuns a ferramentas industriais sofisticadas, como por exemplo sensores.

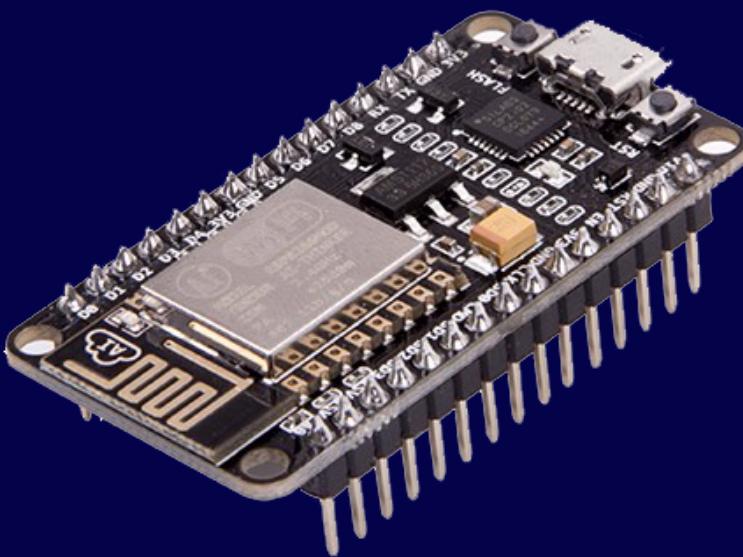
De acordo com o **Instituto Gartner** o número de dispositivos conectados à IoT deve chegar em **50 bilhões** até o ano de **2022!**



O que é NodeMCU ??

Em nossos projetos vamos utilizar o **NodeMCU**, ele nada mais é do que uma placa de desenvolvimento de código aberto que utiliza o chip **ESP8266 (ESP 12-E)**. É muito semelhante ao **Arduino Uno**, visto que ambos são **microcontroladores**. A vantagem de se utilizar o **NodeMCU** é ele possuir **Wi-fi** nativo, dessa forma essa placa é ideal para projetos de **IoT**.

A versão que utilizaremos é a **NodeMCU V3**.



O **NodeMCU** possui 13 pinos digitais e 9 podem ser utilizados como saída PWM (controle de dispositivos variando a intensidade).

Outro diferencial é a possibilidade de fazer a **programação** da placa via **OTA** (Over The Air), assim, através do **Wi-Fi** é possível **programar** o **NodeMCU**.



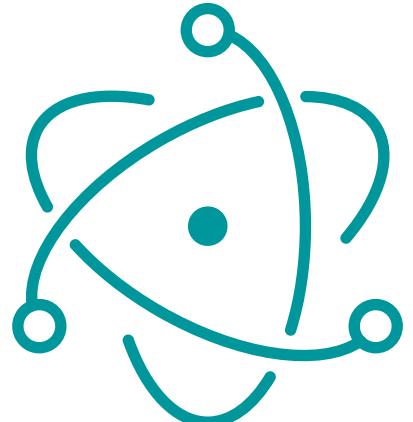
Além disso, é possível programar a placa com as linguagens de programação: **Lua**, **Python**, **JavaScript** ou até mesmo com a **IDE do Arduino**.



Entradas e Saídas

Assim como o **Arduino**, o **NodeMCU** possui entradas e saídas.

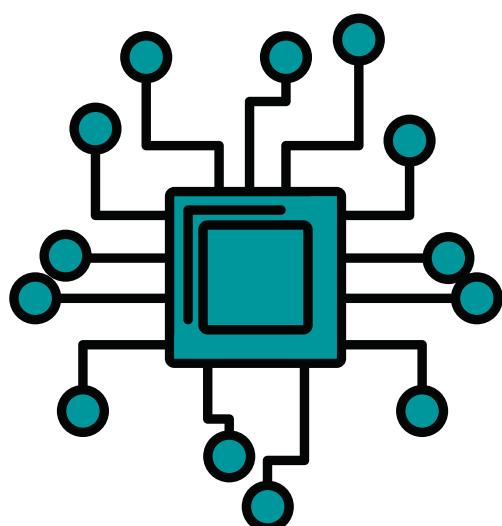
De forma resumida as entradas são por exemplo apertar um botão, e assim, quando ele é acionado é enviado um sinal para o **microcontrolador** do **NodeMCU** que pode ativar algo, como por exemplo um **LED** ou um **motor**.



Já as saídas são exatamente o que é ativado pelo Arduino ou NodeMCU, como um **LED**, **motor**, **buzzer**, entre outros.

Microcontroladores

Os **microcontroladores** são uma junção de Software e Hardware que através da programação conseguimos controlá-los para desempenhar tarefas. Ele é um tipo especial de circuito integrado, visto que conseguimos programar os **microcontroladores** para realizar tarefas específicas. Em breve vamos descobrir diversas coisas que podemos fazer utilizando **microcontroladores**. No caso do **NodeMCU** o microcontrolador utilizado é o **ESP8266**.

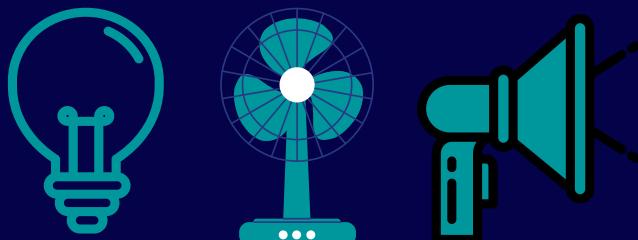


Sinais Digitais

Como visto anteriormente, o **NodeMCU** possui 13 pinos digitais, mas afinal, você sabe o que é um **Sinal Digital**?

O **Sinal Digital** possui uma quantidade limitada, geralmente é representado por dois níveis (Alto ou Baixo). Assim é definido para instantes de tempo e o conjunto de valores que podem assumir é limitada. Podemos usar como **exemplos** de **Saídas Digitais**:

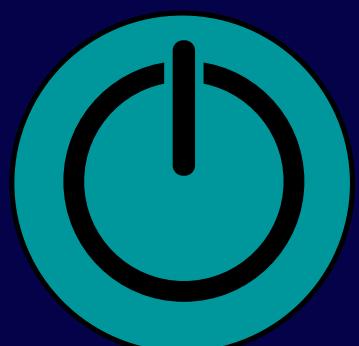
- Lâmpadas;
- Buzinas;
- Ventiladores.



Observe que todos esses equipamentos estão ou ligados ou desligados, não há uma variação contínua.

E também existem as Entradas Digitais, por exemplo:

- Chaves seletoras;
- Fins de Curso;
- Botões.



Sinais Analógicos

Além de portas digitais, o **NodeMCU** também possui uma **Porta Analógica**. O **Sinal Analógico** é um sinal que pode assumir infinitos valores em um intervalo de tempo. **Exemplos de Sinais Analógicos** são:

- [Sensores de Temperatura](#);
- [Sensores de Umidade](#);
- [Sensores de Pressão](#);
- [Potenciômetros](#).



Simplificando...

Para simplificar, pense no interruptor e um dimmer de uma lâmpada...

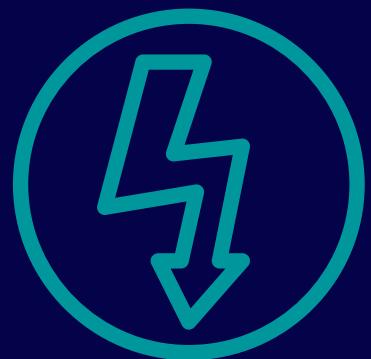
Ou o interruptor está **acionado** e a lâmpada está **acesa** ou está **desacionado** e a lâmpada **apagada**, não existe outro estado. Dessa forma o interruptor é considerado um Sinal Digital.

Já o **dimmer** controla a intensidade da lâmpada, girando o potenciômetro conseguimos regular a intensidade de brilho. Esse é considerado o sinal analógico, pois existem infinitos valores.



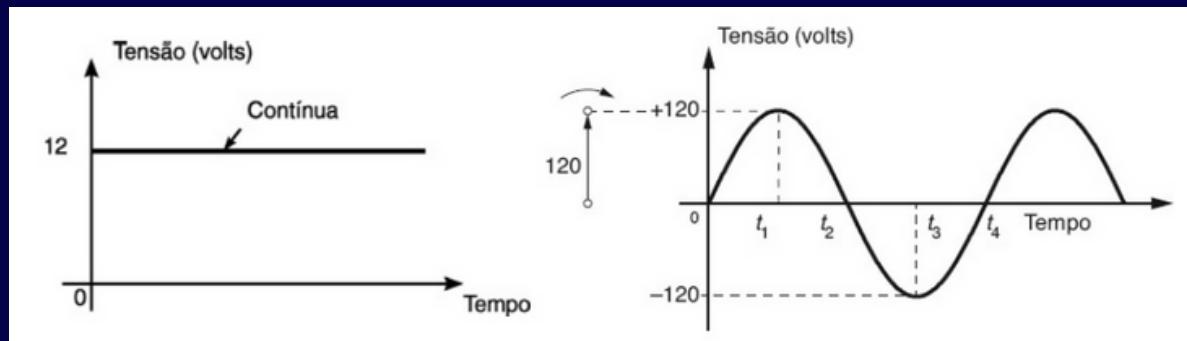
Tensão Elétrica

Também conhecida como **Diferença de Potencial**, a **tensão** é a grandeza que mede a diferença de potencial elétrico entre dois pontos. É a força necessária para mover os elétrons e criar assim uma **Corrente Elétrica**. O equipamento utilizado para medir a tensão é conhecido como **voltímetro**, no **Sistema Internacional** a unidade de medida é **Volts**, onde o símbolo é **V**.



Tipos de Tensão

A tensão pode ser **Contínua** ou **Alternada**. Na contínua ela não muda de polaridade com o passar do tempo, já na alternada é ao contrário, ou seja, muda de polaridade com o passar do tempo.



Exemplos

- A **Contínua** não muda de polaridade com o passar do tempo. A tensão das pilhas é contínua.
- Já a tensão **Alternada** a polaridade será alternada de acordo com a frequência, no caso de uma tomada no Brasil, a frequência normal é de 60Hz, o que quer dizer que a polaridade desta tensão vai alternar 60 vezes por segundo.



Fórmulas

Podemos calcular a **Tensão Elétrica** utilizando a **Lei de Ohm**, onde a tensão é igual a resistência vezes a corrente elétrica.

Assim, temos a seguinte fórmula:

Tensão = Resistência x Corrente



Resumindo: $U = R \times I$

OBS: U = Tensão
 R = Resistência
 I = Corrente

Também é possível calcular a tensão se caso soubermos o valor da **Corrente** e da **Potência** elétrica, onde tensão elétrica é igual a potência dividida pela corrente.

Assim, temos a seguinte fórmula:

Tensão = $\frac{\text{Potência}}{\text{Corrente}}$



Resumindo: $U = \frac{P}{I}$

OBS: U = Tensão
 P = Potência
 I = Corrente

OBS: O termo **Voltagem** apesar de ser muito falado não existe, é apenas um termo popular.

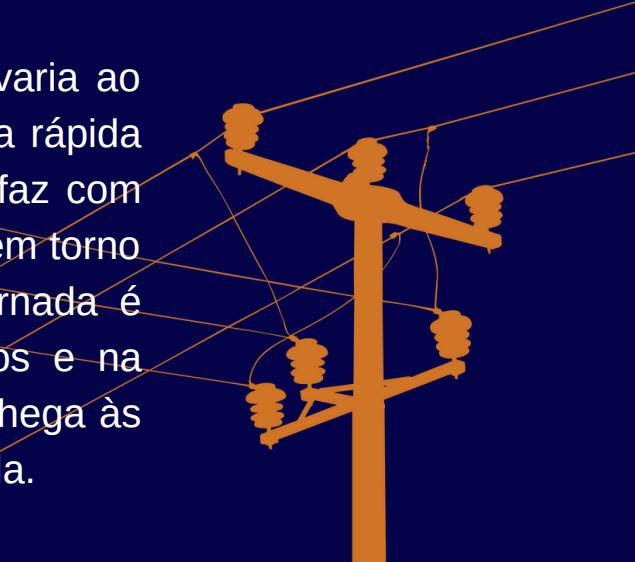
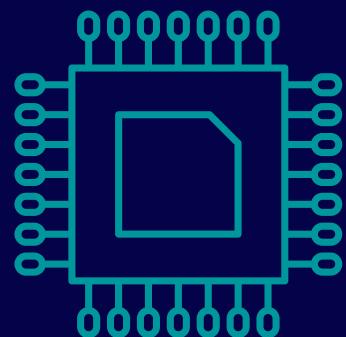
Corrente Elétrica

A **Corrente Elétrica** é um fenômeno físico onde ocorre o movimento de cargas elétricas, como os elétrons, que acontece no interior de diferentes materiais em razão da aplicação de uma diferença de potencial elétrico. A corrente elétrica é uma grandeza escalar, sua unidade de medida de acordo com o Sistema Internacional de Unidades, é o **Ampère**, cujo símbolo é **A**. Essa unidade mede o módulo da carga elétrica que atravessa a secção transversal de um condutor a cada segundo e por isso também pode ser escrita como **Coulombs Por Segundo**, onde o símbolo é **C/s**.

Tipos de Corrente

Assim como a **Tensão Elétrica**, existem dois tipos de corrente, a **Corrente Alternada (CA)** e a **Corrente Contínua (CC)**.

- **Corrente Contínua:** Não tem variação ao longo do tempo e se mantém praticamente constante, os elétrons são forçados a deslocar-se em sentido único. Esse tipo de corrente é comum em dispositivos que utilizam baixas tensões, como eletrônicos em geral.
- **Corrente Alternada:** A corrente alternada varia ao longo do tempo. Nesse tipo de corrente, uma rápida inversão de polaridade do potencial elétrico faz com que os elétrons se desloquem em vai e vem em torno de uma posição fixa. Corrente elétrica alternada é utilizada principalmente em motores elétricos e na transmissão de eletricidade, a corrente que chega às nossas casas é uma corrente elétrica alternada.



Fórmulas

Para se saber qual a intensidade da **Corrente Elétrica** em um condutor, é possível utilizar a equação de carga elétrica pelo tempo.

Assim, temos a seguinte fórmula:

$$\text{Intensidade de Corrente} = \frac{\text{Carga elétrica}}{\text{Intervalo de tempo}}$$



Resumindo: $I = \frac{Q}{\Delta t}$

OBS: Q = Carga elétrica
 Δt = Intervalo de tempo
 I = Corrente

Utilizando a **Tensão Elétrica** e a **Resistência** também é possível calcularmos a corrente, onde corrente é igual tensão elétrica dividido pela resistência.

Assim, temos a seguinte fórmula:

$$\text{Corrente} = \frac{\text{Tensão}}{\text{Resistência}}$$



Resumindo: $I = \frac{U}{R}$

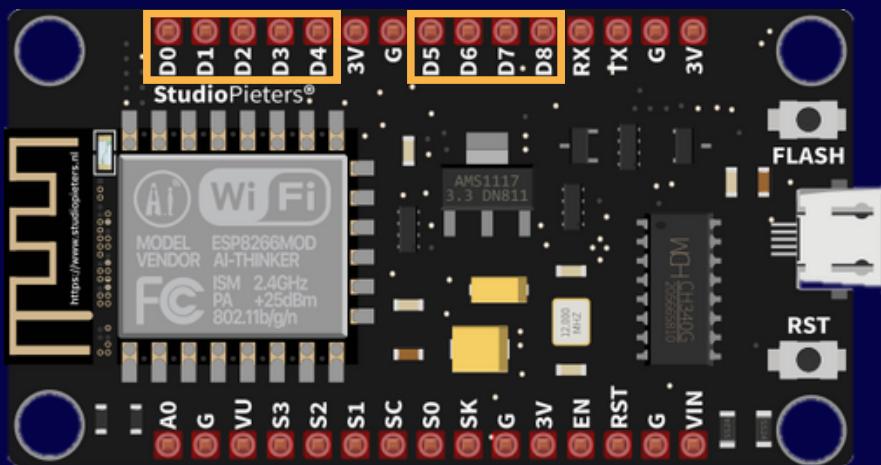
OBS: U = Tensão
 R = Resistência
 I = Corrente

3

Composição do NodeMCU

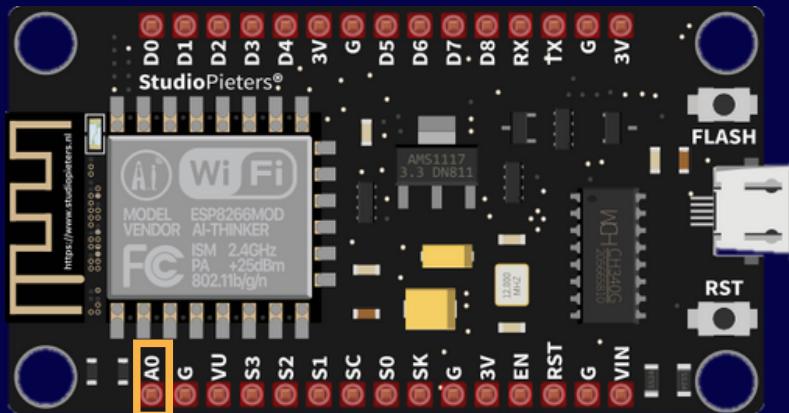
Bem, agora que já conhecemos os principais conceitos que vão ser utilizados, vamos conhecer um pouco mais sobre o **NodeMCU**.

Pinos Digitais



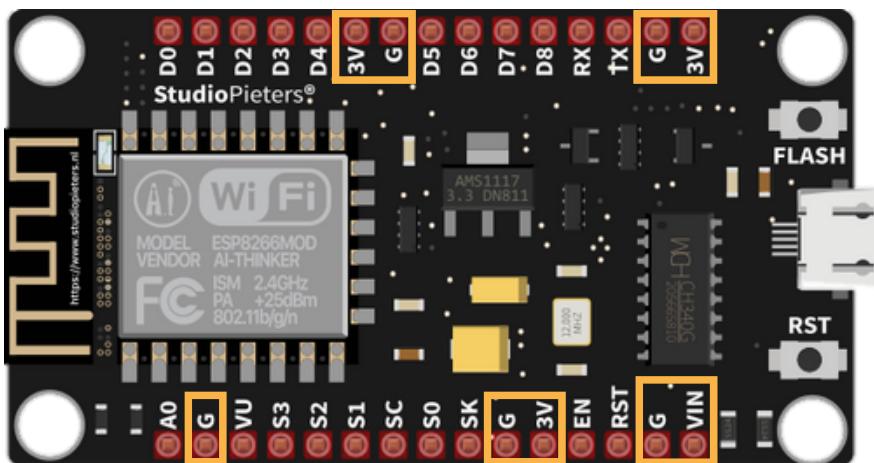
As portas **D0, D1, D2, D4, D5, D6, D7** e **D8** são os pinos de entradas e saídas digitais, usadas para conexões de **módulos** e **sensores**.

Pino Analógico



O pino **A0** é a **entrada** e a **saída analógica** da placa, utilizada para conexões de **módulos** e **sensores**.

Pinos para Alimentação

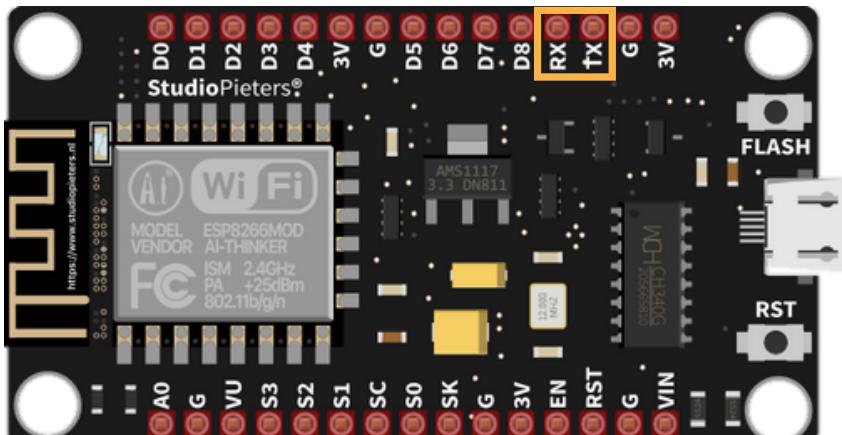


Pinos G: É o terra que funciona como o negativo para todos os circuitos e dispositivos externos. O GND é comum a todas as portas, ou seja, ele é compartilhado, diferente das portas digitais.

Pino 3V: Pino que fornece 3,3 volts para a alimentação de dispositivos externos.

Pino Vin: A porta Vin fornece a mesma tensão que é recebida pelo pino de alimentação externo.

Pinos para Comunicação

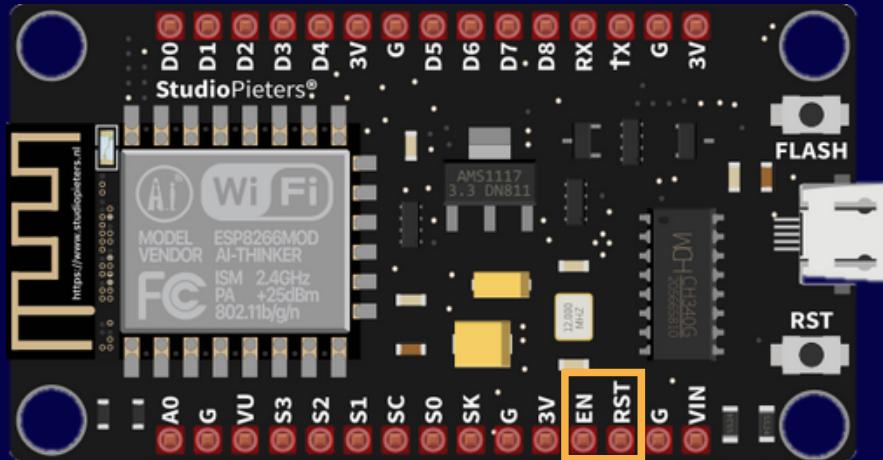


RX: Receiver, recebe bits.

TX: Transmitter, envia bits.

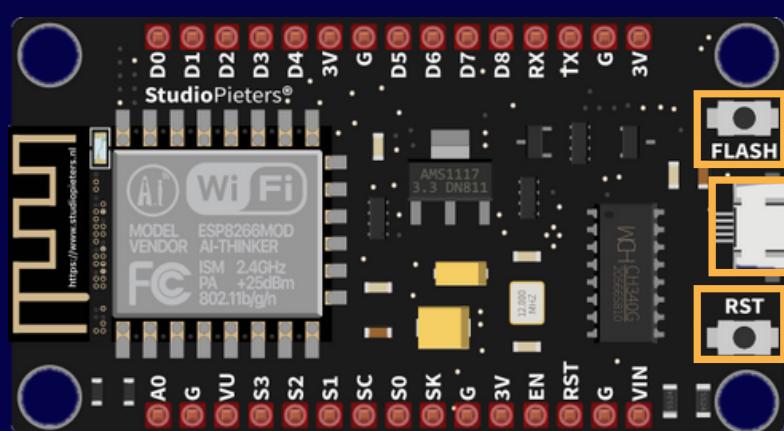
São as portas utilizada para fazer a **Comunicação Serial**.

Outros Pinos



Pino EN: O chip ESP8266 é habilitado quando o pino EN está em nível lógico **alto**. Quando está em nível lógico **baixo**, o chip funciona com a potência mínima.

Pino RST: É usado para **reiniciar o NodeMCU**.



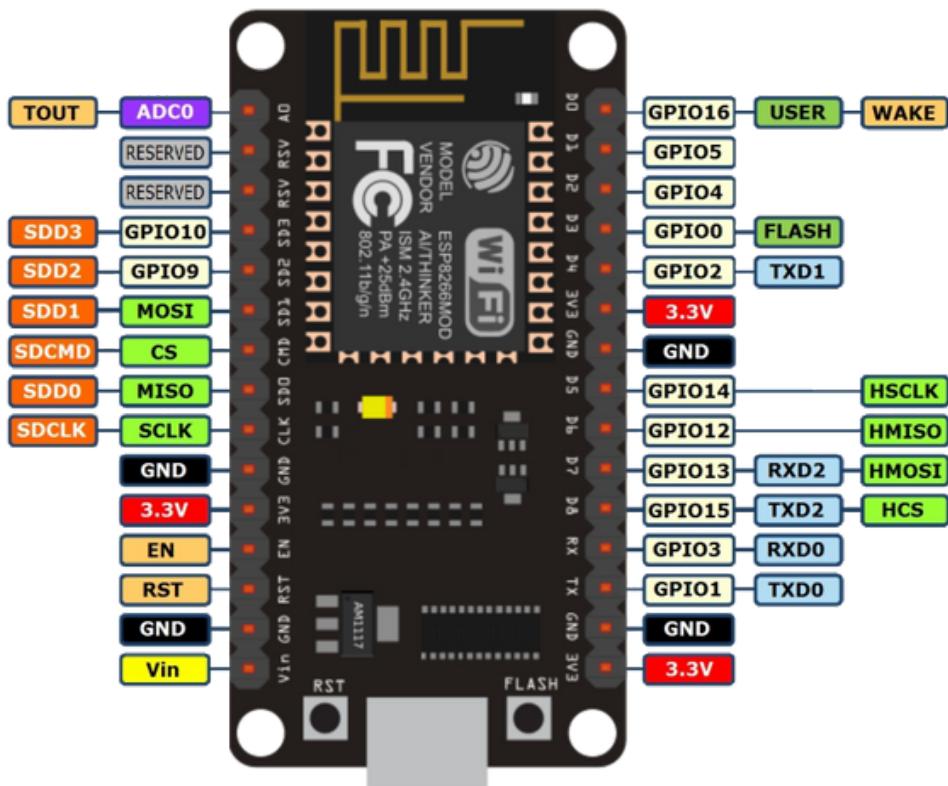
FLASH: Permite a gravação do programa no **ESP-12**.

Entrada Micro Usb: Usada para a alimentação da placa e para enviar a programação.

RST: É usado para reiniciar o NodeMCU.

Pinagem do NodeMCU

Entender os pinos do **NodeMCU** pode ser complicado no início, isso ocorre porque existem duas maneiras de declarar os pinos da placa na programação usando a nomenclatura escrita na placa, como vimos acima, ou pelo **GPIO**.



GPIO é basicamente um conjunto de pinos responsável por fazer a comunicação de entrada e saída de sinais digitais, recebendo funções via programação. O **NodeMCU** tem **17** pinos **GPIO**.

4

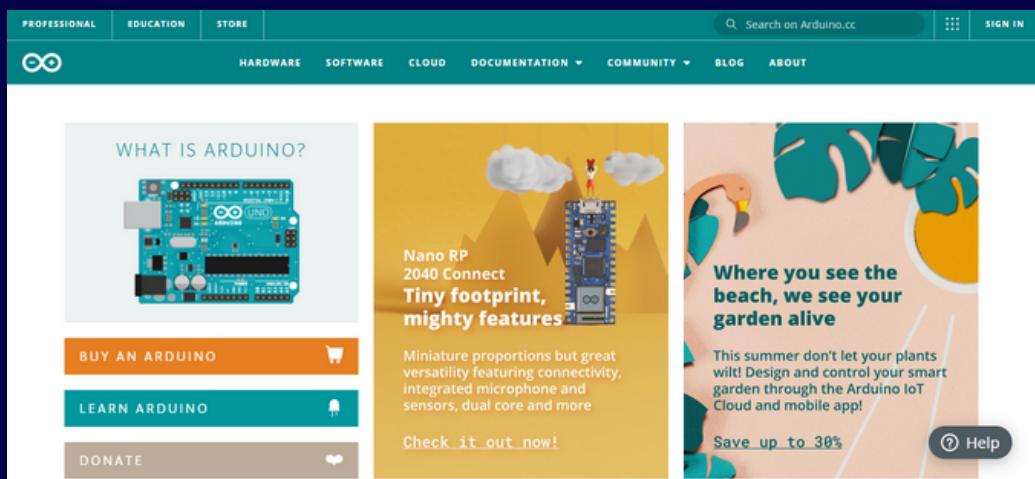
Baixando a IDE do Arduino

Isso mesmo !! Para **programarmos** nossa placa vamos utilizar a **IDE do Arduino** !

Clique Aqui para baixar a **IDE do Arduino**. Caso tenha dúvidas, siga o passo a passo abaixo.

1

Acesse o site oficial do Arduino [Clicando aqui](#).



2

Clique em **Software**.

A screenshot of the Arduino Software page. The top navigation bar has a red box around the 'SOFTWARE' tab. Below it, there's a 'Downloads' section featuring the 'Arduino IDE 1.8.15' download. To the right, a 'DOWNLOAD OPTIONS' box is highlighted with a red box, listing download links for Windows (Win 7 and newer, ZIP file, app), Linux (32 bits, 64 bits, ARM 32 bits, ARM 64 bits), and Mac OS X (10.10 or newer). Below this box are links for Release Notes and Checksums.

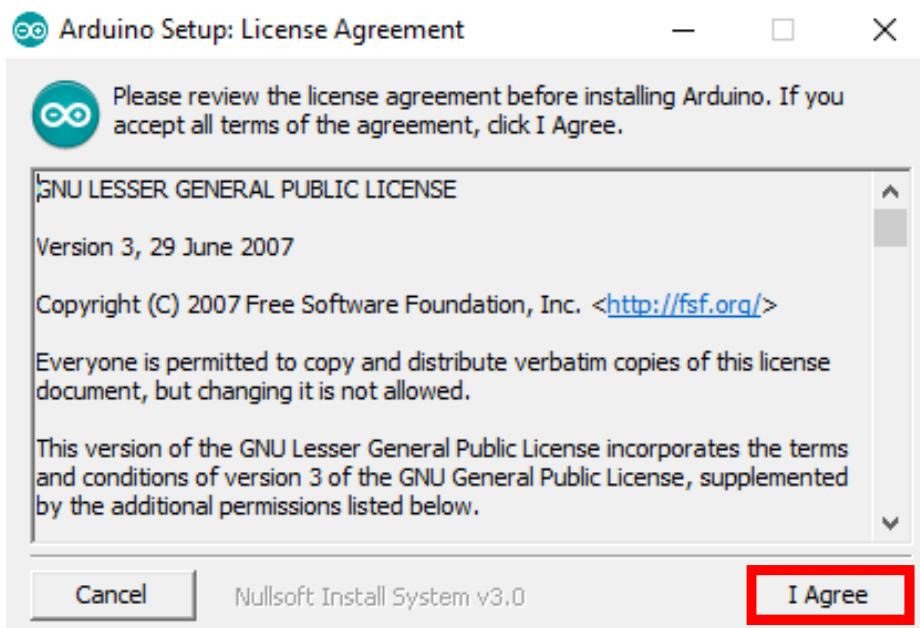
Aqui é onde vamos fazer o Download da **IDE do Arduino**. Selecione qual sistema operacional você utiliza.

Caso possua **Windows 7** ou uma **versão superior**, selecione a primeira opção: **Windows Win 7 and newer**.

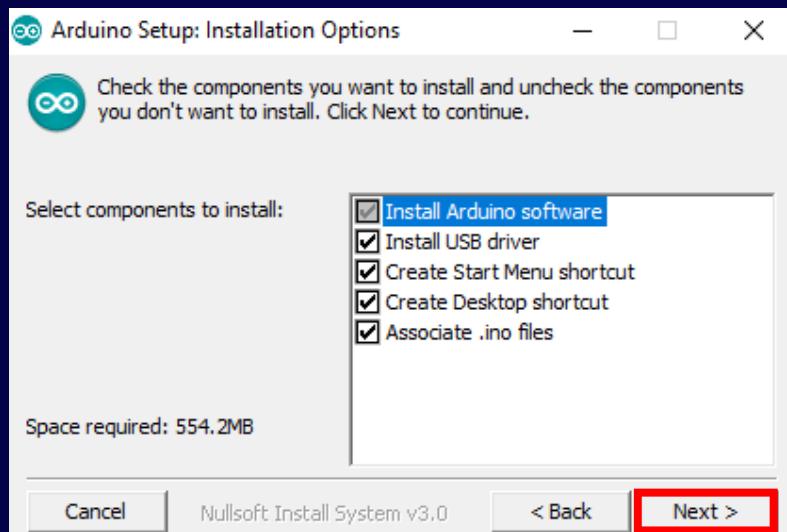
The screenshot shows the Arduino IDE download page. At the top, there's a navigation bar with links for VARE, SOFTWARE, CLOUD, DOCUMENTATION, COMMUNITY, and BLOG. Below the navigation bar, a section titled "Support the Arduino IDE" displays statistics: "Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **53.359.212** times — impressive! Help its development with a donation." Below this, there are several donation buttons with amounts: \$3, \$5, \$10, \$25, \$50, and Other. Two buttons are highlighted: "JUST DOWNLOAD" (in red) and "CONTRIBUTE & DOWNLOAD" (in green). Below the buttons is a small illustration of a computer monitor with a circuit board and two stylized characters.

Se quiser, você pode fazer uma doação para ajudar o desenvolvimento da **Arduino IDE**. Caso contrário, basta clicar em **Just Download** e o programa começará a baixar.

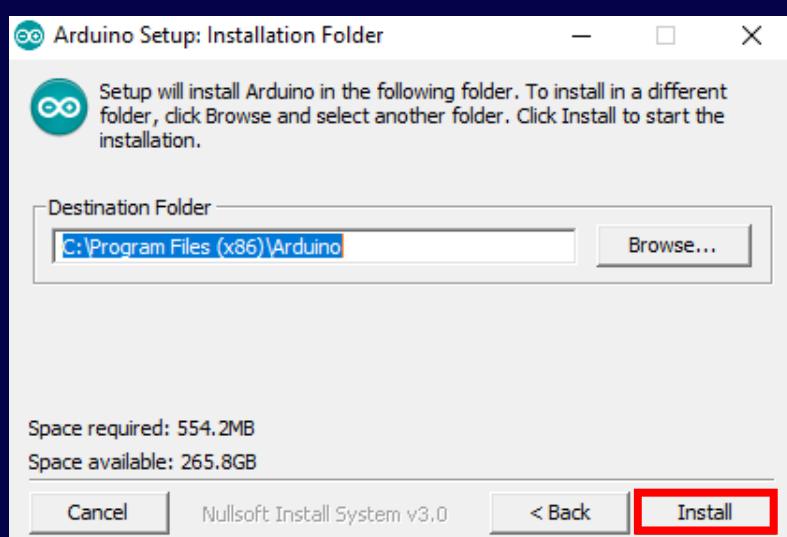
3 Após o Download do instalador da IDE terminar, **clique duas vezes sobre o ícone** gerado para abri-lo.



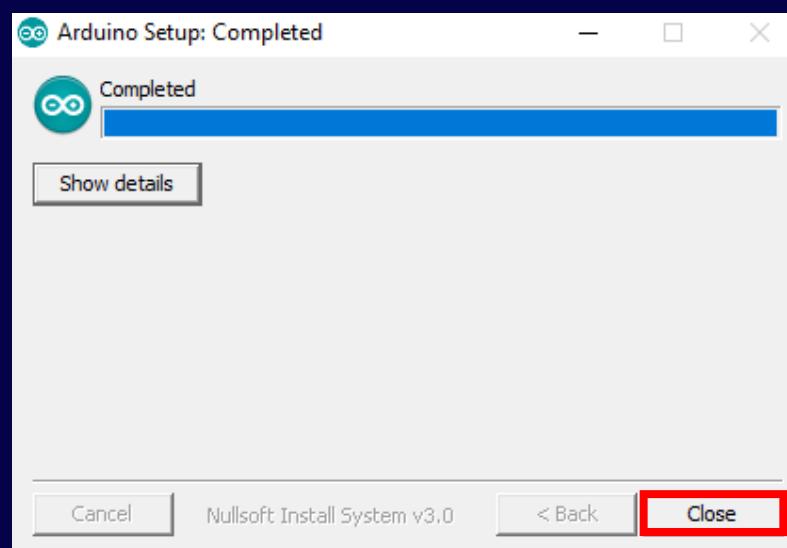
Clique em **I Agree**.



Selecione todas as caixas e clique em **Next**.



Escolha o local onde a IDE vai ser instalada em seu computador e clique em **Install**. A instalação começará.



Quando a instalação for concluída clique em **Close**.

Pronto, agora é só programar !

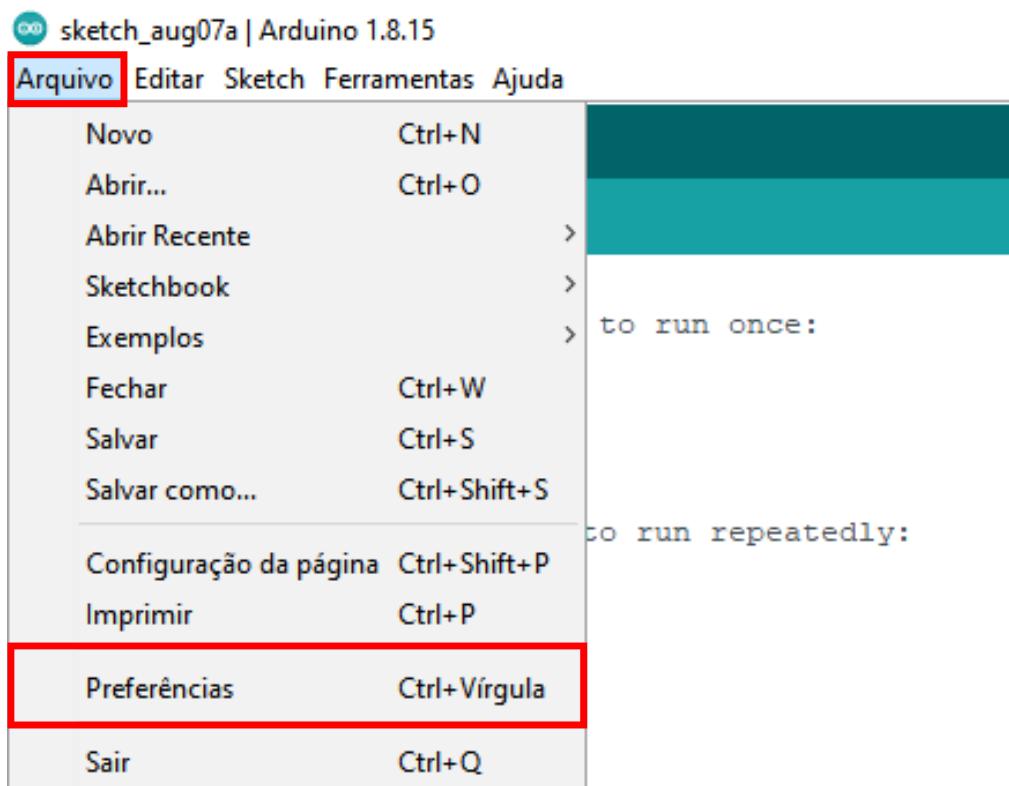
5

Configurando o NodeMCU na IDE

Precisamos configurar a **IDE do Arduino** para conseguirmos programar o **NodeMCU**.

1

Abra a **IDE do Arduino** e clique em **Arquivo** e depois em **Preferências**.

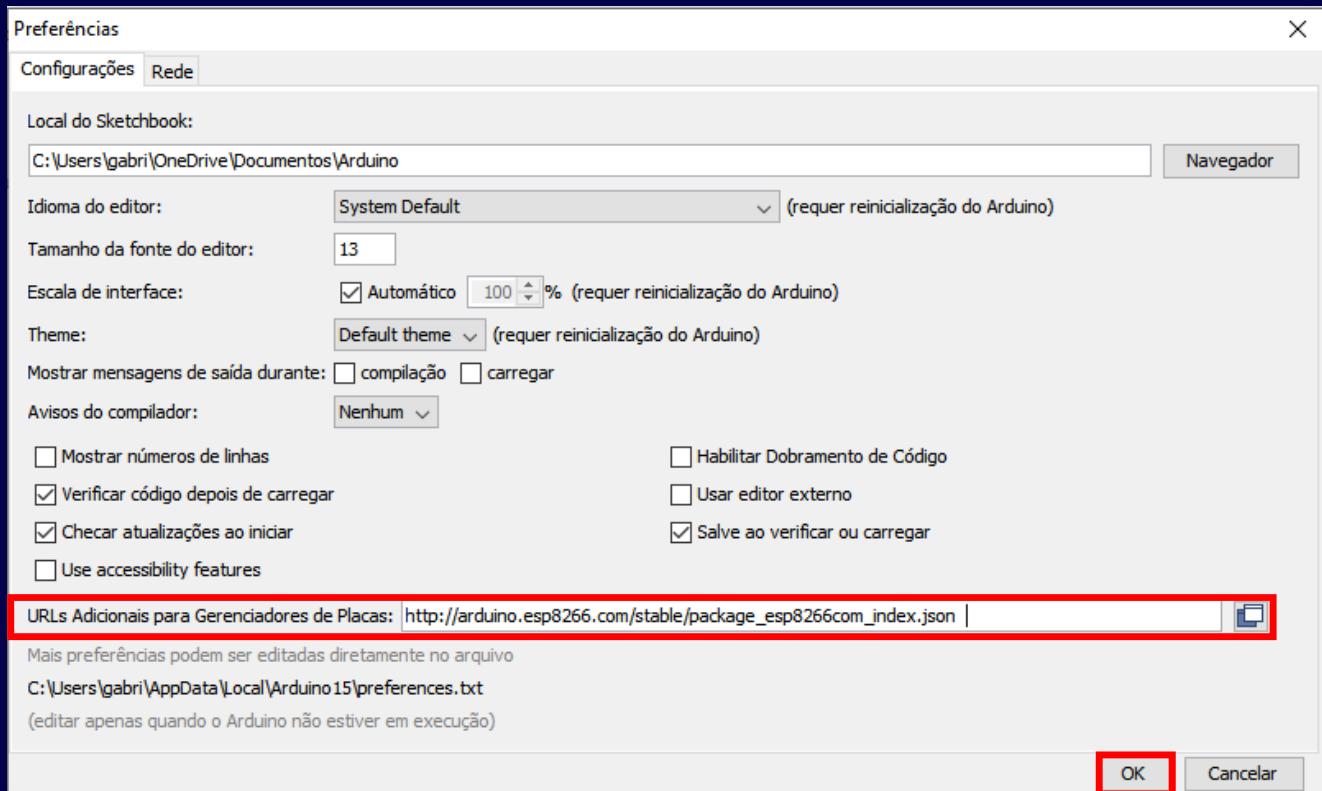


2

Copie o link:

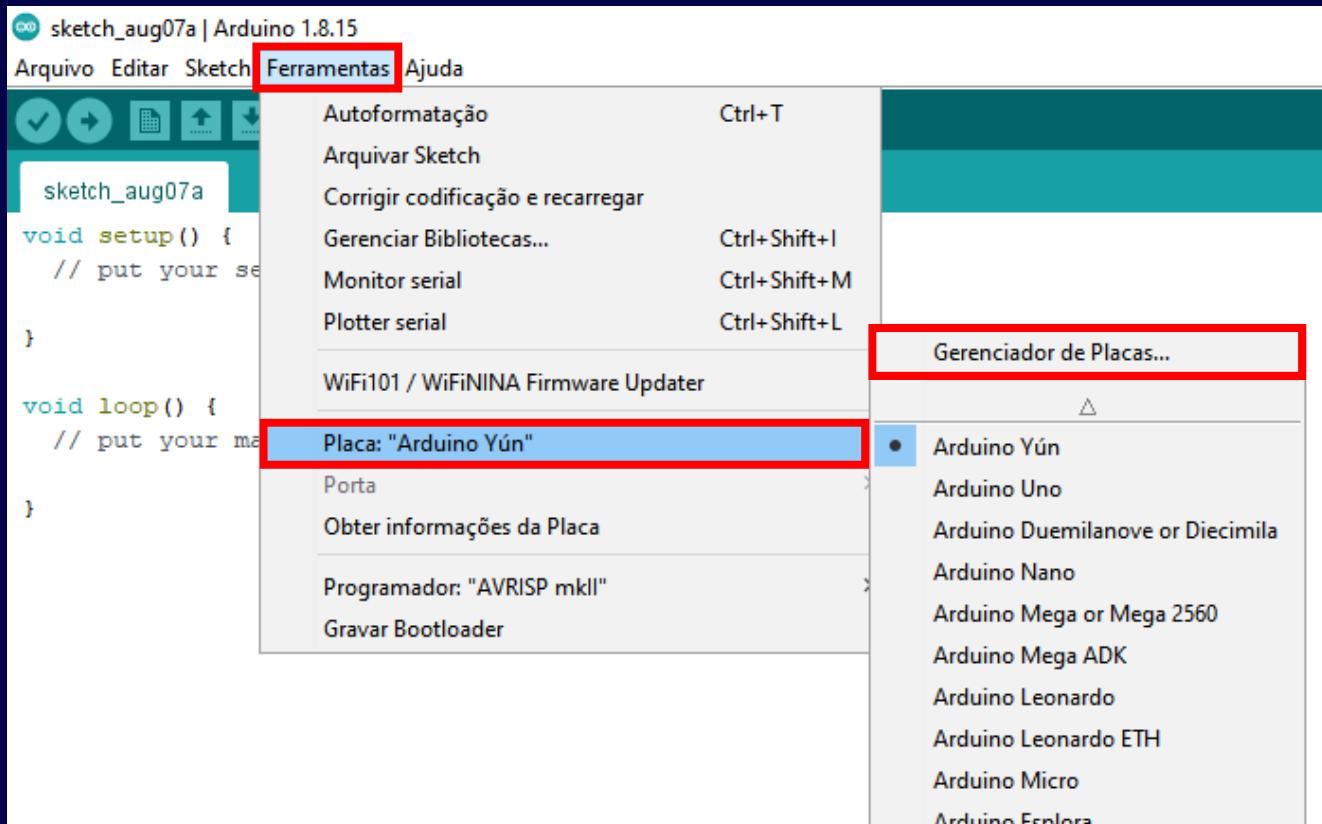
http://arduino.esp8266.com/stable/package_esp8266com_index.json

Cole no campo **URLs adicionais de Gerenciadores de Placas**, depois clique em **ok**.



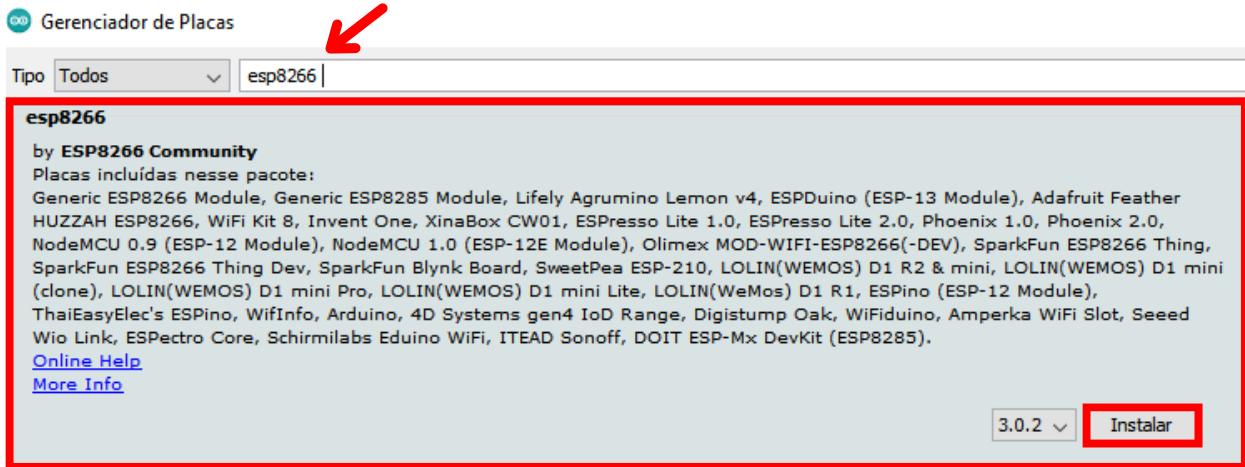
3

Clique em **Ferramenta**, depois em **Placa** e em **Gerenciador de Placas**.



4

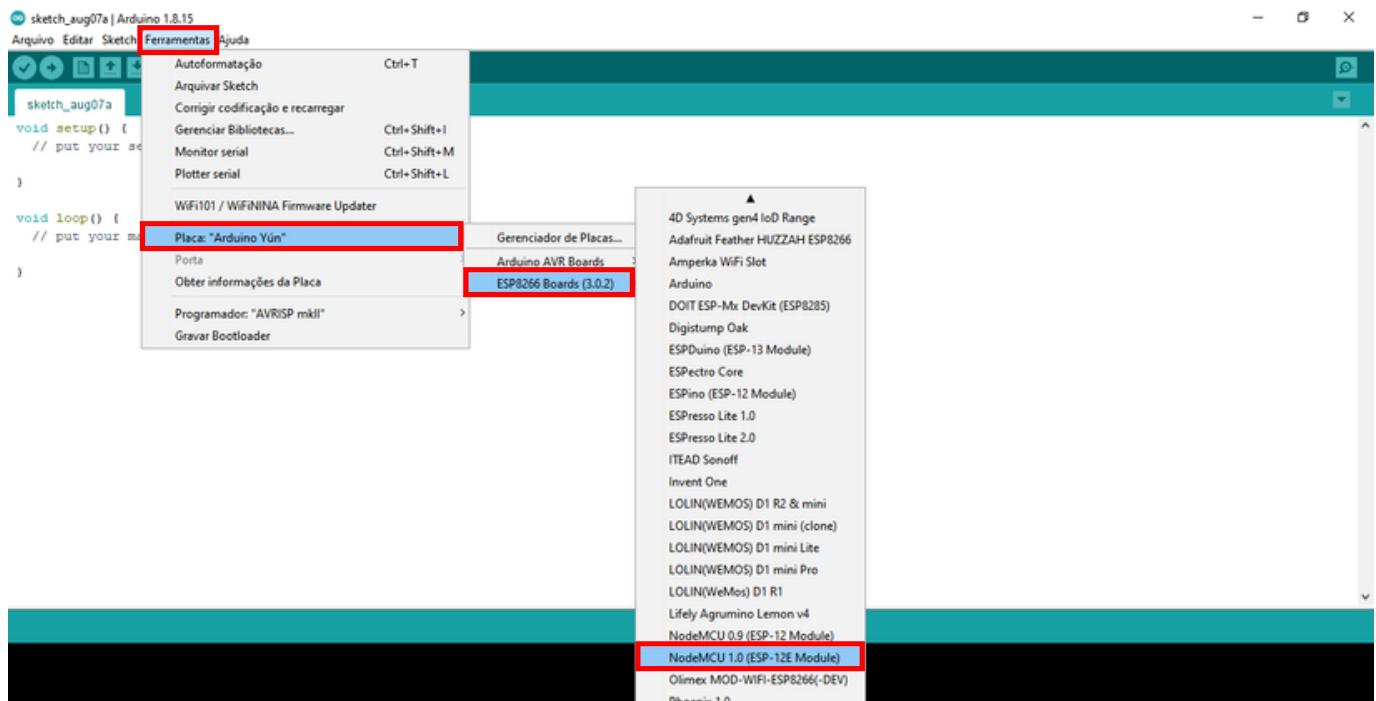
Busque na **barra de pesquisa** por **ESP8266**.



Clique em **Instalar**.

5

Clique em **Ferramentas**, depois em **Placa**, selecione **ESP8266 Boards (3.0.2)**, clique em **NodeMCU 1.0 (ESP 12-E module)**.



Possíveis Erros

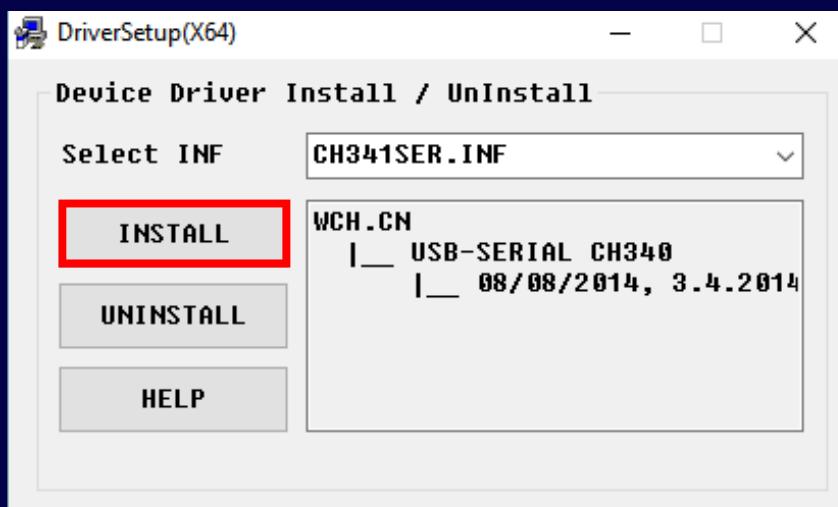
1

O Computador não reconhece a placa

Nesse caso é necessário instalar um **driver** adequado para o computador reconhecer a placa.

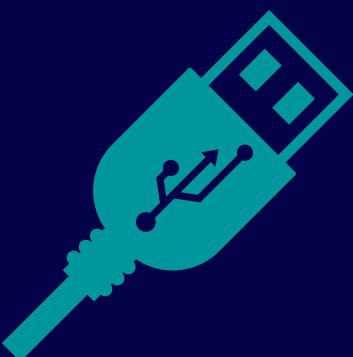
[Clique aqui](#) para baixar o driver.

Abra o Instalador do Driver e clique em **INSTALL**.



Pronto, agora o computador deve reconhecer o **NodeMCU**.

OBS: Esse driver é recomendado para placas **NodeMCU V3**, a qual é enviada no Kit, se você possui uma placa **NodeMCU V2**, [Clique Aqui](#) para baixar o driver correto para esse modelo.





Mesmo com o driver instalado, o NodeMCU não é reconhecido

Observe se ao plugar a placa no **USB** do computador um **LED** se **acende**, caso não acenda **o problema está na porta USB do computador.**

Para resolver esse problema será necessário, quando for programar o **NodeMCU, energiza-lo com uma fonte externa e depois plugar o cabo USB na porta do computador.**



6 Conhecendo o Blynk

Em muitos projetos nesta apostila vamos utilizar o aplicativo **Blynk**. O **Blynk** foi desenvolvido para ser utilizado em **projetos IoT**, com ele conseguimos comunicar através do celular com nossa placa **ESP-32** e controlá-la via **Wi-fi** ou **Bluetooth**.

É possível até mesmo **programarmos** nosso **microcontrolador** sem escrever uma linha de códigos se quer !!!



[Clique Aqui](#) para baixar o aplicativo para **Android** !!

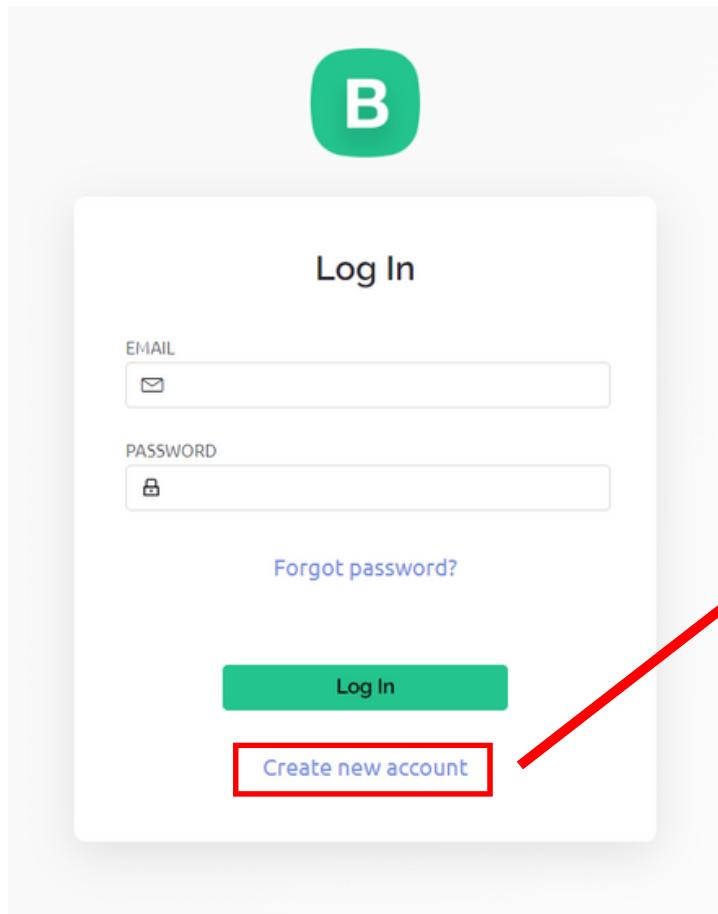
Caso utilize **IOS**, [Clique Aqui](#) para baixar.

Criando uma Conta

Para utilizar o **Blynk**, é preciso criar uma **conta**. É possível cria-la pelo site da **Blynk**, [clique aqui](#) para acessar.

OBS: É muito importante inserir um **e-mail** que você tem acesso !!

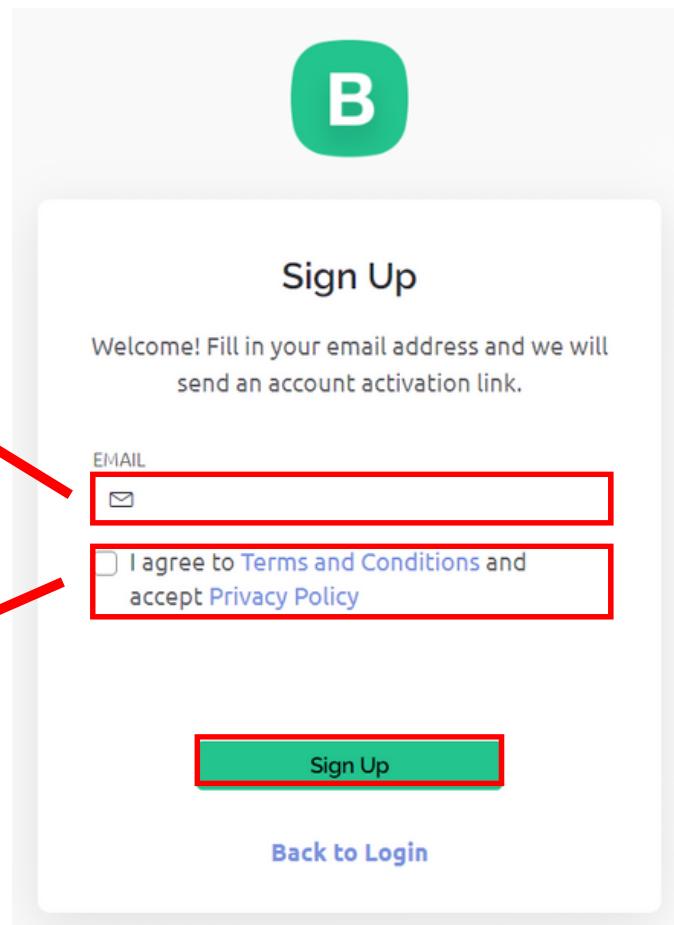




Clique em **Create new account**.

Você deve inserir seu e-mail, o **Blynk** enviara uma mensagem nesse **e-mail** para confirmar que ele é realmente seu. Logo após a confirmação, será necessário criar um senha.

Não se esqueça de **aceitar os Termos** e **Condições** de uso do site !

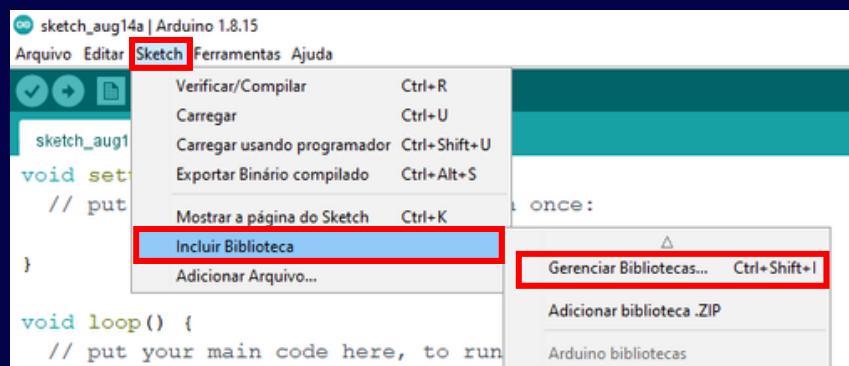


Configurando o Blynk na IDE

É necessário instalar a **biblioteca** do **Blynk** na IDE do Arduino para as programações funcionarem **corretamente**.

1

Para instalar a biblioteca clique em **Sketch**, depois em **Incluir Biblioteca** e **Gerenciar Bibliotecas**.



2

Na barra de pesquisa busque por **Blynk**, irá aparecer um botão para instalar a **biblioteca**, o botão só é habilitado após a escolha da versão da biblioteca, selecione a versão feita por Volodymyt Shymankyy e **clique no botão instalar**.



OBS: Instale a biblioteca feita por **Volodymyt Shymankyy** e **seleccione a versão mais recente**.



Monitoramento de Temperatura

Finalmente chegamos em nosso **primeiro projeto** !!! Nós vamos **monitorar a temperatura e a umidade** do ambiente utilizando o **Blynk**, a placa **NodeMCU** e um **sensor** !!!

Mas antes é preciso conhecer um pouco mais sobre os componentes que serão utilizados no projeto.

Placa Borne para NodeMCU

A **Placa Borne** foi projetada com o objetivo de facilitar a conexão de circuitos externos na placa.

Nosso **NodeMCU** é a **versão 3**, ele é a placa que tem o maior tamanho, dessa forma ela não cabe na **Protoboard**. Assim, é necessário o **adaptador**.

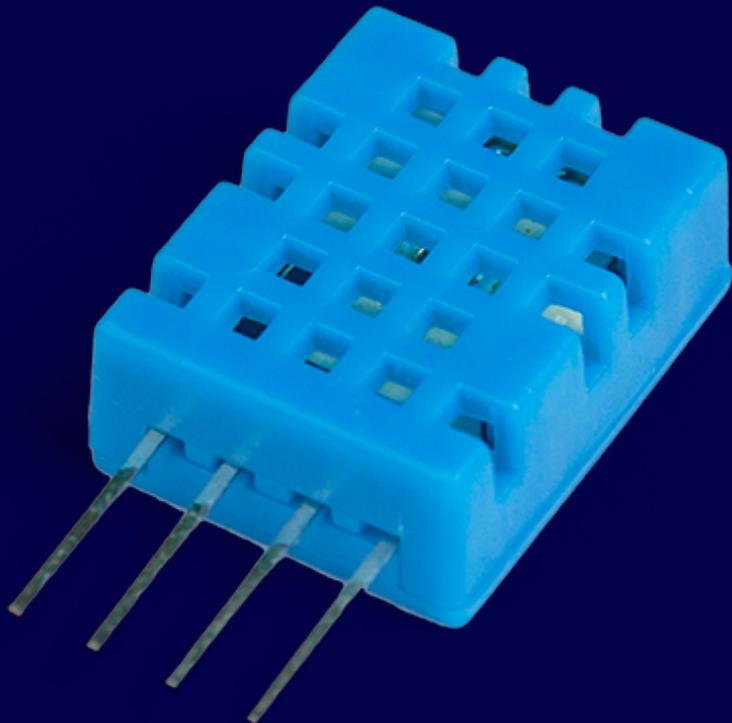
Basta encaixar o **NodeMCU** na **Placa Borne**.



Sensor DHT11

O **Sensor DHT11** é um sensor de **temperatura** e **umidade** que permite fazer leituras de temperaturas entre **0** a **50** Celsius e umidade entre **20** a **90%**, é muito utilizado em projetos com **microcontroladores**.

O elemento sensor de temperatura é um **termistor** do tipo **NTC** e o sensor de Umidade é do tipo **HR202**, o circuito interno faz a leitura dos sensores e se comunica a um **microcontrolador** através de um sinal serial de uma via.

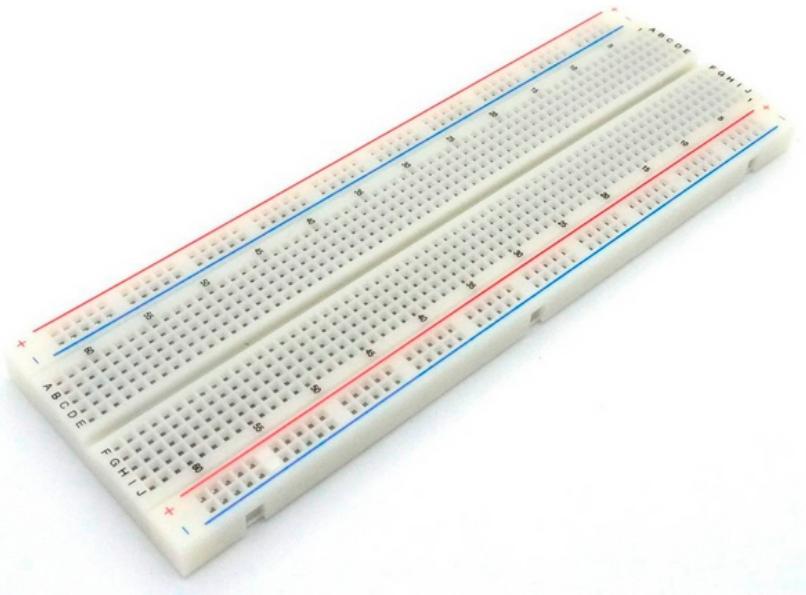


Cada elemento **DHT11** é calibrado em laboratório, isso proporciona extrema precisão de leitura. O coeficiente de calibração é armazenado em forma de programa na memória **OTP** (One Time Programmed), que é utilizado pelo processo interno de leitura do sensor.

Com seu **tamanho reduzido**, sua **baixo consumo** de potência e sua alta capacidade de transmissão de sinal de até **20 metros** o torna uma ótima opção para diversas aplicações inclusive as mais exigentes.

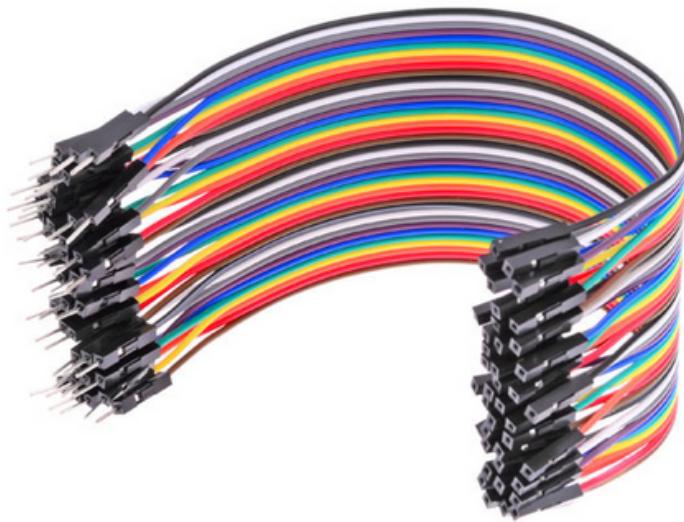
Protoboard

Nada mais é do que uma placa com furos e conexões condutoras utilizada para a montagem de protótipos e projetos que estão na fase inicial. A vantagem de se utilizar uma **Protoboard** é que podemos montar os componentes direto nele, assim eliminamos a necessidade de utilizar a solda.



Jumpers

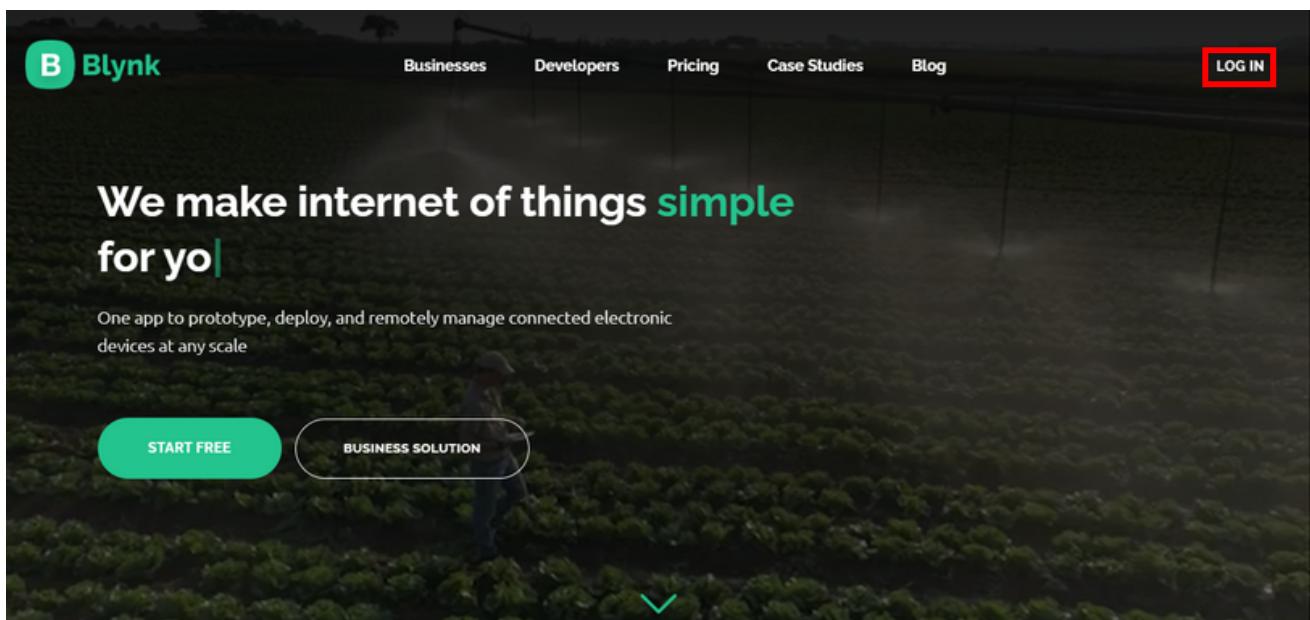
Os Jumpers nada mais são do que condutores para ligar um ponto a outro.



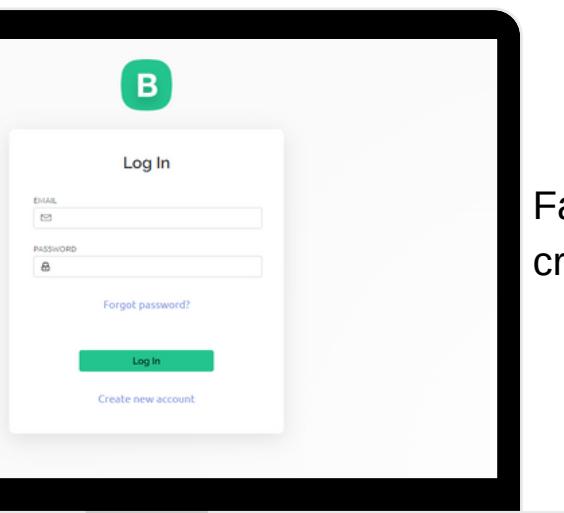
Configurando o Blynk



Vamos acessar o site do [Blynk](#) e acessar nossa conta que criamos anteriormente. [Clique aqui](#) para acessar.



Clique sobre **LOG IN**.



Faça o login com o **e-mail** e a **senha** criada anteriormente.



Clique sobre **New Templates**.

The screenshot shows the Blynk platform's main dashboard. On the left is a sidebar with icons for Devices, Locations, and Users. The main area has a heading "Start by creating your first template" and a sub-instruction: "Template is a digital model of a physical object. It is used in Blynk platform as a template to be assigned to devices." A red box highlights the "New Template" button at the bottom right of the main area.



Crie um novo **Template**.

Create New Template

NAME
Temperatura e Umidade

HARDWARE
ESP8266

CONNECTION TYPE
WiFi

DESCRIPTION
This is my template

19 / 120

Cancel Done

Adicione um nome para o **Template**.
Em **Connection Type** selecione **WiFi**.
Em **Hardware** selecione **ESP8266**.
Por fim, clique em **Done**.

4

Clique sobre **Datastreams**.

Temperatura e Umidade

Datastreams (destacado com um retângulo vermelho)

TEMPLATE NAME: Temperatura e Umidade

HARDWARE: ESP8266 | **CONNECTION TYPE:** WiFi

DESCRIPTION: This is my template

TEMPLATE ID: TMPLugnop2Bp | **MANUFACTURER:** My organization 9402DL

OFFLINE IGNORE PERIOD: 00 hrs 00 mins 00 secs

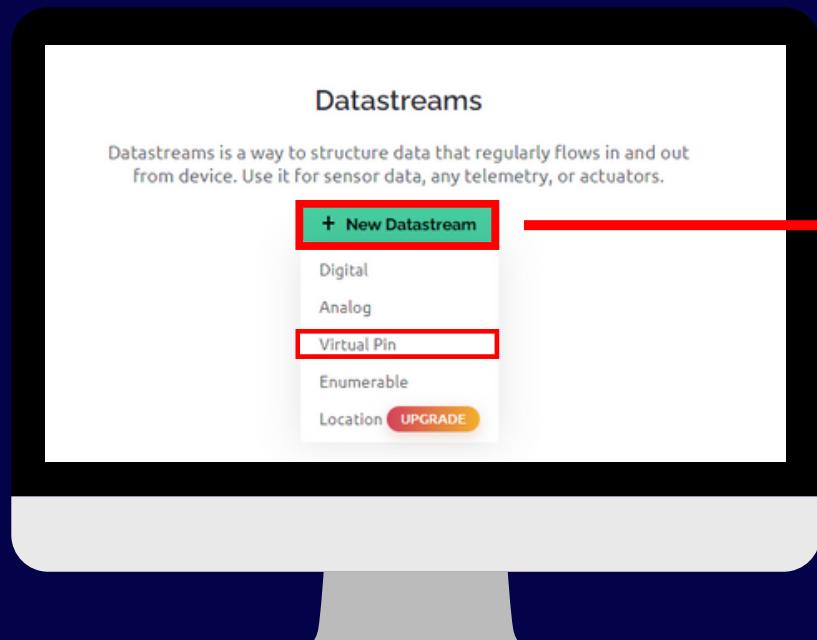
HOTSPOT PREFIX: Hotspot Prefix

FIRMWARE CONFIGURATION:

```
#define BLYNK_TEMPLATE_ID "TMPLugnop2Bp"
#define BLYNK_DEVICE_NAME "Temperatura e Umidade"
```

Template ID and Device Name should be included at the top of your main firmware.

Cancel | **Save**

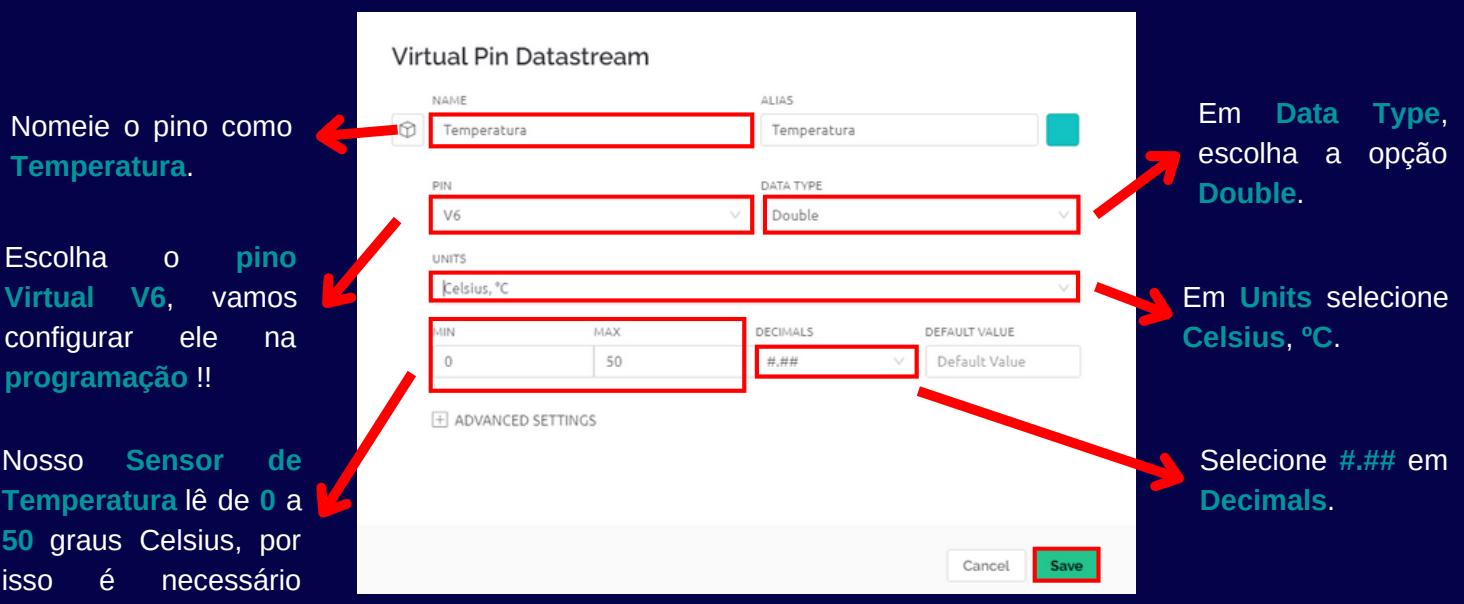


Clique sobre **New Datasteam** e selecione a opção **Virtual Pin** (pinos virtuais).

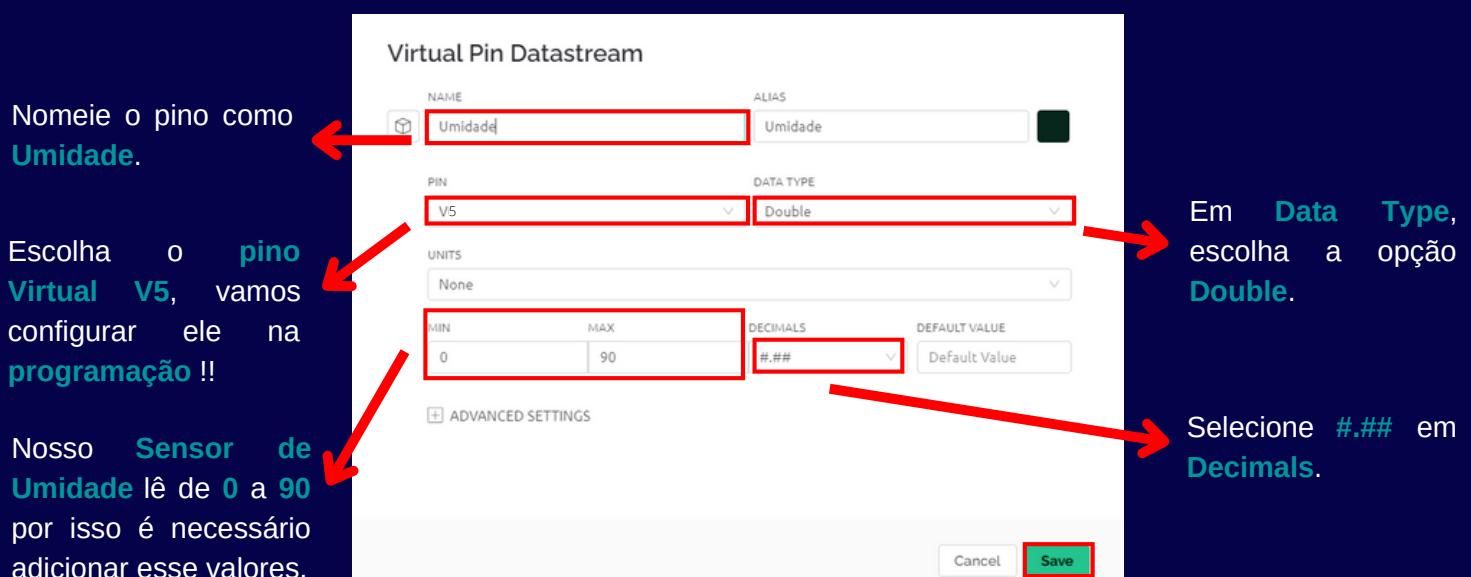
Configurando os Pinos Virtuais

Vamos precisar de dois **pinos virtuais**, um para a **temperatura** e um para a **umidade**.

Depois, clique em **Datastreams**, em seguida **New Datastream** e escolha a opção **Virtual Pin**. Vamos iniciar configurando o **pino virtual** para a **Temperatura**.



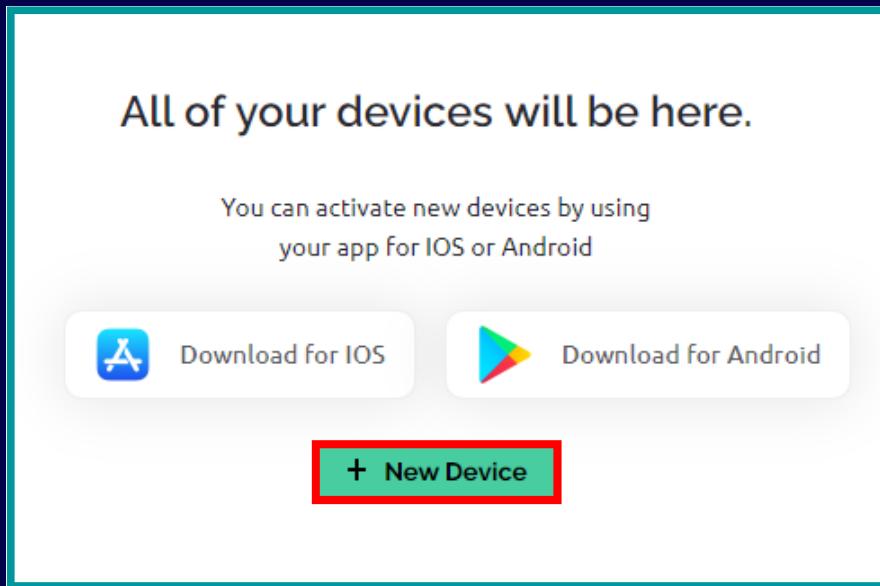
Agora vamos configurar o **pino virtual** responsável pela **umidade**.



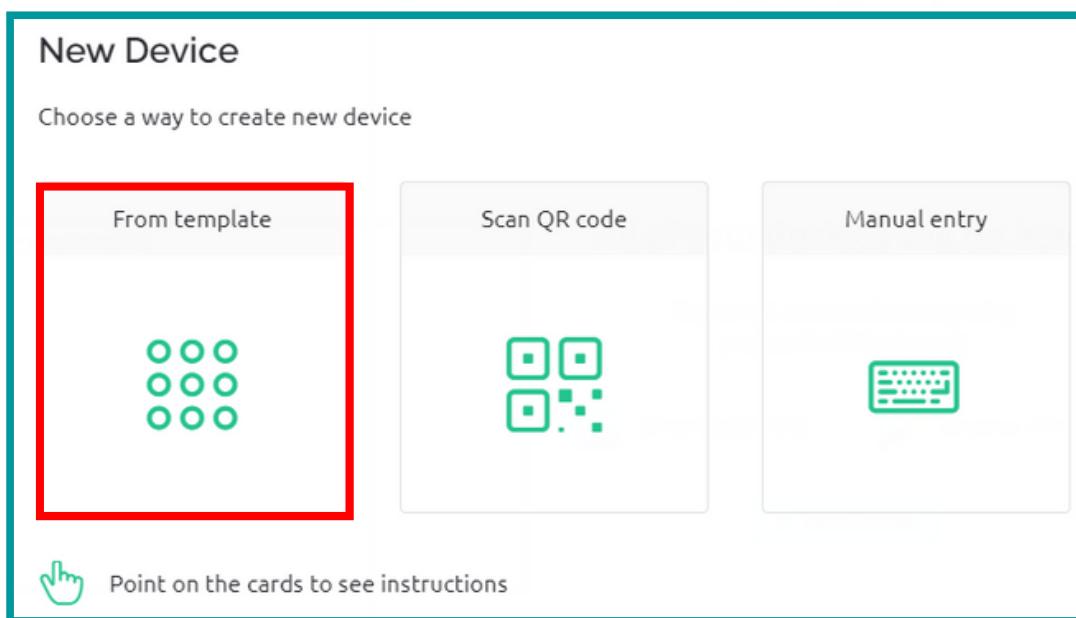
Após finalizar as configurações dos pinos, clique em **Save** para salvar e após clique sobre a **Lupa** (Search).

The screenshot shows a software interface for managing datastreams. On the left is a sidebar with icons for Home, Search (highlighted with a red box), Metadata, Datastreams (selected), Events, Automations, Web Dashboard, and Mobile Dashboard. The main area has a title 'Temperatura e Umidade' and tabs for Info, Metadata, Datastreams, Events, Automations, Web Dashboard, and Mobile Dashboard. A search bar 'Search datastream' and a 'New Datastream' button are visible. Below is a table titled '2 Datastreams' with columns: Id, Name, Alias, Color, Pin, Data Type, Units, Is Raw, Min, Max, and Actions. Two rows are listed: '1 Temperatura Temperatura V6 Integer false 0 1' and '2 Umidade Umidade V0 Integer false 0 1'. The 'Datastreams' tab is highlighted.

Ao clicar sobre a lupa, abrirá uma nova janela. Clique sobre **New Device**.



Selecione **From Template**.



Em **Template**, escolha o criando anteriormente. Após, clique em **Create**.

New Device

Create new device by filling in the form below

TEMPLATE

Temperatura e Umidade

DEVICE NAME

Temperatura e Umidade

Cancel Create

Em **Device Info** será apresentado informações importantes na hora da programação !!

The screenshot shows the 'Device Info' tab selected in the Blynk interface. Key information displayed includes:

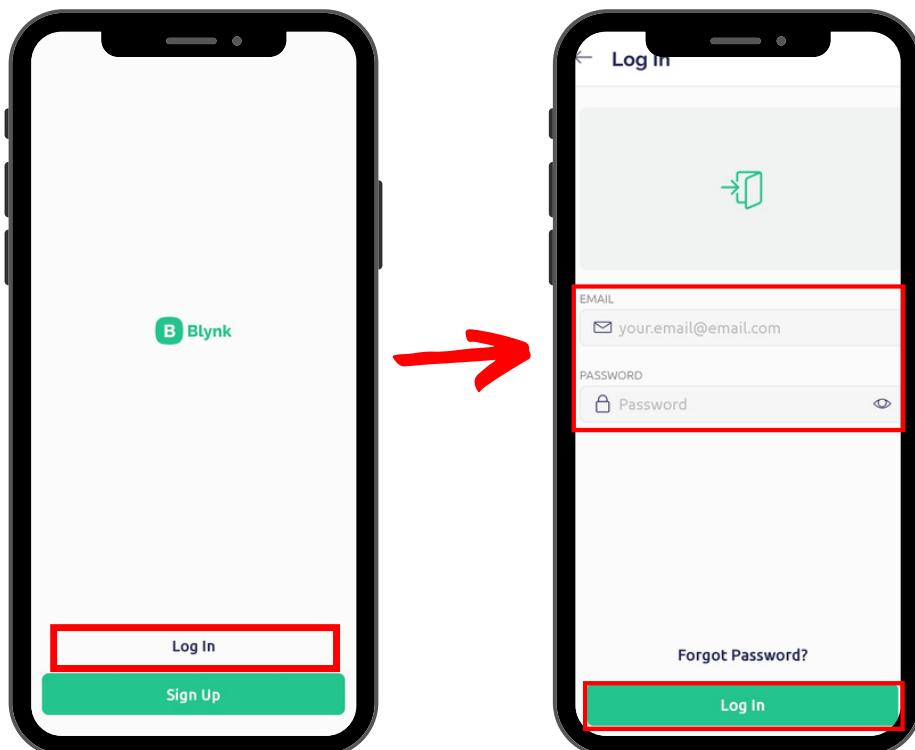
- STATUS:** Offline
- LAST UPDATED:** 4:31 PM Today
- DEVICE ACTIVATED:** 4:31 PM Today by rangelarena@gmail.com
- AUHTOKEN:** GkZ - **** - **** - ****
- MANUFACTURER:** (empty)
- SSL:** No SSL
- BOARD TYPE:** ESP8266

In the 'FIRMWARE CONFIGURATION' section, the following code is shown:

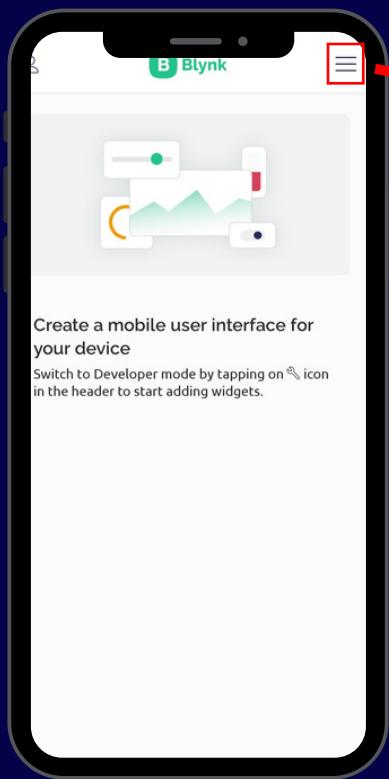
```
#define BLYNK_TEMPLATE_ID "Tn"      ip"
#define BLYNK_DEVICE_NAME "Temperatura e Umidade"
#define BLYNK_AUTH_TOKEN "GkZ - **** - **** - ****";"
```

A note below the code states: "Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code."

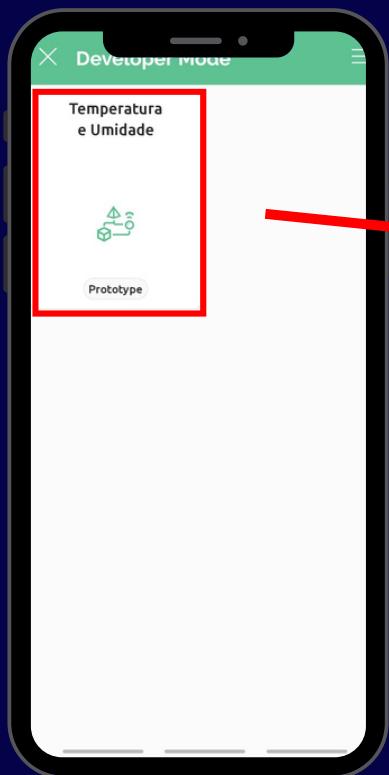
Configurando o Blynk no celular



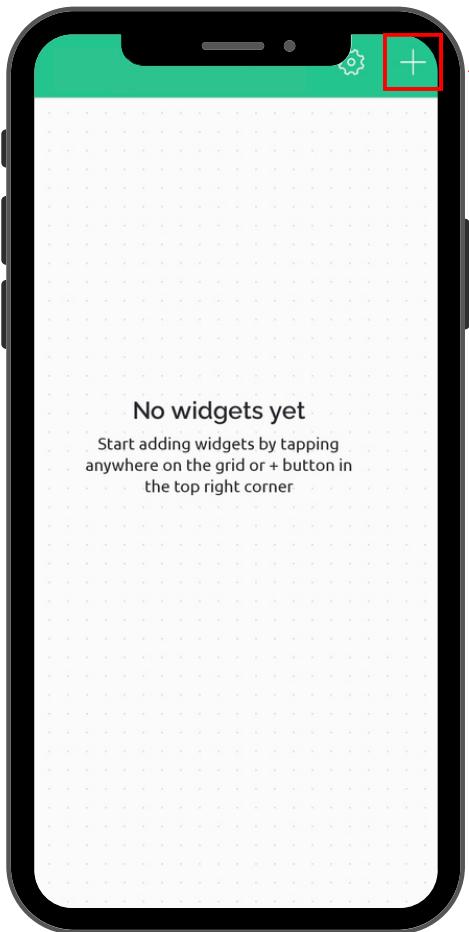
Abra o aplicativo baixado anteriormente, faça login com o mesmo **e-mail** e **senha** usado no site.



Na página inicial, clique sobre as **três barras** e depois selecione a opção **Developer Mode**.

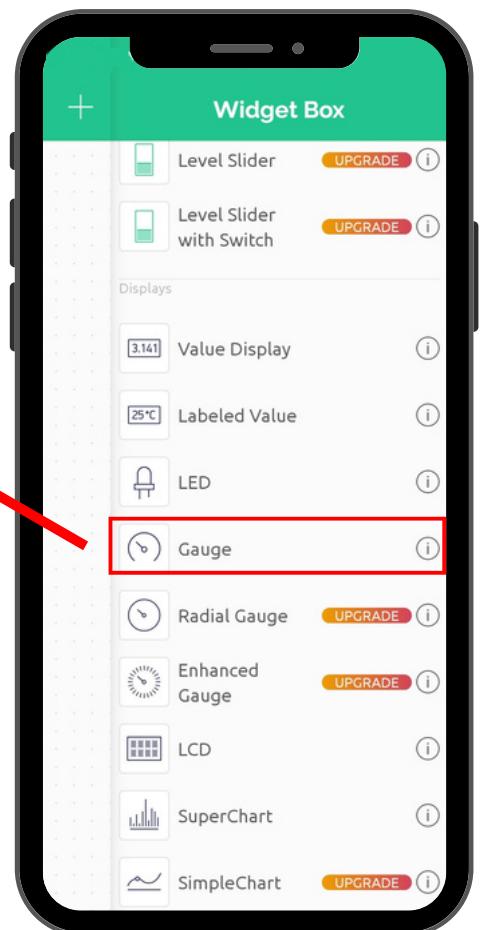


Selecione o **projeto** que criamos pelo **site**.



Clique sobre o **+** para adicionar os **Medidores**.

Adicione dois Medidores (**Gauge**), um para monitorarmos a **temperatura**, e outro a **umidade** do ambiente.

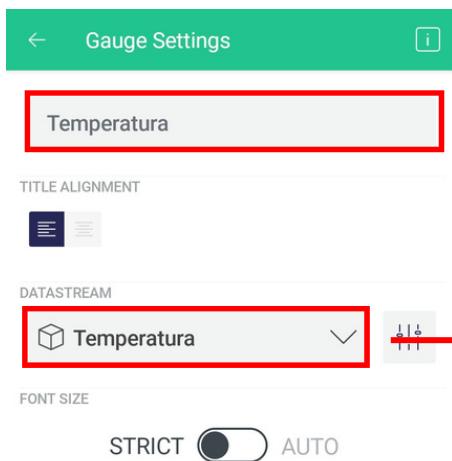


Configurando os Medidores



Medidor de Temperatura

Para iniciar a configuração, basta clicar sobre o **medidor**.

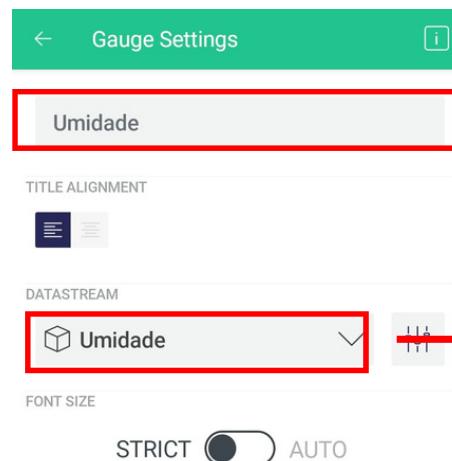


Coloque o **nome** no medidor de **temperatura**.

Selecione o **pino virtual** da **temperatura** (definimos pelo **site**).



Medidor de Umidade

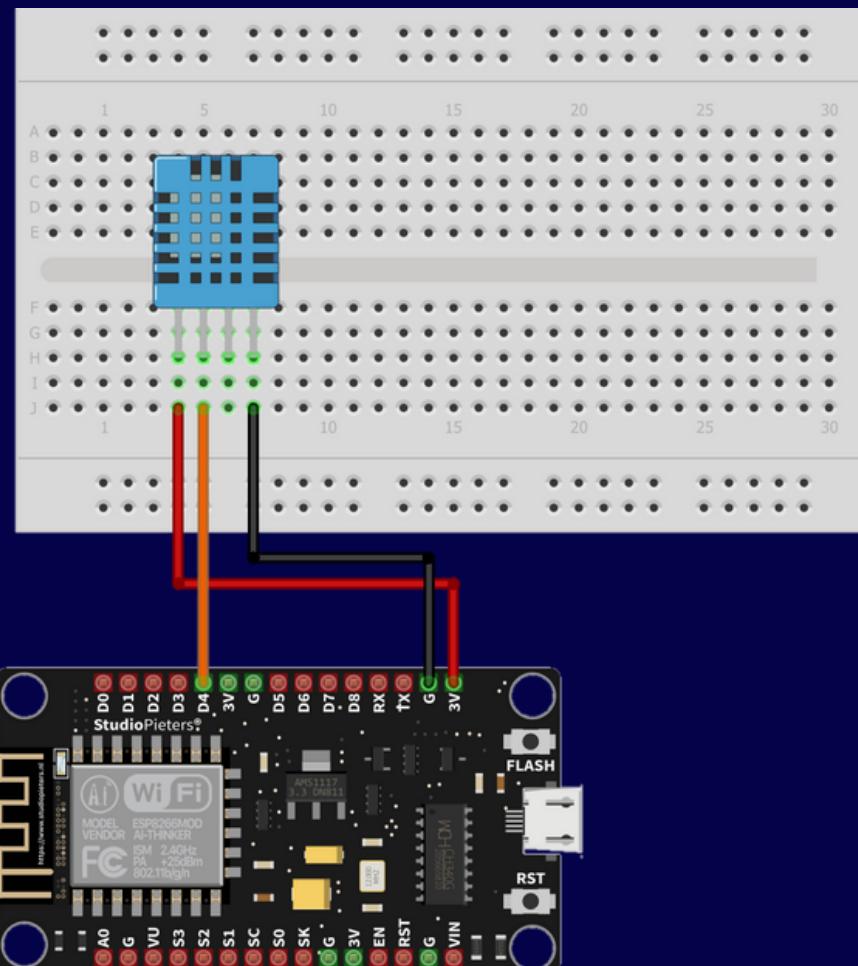


Coloque o **nome** no medidor de **Umidade**.

Selecione o **pino virtual** da **temperatura** (definimos pelo **site**).

Após finalizar as **configurações**, basta **salvar** e voltar para a **página inicial** do **Blynk** e selecionar o **projeto**.

Circuito



O primeiro pino do sensor é utilizado para alimentação, conecte ele nos **3.3V** do **NodeMCU**.

Conecte o segundo pino do **DTH11** no pino **D4** da nossa placa, ele é responsável por enviar a **leitura** da temperatura e da umidade.

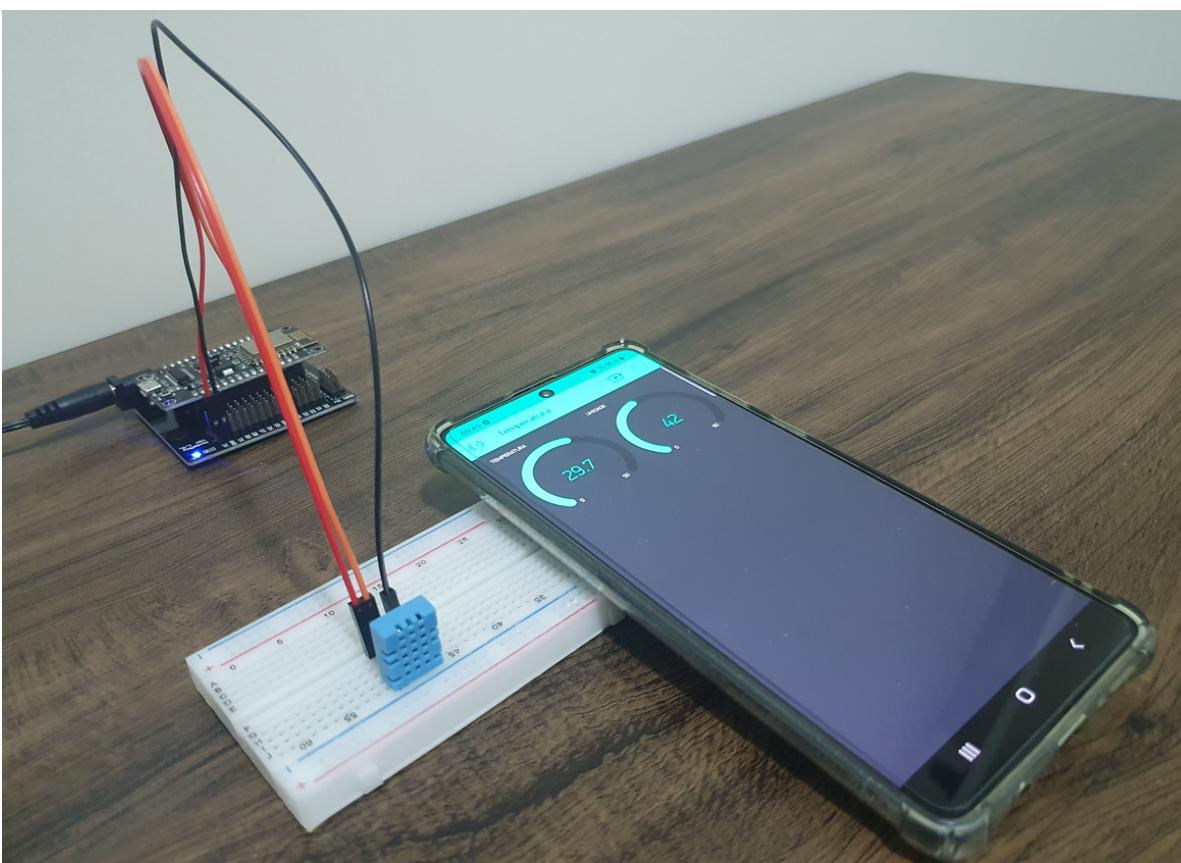
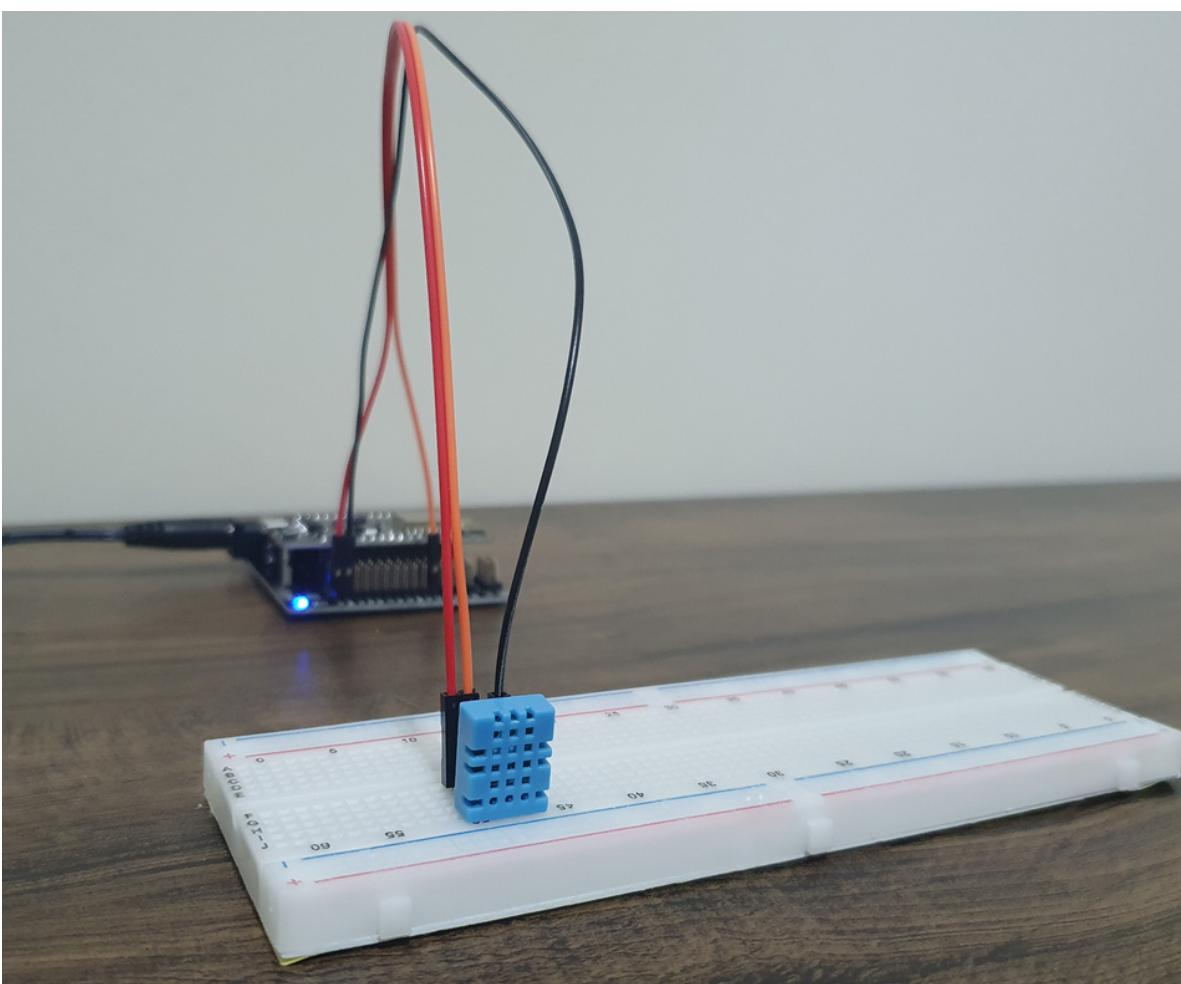
O terceiro pino não será utilizado.

Por fim, conecte o ultimo pino no pino **GND** da placa.

Resumindo

Pino 1 ---- 3V
Pino 2 ---- D4
Pino 3 ---- -
Pino 4 ---- GND

Círcito na Prática



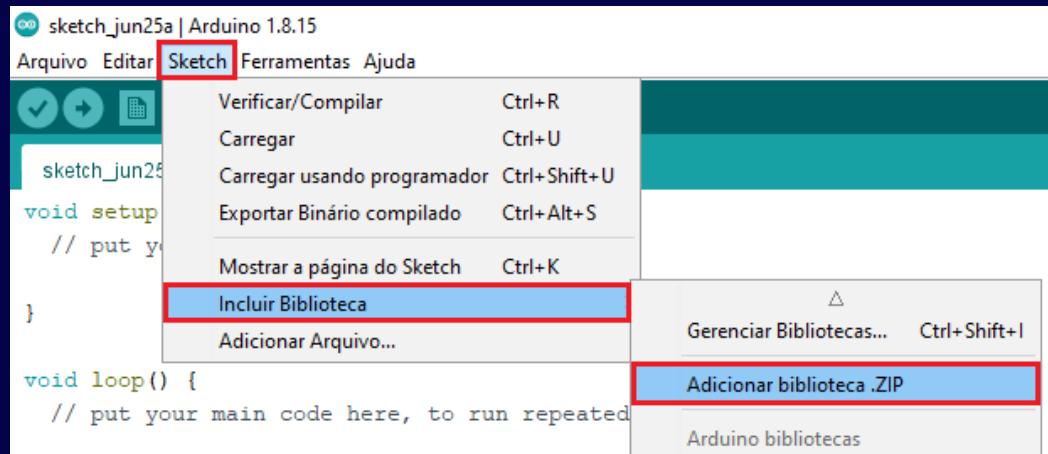
Instalando Bibliotecas

Para conseguirmos programar nosso sensor, é necessário adicionarmos a biblioteca “**DHT.h**”, você pode baixá-la [Clicando Aqui](#).

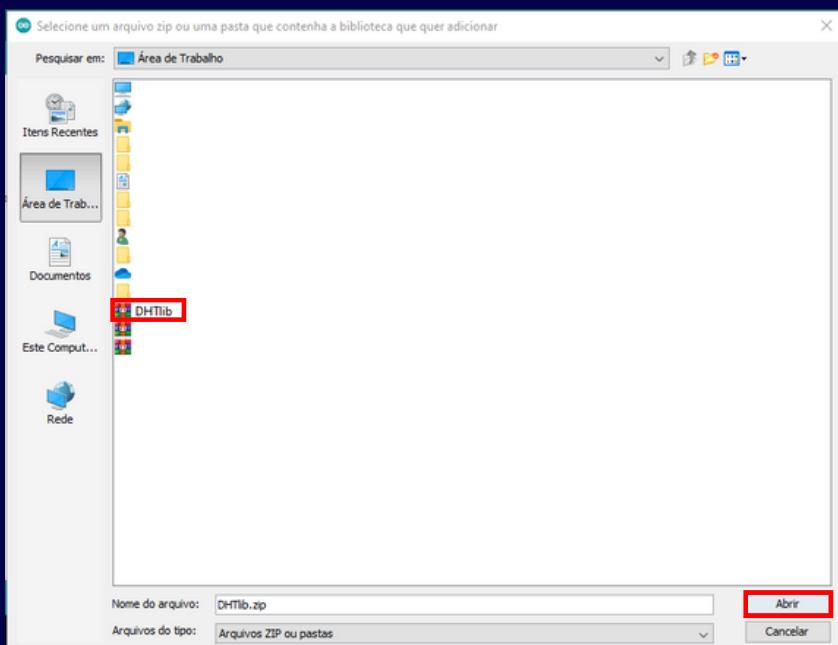
Para Adicionar a **biblioteca** no Arduino, **siga o passo a passo abaixo:**



Abra a **IDE do Arduino**, clique em **Sketch**, selecione **Incluir Biblioteca** e depois clique em **Adicionar Biblioteca.zip**.



Selecione o arquivo **.zip** e clique em **Abrir** e pronto, a biblioteca necessária já está instalada.



Programação

```
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "ID do Template"
#define BLYNK_DEVICE_NAME "Nome do projeto"

// Biblioteca necessárias

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>

char auth[] = "Insira o Token"; // Aqui é necessário inserir o token mostrado no site
char ssid[] = "Insira o nome da Rede"; // Insira o nome da rede Wi-fi utilizada
char pass[] = "Insira a senha da Rede"; // Insira a senha da rede Wi-fi utilizada

#define DHTPIN 2 // Aqui é o pino digital que estamos utilizando, no nosso caso D4 (GPIO 2)

#define DHTTYPE DHT11 // Declarando o sensor

DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;

// Esta função envia o tempo de atividade do Arduino a cada segundo para o Pino Virtual (5).
// você também pode definir com que frequência enviar dados para o aplicativo Blynk.

void sendSensor() {
```

```

float h = dht.readHumidity();
float t = dht.readTemperature();

if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
}

Blynk.virtualWrite(V5, h); // Pino Virtual 5 para umidade
Blynk.virtualWrite(V6, t); // Pino Virtual 6 para temperatura
}

void setup()
{
    Serial.begin(9600);

    Blynk.begin(auth, ssid, pass);

    dht.begin();

    timer.setInterval(1000L, sendSensor);
}
void loop()
{
    Blynk.run();
    timer.run();
}

```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento na prática ?? [Clique aqui!!](#)





Alerta de Chuva

Nesse próximo projeto, vamos fazer um **sistema de alerta de chuva**. Para isso, vamos utilizar um **módulo sensor de chuva**, nossa placa **NodeMCU** e o aplicativo **Blynk**. Nosso projeto funcionará da seguinte maneira: sempre que começar a chover, uma **notificação** aparecerá no celular alertando.

Módulo Sensor de Chuva

Este **Sensor de Chuva** pode ser usado para monitorar uma variedade de condições climáticas como gotas de chuva ou neve. Quando o clima está seco a saída do sensor fica em estado **alto** e quando há uma gota de chuva, a saída fica em estado **baixo**. O limite entre tempo seco e chuva pode ser ajustado através do potenciômetro presente no sensor que regulará a saída digital D0.

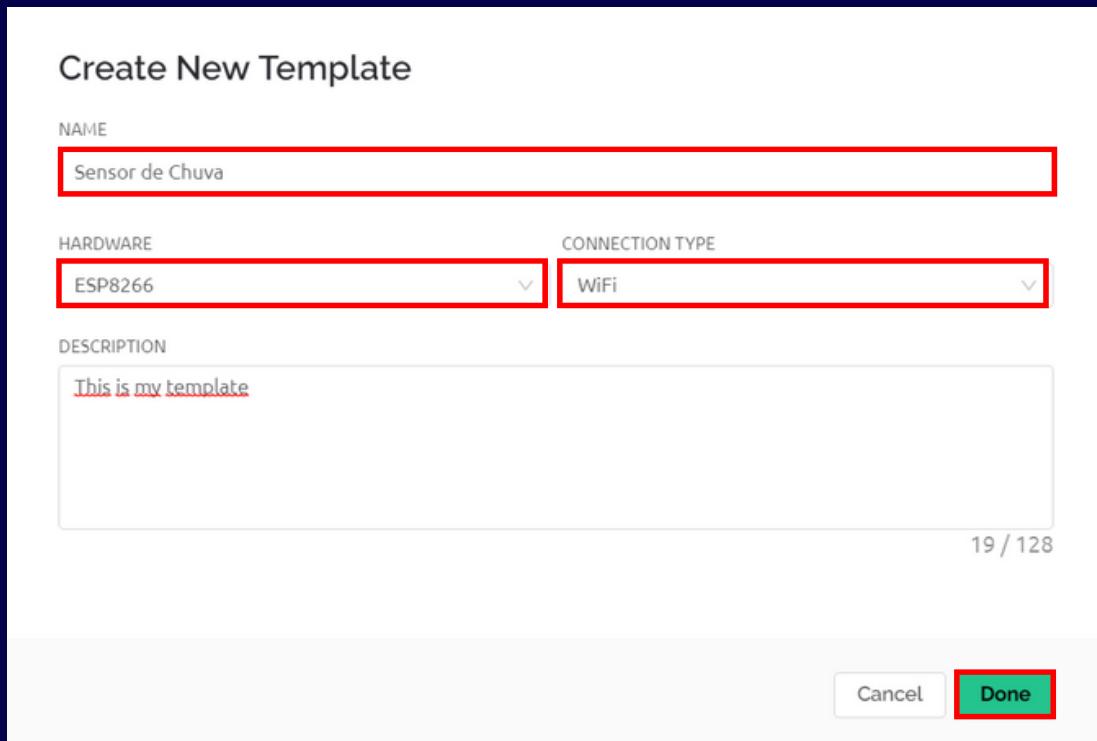


A placa do **Sensor de Chuva** é revestida em ambos os lados com um tratamento de **níquel contra oxidação**, melhorando assim a condutividade, desempenho e duração.

Configurando o Blynk

Primeiro vamos criar um novo **template** pelo site do [Blynk](#). Se houver dúvidas de como criar, siga o passo a passo nos projetos acima.

Adicione um nome para o **template**, em **Hardware** selecione **ESP8266**, em **Connection Type** selecione **WiFi**. Após concluir clique em **Done**.



Depois, clique em **Datastreams**, em seguida **New Datastream** e escolha a opção **Virtual Pin**. Vamos iniciar configurando o **pino virtual** para o **Sensor de Chuva**.



Essa mensagem aparecerá enquanto o celular se conecta ao nosso **hardware**.

Após finalizar as configurações do pino virtual, clique em **Events** e depois em **Add New Event**, para adicionarmos um novo evento.

The screenshot shows the 'Sensor de Chuva' configuration page. At the top, there are tabs for Info, Metadata, Datastreams, **Events** (which is highlighted with a red box), Automations, Web Dashboard, and Mobile Dashboard. Below the tabs is a search bar labeled 'Search event'. To the right of the search bar is a green button with a plus sign and the text '+ Add New Event' (also highlighted with a red box). On the far right, there are 'Cancel' and 'Save And Apply' buttons. The main area displays a table with columns: Name, Code, Color, Type, Description, and Actions. Two rows are shown: 'Online' (Code: ONLINE, Color: Green, Type: Online) and 'Offline' (Code: OFFLINE, Color: Red, Type: Offline). A vertical sidebar on the left contains icons for search, refresh, and other settings.

Em **General**, de um nome ao evento, em **Type** selecione a opção **Warning**. Deixe ativado **Send event to Notifications tab** e **Send event to Timeline**.

The screenshot shows the 'Edit Event' dialog. The 'General' tab is selected (highlighted with a green underline). In the 'EVENT NAME' field, 'Chuva' is entered (highlighted with a red box). In the 'EVENT CODE' field, 'chuva_' is entered. Under 'TYPE', the 'Warning' option is selected (highlighted with a red box). In the 'DESCRIPTION (OPTIONAL)' field, 'Começou a chover' is written. At the bottom, two checkboxes are shown: 'Send event to Notifications tab' (checked) and 'Send event to Timeline' (checked). Both checkboxes have descriptions below them: 'Make event visible in the notifications tab in the mobile app' and 'Make event visible in the Timeline'. A red box highlights the 'Send event to Notifications tab' checkbox and its description.

Após concluir as configurações em **General**, clique sobre **Notifications**.

Ative a opção **Enable notifications**. Em **Default recipients** na opção **E-mail to** e **Push notifications to** selecione a opção **Device Owner**. Em **Notifications limit**, selecione **1 minuto** em **Limit Period** e **1** em **Event Counter**. Depois, basta salvar.

Enable notifications

Default recipients

E-MAIL TO
Device Owner X

PUSH NOTIFICATIONS TO
Device Owner X

SMS TO
Select contact

Notifications limit

LIMIT PERIOD
1 minute ▾

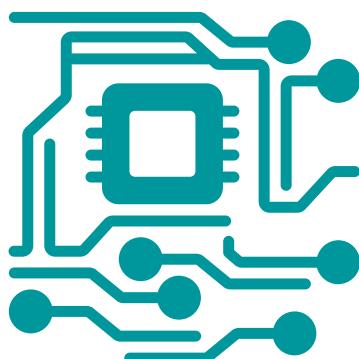
EVENT COUNTER
1

Notifications Management
When turned ON, end-users will access advanced notification management for this event

Enable notifications management

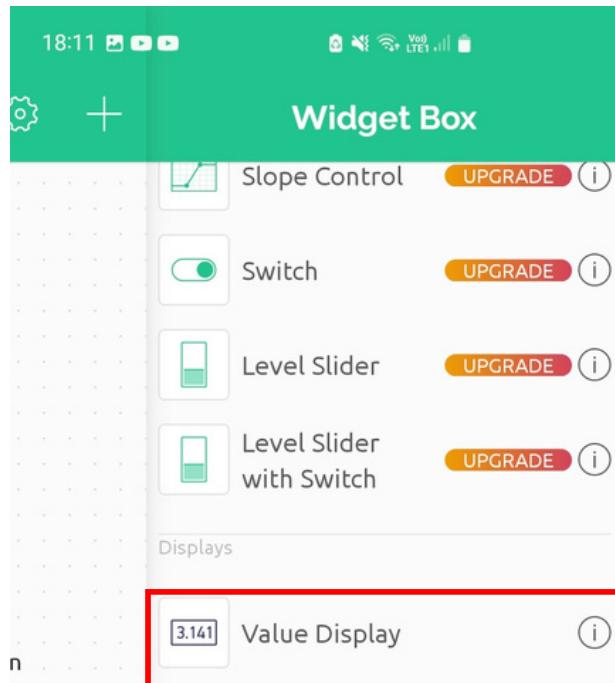
Cancel Save

Após finalizadas as configurações, clique sobre a lupa e depois sobre **New Device**. Selecione a opção **From Template** e escolha o projeto que acabamos de criar. Em **Device Info** é possível ver informações **necessárias** para a programação. Se causo houver dúvida siga o passo a passo na página 37.



Configurando o Blynk no celular

Primeiro é necessário abrir o aplicativo do **Blynk**. Após entre no **modo de desenvolvedor** e selecione o projeto do **sensor de chuva**. Vamos precisar de um **display**.

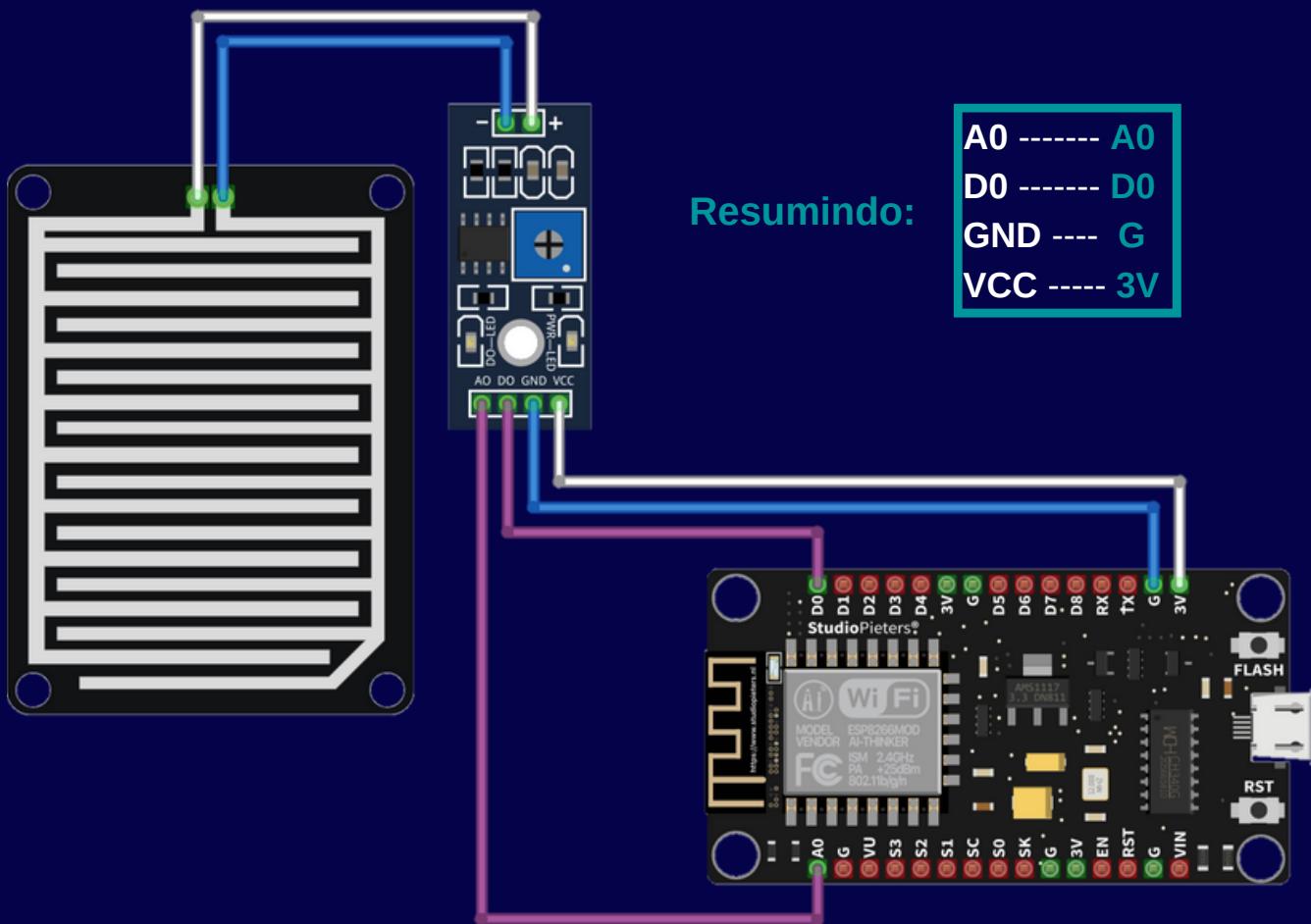


A screenshot of the 'Value Display Settings' screen. It includes fields for 'Title (optional)', 'TITLE ALIGNMENT' (with two icons), 'DATASTREAM' (set to 'Chuva'), 'FONT SIZE' (with 'STRICT' and 'AUTO' options and a slider set to 'Medium'), and 'DESIGN' (with a 'TEXT' icon).

Para configura-lo basta somente em **Datastream** selecionar o pino virtual que criamos pelo site.

Após realizar a **configuração**, basta salvar e voltar para a página inicial do **App**.

Círcuito



Primeiro é necessário conectar dois jumper no sensor e conectar na **placa de controle**, no negativo e positivo.

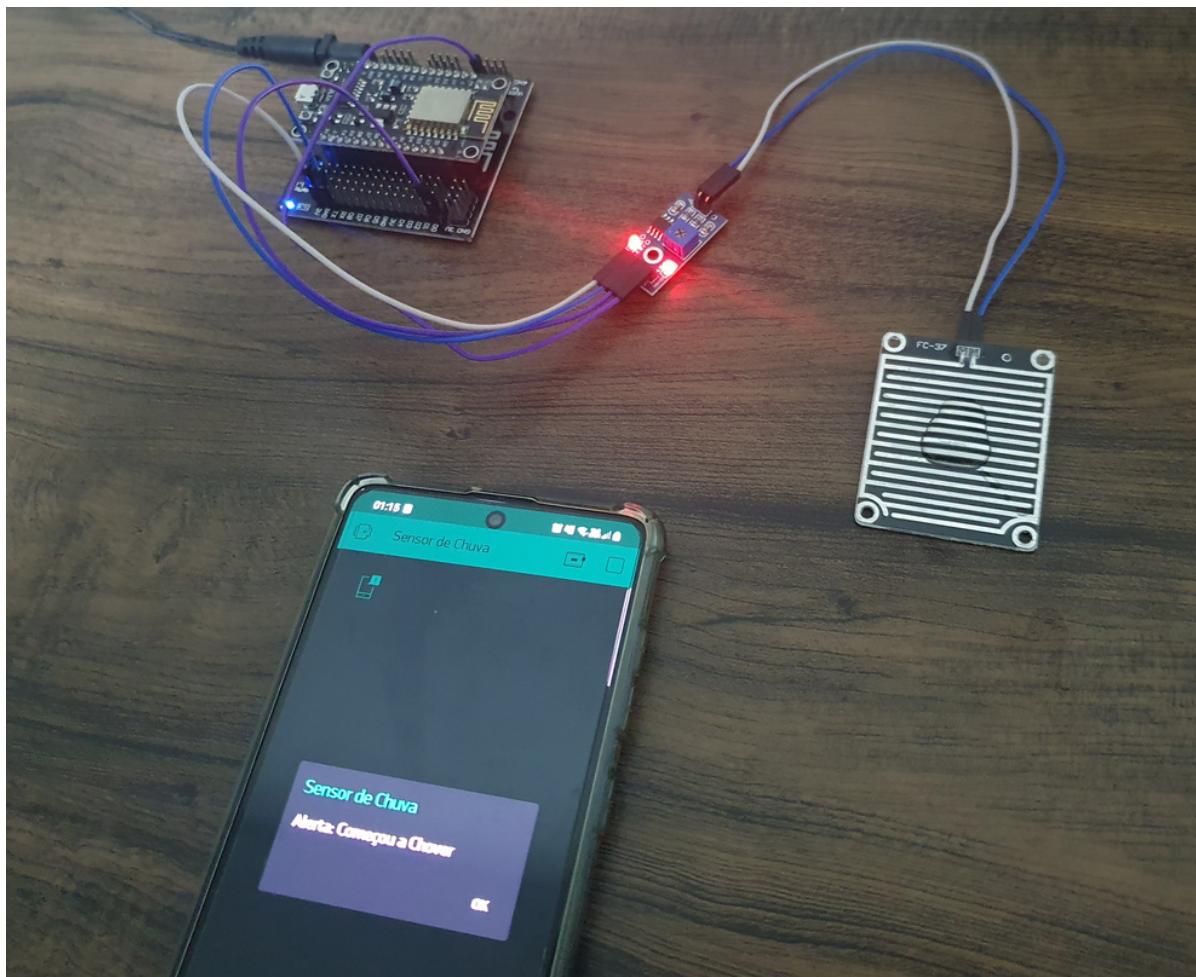
Conecte o pino analógico **(A0)** da **placa de controle** no pino analógico do **NodeMCU (A0)**.

Conecte o pino digital **(D0)** da **placa de controle** no pino analógico do **NodeMCU (D0)**.

O pino **GND** da **placa de controle** deve ser conectado no pino **G** do microcontrolador.

Por fim, é necessário conectar o pino **VCC** da **placa de controle** no pino **3V** do **NodeMCU**.

Círcito na Prática



Programação

```
#define BLYNK_TEMPLATE_ID "Insira o template ID"
#define BLYNK_DEVICE_NAME "Insira o Nome do Projeto"
#define BLYNK_AUTH_TOKEN "Insira o Token"

char ssid[] = "Nome da Rede";
char pass[] = "senha";

#define RAIN_SENSOR 16

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
BlynkTimer timer;

int RAIN_SENSOR_Value = 0;
bool isconnected = false;
char auth[] = BLYNK_AUTH_TOKEN;

#define VPIN_BUTTON_2 V0

void checkBlynkStatus() {
    isconnected = Blynk.connected();
    if (isconnected == true) {

        sendData();

    }
    else{

        Serial.println("Blynk Not Connected");
    }
}
```

```

void getSensorData()
{
    RAIN_SENSOR_Value = digitalRead(RAIN_SENSOR);
    if (RAIN_SENSOR_Value == 0 ){
    }
    else{
    }
}

void sendData()
{
    Blynk.logEvent("rain", "Water Detected!");
    Blynk.virtualWrite(VPIN_BUTTON_2, "Começou a Chover !!");

    if (RAIN_SENSOR_Value == 1 )
    {
        Blynk.virtualWrite(VPIN_BUTTON_2, "Não está chovendo.");
    }
}

void setup()
{
    Serial.begin(9600);
    pinMode(RAIN_SENSOR, INPUT);
    WiFi.begin(ssid, pass);
    timer.setInterval(2000L, checkBlynkStatus);
    Blynk.config(auth);
    delay(1000);
}

void loop()
{
    getSensorData();
    Blynk.run();
    timer.run();
}

```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento ?? [Clique aqui!!](#)



Nesse projeto continuaremos utilizando o aplicativo **Blynk**, agora vamos aprender como controlar um **Módulo Relé de 4 canais** e em breve, utilizando esse mesmo relé controlaremos **lâmpadas** e **ventiladores** !!

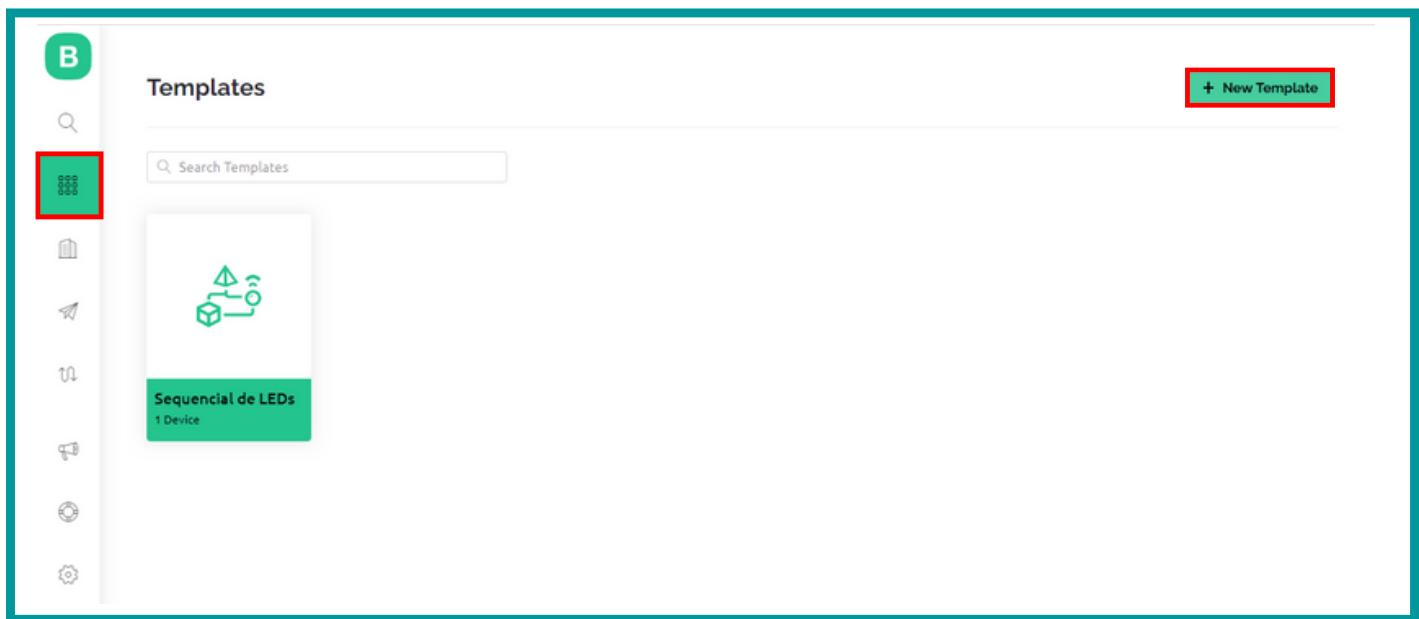
Módulo Relé

O **Módulo Relé** permite uma integração com a maioria dos **microcontroladores**. A partir das saídas digitais pode-se, através do relé, controlar cargas maiores e dispositivos como **motores AC ou DC**, **eletroímãs**, **solenoides** e **lâmpadas incandescentes**. Este módulo tem dois canais sendo assim concebido para ser integrado para controlar até **4 relés**. O módulo é equipado com um relé de **alta qualidade**, com carga nominal **10A/250VAC** e **10A/30VDC**.



Configurando o Blynk

Acesse o site blynk.io, clique sobre **Templates** e depois em **New Template**.



Nas configurações do novo **template**, adicione um **nome**, em **Hardware** selecione **ESP8266** e em **Connection Type** escolha **WiFi**. Após finalizar clique em **Done**.

A screenshot of the 'Create New Template' dialog box. It has fields for NAME (containing 'Relés'), HARDWARE (containing 'ESP8266'), CONNECTION TYPE (containing 'WiFi'), and DESCRIPTION (containing 'This is my template'). At the bottom right are 'Cancel' and 'Done' buttons, with 'Done' highlighted with a red box.

Para iniciar a **configuração** dos botões, clique sobre **Datastreams**, depois em **New Datastreams** e selecione a opção **Virtual Pin**.

The screenshot shows the Datastream configuration interface. At the top, there are tabs: Info, Metadata, **Datastreams** (which is highlighted with a red box), Events, Automations, Web Dashboard, and Mobile Dashboard. On the left, there is a sidebar with various icons. In the center, under the heading "Datastreams", it says: "Datastreams is a way to structure data that regularly flows in and out from device. Use it for sensor data, any telemetry, or actuators." Below this is a button labeled "+ New Datastream". A dropdown menu is open, showing options: Digital, Analog, **Virtual Pin** (which is highlighted with a red box), Enumerable, and Location. At the bottom right of the screen is a "Save" button.

Configurando os Botões

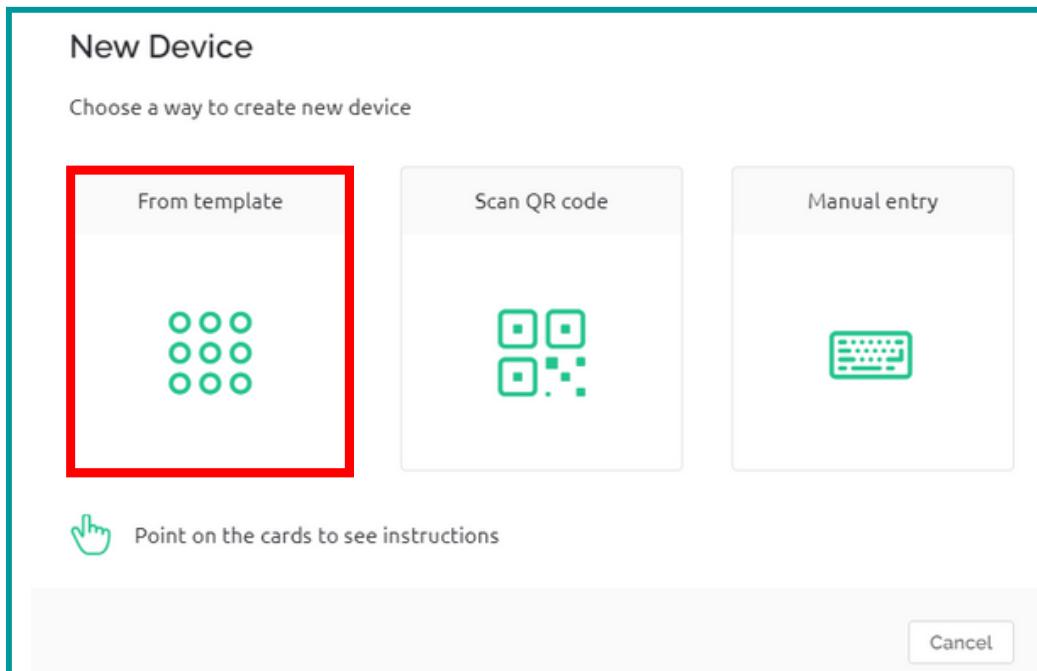
As configuração dos botões que vão ativar os relés é igual. Basta alteras os pinos virtuais. OBS: **Relé 1 = V0**, **Relé 2 = V1**, **Relé 3 = V2**, **Relé 4 = V3**.

The screenshot shows the "Virtual Pin Datastream" configuration dialog. It has fields for NAME (with "Relé 1" entered), ALIAS (with "Rel"), PIN (with "V0" selected), DATA TYPE (set to Integer), UNITS (None), MIN (0), MAX (1), and DEFAULT VALUE (0). There is also an "ADVANCED SETTINGS" button. At the bottom are "Cancel" and "Create" buttons. Red arrows and text annotations are present: one arrow points to the "NAME" field with the text "Adicione o nome"; another arrow points to the "PIN" field with the text "Seleciono o Pino Virtual."; a third arrow points to the "Create" button with the text "Após finalizar clique em Create".

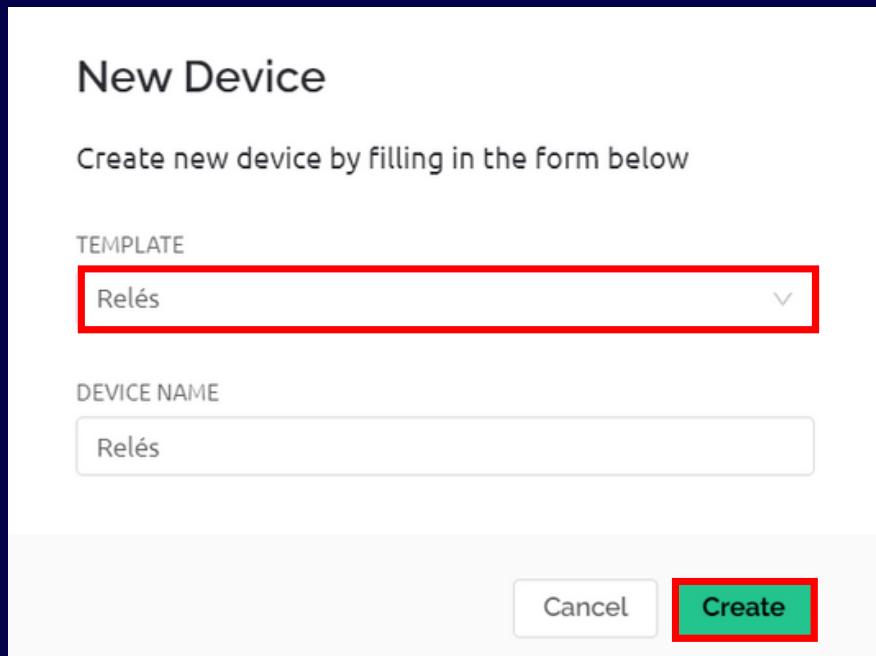
Após finalizar a **configuração** dos botões, clique sobre a **Lupa** e depois em **New Device**.

The screenshot shows the 'My Devices' section of a management interface. On the left, there's a sidebar with icons for organization, devices, locations, users, and settings. A red box highlights the search icon (magnifying glass) at the top of the sidebar. On the right, the main area is titled 'My Devices' and shows '1 Device'. There are filters for 'Device name', 'Device owner', 'Status', and 'Device model'. A red box highlights the '+ New Device' button in the top right corner.

Selecione a opção **From Template**.



Em **Template** selecione o projeto que criamos anteriormente. Depois clique em **Create**.



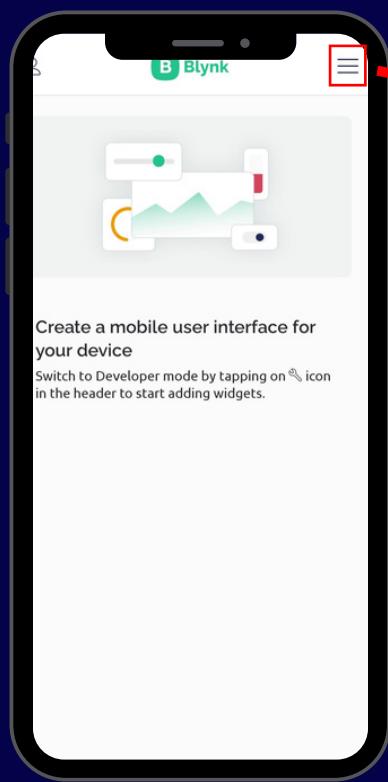
Em **Device Info** é possível ver **informações necessárias** na hora da programação.

The screenshot shows the 'Device Info' tab of a device configuration page. On the left is a sidebar with various icons. The main area shows device details: STATUS (Offline), LAST UPDATED (12:26 AM Today), DEVICE ACTIVATED (12:26 AM Today by rangelarena@gmail.com), AUTH TOKEN (MzGU - **** - **** - ****), MANUFACTURER (My organization 9402DL), and a FIRMWARE CONFIGURATION block. The FIRMWARE CONFIGURATION block contains the following code, with the entire block highlighted with a red box:

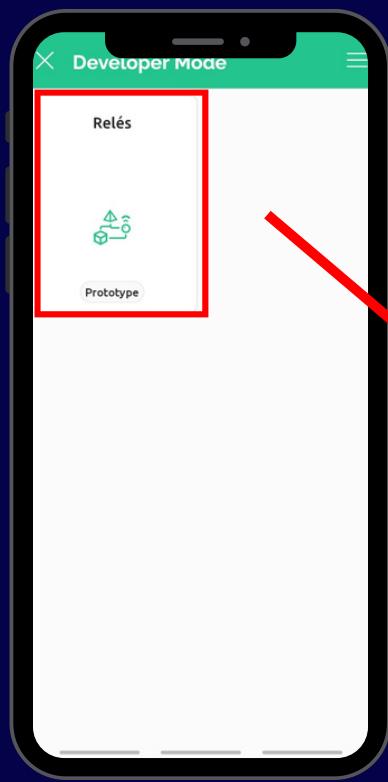
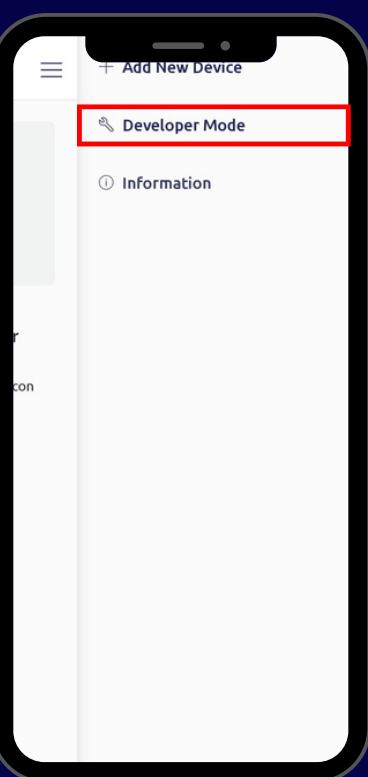
```
#define BLYNK_TEMPLATE_ID "T1"
#define BLYNK_DEVICE_NAME "Ventilador e Lâmpada"
#define BLYNK_AUTH_TOKEN
"Ma"
```

Below the configuration block, a note states: 'Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.'

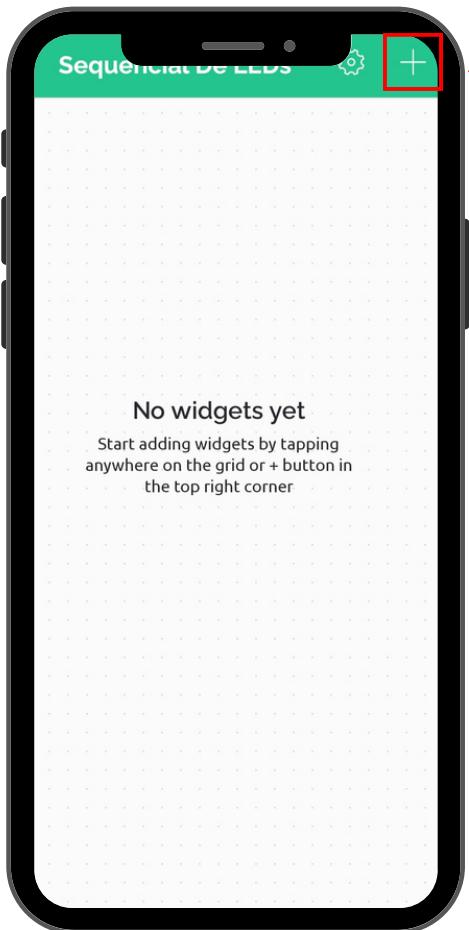
Configurando o Blynk no celular



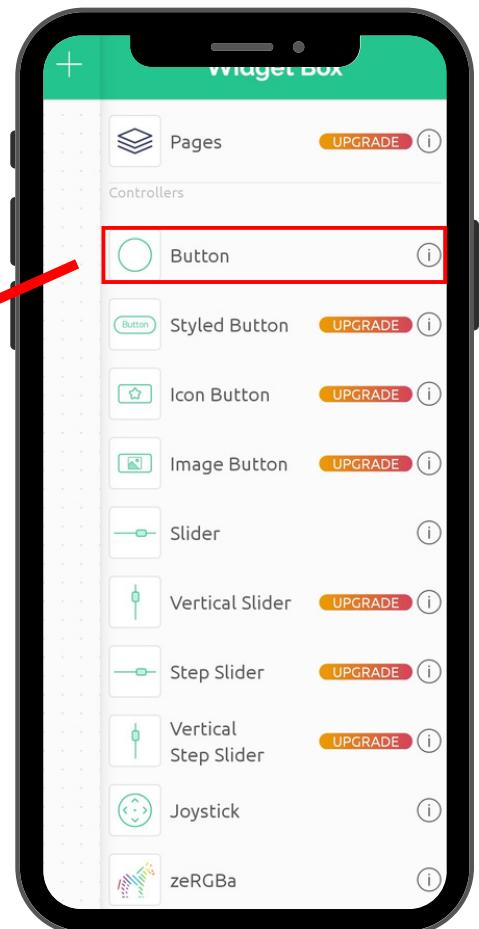
Na página inicial, clique sobre as **três barras** e depois selecione a opção **Developer Mode**.



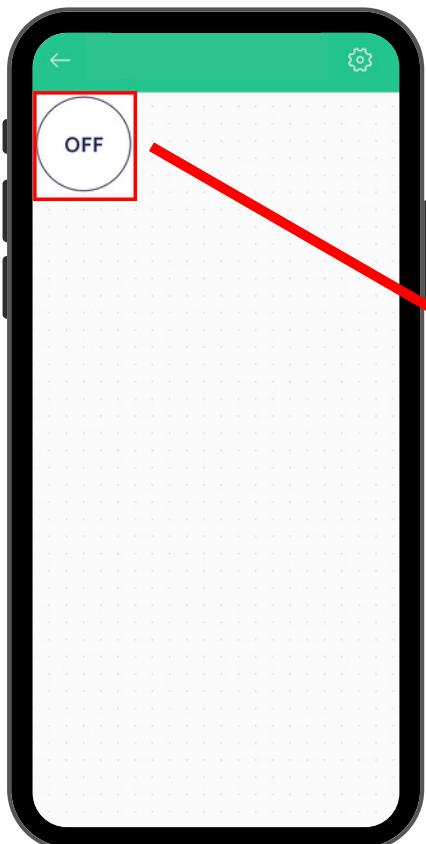
Selecione o **projeto** que criamos pelo **site**.



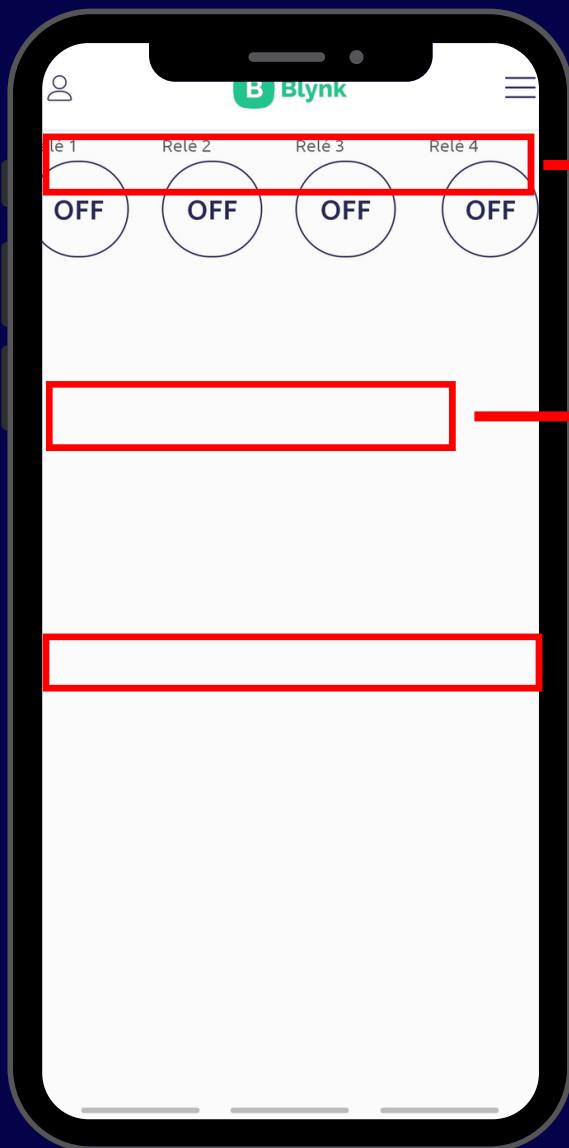
Clique sobre o **+** para adicionar os botões que farão com que os **LEDs** seja acessos.



Seleciona **Button**,
vamos precisar de
quatro.



Clique sobre o **botão** para realizar as
configurações necessárias.



Coloque o **nome** do botão.

Selecione o **Pino Virtual** que vai acender o **Relé** (já definimos ele pelo site).

Selecione a opção **Switch**, dessa forma um toque liga e outro desliga o **Relé**.

OBS: As configurações dos **relés** são iguais, basta alterar o nome e o **Pino Virtual**. Lembre-se:

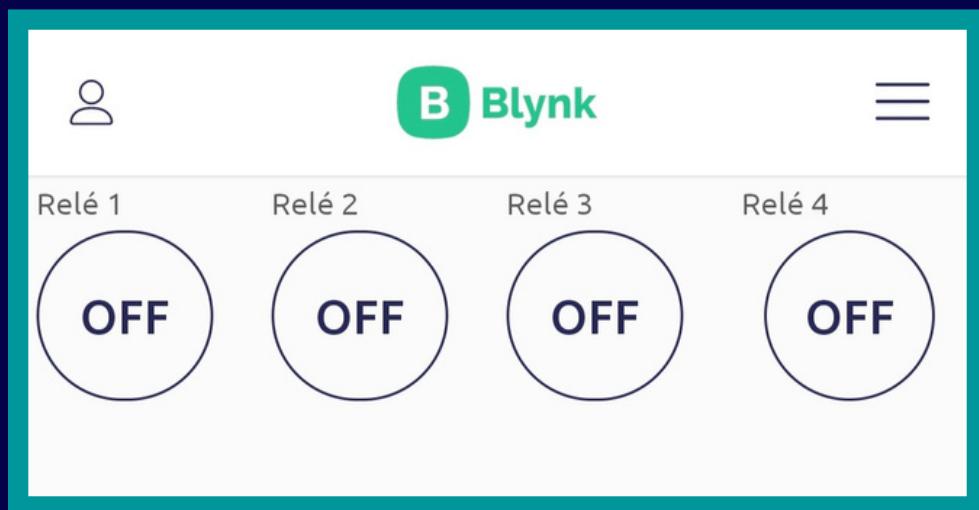
Relé 1: **V0**

Relé 2: **V1**

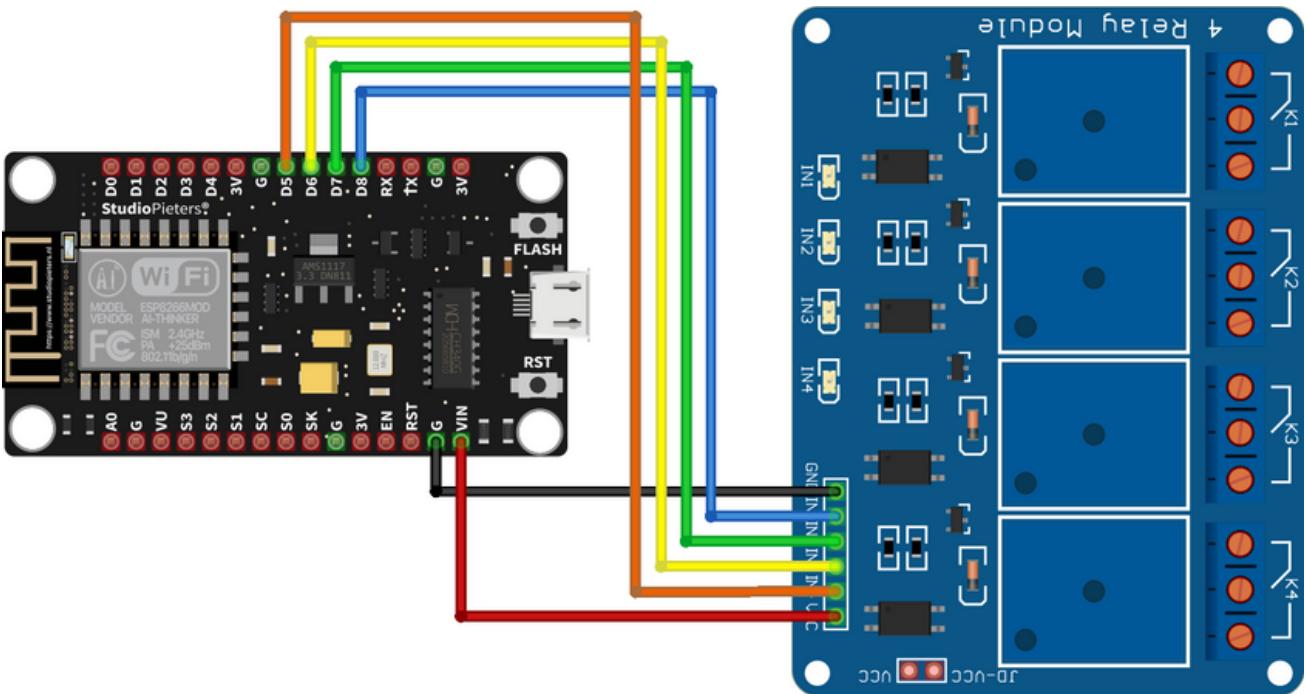
Relé 3: **V2**

Relé 4: **V3**

Após finalizar a configuração dos **botões**, basta voltar para a **tela inicial** do App.



Círcuito



É necessário conectar o pino **Vin** da placa no **VCC** do Relé, visto que ele precisa de uma tensão de **5V** para funcionar.

Conecte o pino **G** do **NodeMCU** no pino **GND** do **Módulo Relé**, esse é o **negativo** do circuito.

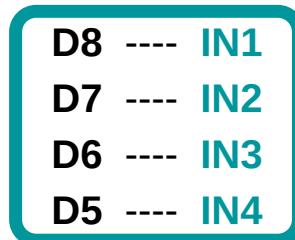
O **Pino Digital D8 (GPIO 15)** deve estar no **IN1** do Relé.

O **Pino Digital D7 (GPIO 13)** deve estar no **IN2** do Relé.

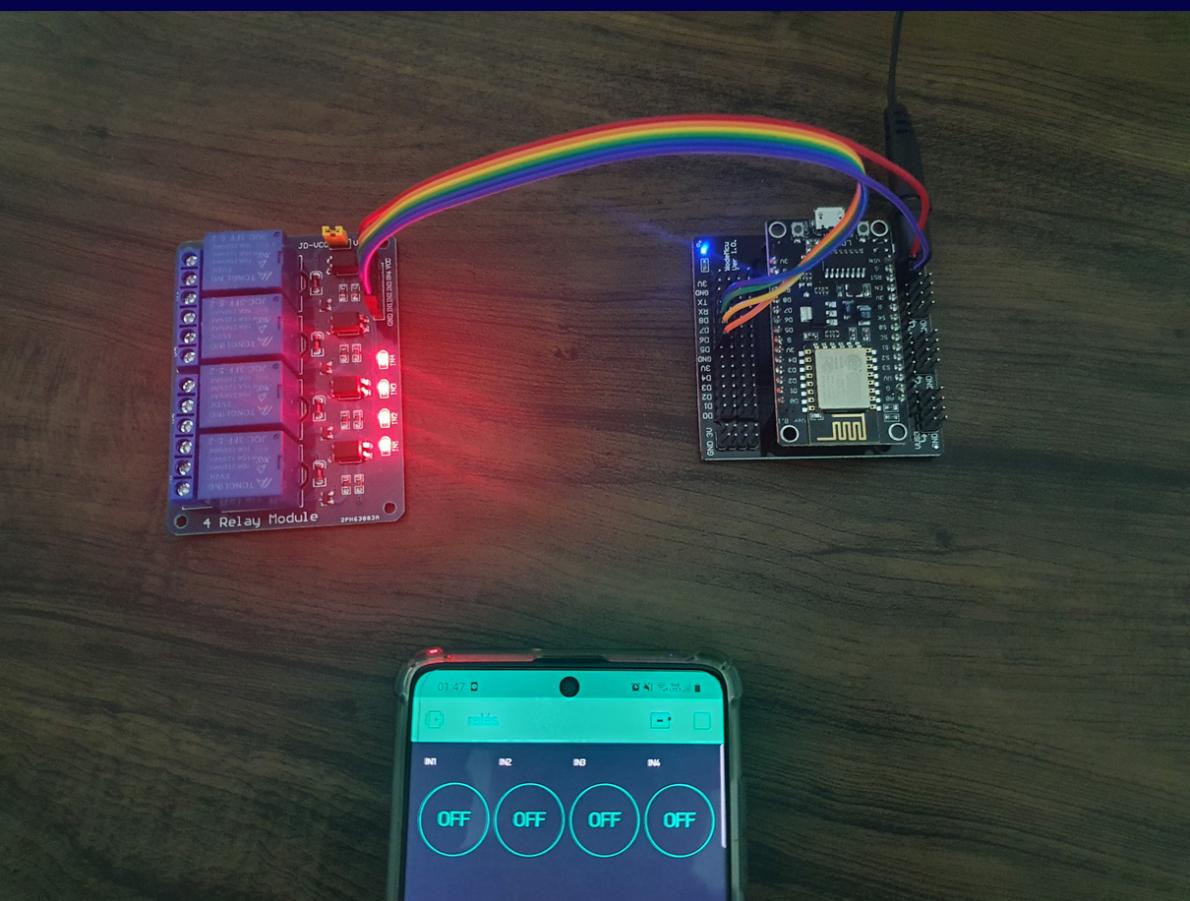
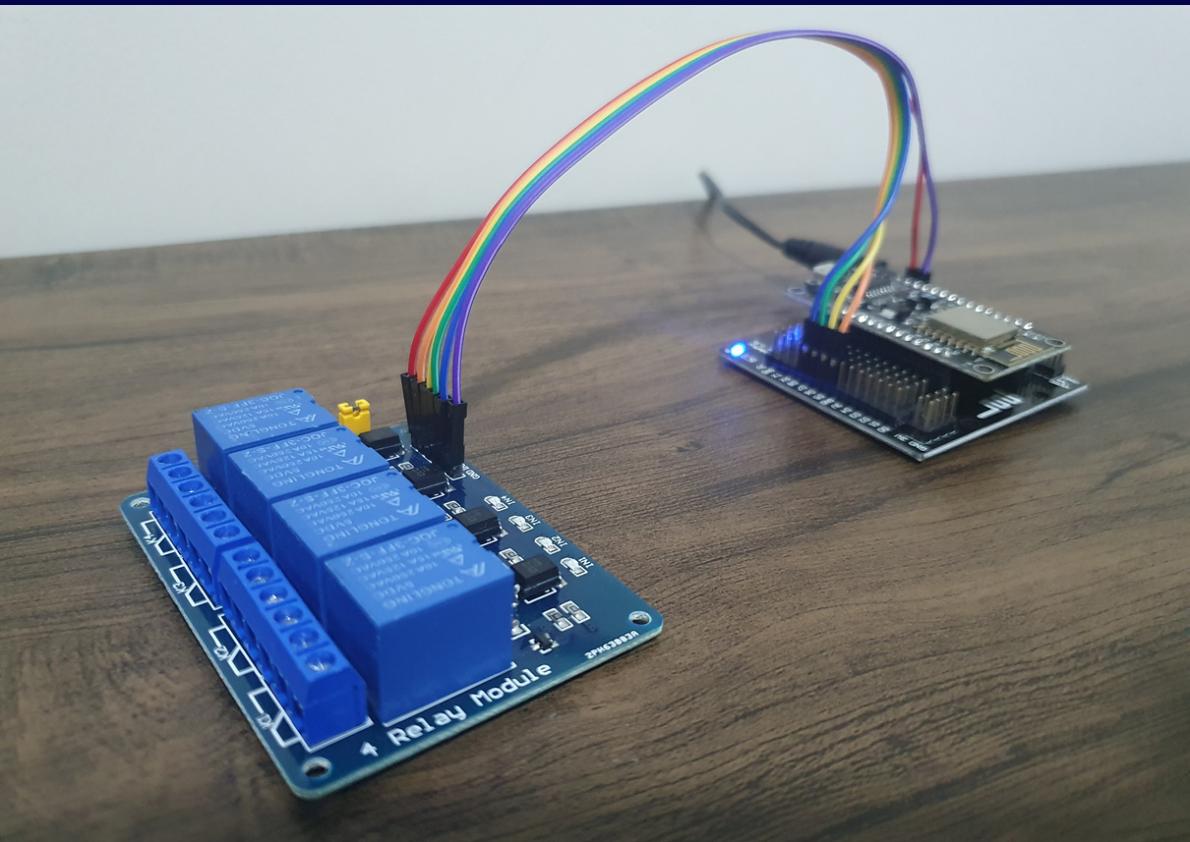
O **Pino Digital D6 (GPIO 12)** deve estar no **IN3** do Relé.

O **Pino Digital D5 (GPIO 14)** deve estar no **IN4** do Relé.

Resumindo:



Círcito na Prática



Programação

```
#define BLYNK_PRINT Serial
#define BLYNK_DEVICE_NAME "Nome do projeto"
#define BLYNK_TEMPLATE_ID "Insira o código do Template"

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = "Insira o Token";
char ssid[] = "Insira o nome da rede";
char pass[] = "Insira a senha da rede";

int rele1= D8;
int rele2= D7;
int rele3= D6;
int rele4= D5;

BLYNK_WRITE (V0){

int valor = param.asInt();
digitalWrite(rele1, valor);
}

BLYNK_WRITE (V1){

int valor = param.asInt();
digitalWrite(rele2, valor);

}
```

```
BLYNK_WRITE (V2){
```

```
    int valor = param.asInt();
    digitalWrite(rele3, valor);
}
```

```
BLYNK_WRITE (V3){
```

```
    int valor = param.asInt();
    digitalWrite(rele4, valor);
}
void setup()
{
    Serial.begin (115200);
    Blynk.begin(auth, ssid, pass);

    pinMode(rele1, OUTPUT);
    pinMode(rele2, OUTPUT);
    pinMode(rele3, OUTPUT);
    pinMode(rele4, OUTPUT);
}
void loop()
{
    Blynk.run();
}
```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento ?? [Clique aqui!!](#)



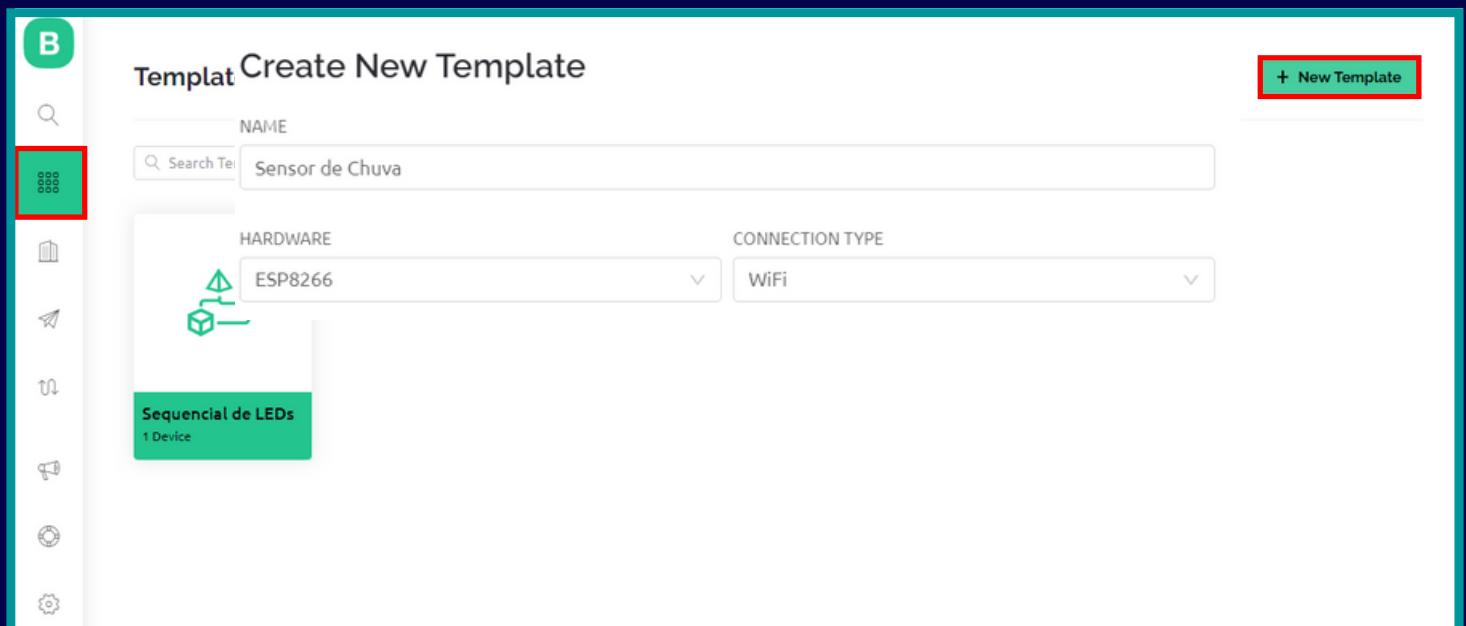
10

Ligando Lâmpada e Ventilador

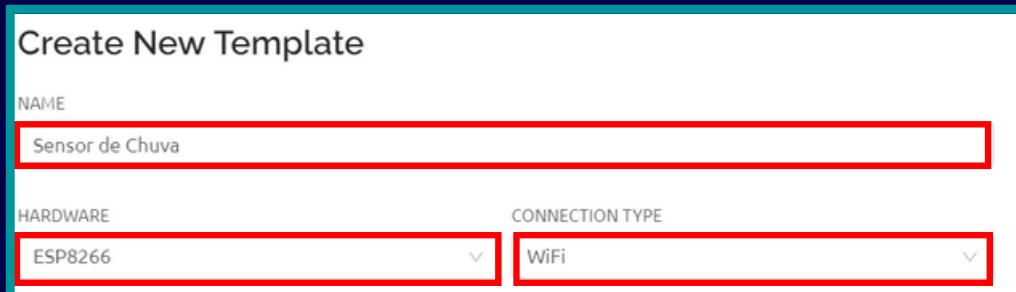
Vamos utilizar o **Relé** e o **Blynk** para acender e apagar lâmpadas e ligar um ventilador, não será necessário realizar grandes alterações no circuito e no aplicativo !!

Configurando o Blynk

Acesse o site blynk.io, clique sobre **Templates** e depois em **New Template**.



Nas configurações do novo **template**, adicione um **nome**, em **Hardware** selecione **ESP8266** e em **Connection Type** escolha **WiFi**. Após finalizar clique em **Done**.



Para iniciar a **configuração** dos botões, clique sobre **Datastreams**, depois em **New Datastreams** e selecione a opção **Virtual Pin**.

The screenshot shows a device configuration interface with a sidebar containing icons for basic settings like Info, Metadata, Events, Automations, Web Dashboard, and Mobile Dashboard. The main header is 'Ventilador e Lâmpada'. The 'Datastreams' tab is selected and highlighted with a red box. Below it, a modal window titled 'Datastreams' is displayed, containing the text: 'Datastreams is a way to structure data that regularly flows in and out from device. Use it for sensor data, any telemetry, or actuators.' It features a button '+ New Datastream' and a dropdown menu with options: Digital, Analog, Virtual Pin (which is also highlighted with a red box), Enumerable, and Location. There is also an 'UPGRADE' button.

Configurando os Botões

- Vamos iniciar realizando as **configurações** do **botão** que vai acionar o **ventilador**.

The screenshot shows the 'Virtual Pin Datastream' configuration dialog. On the left, there is a note 'Adicione o nome' with an arrow pointing to the 'NAME' field, which contains 'Ventilador'. Another note 'Selezione o pino V0' with an arrow points to the 'PIN' field, which contains 'V0'. At the bottom right, there is a note 'Após finalizar clique em Create.' with an arrow pointing to the 'Create' button. The dialog also includes fields for 'ALIAS' (set to 'Ventilador'), 'DATA TYPE' (set to 'Integer'), 'UNITS' (set to 'None'), 'MIN' (set to '0'), 'MAX' (set to '1'), and 'DEFAULT VALUE' (set to '0'). There is also an 'ADVANCED SETTINGS' button.

- Agora vamos **configurar** o **botão** que acenderá a **lâmpada**. Em **Name** escolha um nome e em **Pin** selecione **V1**. Após terminar a configuração clique em **Create**.

Virtual Pin Datastream

NAME	ALIAS	
<input type="text" value="Lâmpada"/> Lâmpada	<input type="text" value="L"/> L	
PIN	DATA TYPE	
<input type="text" value="V1"/> V1	<input type="text" value="Integer"/> Integer	
UNITS		
<input type="text" value="None"/> None		
MIN	MAX	DEFAULT VALUE
<input type="text" value="0"/> 0	<input type="text" value="1"/> 1	<input type="text" value="0"/> 0
+ ADVANCED SETTINGS		
		Cancel Create

Após finalizar a **configuração** dos botões, clique sobre a **Lupa** e depois em **New Device**.

My organization - 9402DL

My Devices

	Device name	Device owner	Status	Device model	Actions
	Sequencial de LEDs	Rangel	● Offline		 

+ New Device

DEVICES

- My Devices** 1
- All 1

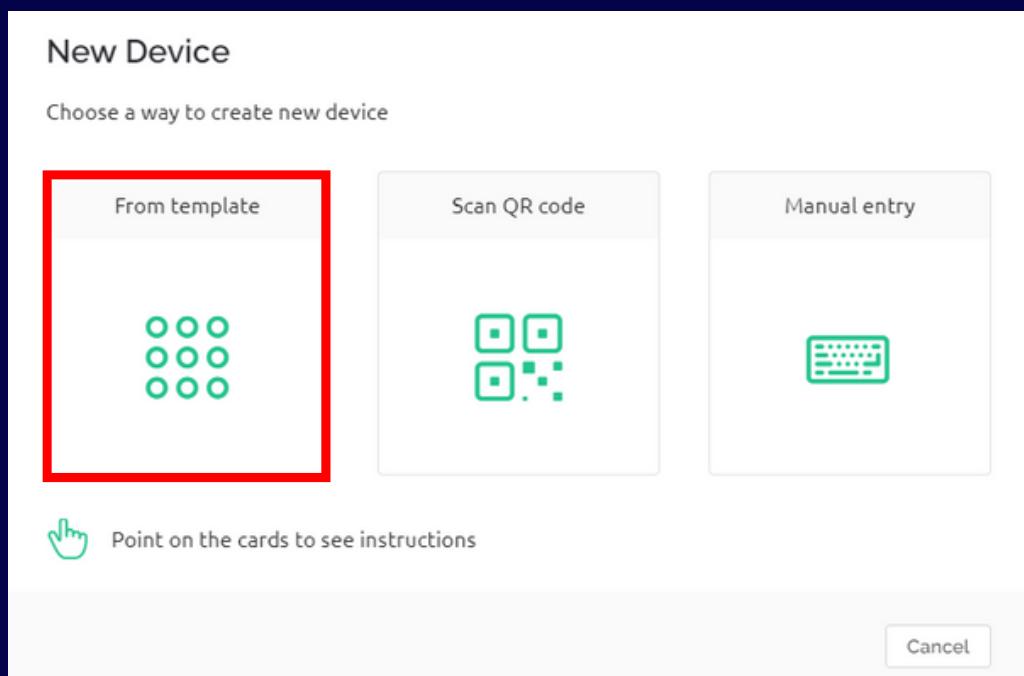
LOCATIONS

- My locations** 0
- All 0

USERS

- My organization members** 1
- All 1
- With no devices 0

Selecione a opção **From Template**.



Em **Template** selecione o projeto que criamos anteriormente. Depois clique em **Create**.

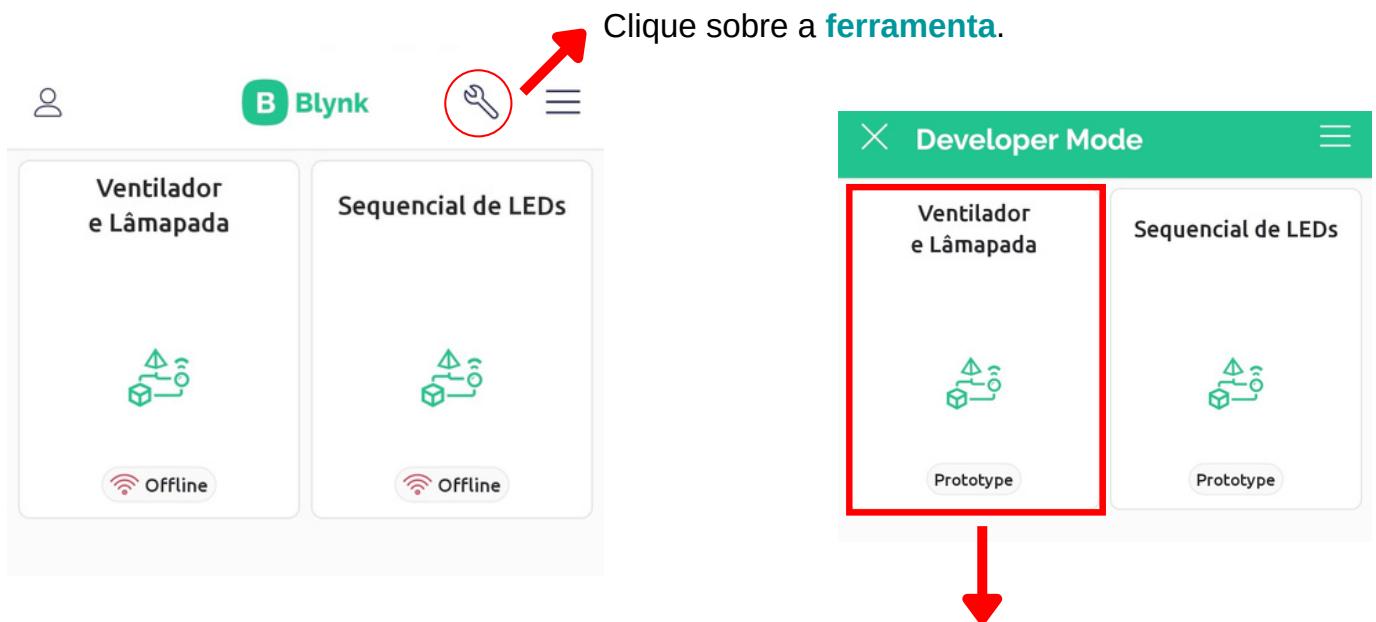
Em **Device Info** é possível ver **informações necessárias** na hora da programação.

The screenshot shows the Blynk web interface. On the left, there's a sidebar with icons for search, devices, and settings. The main area shows '2 Devices': 'Sequencial de LEDs' and 'Ventilador e Lâmapada'. The 'Ventilador e Lâmapada' card is selected. The right side displays 'Ventilador e Lâmapada' status as 'Offline'. Below it are sections for 'DEVICE ACTIVATED' (12:26 AM Today by rangelarena@gmail.com), 'AUTHTOKEN' (MzGU - **** - **** - ****), 'MANUFACTURER' (My organization 9402DL), and 'LAST UPDATED' (12:26 AM Today). A red box highlights the 'Device Info' tab. Another red box highlights the 'FIRMWARE CONFIGURATION' section, which contains the following code:

```
#define BLYNK_TEMPLATE_ID "T1"
#define BLYNK_DEVICE_NAME "Ventilador e Lâmapada"
#define BLYNK_AUTH_TOKEN
"Ma"
```

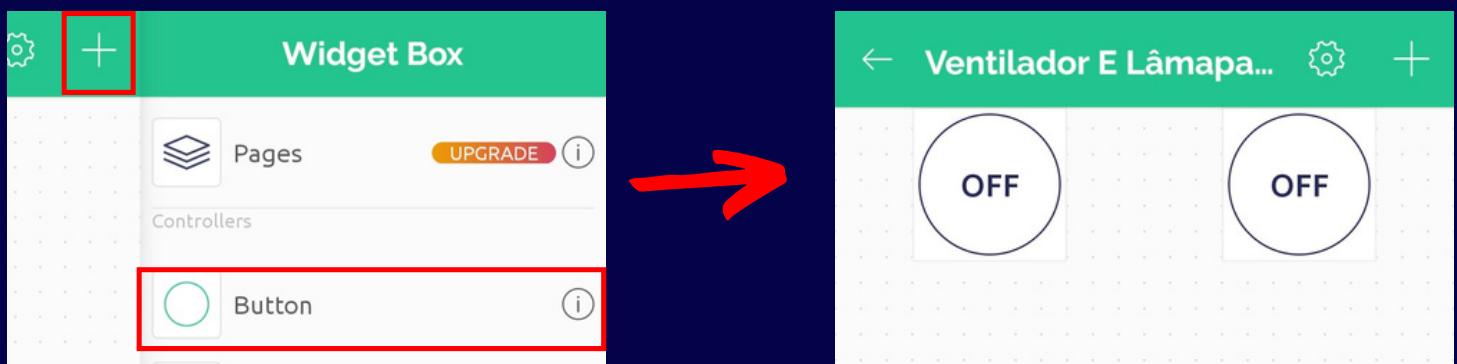
A note below states: 'Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.'

Configurando o Blynk no celular

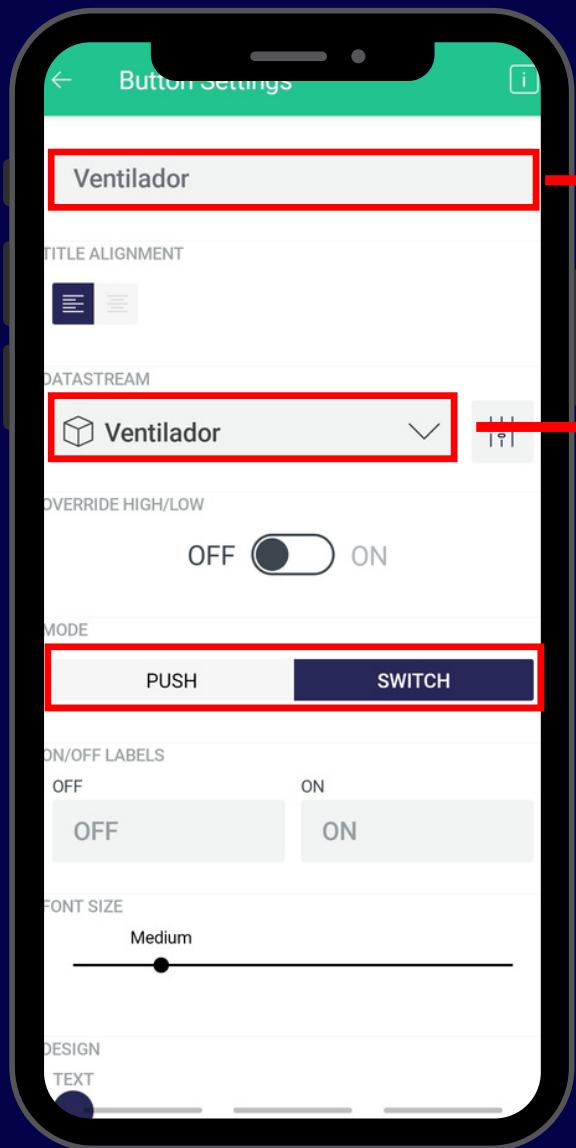


Selecionar o **projeto** criado anteriormente no site.

Clique sobre o + e selecione dois Button.



Configurando os botões



Coloque o **nome** do botão.

Selecione o **Pino Virtual** que vai acionar o **dispositivo** (já definimos ele pelo site).

Selecione a opção **Switch**, dessa forma um toque liga e outro desliga o **LED**.

OBS: As configurações dos **botões** são iguais, basta alterar o nome e o **Pino Virtual**. Lembre-se:

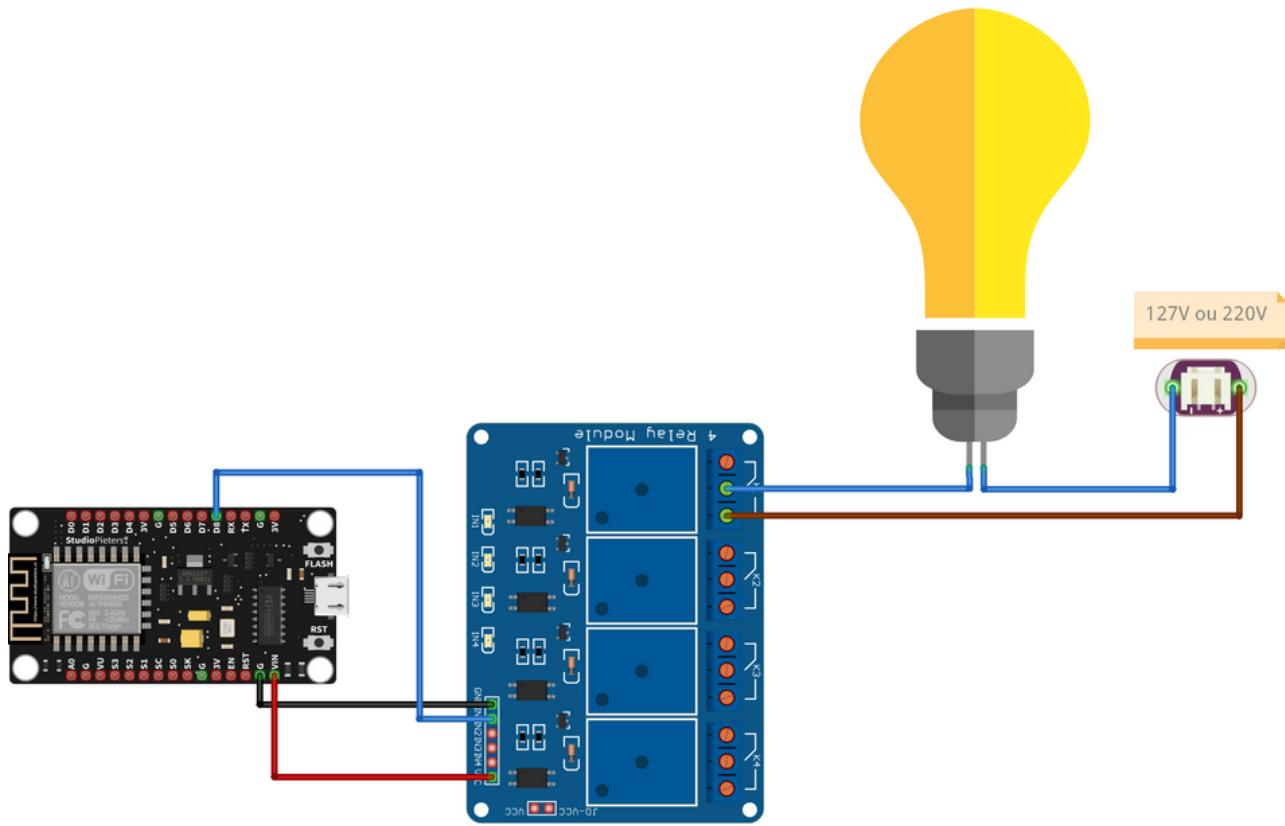
Ventilador: **V0**

Lâmpada: **V1**

Após finalizar a configuração dos **botões**, basta voltar para a **tela inicial** do App.

Círculo com a Lâmpada

O circuito não vai mudar muito do outro projeto, apenas vamos remover as conexões dos relés que não vão ser utilizadas e adicionar uma **Lâmpada**.



É necessário conectar o pino **Vin** da placa no **VCC** do Relé, visto que ele precisa de uma tensão de **5V** para funcionar.

Conecte o pino **G** do **NodeMCU** no pino **GND** do **Módulo Relé**.

O **Pino Digital D8 (GPIO 15)** deve estar no **IN1** do Relé.

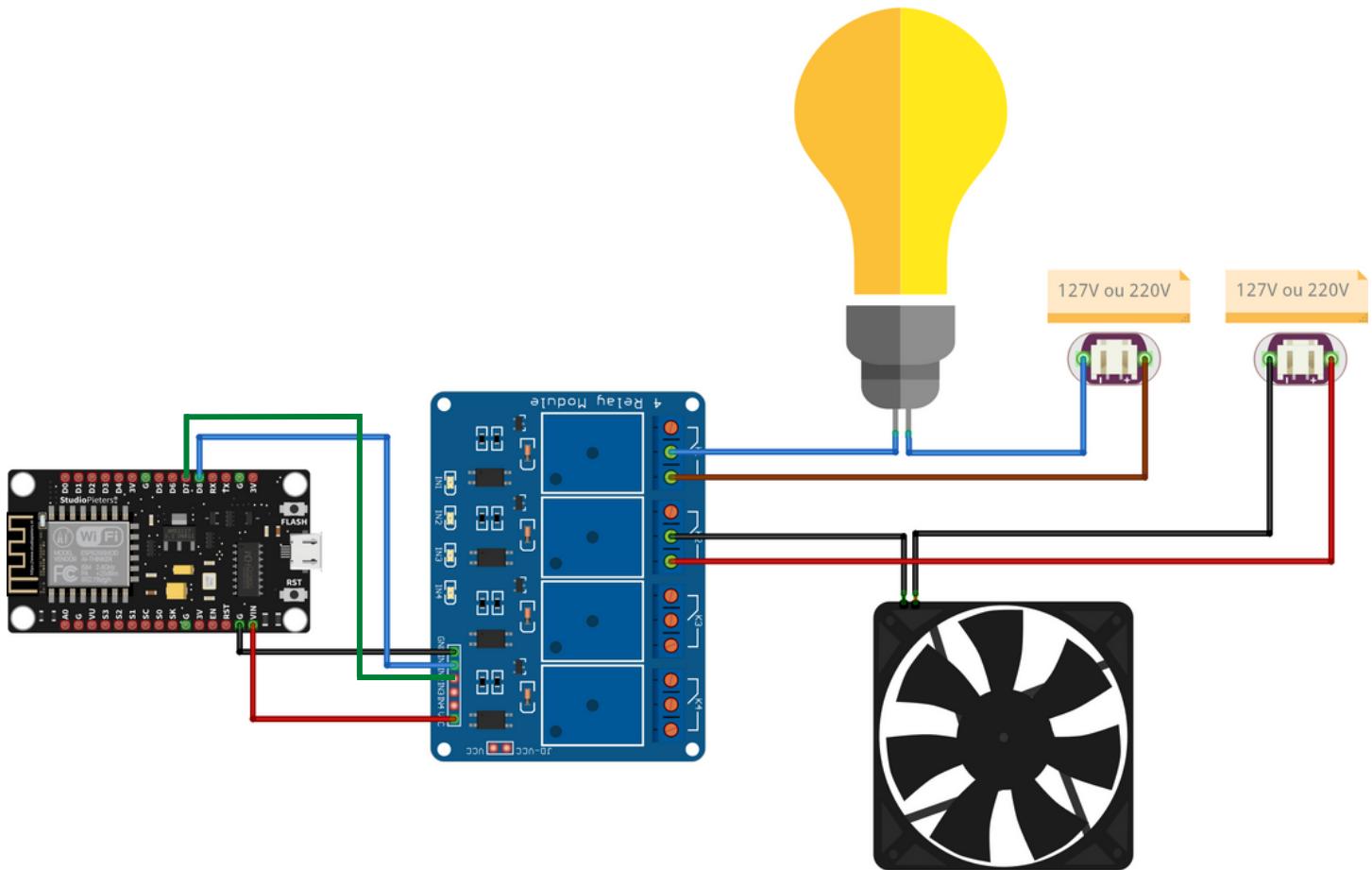
Um terminal da lâmpada deve estar conectado no **contato aberto do relé K1** e outro na **rede elétrica**.

O **contato fechado do relé K1** deve estar conectado diretamente na **rede elétrica**.

Atenção: Se for menor de idade, peça ajuda a um adulto para realizar as ligações na rede elétrica para evitar o risco de choque !!

Círculo Completo

Para simular um ventilador, será utilizado uma **ventoinha**. Caso você queira utilizar um ventilador, as ligações são as mesmas !!



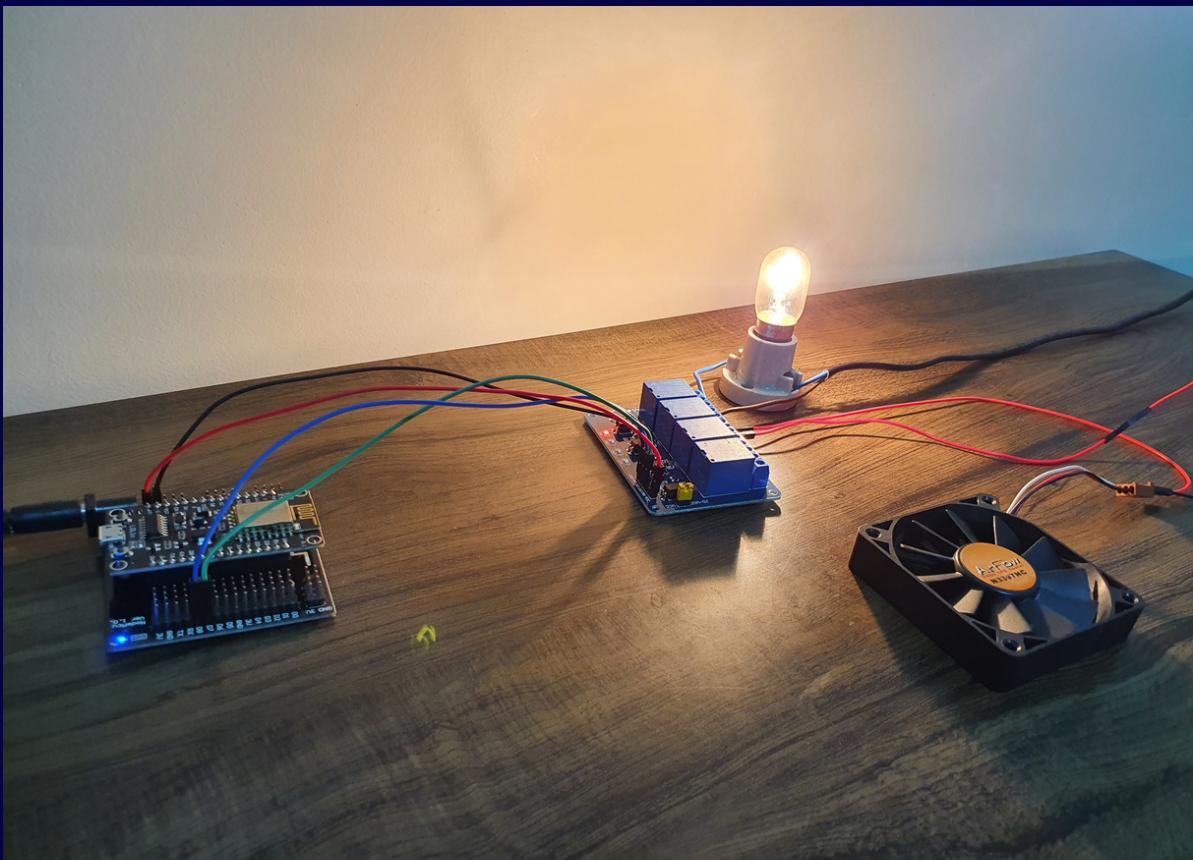
O **Pino Digital D7** deve estar no **IN2** do Relé.

Um terminal do ventilador de estar conectado no **contato aberto do relé K2** e outro na **rede elétrica**.

O **contato fechado do relé K2** deve estar conectado diretamente na **rede elétrica**.

Atenção: Se for menor de idade, peça ajuda a um adulto para realizar as ligações na rede elétrica para evitar o risco de choque !!

Círcuito na Prática



Programação

```
#define BLYNK_PRINT Serial
#define BLYNK_DEVICE_NAME "Nome do projeto"
#define BLYNK_TEMPLATE_ID "Insira o código do Template"

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

char auth[] = "Insira o Token";
char ssid[] = "Insira o nome da rede";
char pass[] = "Insira a senha da rede";

int ventilador= D7;
int lampada= D8;

BLYNK_WRITE (V0){

    int valor = param.asInt();
    digitalWrite(ventilador, valor);
}

BLYNK_WRITE (V1){

    int valor = param.asInt();
    digitalWrite(lampada, valor);

}
```

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

    Blynk.begin(auth, ssid, pass);

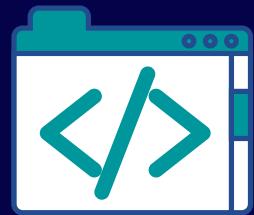
    pinMode(ventilador, OUTPUT);
    pinMode(lampada, OUTPUT);

}

void loop()
{
    Blynk.run();
}
```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento ?? [Clique aqui!!](#)



11

Controlando LEDs com IR

Vamos deixar um pouco de lado os projetos com o **Blynk**. Agora, através de um **controle remoto**, vamos utilizar o **infravermelho** para controlar a cor de um **LED RGB !!**

Controle Remoto Ir e Receptor IR

Este é um módulo de comunicação IR (**infrared** ou simplesmente **infravermelho**) opera na faixa de **38KHz**, é capaz de **decodificar** o sinal de um **controle remoto IR** através de um **microcontrolador** como o **NodeMCU**, **Arduino**, **PIC** e outros.



OBS: O controle não acompanha bateria.

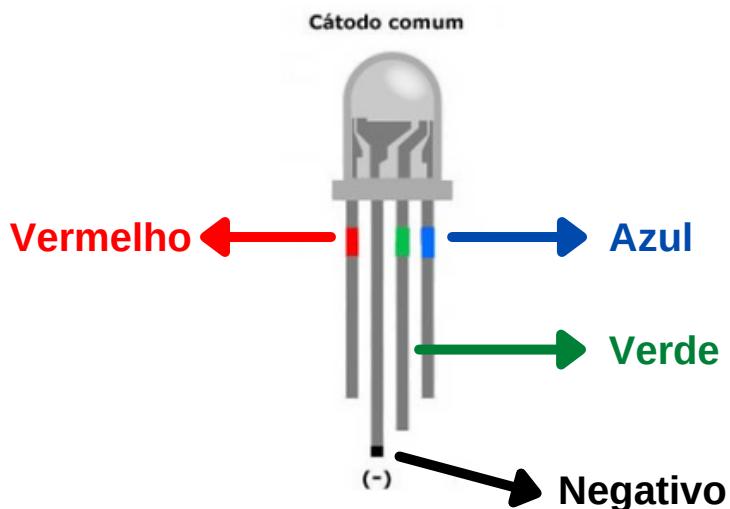
LED RGB

O **LED RGB** consegue exibir diversas cores que são originadas a partir de três cores primárias, o vermelho (**Red**), verde (**Green**) e azul (**Blue**). Este tipo de LED possui três LEDs encapsulados em um mesmo dispositivo, onde cada LED possui uma **cor distinta** e pode ser controlado de **forma individual**. Com esse componente, você pode montar um sistema de sinalização **sem precisar de vários componentes no mesmo circuito**.



Esse LED é do tipo **Cátodo Comum**, o que significa que o terminal maior deve ser ligado ao **Negativo (-)**. Para evitar danos ao LED, você deve utilizar **resistores** adequados à cada cor, não ultrapassando os limites de tensão.

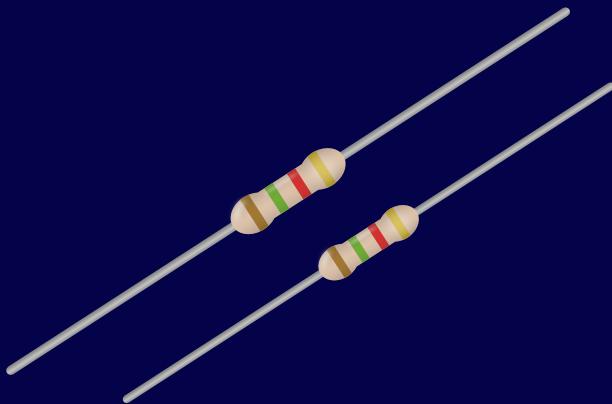
Terminais do LED



Resistor

Resistores são dispositivos muito utilizados para controlar a passagem de corrente elétrica em circuitos elétricos por meio do **efeito Joule**, que converte **energia elétrica** em **energia térmica**.

Vamos utilizar ele pois se ligarmos um **LED** direto no 5 **volts** fornecido pelo Arduino, ele irá **queimar** !!



Como saber qual a resistência correta ??

Existe uma **fórmula** para saber qual a **resistência** deve ser utilizada em um **LED**. Mas antes é preciso saber qual é a **Tensão** e a **Corrente** de cada **LED**.

LEDs		
Cor do LED	Tensão em Volts (V)	Corrente em Miliampères (mA)
Vermelho	1,8 - 2,0V	20 mA
Amarelo	1,8 - 2,0V	20 mA
Laranja	1,8 - 2,0V	20 mA
Verde	2,0 - 2,5V	20 mA
Azul	2,5 - 3,0V	20 mA
Branco	2,5 - 3,0V	20 mA

Cálculos

Para se calcular o **resistor** adequado para o **LED** utilizaremos a seguinte fórmula:

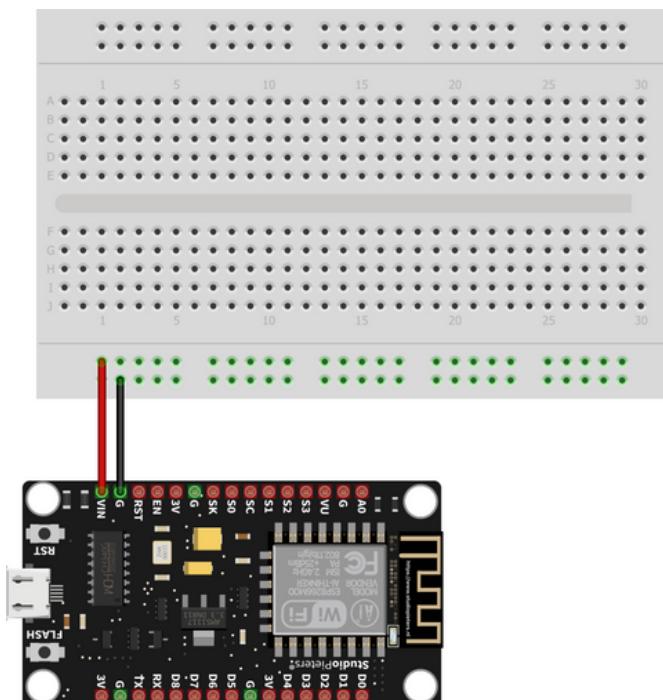
$$\text{Resistência do LED} = \frac{\text{Tensão de Alimentação} - \text{Tensão do LED}}{\text{Corrente do LED}}$$

OBS: O valor do resistor pode ser acima do resultado, porém nunca abaixo, por exemplo, se o valor for igual a **150 Ω**, podemos usar um resistor de valor mais alto, como **200 Ω**, ou até mesmo **1KΩ**, que são 1000 Ω.

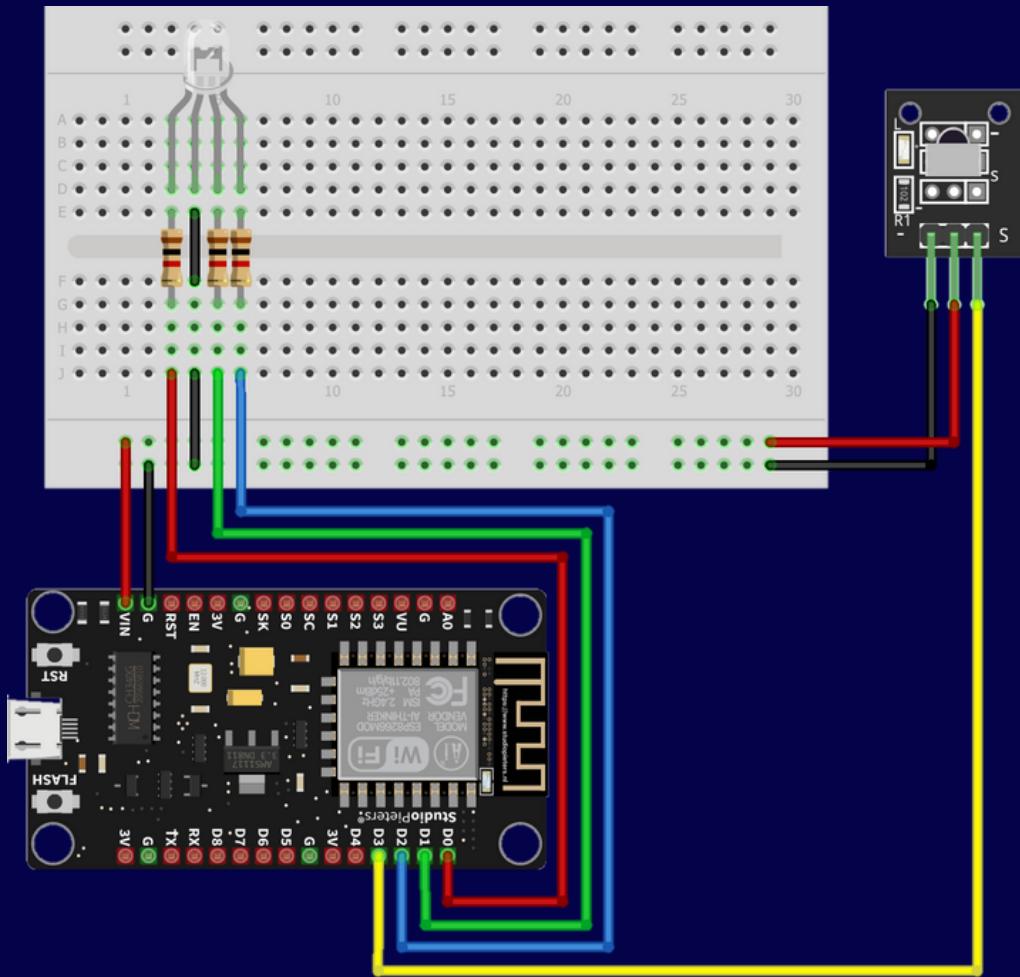
OBS: O símbolo **Ω** significa **Ohms**, que é a nossa unidade de resistência elétrica.

Circuito

Primeiro é necessário energizarmos a **Protoboard**.



Conekte o **Pino Vin** e o **Pino GND** do **NodeMCU** na **Protoboard**.



Em cada terminal do LED RGB onde estão os LEDs vamos conectar resistores de **1K Ω**.

O Maior Terminal deve estar conectado no **Negativo**.

LED Vermelho: Pino Digital **D0**.

LED Verde: Pino Digital **D1**.

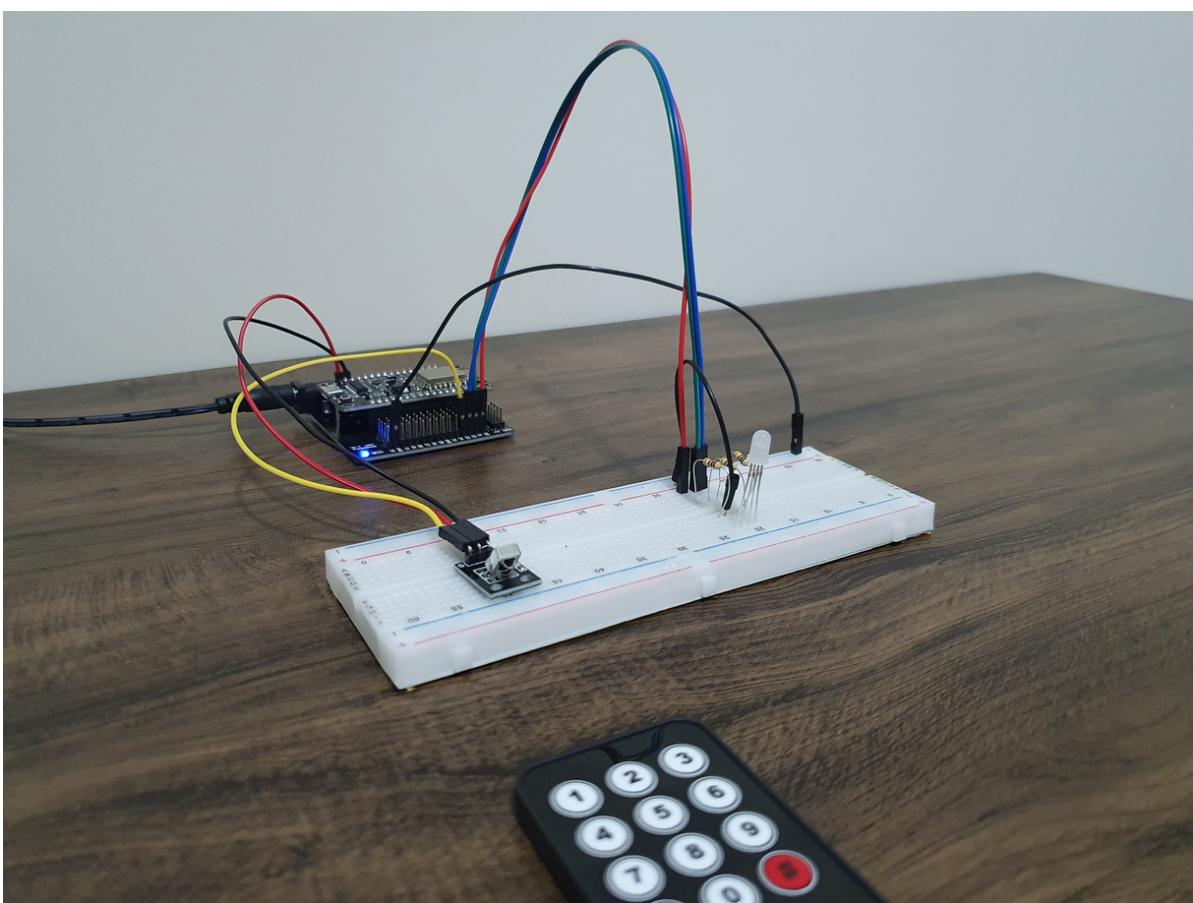
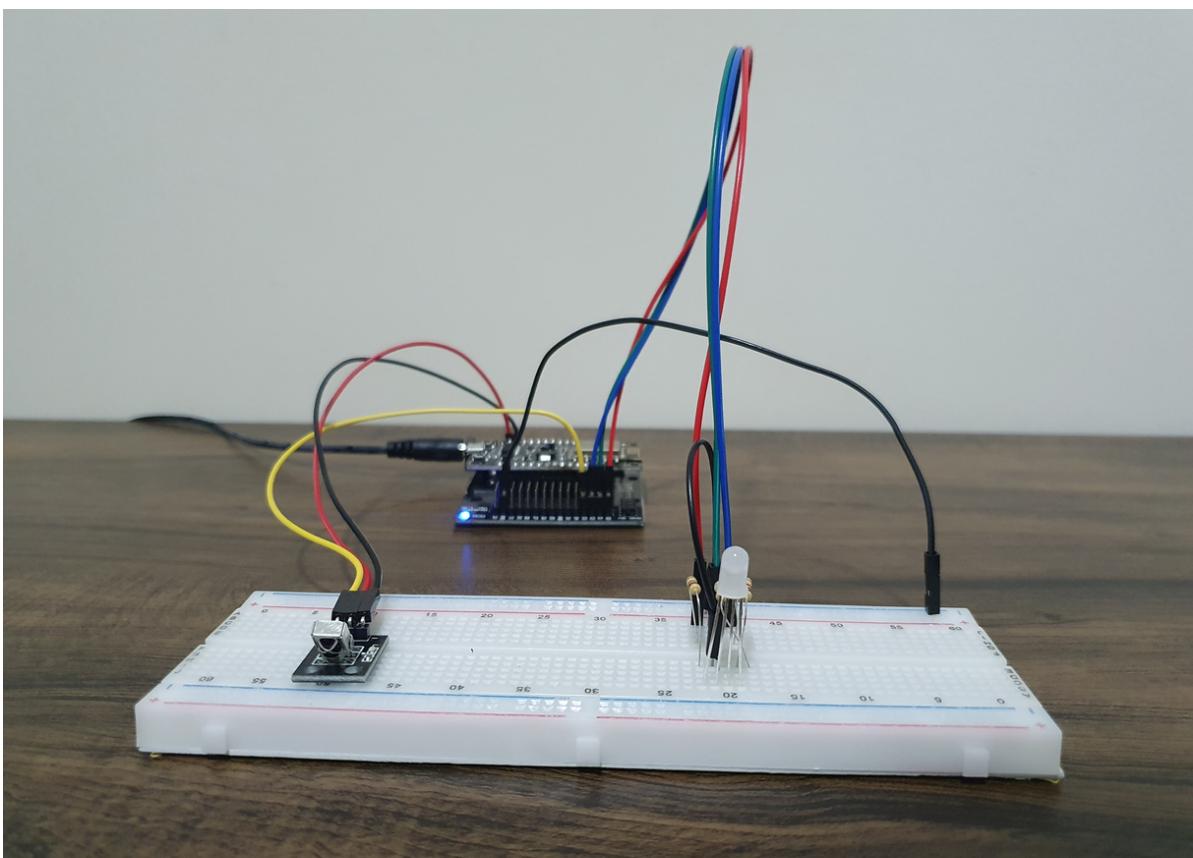
LED Azul: Pino Digital **D2**.

O primeiro pino do receptor deve estar no **negativo**.

O Segundo pino é o **VCC**, deve estar conectado no **positivo** do circuito.

O Terceiro pino do Receptor, ao lado da legenda S, é o **sinal**, precisa estar conectado no **Pino Digital D3**.

Círcuito na Prática



Clonagem das Teclas

Precisamos saber qual código o controle emite ao acionarmos determinadas teclas, vamos utilizar as teclas **1, 2, 3, 4, 5, 6, 7 e 8**.

Para isso, precisamos criar uma programação para clonar as teclas.

Será necessário instalarmos na **IDE do Arduino** a biblioteca **IRremote ESP8266-master**, [clique aqui](#) para baixar.

Programação Clonagem

```
#include <IRremoteESP8266.h> // Incluido bibliotecas necessária

int RECV_PIN = D3; // Pino Digital onde conectamos o receptor infravermelho

IRrecv irrecv(RECV_PIN);

decode_results results; // Armazena os resultados

void setup (){
    Serial.begin(9600); //Da inicio a serial

    irrecv.enableIRIn(); // Inicia a recepção de sinais IR
}

void loop(){
    //Capta o sinal IR

    if (irrecv.decode(&results)) {
```

```

Serial.print ("Código DEC: "); //Texto no monitor serial

Serial.println (results.value); //Código em Decimal

Serial.println("");
irrecv.resume();
}

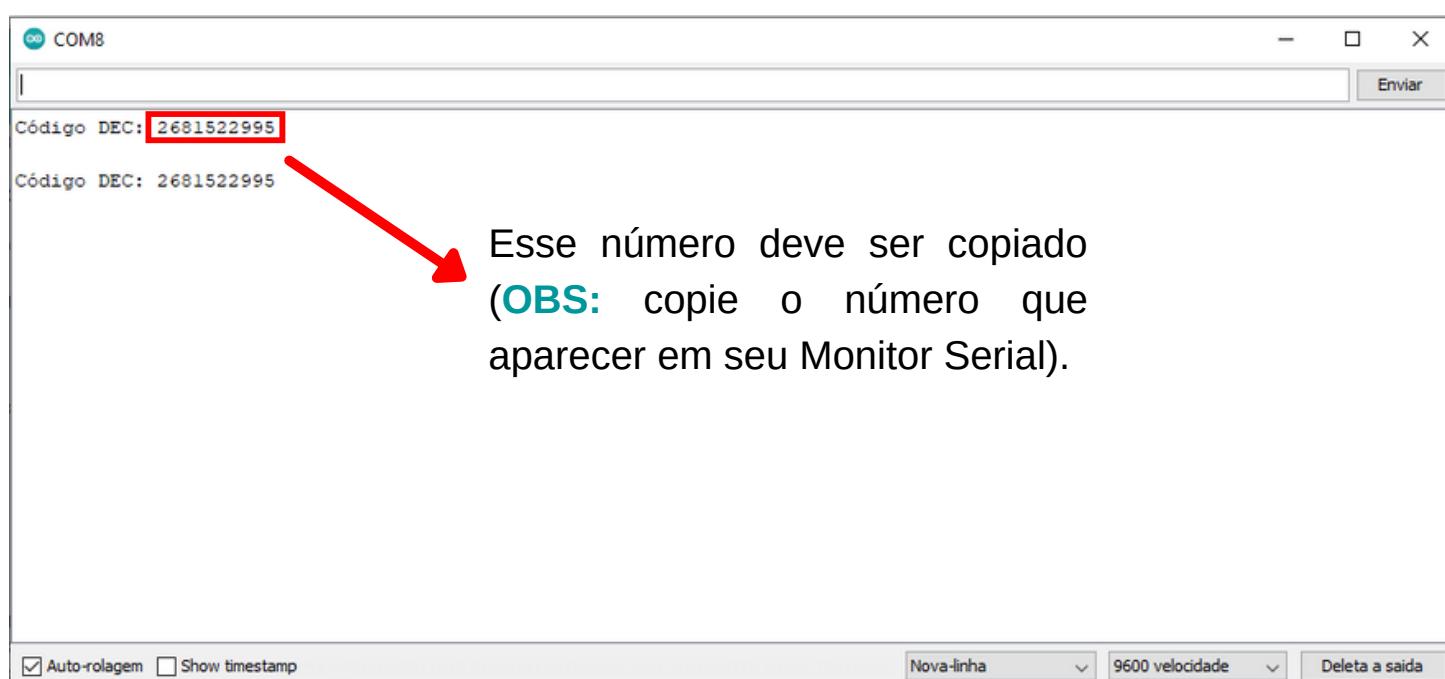
delay(100);
}

```

Baixe o código do projeto [Clicando Aqui!](#)

Envie a programação para o **NodeMCU**, deixe ele conectado via **cabo USB** ao computador e pressione as teclas **Ctrl + Shift + m** para abrir o **Monitor Serial** na **IDE do Arduino**.

Após abrir o **Monitor Serial** clique sobre as **teclas** do controle que vamos clonar e **irá aparecer números para cada tecla**, copie esse numero, pois iremos utilizar na programação com o **LED**.



Lembre-se: Precisamos clonar as teclas **1, 2, 3, 4, 5, 6, 7** e **8**.

Programação

Agora sim, vamos criar o **programa** do nosso projeto !!

```
#include <IRremoteESP8266.h>

int const PINO_RECECTOR = D3; // Pino Digital onde está conectado o receptor
infravermelho

int LED_VERMELHO = D0; // Pino Digital onde está conectado o LED Vermelho
int LED_VERDE = D1; // Pino Digital onde está conectado o LED Verde
int LED_AZUL = D2; // Pino Digital onde está conectado o LED Azul

// Aqui você deve colocar os números das teclas clonadas na programação
anterior

int tecla1 = 0000000000; // insira o número clonado da tecla 1
int tecla2 = 0000000000; // insira o número clonado da tecla 2
int tecla3 = 0000000000; // insira o número clonado da tecla 3
int tecla4 = 0000000000; // insira o número clonado da tecla 4
int tecla5 = 0000000000; // insira o número clonado da tecla 5
int tecla6 = 0000000000; // insira o número clonado da tecla 6
int tecla7 = 0000000000; // insira o número clonado da tecla 7
int tecla8 = 0000000000; // insira o número clonado da tecla 8

IRrecv receptor (PINO_RECECTOR);

decode_results valorSaida;

void setup ()
{
    Serial.begin (9600);
    pinMode (LED_VERMELHO, OUTPUT); // O LED é uma Saída Digital
```

```

pinMode(LED_VERDE, OUTPUT); // O LED é uma Saída Digital
pinMode(LED_AZUL, OUTPUT); // O LED é uma Saída Digital

receptor.enableIRIn(); // Inicia o receptor

}

void loop () {

if (receptor.decode(&valorSaida)) {
    receptor.resume (); // Recebe o próximo valor

    if (valorSaida.value == tecla1){

        // Cor Vermelha

        digitalWrite (LED_VERMELHO, HIGH); // LED Vermelho ligado
        digitalWrite (LED_VERDE, LOW); // LED Verde desligado
        digitalWrite (LED_AZUL, LOW); // LED Azul desligado

    } else if (valorSaida.value == tecla2) {

        // Cor Verde

        digitalWrite (LED_VERDE, HIGH); // LED Verde ligado
        digitalWrite (LED_VERMELHO, LOW); // LED Vermelho desligado
        digitalWrite (LED_AZUL, LOW); // LED Azul desligado

    } else if (valorSaida.value == tecla3){

        // Cor Azul

        digitalWrite (LED_AZUL, HIGH); // LED Azul ligado
        digitalWrite (LED_VERMELHO, LOW); // LED Vermelho Desligado
    }
}
}

```

```
digitalWrite (LED_VERDE, LOW); // LED Verde Desligado

} else if (valorSaida.value == tecla4){

    // Cor Amarela

    digitalWrite (LED_VERMELHO, HIGH); // LED Vermelho ligado
    digitalWrite (LED_VERDE, HIGH); // LED Verde ligado
    digitalWrite (LED_AZUL, LOW); // LED Azul desligado

} else if (valorSaida.value == tecla5){

    // Lilás

    digitalWrite (LED_VERMELHO, HIGH); // LED Vermelho ligado
    digitalWrite (LED_AZUL, HIGH); // LED Azul ligado
    digitalWrite (LED_VERDE, LOW); // LED Verde desligado

} else if (valorSaida.value == tecla6){

    // Ciano

    digitalWrite (LED_VERDE, HIGH); // LED Verde ligado
    digitalWrite (LED_AZUL, HIGH); // LED Azul ligado
    digitalWrite (LED_VERMELHO, LOW); // LED Vermelho desligado

} else if (valorSaida.value == tecla7){

    // Branco

    digitalWrite (LED_VERDE, HIGH); // LED Verde ligado
    digitalWrite (LED_AZUL, HIGH); // LED Azul ligado
    digitalWrite (LED_VERMELHO, HIGH); // LED Vermelho ligado

} else if (valorSaida.value == tecla8){
```

```
// LED apagado

digitalWrite (LED_Verde, LOW);      // LED Vermelho desligado
digitalWrite (LED_Azul, LOW);      // LED Vermelho desligado
digitalWrite( LED_Vermelho, LOW); // LED Vermelho desligado

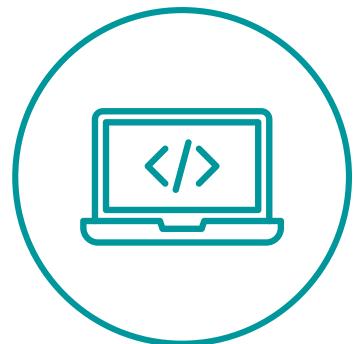
}

delay(100);

}
}
```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento ?? [Clique aqui!!](#)



12 Alarme

Nesse projeto, utilizando um **Sensor de Movimentos e Presença** vamos construir um alarme. Sempre que o Sensor detectar o movimento de alguém, um **LED** da cor vermelha será acesso e um **Buzzer** emitirá um som.

Vamos conhecer os novos componentes que vão ser utilizados:

Sensor de Movimento e Presença

Também conhecido como **Sensor PIR**, seu funcionamento se baseia a partir detecção de **calor** emitido pelo corpo humano. O **Sensor PIR** consegue detectar o movimento de objetos que estejam em uma área de até **7 metros**, caso algo ou alguém emitindo calor acima de 0 graus se movimentar nessa área o pino de alarme é ativado. É possível ajustar a duração do tempo de espera para estabilização do **PIR** através do potenciômetro amarelo em baixo do sensor assim como sua sensibilidade. A estabilização pode variar entre 5-200 seg.



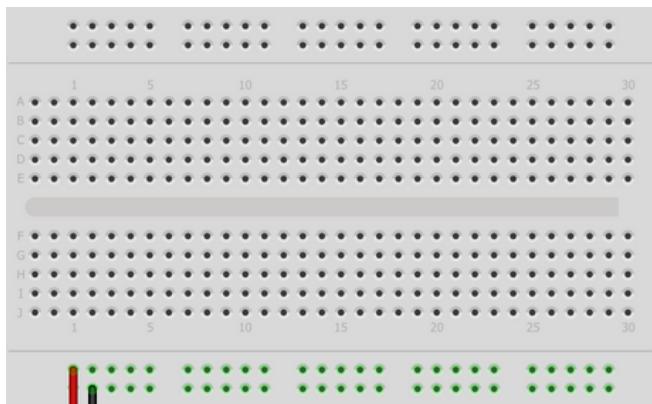
Buzzer

O **Buzzer 5V** é um componente para adicionar efeitos sonoros em projetos eletrônicos como por exemplo, alarmes, sistemas de sinalização, jogos, brinquedos, etc. O **Buzzer do tipo Ativo** contém um circuito oscilador embutido, dessa forma basta você energizar o componente para que o mesmo comece a emitir som.

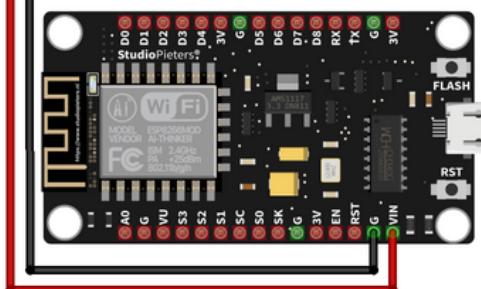


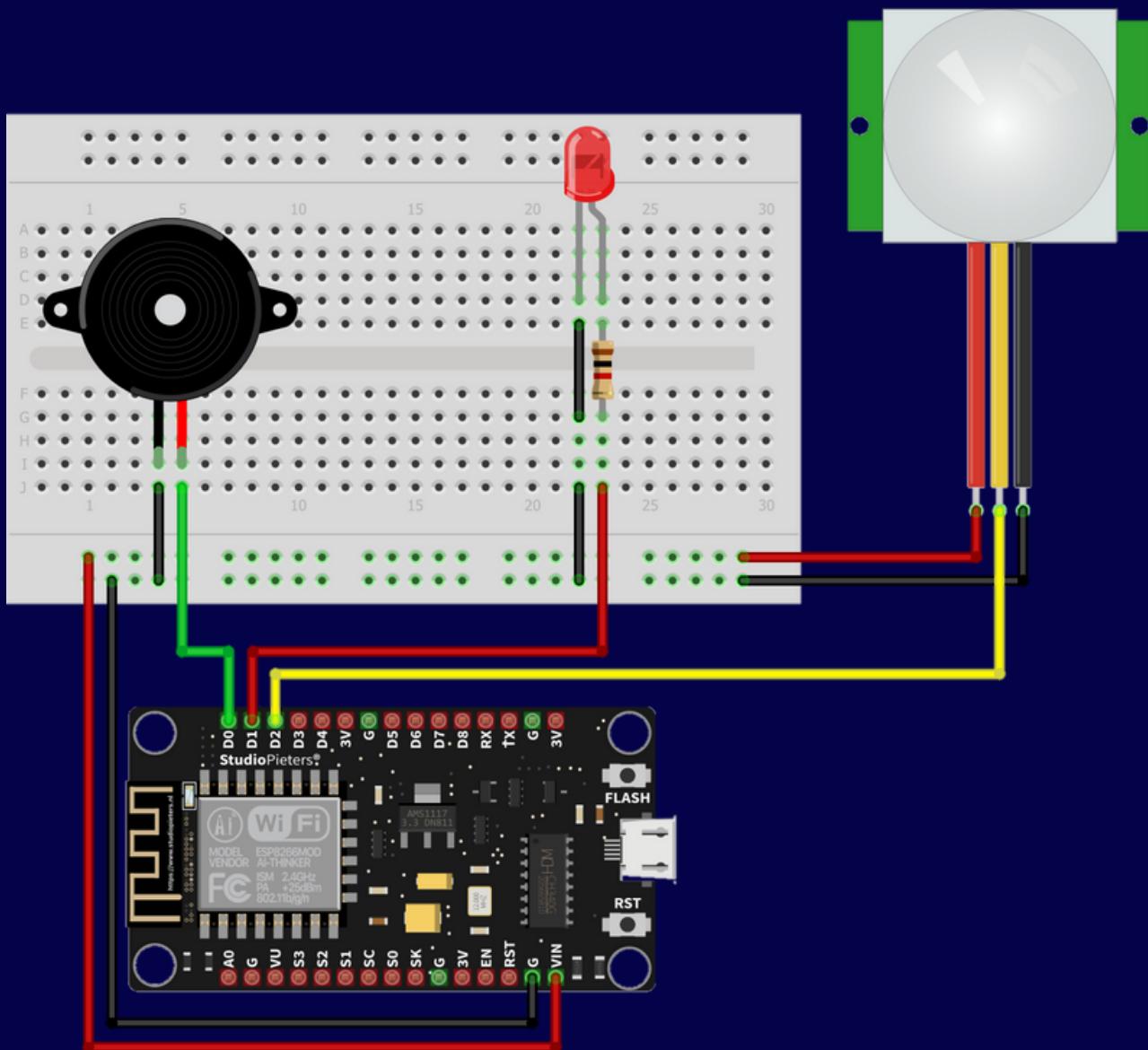
OBS: O maior pino do Buzzer é positiva.

Circuito



Conekte o **Pino Vin** (que fornece **5V**) e o **Pino GND** do **NodeMCU** na **Protoboard**.





O menor pino do **Buzzer** deve estar conectado ao **GND** (negativo) do circuito e o maior no **Pino Digital D0** do **NodeMCU**.

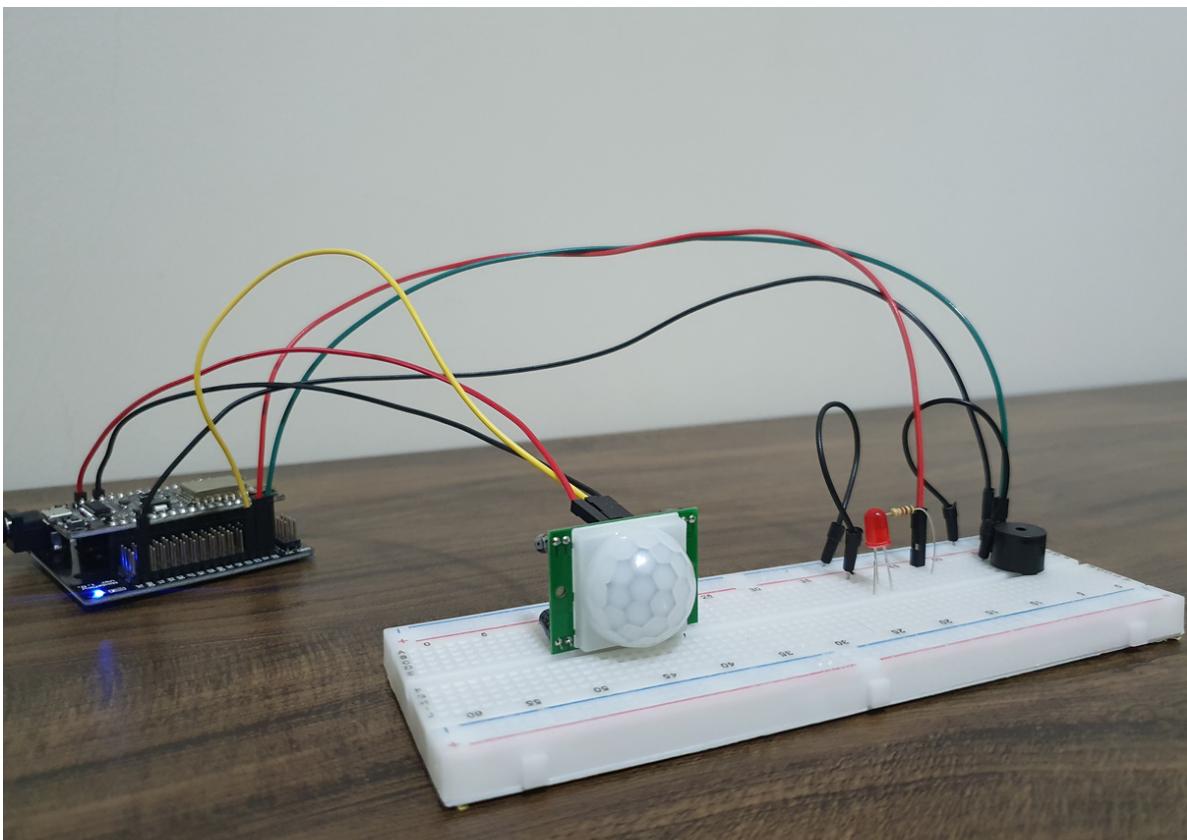
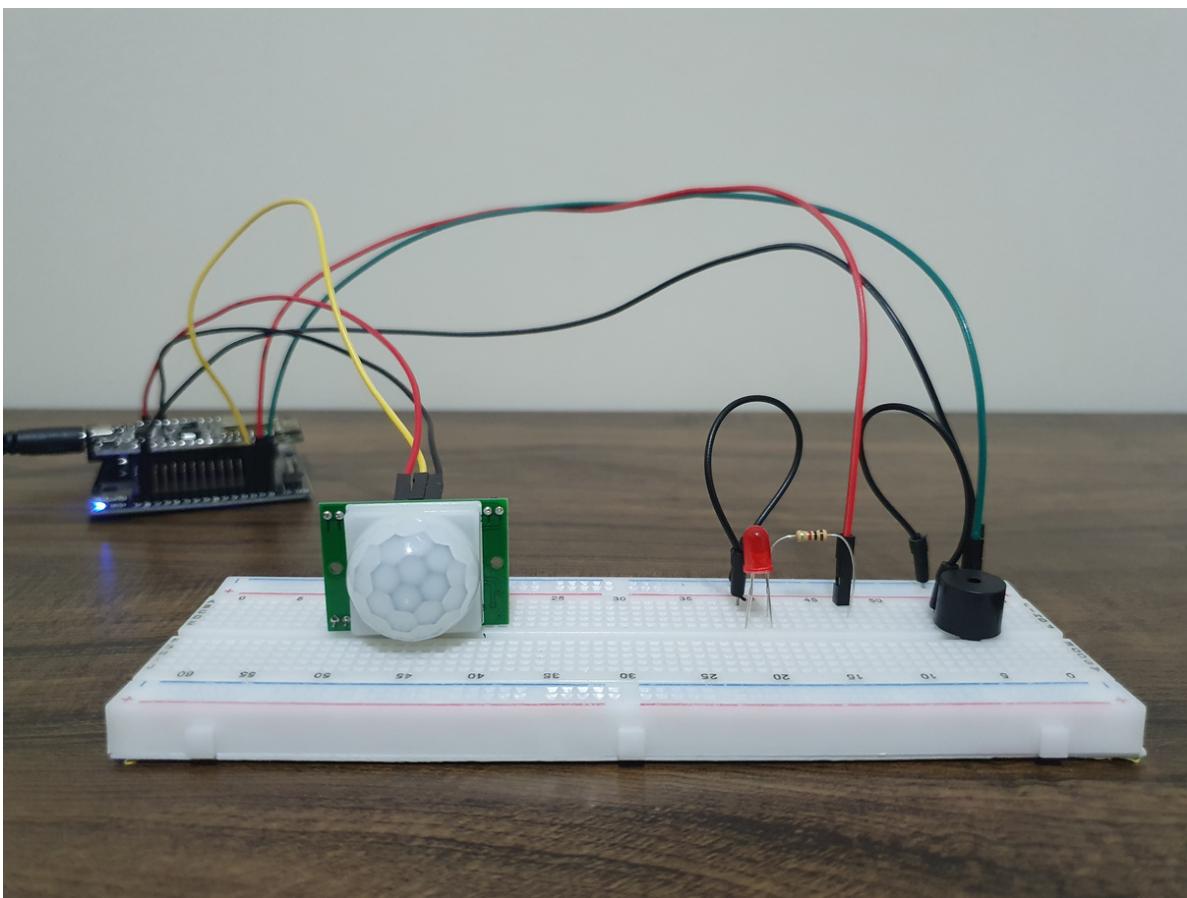
O menor terminal do **LED** precisa ser ligado ao **GND**, e o maior em um **resistor** de **1K Ω** e no **Pino Digital D1** da nossa placa.

O Pino **VCC** do sensor deve ser conectado ao **Vin** do circuito.

O Pino **OUT** do sensor é conectado ao **Pino Digital D2**.

O Pino **GND** do sensor precisa ser ligado ao **GND** do circuito.

Círcuito na Prática



Programação

```
// Declaração das variáveis dos pinos digitais.

int Buzzer = 16;      // Buzzer no pino D0 (GPIO 16)
int Led = 5;           // Led no pino D1    (GPIO 5)
int SensorPIR = 4;   //Sensor no pino D2  (GPIO 4)

int valorSensor = 0;

void setup () {

    Serial.begin (9600); // Inicializando o serial monitor

    // Definido pinos como saídas

    pinMode(Buzzer,OUTPUT);
    pinMode(Led,OUTPUT);
    pinMode(SensorPIR,INPUT);

}

void loop () {

    valorSensor = digitalRead(SensorPIR); // Faz a leitura do sensor

    Serial.print("Valor do Sensor: "); // imprime no monitor serial

    Serial.println(valorSensor); // imprime o valor do sensor no monitor.

    //Se o sensor detectar movimentos

    if (valorSensor == 1) {
```

```
// Alarme Ligado
```

```
tone(Buzzer,1000);  
digitalWrite(Led, HIGH);  
  
delay(5000); // tempo de 5 segundos para desligar o Alarme
```

```
noTone(Buzzer);  
digitalWrite(Led, LOW);
```

```
} else {
```

```
// Alarme Desligado
```

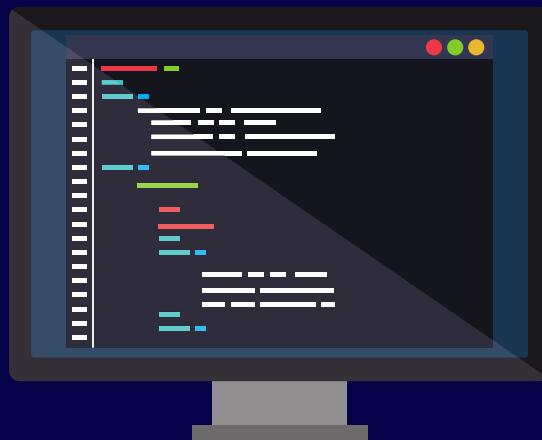
```
noTone(Buzzer);  
digitalWrite(Led, LOW);
```

```
}
```

```
}
```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento ?? [Clique aqui!!](#)



13

Controlando um Display LCD

Nesse próximo projeto, vamos aprender a controlar um **Display LCD** utilizando o **NodeMCU** !!

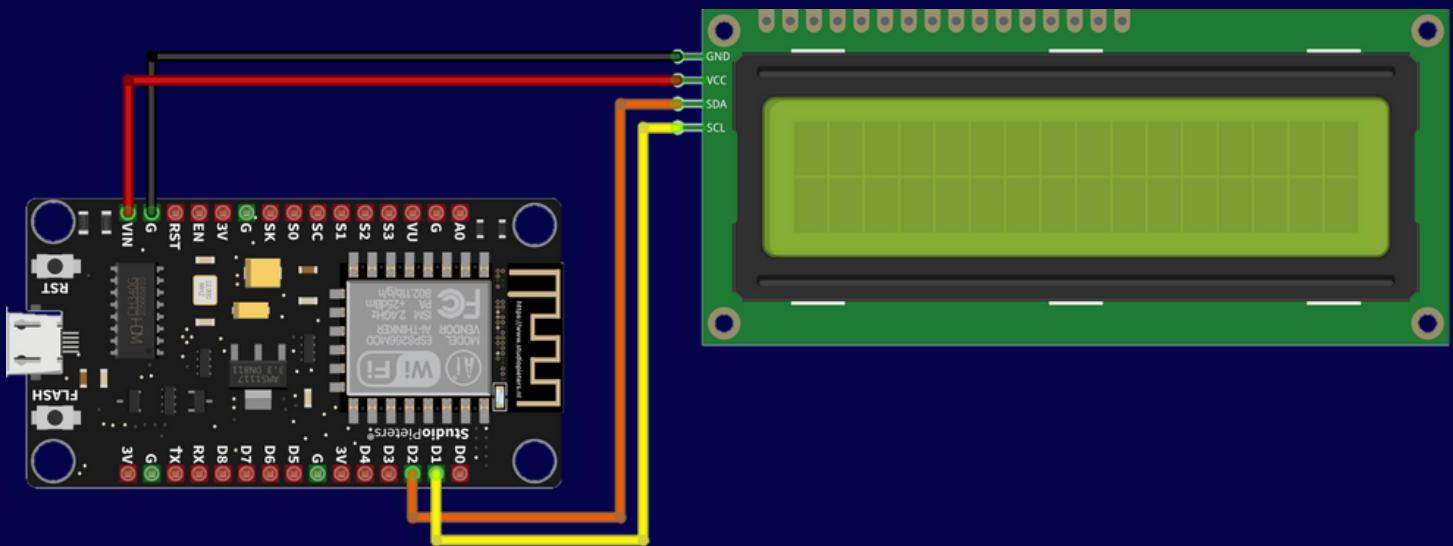
Display LCD 16x2 e I2C

O Display LCD serve para escrevermos textos e sinalizações, tem outras diversas aplicações com **microcontroladores**, como **NodeMCU** e **Arduino**. São 16 colunas por 2 linhas, backlight azul e escrita branca. Possui o controlador HD44780 usado em toda indústria de LCD's como base de interface.



Uma vantagem que esse display traz é o **Módulo I2C** integrado, dessa forma você faz a conexão entre o **microcontrolador** e o display utilizando apenas os pinos **SDA** e **SCL**, deixando as outras portas livres para o desenvolvimento do seu projeto.

Círcuito



GND do **Módulo I2C** deve estar conectado no **Pino G** da placa.

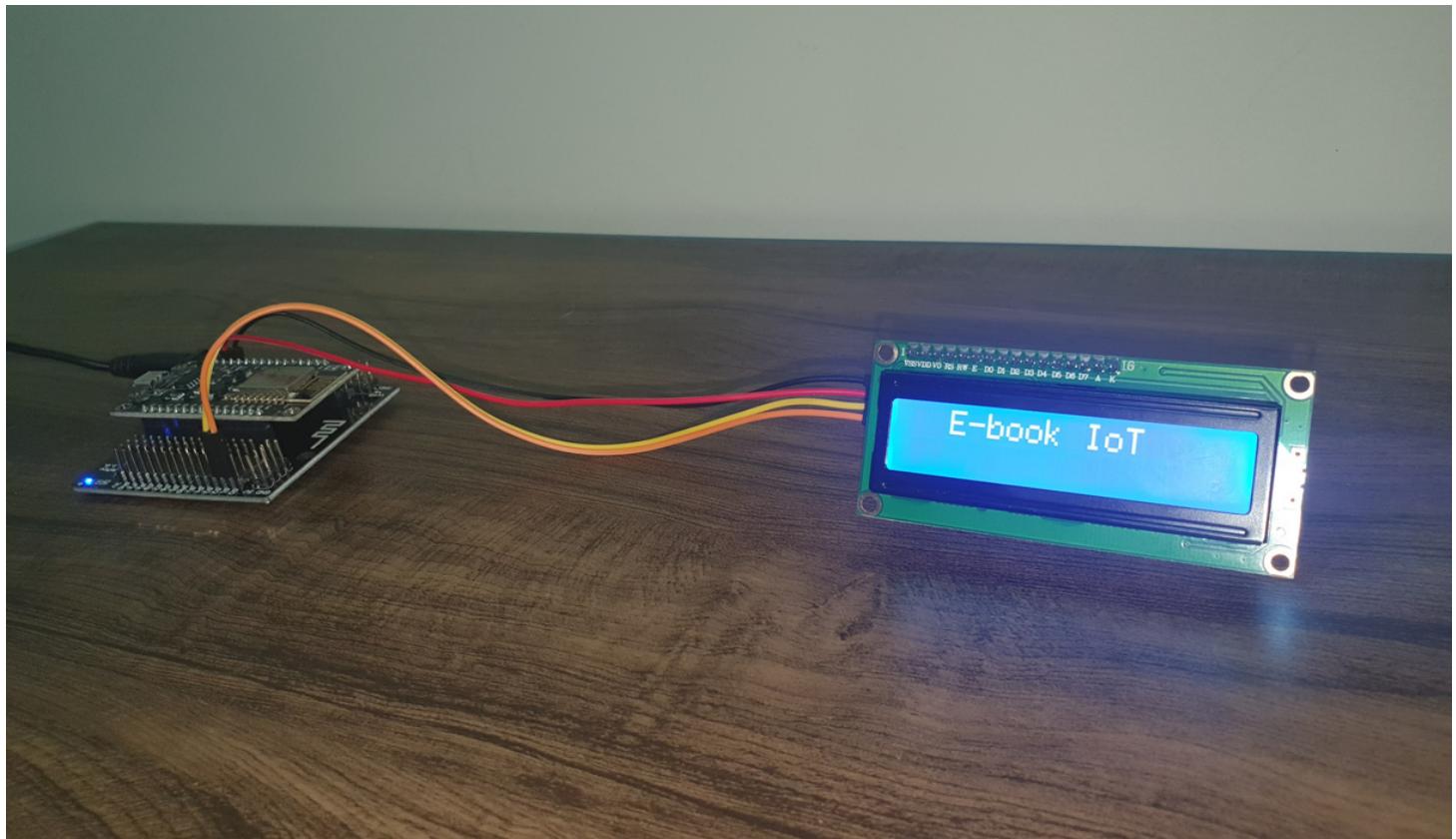
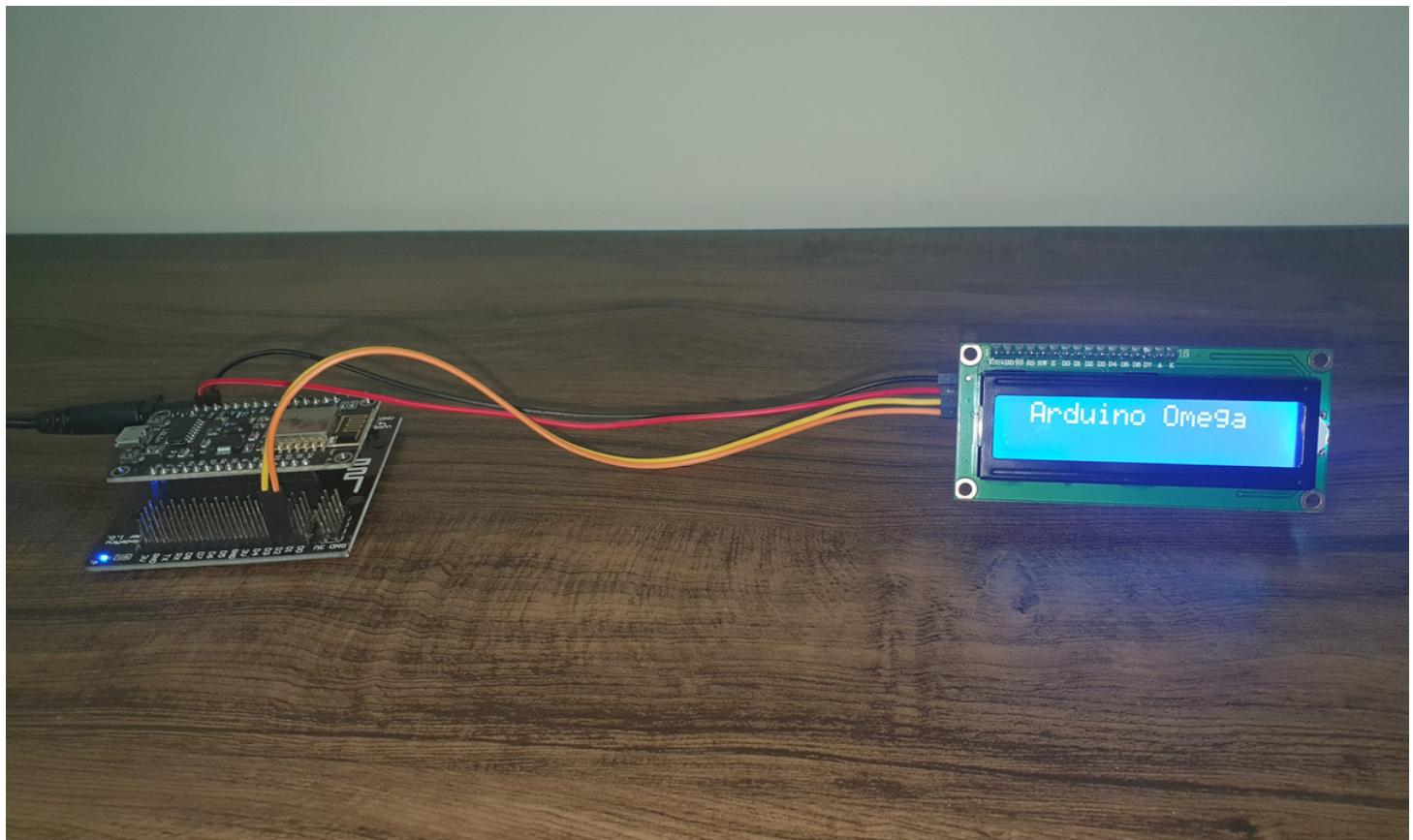
O **Pino VCC** deve estar no **Vin** (A placa Borne do **NodeMCU** possui um regulador de tensão de **5V**, dessa forma mesmo utilizando uma fonte de **9V** a tensão no pino **Vin** será **5V**).

SDA precisa estar conectado no **Pino digital D2** e o **SCL** no **Pino Digital D1**.

Resumo

GND ---- Pino G
VCC ---- Vin
SDA ---- D2
SCL ---- D1

Círcuito na Prática



Programação

Para nossa programação funcionar corretamente, é necessário instalarmos a biblioteca **LiquidCrystal I2C** na **IDE do Arduino**, para baixar a biblioteca, [Clique Aqui](#).

```
// Incluindo bibliotecas necessárias

#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd = LiquidCrystal_I2C(0x27, 16, 2);

void setup () {

lcd.begin(16,2); // linhas e colunas do display
lcd.init();
lcd.backlight();
}

void loop () {

lcd.setCursor(0,0); // coluna 0, linha 0
lcd.print(" E-book IoT"); // Insira seu texto Aqui

delay(5000); // Espera de 5 segundos

lcd.clear(); // apaga o texto escrito no Display

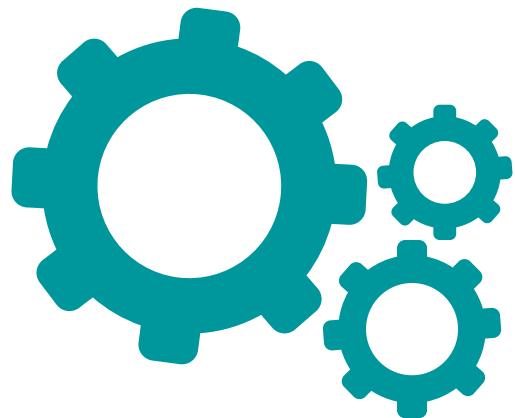
lcd.setCursor(0,0); // coluna 0, linha 0
lcd.print(" Arduino Omega"); // Insira seu texto Aqui

delay(5000); // Espera de 5 segundos

lcd.clear(); // apaga o texto escrito no Display
}
```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento ?? [Clique aqui!!](#)



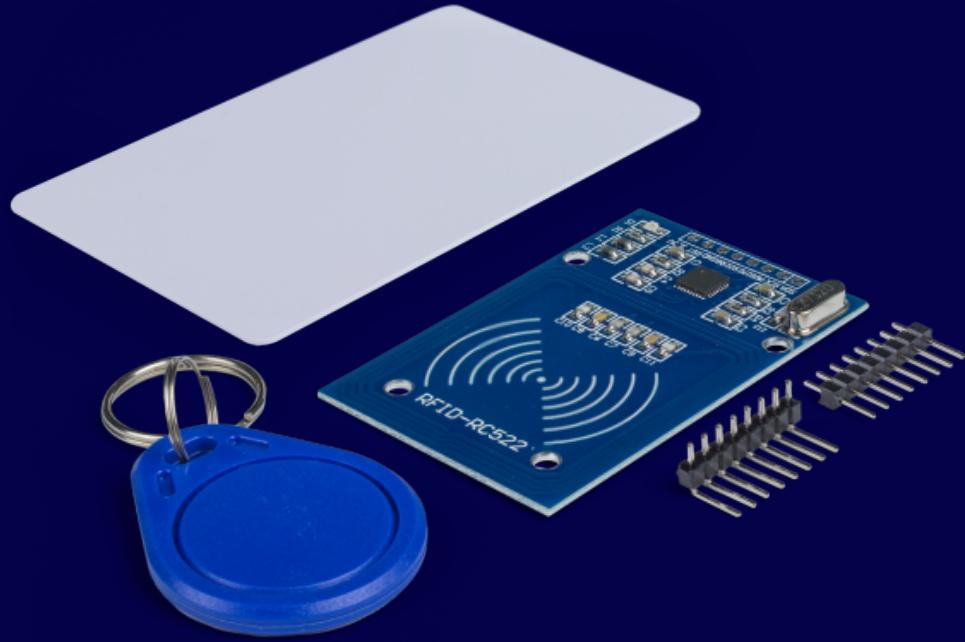
14

Controle de Acesso com RFID

Nesse projeto vamos desenvolver um **controle de acesso** utilizando o **Módulo RFID** e outros componentes que já conhecemos, como o **Buzzer**, o **Display** e outros.

Módulo RFID

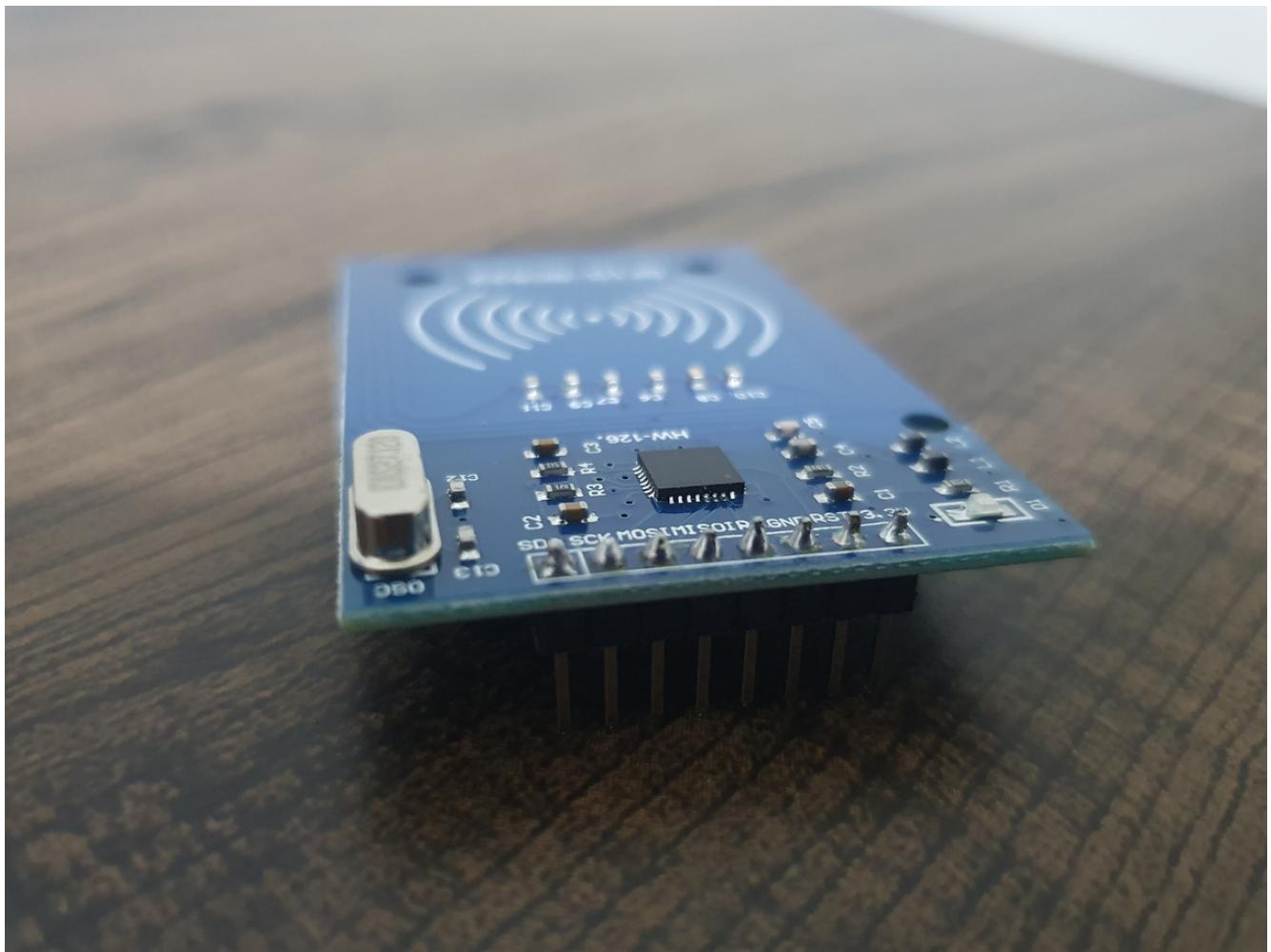
Este **Kit Módulo Leitor RFID** baseado no chip **MFRC522** da empresa NXP é altamente utilizado em comunicação sem contato a uma frequência de 13,56MHz. Este chip, de baixo consumo e pequeno tamanho, permite sem contato ler e escrever em cartões que seguem o padrão **Mifare**, muito usado no mercado.



Possui ferramentas que é preciso para projetos de controle de acesso ou sistemas de segurança. As tags (ou etiquetas) **RFID**, podem conter **vários dados** sobre o proprietário do cartão, como nome e endereço e, no caso de produtos, informações sobre procedência e data de validade, entre outros exemplos.

Soldando os Pinos

É necessário soldar a barra de pinos no [Módulo Leitor RFID-RC522](#). Como vamos utilizar a [Protoboard](#), é recomendado soldar a barras de pinos de 180 graus. Para realizar a solda é preciso utilizar um [ferro de solda](#) e [estanho](#).

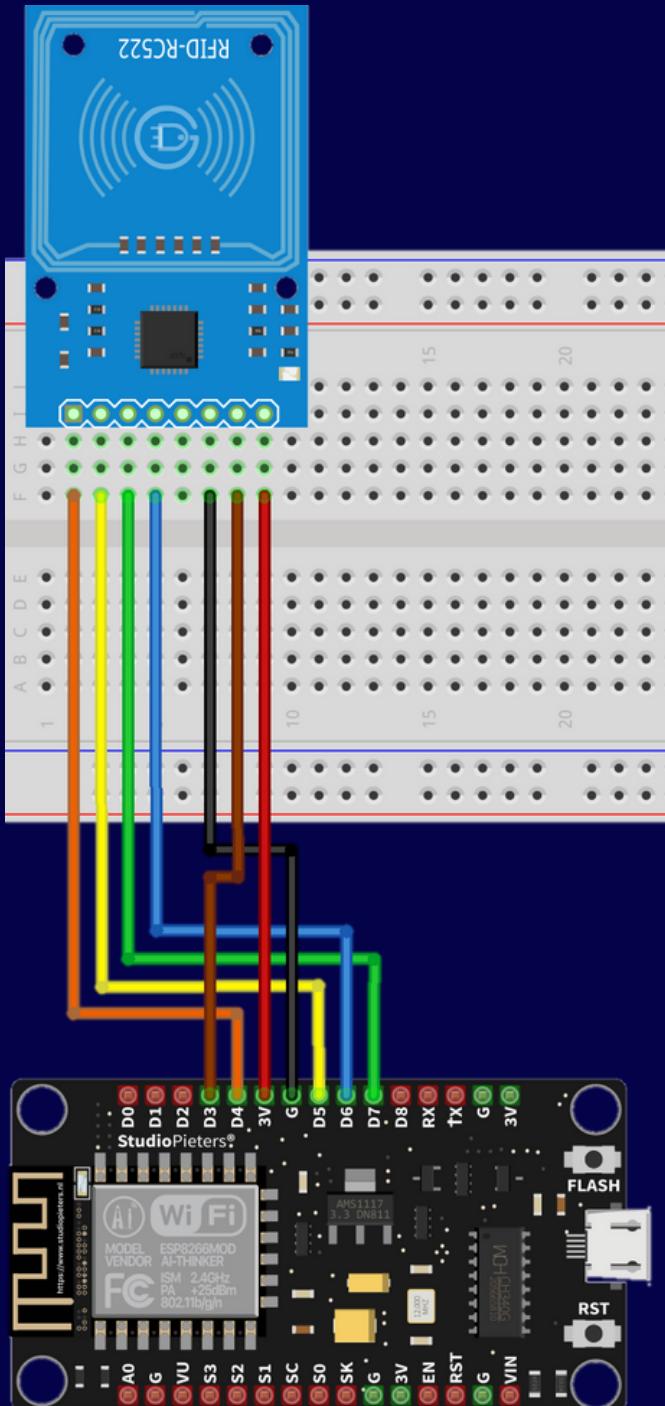


Caso você seja [**menor de idade**](#), peça ajuda a um [**adulto**](#) para a realização da solda dos fios nos motores !!



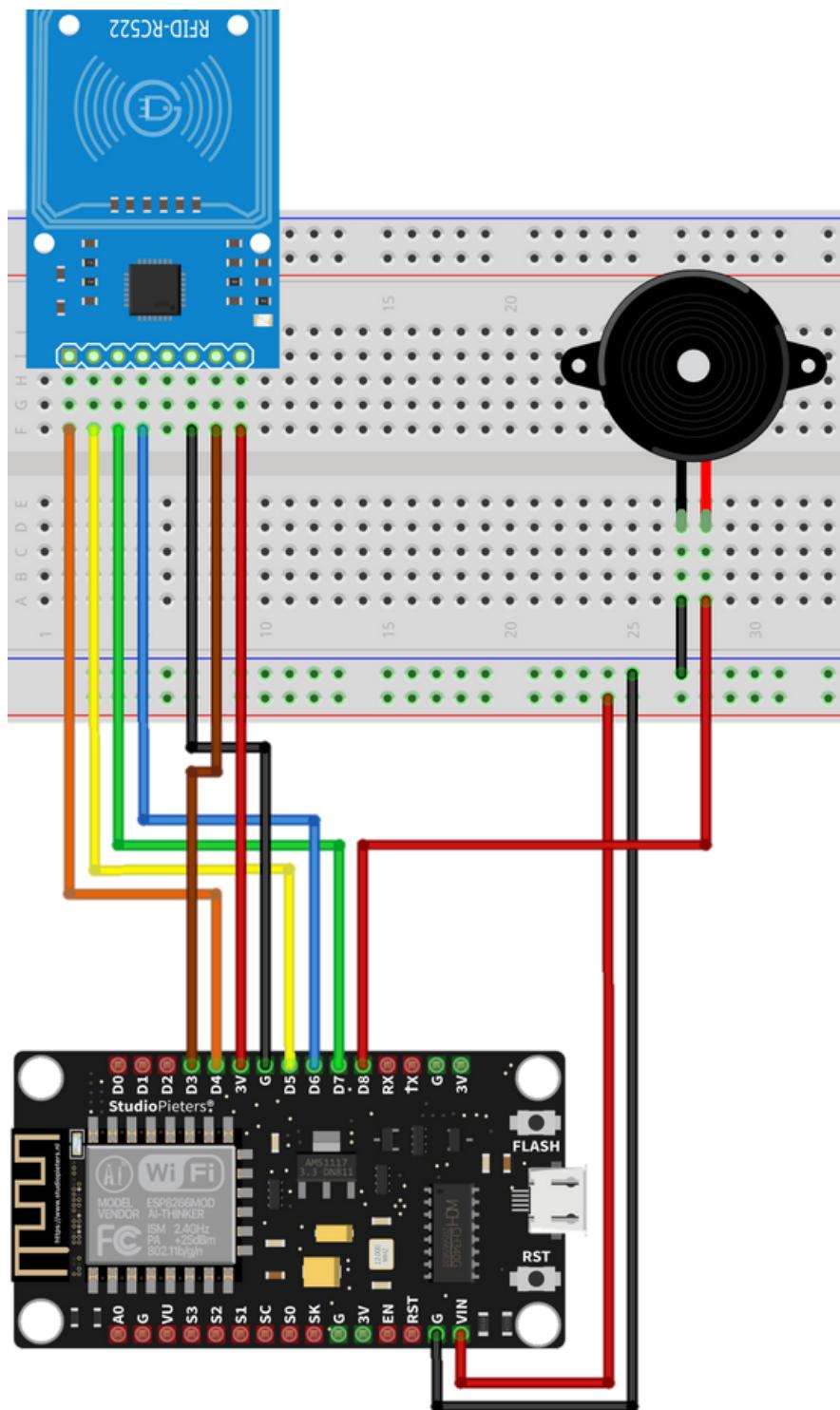
Círcuito

Primeiro vamos realizar as **conexões necessárias** no **Módulo RFID**.



- O **Pino SDA** deve estar conectado ao **Pino Digital D4**.
- O **Pino SCK** deve estar conectado ao **Pino Digital D5**.
- O **Pino MOSI** deve estar conectado ao **Pino Digital D7**.
- O **Pino MISO** deve estar conectado ao **Pino Digital D6**.
- O Pino IRQ não deve ser conectado a nada.
- O **Pino GND** deve estar conectado ao **Pino G**.
- O **Pino RST** deve estar conectado ao **Pino Digital D3**.
- O **Pino 3.3V** deve estar conectado ao **Pino 3V**.

Agora vamos adicionar o **Buzzer** em nosso circuito.

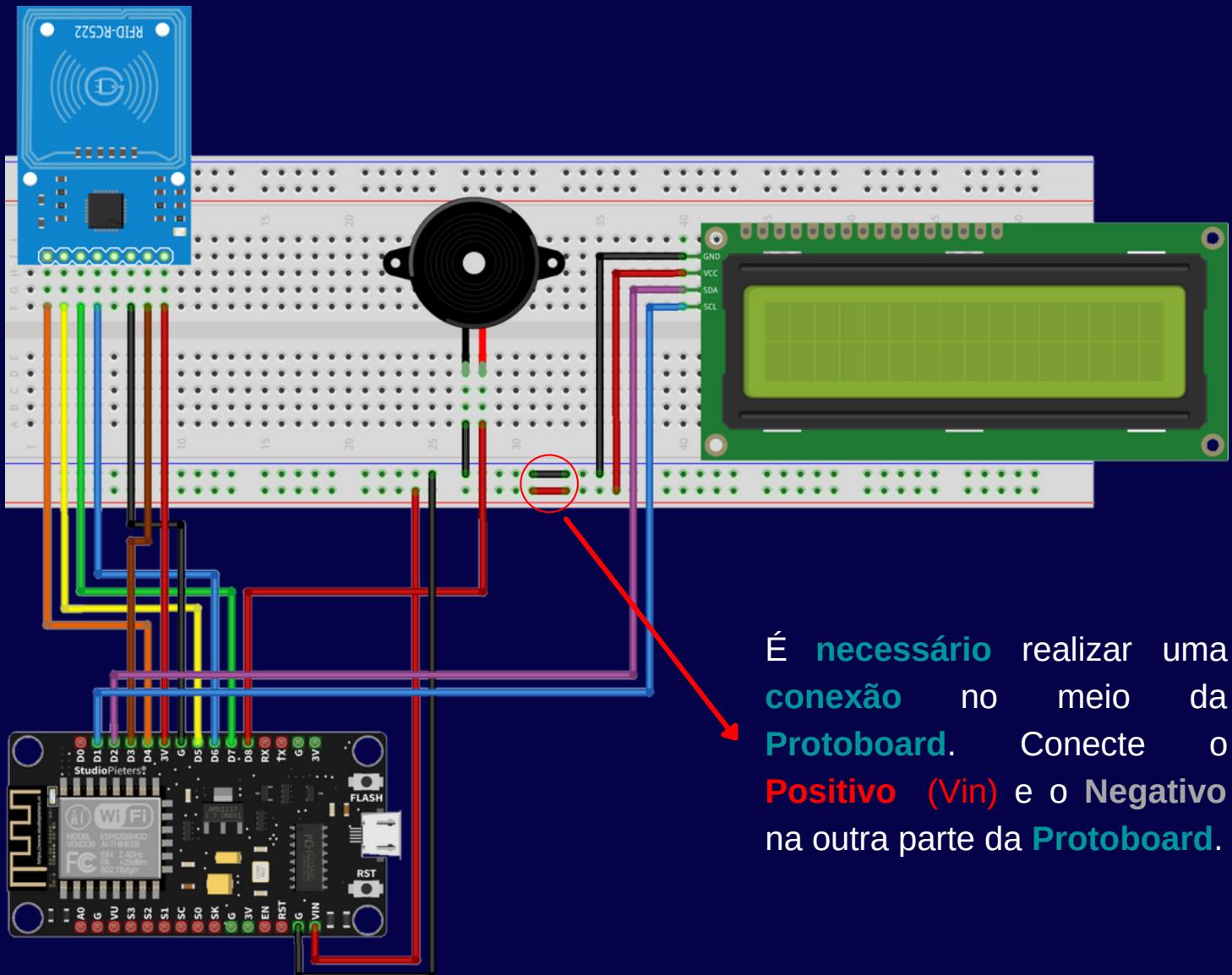


É necessário conectar o **Pino G** (GND) e o **Pino Vin** do **NodeMCU** na **Protoboard**.

O **Maior terminal** do **Buzzer** deve ser conectado no **Pino Digital D8** e o **menor** no **GND** da **Protoboard**.

Círcuito Final

Por fim, vamos incluir no nosso projeto o **Display LCD 16x2** com o **Módulo I2C**.



É necessário realizar uma conexão no meio da **Protopboard**. Conecte o **Positivo (Vin)** e o **Negativo** na outra parte da **Protopboard**.

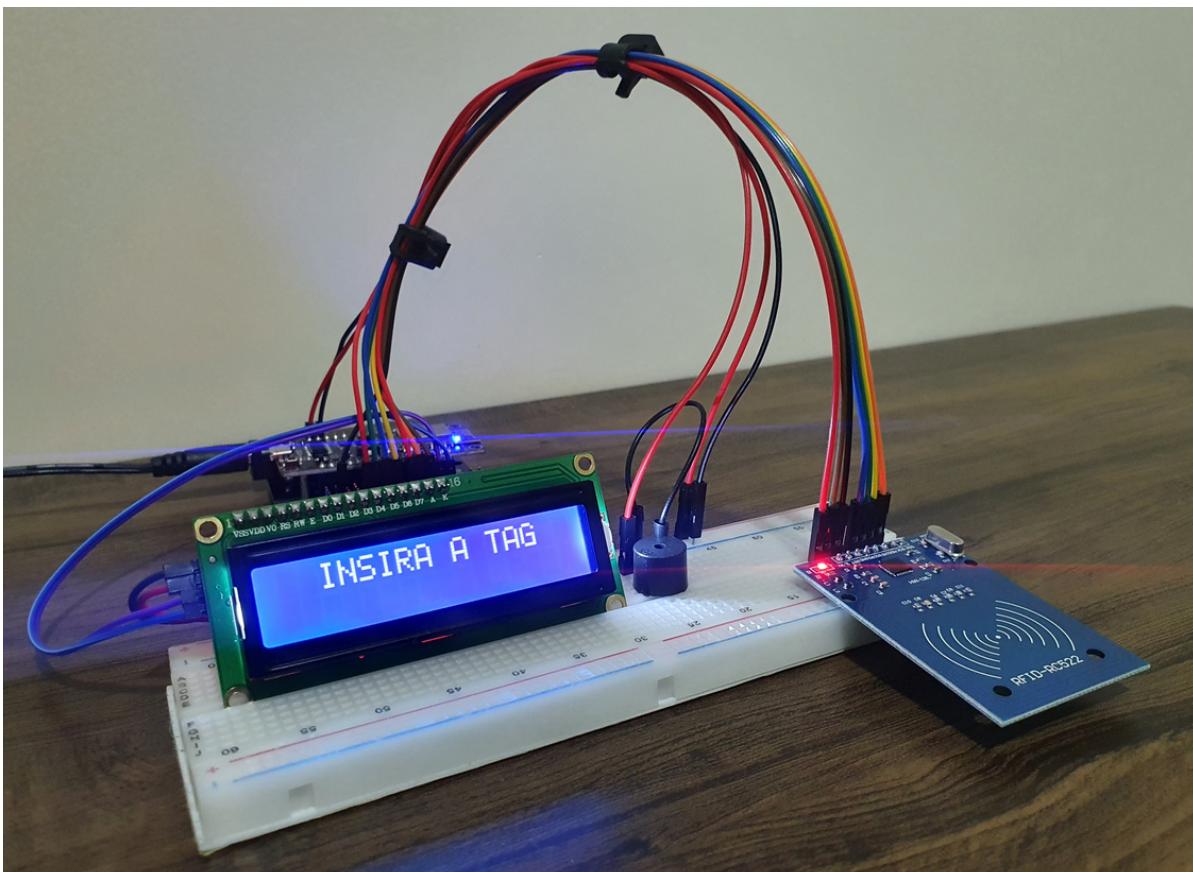
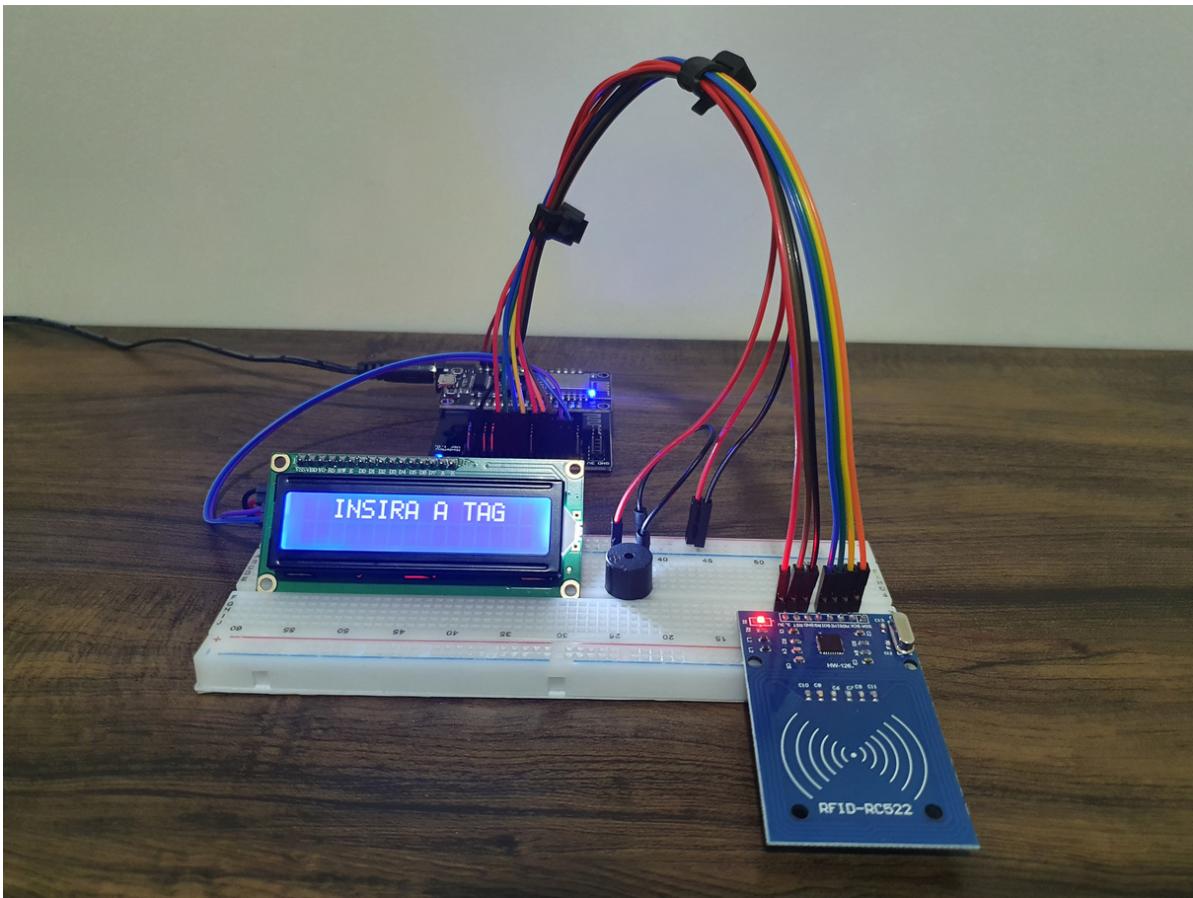
O Pino **GND** do **Display** deve ser conectado ao **GND** do circuito.

O **Pino VCC** no **positivo** do circuito (**Vin**).

O **Pino SDA** deve estar conectado no **Pino Digital D2**.

O **Pino SCL** deve estar conectado no **Pino Digital D1**.

Círcito Na Prática



Capturando as UIDs

Precisamos saber a **UID** da **Tag** para cadastrarmos na programação de **validação de Tags**.

Para isso, vamos utilizar a programação abaixo:

OBS: É necessário instalar a **biblioteca MFRC522**, [Clique Aqui](#) para baixar.

```
// Bibliotecas Necessárias

#include <SPI.h>
#include <MFRC522.h>

#define SS_PIN D4 //Pino SDA
#define RST_PIN D3 //Pino de Reset

MFRC522 rfid(SS_PIN, RST_PIN);

void setup () {

    Serial.begin(9600); //Iniciando a Serial

    SPI.begin();
    rfid.PCD_Init();
}

void loop () {
    if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial())
        return;

    // Código Responsável por gerar a Tag

    String strID = "";
    for (byte i = 0; i < 4; i++) {
```

```

strID +=  

    (rfid.uid.uidByte[i] < 0x10 ? "0" : "") +  

    String(rfid.uid.uidByte[i], HEX) +  

    (i!=3 ? ":" : "");  

}  

strID.toUpperCase();  
  

Serial.print("Identificador da tag: "); //Imprime texto na Serial  

Serial.println(strID); //Imprime o UID da Tag  
  

rfid.PICC_HaltA();  

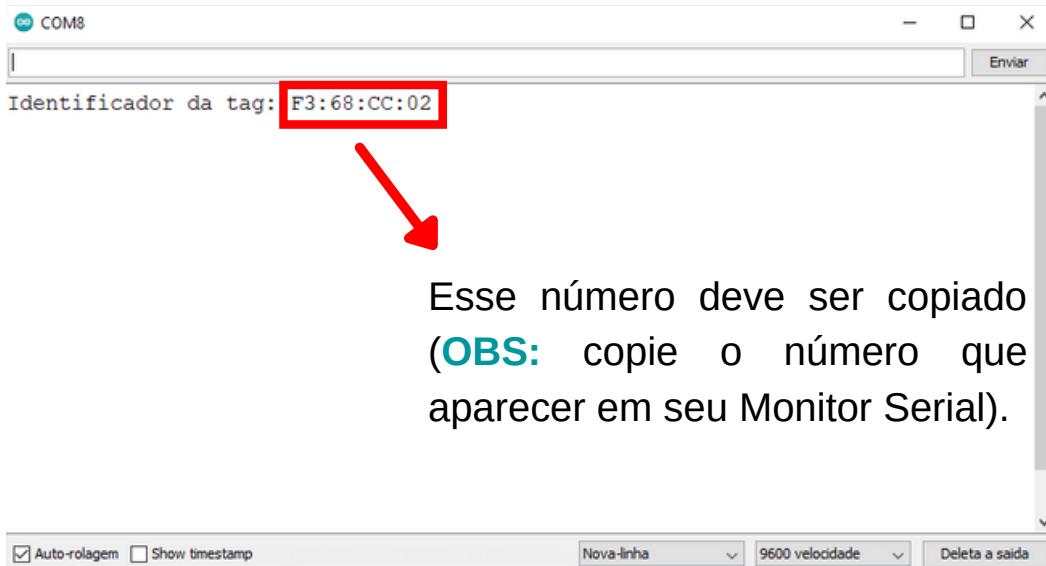
rfid.PCD_StopCrypto1();
}

```

Baixe o código do projeto [Clicando Aqui!](#)

Envie a programação para o **NodeMCU**, deixe ele conectado via **cabo USB** ao computador e pressione as teclas **Ctrl + Shift + m** para abrir o **Monitor Serial** na **IDE do Arduino**.

Após abrir o **Monitor Serial** aproxime as **tags** no **Módulo RFID** e aparecerá no Monitor Serial a **UID**, escolha uma e **copie**, vamos usar na **programação final**.



Esse número deve ser copiado
(OBS: copie o número que
aparecer em seu Monitor Serial).

Programação

// Incluindo Bibliotecas Necessárias

```
#include <Wire.h>
#include <SPI.h>
#include <MFRC522.h>
#include <LiquidCrystal_I2C.h>
```

```
LiquidCrystal_I2C Display = LiquidCrystal_I2C(0x27, 16, 2);
```

```
#define SS_PIN D4 //Pino SDA no Pino Digital D4
#define RST_PIN D3 //Pino RST Pino Digital D4
```

```
MFRC522 rfid(SS_PIN, RST_PIN);
```

```
int Buzzer = D8;
```

```
void setup(){
```

```
lcd.begin(16,2); // linhas e colunas do display
lcd.init();
lcd.backlight();
```

```
// Iniciando as bibliotecas
```

```
Wire.begin();
SPI.begin();
rfid.PCD_Init();
```

```
// Definindo o Buzzer como Saída Digital
```

```
pinMode(Buzzer, OUTPUT);
digitalWrite(Buzzer, LOW);
```

```

// Inicia o Display

lcd.setCursor(0,0); // coluna 0, linha 0
lcd.print(" INSIRA A TAG"); // Escreve no Display

}

void loop() {
    leituraRfid();
}

//Função de validação da TAG
void leituraRfid(){
    if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial())
        return;

    String strID = "";
    for (byte i = 0; i < 4; i++) {
        strID += (rfid.uid.uidByte[i] < 0x10 ? "0" : "") +
            String(rfid.uid.uidByte[i], HEX) +
            (i!=3 ? ":" : "");
    }
    strID.toUpperCase();

    // Insira a UID da tag capturada anteriormente

    if (strID.indexOf("Insira a UID da tag capturada anteriormente") >= 0) { // Se a
TAG for correta

        int qtd_bips = 2; //definindo a quantidade de bips
        for(int j=0; j<qtd_bips; j++){

        // Ligando o Buzzer com uma frequência de 1500 hz

```

```

lcd.clear();
lcd.setCursor(0,0); // coluna 0, linha 0
lcd.print("ACESSO LIBERADO"); // Escreve no Display

tone(Buzzer,1500);
delay(100);
noTone(Buzzer);
delay(100);
}

delay(2000);

lcd.clear(); // Limpa Display
lcd.setCursor(0,0); // coluna 0, linha 0

lcd.print(" INSIRA A TAG"); // Escreve no Display

}else{ //SENÃO, FAZ (CASO A TAG LIDA NÃO SEJÁ VÁLIDA)
  int qtd_bips = 1; //definindo a quantidade de bips
  for(int j=0; j<qtd_bips; j++){

//Ligando o Buzzer com uma frequênciade 500 hz

lcd.clear(); // Limpa Display
lcd.setCursor(0,0); // coluna 0, linha 0
lcd.print(" ACESSO NEGADO "); // Escreve no Display

tone(Buzzer,500);
delay(500);

// Desligando o Buzzer

noTone(Buzzer);
delay(1000);
}

```

```

lcd.clear(); // Limpa Display

lcd.setCursor(0,0); // coluna 0, linha 0
lcd.print(" INSIRA A TAG");

}

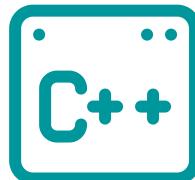
}

rfid.PICC_HaltA();
rfid.PCD_StopCrypto1();
}

```

Baixe o código do projeto [Clicando Aqui!](#)

Quer ver o funcionamento ?? [Clique aqui !!](#)



Sugestões

Nesse projeto você também pode adicionar uma **mini trava elétrica** e incluir um código para que uma **notificação** seja enviada para o **celular** caso alguém tente acessar com uma **Tag errada**.

Seu funcionamento será da seguinte maneira: quando a **Tag** for validada a **trava é acionada**, caso contrário a trava não é acionada e um **SMS** ou **e-mail** é enviado para o celular alertando sobre o ocorrido !!!

O que você achou dessa sugestão, Gostou? Acha que devemos criar esse projeto ?

Caso você queira ver um tutorial dessa sugestão envie seu e-mail para [contato@arduinoomega.com](mailto: contato@arduinoomega.com), Se recebermos **muitos e-mails** vamos fazer esse projeto !!

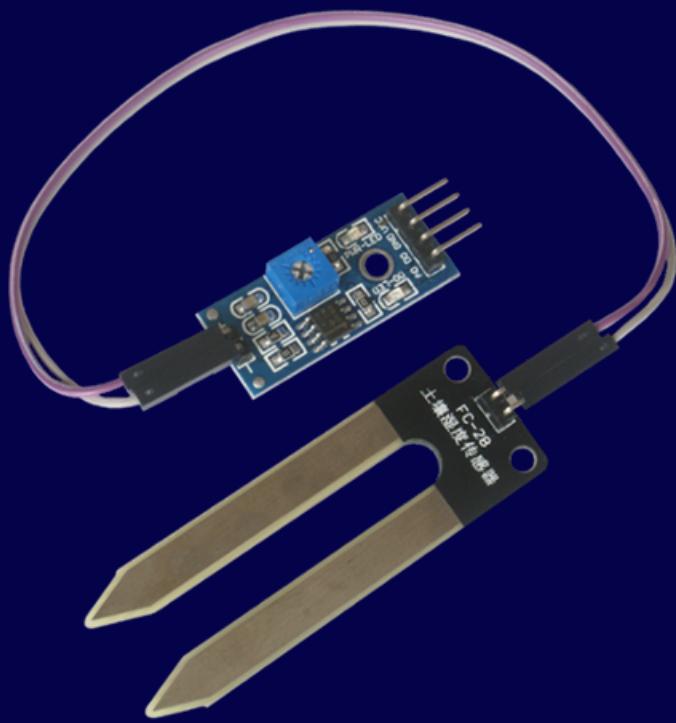


15 Irrigação Automática

Chegamos no último projeto do nosso E-book, vamos desenvolver um sistema de irrigação automático para plantas, para isso será necessário utilizarmos um **Módulo Sensor de Umidade de Solo** e uma **Mini Bomba de água**.

Sensor de Umidade de Solo

Este **Sensor de Umidade do Solo Higrômetro** foi feito para detectar as variações de umidade no solo, sendo que quando o solo está seco a saída do sensor fica em estado alto (**HIGH**), e quando úmido em estado baixo (**LOW**). Sua tensão de operação é de **3,3 a 5V**.



O limite entre **seco** e **úmido** pode ser ajustado através do potenciômetro presente no sensor que regulará a **saída digital D0**. Contudo para ter uma resolução melhor é possível utilizar a saída analógica A0 e conectar a um conversor AD, como a presente no **Arduino** por exemplo.

Mini Bomba de água

A **Mini Bomba de Água RS385** foi criada especialmente para o desenvolvimento de projetos de prototipagem, incluindo automação residencial e protótipos robóticos baseados em plataformas **microcontroladoras** como por exemplo o **NodeMCU** e o **Arduino**.

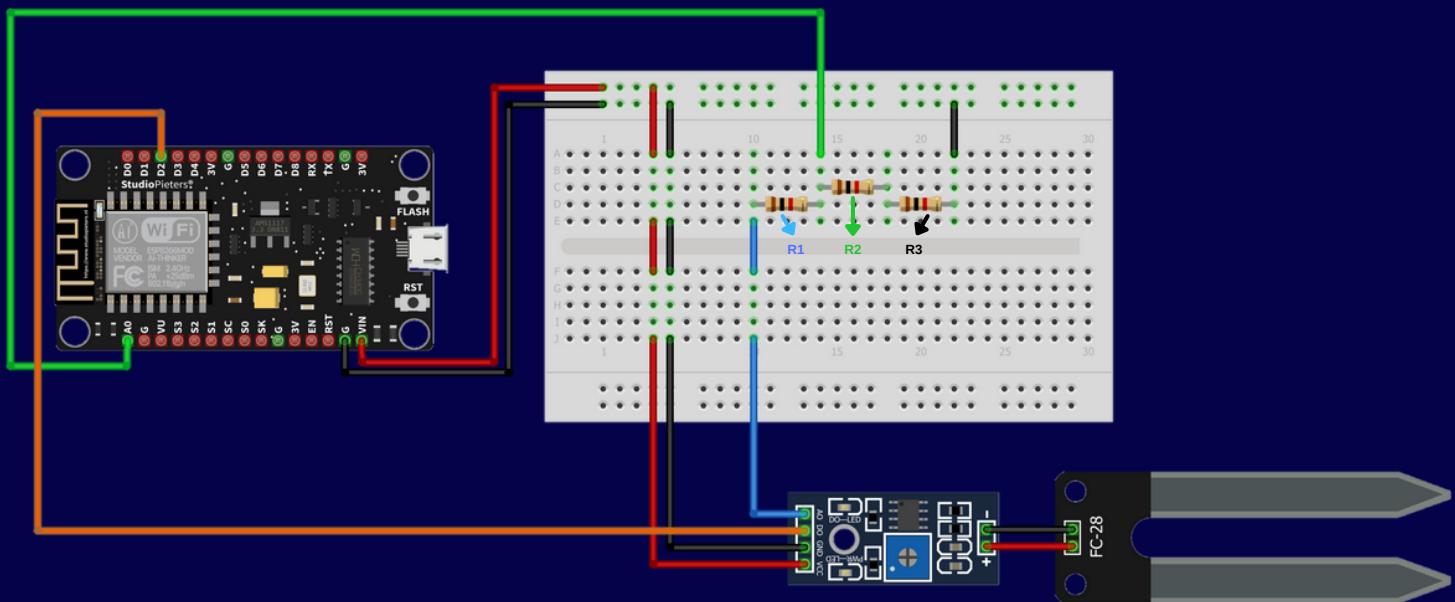


Com um motor de tamanho adequado a **mini bomba** é capaz de impulsionar entre **1500 ml** a **2000 ml** por minuto, sendo destacada pela sua eficiência e precisão durante sua execução em conjunto com o **NodeMCU**, por exemplo.

É frequentemente utilizada em desenvolvimento de **carrinhos** ou **robôs bombeiros**, **robôs hidráulicos**, **irrigadores automáticos** no caso de automação residencial, enfim sua criatividade que dará a aplicação final para este incrível acessório.

A **mini bomba** opera com tensão entre **9V** a **15V** e permite elevação máxima de até **3 metros** e altura de aspiração de até **2 metros**.

Círcito



Primeiro é necessário conectar dois jumper no sensor e conectar na **placa de controle**, no **negativo** e **positivo**.

O pino **GND** da **placa de controle** deve ser conectado no pino **G** do microcontrolador.

O pino **VCC** da **placa de controle** no pino **Vin** do **NodeMCU**.

O **Pino D0** do sensor deve estar conectado ao **Pino D2** do **NodeMCU**.

O **Pino A0** do sensor fornece uma tensão de **5V**, como o **NodeMCU** suporta somente **3.3V** é necessário fazer um **regulador de tensão**, para isso vamos utilizar **3 resistores** de **1KΩ** em **série**, dessa forma a tensão de saída será de **3.3V**.

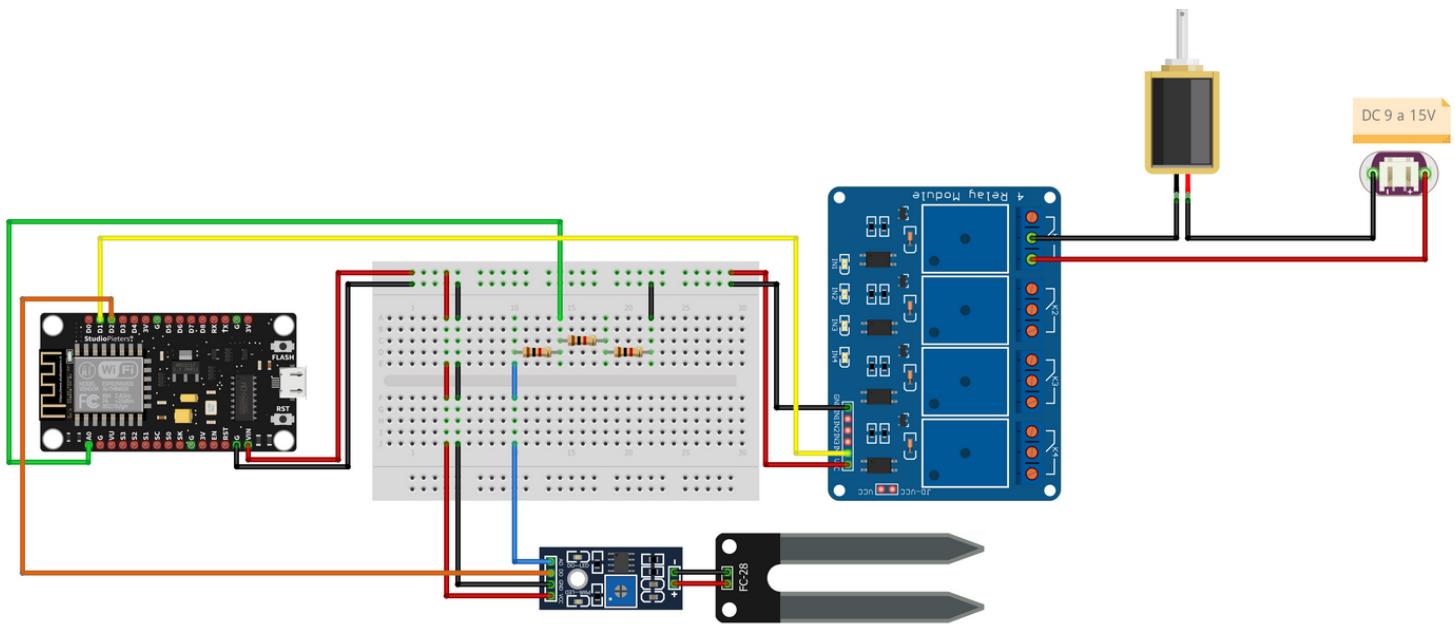
O **Pino A0** do **sensor** deve ser conectado ao primeiro **resistor (R1)**.

O segundo **resistor (R2)** precisa estar ligado ao pino **A0** do **NodeMCU**.

O terceiro **resistor (R3)** deve ser ligado ao **GND** do **círcito**.

Círculo Completo

Vamos adicionar o **Relé** e a **Mini Bomba** no circuito.



É necessário conectar **Vin** do circuito no pino **VCC** do Relé.

Conecte o GND do circuito no pino **GND** do **Módulo Relé**.

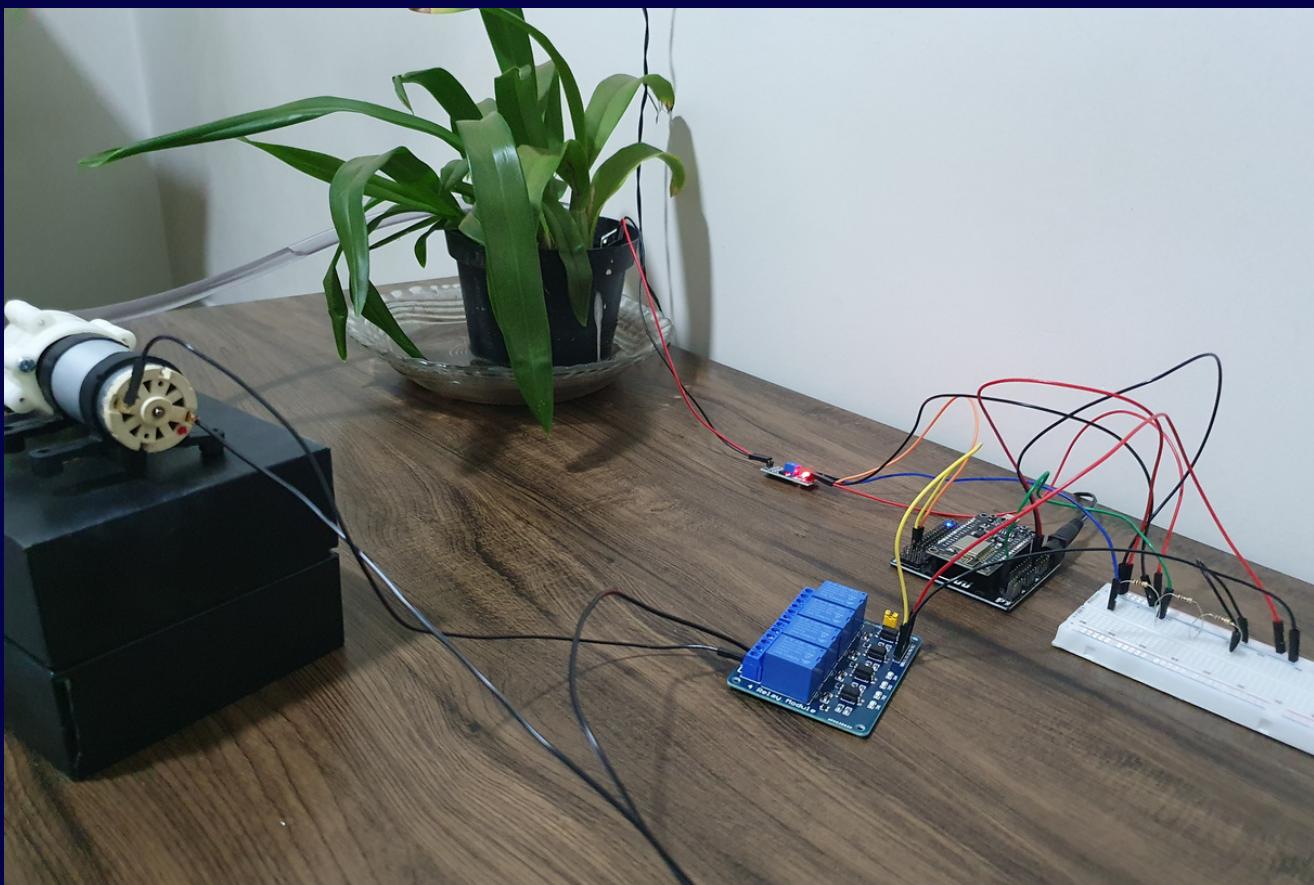
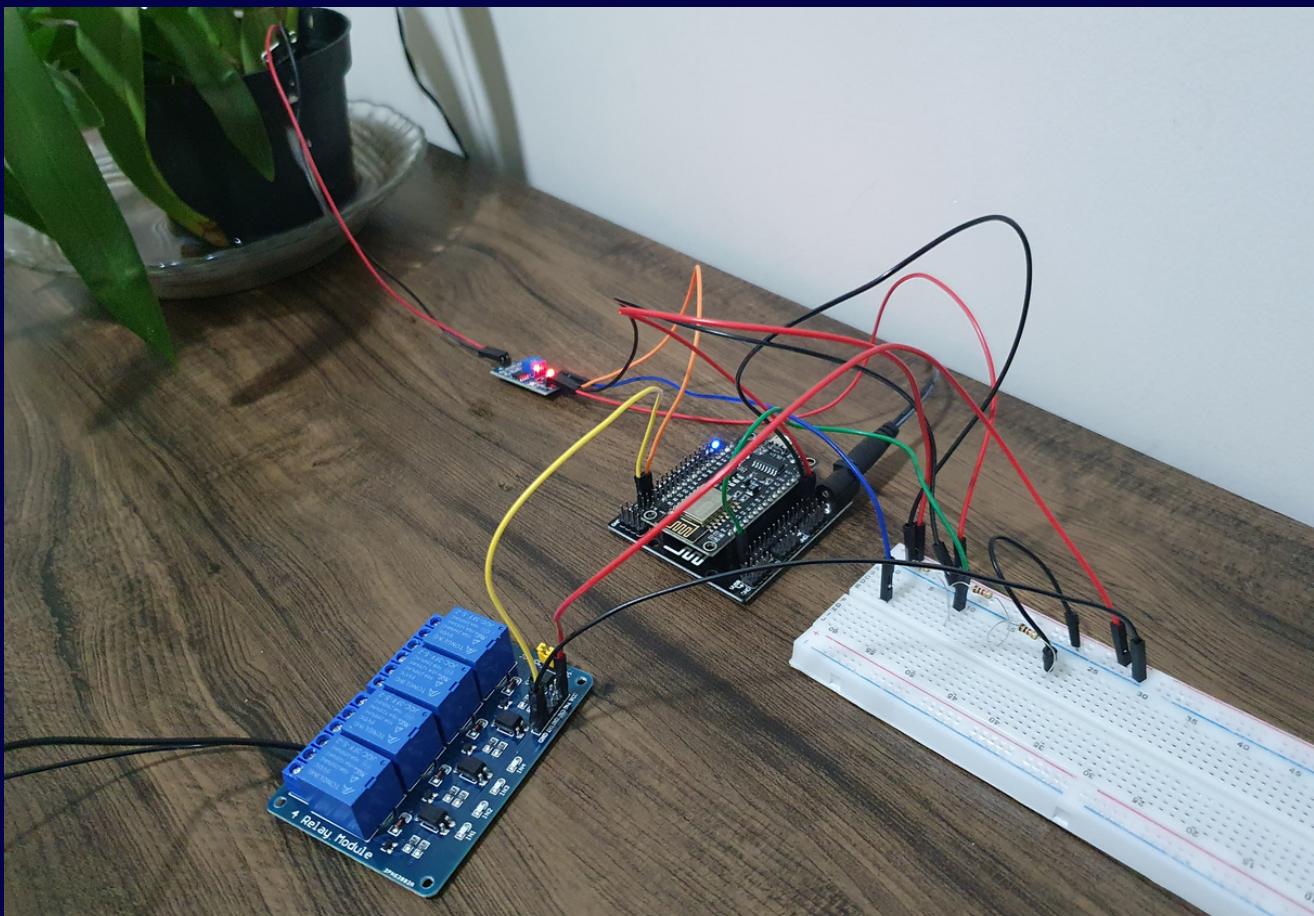
O **Pino Digital D1** do **NodeMCU** deve estar no **IN1** do Relé.

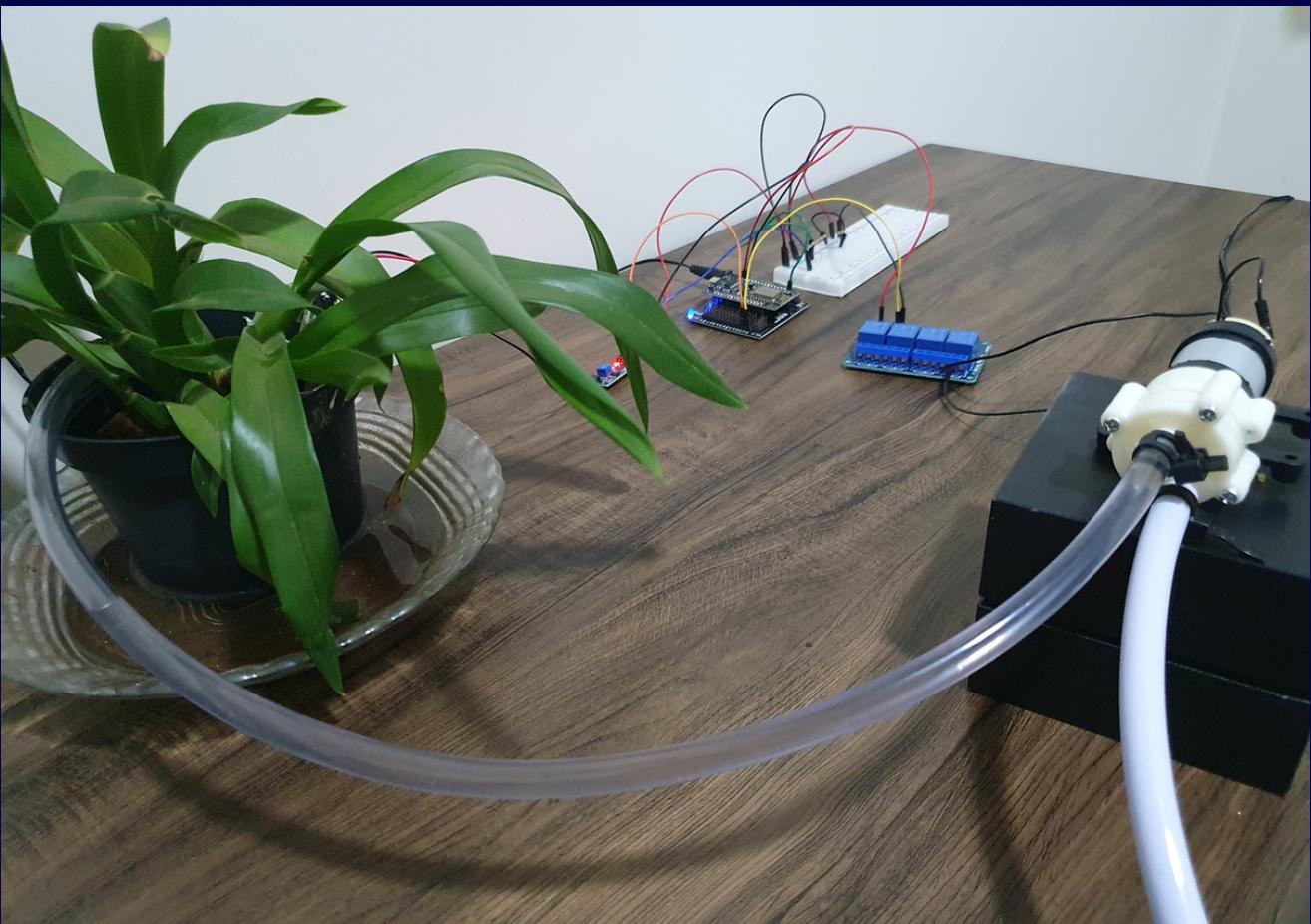
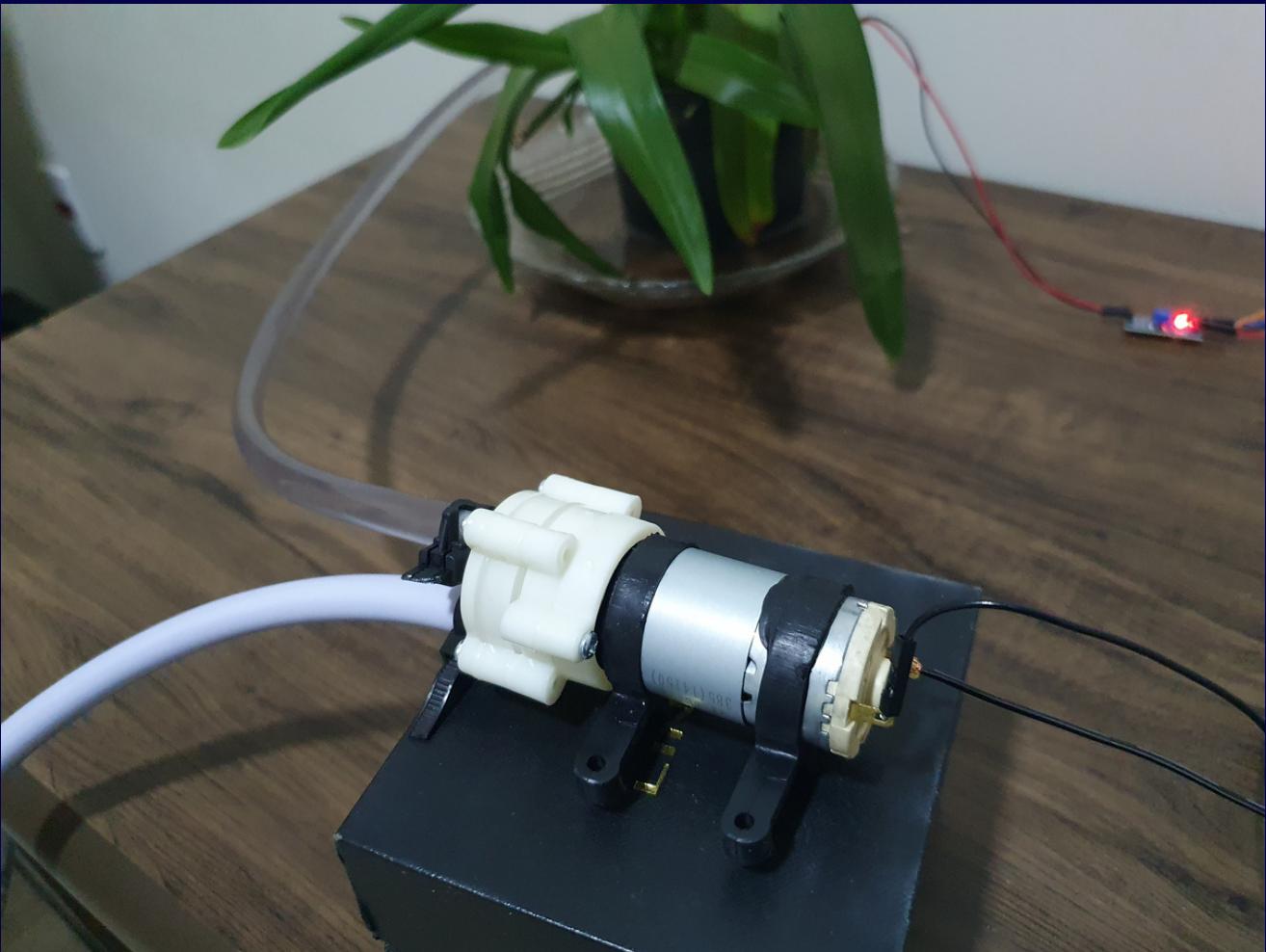
Um terminal da **Mini Bomba** deve estar conectado no **contato aberto do relé K1** e outro em uma fonte de **9 a 15V**.

O **contato fechado do relé K1** deve estar conectado diretamente na fonte de **9 a 15V**.

Nesse projeto, também vamos utilizar o aplicativo **Blynk**, para nos avisar quando a umidade da terra está baixa e a bomba será ligada. **Você pode utilizar o mesmo projeto feito no sensor de chuva !!**

Círcito na Prática





Programação

```
int PinoAnalogico = A0; // Define o pino A0 como Pino Analógico do sensor
int PinoDigital = D2; // Define pino D2 como Pino Digital do Sensor

int Rele = D1; // Pino Digital D1 como Relé

int EstadoSensor = 0;
int UltimoEstSensor = 0;

int ValAnalogIn; // Valor analógico no código

void setup() {

Serial.begin(9600);
pinMode(Rele, OUTPUT); // Declara o Rele como Saída Digital
pinMode(PinoDigital, INPUT);
}

void loop() {

ValAnalogIn = analogRead(PinoAnalogico);
int Porcento = map(ValAnalogIn, 1023, 0, 0, 100); // Traforma o valor analógico
em porcentagem

Serial.print("Umidade: "); // Imprime o símbolo no valor
Serial.print(Porcento); // Imprime o valor em Porcentagem no monitor Serial
Serial.println("%");

if (Porcento <= 76) { // Se a porcentagem for menor ou igual à 76%. OBS: Você
pode alterar essa porcentagem

Serial.println("Irrigando Planta"); // Imprime no monitor serial
```

```
digitalWrite(Rele, LOW); // Aciona Relé  
  
}else { // Caso contrario  
  
Serial.println("Planta Irrigada"); // Imprime a no monitor serial  
digitalWrite(Rele, HIGH); // Desliga Relé  
  
delay (1000);  
}  
}
```

Baixe o código do projeto [Clicando Aqui !](#)

Quer ver o funcionamento ?? [Clique aqui !!](#)

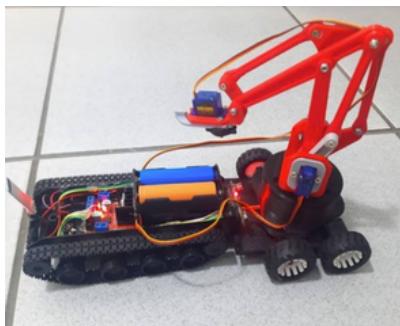


16 Finalização

Nessa apostila abordamos a **introdução a Internet das coisas** para iniciantes, com diversos projetos, nosso intuito foi desvendar um pouco sobre o mundo da Internet das Coisas e te instigar a conhecer cada vez mais sobre essa área que a cada dia vem crescendo.

Caso você tenha interesse em conhecer mais projetos de eletrônica usando o **NodeMCU**, a **Eletrônica Ômega** disponibiliza vários outros tutoriais de forma gratuita em nosso blog, temos vários outros tutoriais para você mergulhar no mundo da IoT!

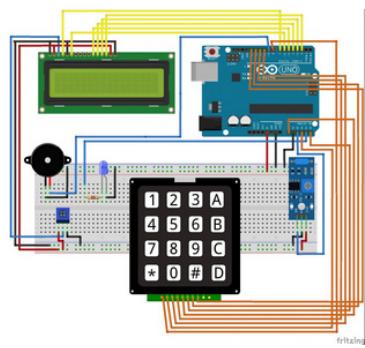
Veja abaixo exemplos de alguns tutoriais que você encontra no blog:



[Tanque Reboque Bluetooth](#)



[Braço Robótico Controlado por voz](#)



[Sistema de Alarme Codificado](#)



[Sistema de Acesso com RF ID](#)

E-book Robótica

Venha aprender robótica com a gente! Com o [Kit Arduino Robôs](#) e o nosso [E-book Robótica para Iniciantes](#) você irá aprender sobre [Arduino](#), eletrônica, física, programação e robótica! Bora lá?

Elaboramos o [E-book](#) e o [Kit Arduino Robôs](#), para serem totalmente amigáveis aos iniciantes, todo o material descrito no [E-book](#) está contido neste Kit, todos os projetos descritos no [E-book](#) são possíveis de montar com os componentes deste kit!



Em nosso [E-book](#) há o detalhamento de **4 projetos de robôs**: o [Veículo Autônomo](#), o [Veículo Controlado por App](#), [Braço Robótico Controlado por Joystick](#) e [Braço Robótico Controlado por App](#)! Você irá encontrar o código fonte completo, com explicação de todos os pontos do código, diagrama elétrico mostrando como fazer a ligação de cada componentes, detalhamentos dos principais problemas que podem ocorrer na montagem, entre várias dicas.

Quem escreveu o E-book ?



Rangel Gabriel

Formado em **Eletroeletrônica** pelo **SENAI Shunji Nishimura**, tem experiência com eletrônica e microcontroladores.

E-mail: rangelarena@gmail.com



Eletrônica Ômega



A **Eletrônica Ômega** é uma loja virtual sediada em BH/MG, especializada em **Arduino** e **componentes eletrônicos**.

Temos a missão de **promover transformação cultural e social através da educação**, inserindo o maior número possível de pessoas no movimento maker, levando acesso a tecnologia para todos os interessados, e **mostrando que é possível desenvolver suas ideias através da tecnologia**.

Caso queria falar com a gente não deixe de mandar seu email para [contato@arduinoomega.com](mailto: contato@arduinoomega.com).

Redes Sociais:



ARDUINO ÔMEGA
BLOG

Referências

Tensão:

<https://www.todamateria.com.br/tensao-eletrica/> - acesso em 16/06/2021

<https://www.mundodaeletrica.com.br/tensao-eletrica-x-voltagem/> - acesso em 16/06/2021.

Corrente:

<https://mundoeducacao.uol.com.br/fisica/corrente-eletrica.htm> - acesso em 16/06/2021.

<https://brasilescola.uol.com.br/fisica/corrente-eletrica.htm> - acesso em 16/06/2021.

<https://www.mundodaeletrica.com.br/o-que-e-corrente-eletrica/> - acesso em 16/06/2021.

LED:

<https://athoselectronics.com/o-que-e-led-diodo-emissor-luz/#:~:text=O%20LED%C3%A9%20um%20Diodo,energia%20el%C3%A9trica%20em%20energia%20luminosa.&text=Como%20se%20trata%20de%20um,corrente%20el%C3%A9trica%20em%20um%20sentido.> - acesso em 25/06/2021.

Resistores:

<https://brasilescola.uol.com.br/fisica/resistores.htm#:~:text=Resistores%20s%C3%A3o%20dispositivos%20usados%20para,esses%20s%C3%A3o%20conhecidos%20como%20diel%C3%A9tricos.> - acesso em 25/06/2021.

IoT

<https://inforchannel.com.br/2021/04/27/internet-das-coisas-e-protecao-de-dados/>
- acesso em 10/08/2021.

DHT11

<https://www.baudaelectronica.com.br/sensor-de-umidade-e-temperatura-dht11.html> - **acesso em 15/08/2021.**

IR

<https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-kit-controle-remoto-infravermelho> - **acesso em 17/08/2021.**

