

Engenharia de Computação

LABORATÓRIO

Coordenação: Prof. Dr. Angelo Sebastião Zanini

INSTITUTO MAUÁ DE TECNOLOGIA



Time de Computação

Coordenação:

Prof. Dr. Ângelo Sebastião Zanini

Professores:

Prof. Nuncio Perrella

Prof. Everson Denis

Corpo Técnico:

Eng. Gabriel Godoy

Enga. Ana Helena

Eng. Matheus Coelho

Meio Ambiente – Poluição Sonora

Poluição sonora: excesso de sons que afetam os seres humanos e o ambiente.

Fontes Sonoras: Trânsito, indústrias, construção civil, eventos, aeroportos, entre outros.

Impactos

Saúde

- Fisiológicos: perda auditiva, aumento da pressão arterial, estresse.
- Psicológicos: irritabilidade, ansiedade, dificuldades de concentração e distúrbios do sono.
- Desempenho Cognitivo: redução da capacidade de aprendizado e produtividade.

Meio Ambiente

- Fauna: alteração nos padrões de comunicação dos animais, afetando suas interações sociais e reprodutivas.
- Flora: embora menos evidente, há estudos que indicam impactos no crescimento de plantas expostas a altos níveis de ruído.

Níveis Sonoros

A medição de níveis sonoros é feita utilizando escala em **decibéis (dB)**.

A **escala em decibéis (dB)** é usada para quantificar níveis sonoros e comparar a intensidade do som. Como o ouvido humano percebe sons de forma não linear, a escala dB é baseada em **razões logarítmicas**, tornando-a mais adequada para representar nossa percepção auditiva.

O equipamento mais comum para essa medição é o **medidor de nível de pressão sonora (Decibelímetro ou Sonômetro)**, que capta as variações de pressão do ar causadas pelas ondas sonoras.

Decibel

O **decibel (dB)** é uma unidade de medida relativa que expressa a razão entre dois valores de pressão sonora.

Ele é definido pela seguinte fórmula:

$$L = 10 \cdot \log \left(\frac{P^2}{P_0^2} \right) = 20 \cdot \log \left(\frac{P}{P_0} \right)$$

Onde:

L = Nível de pressão sonora em **dB**

P = Pressão sonora medida (**Pa**)

P₀ = Pressão de referência (geralmente 20 μPa, o limiar da audição humana)

Características da Escala Logarítmica

A escala de decibéis é **logarítmica**, o que significa que:

- Um **aumento de 10 dB** representa um som **10 vezes mais intenso**.
- Um **aumento de 20 dB** representa um som **100 vezes mais intenso**.
- Um **aumento de 30 dB** representa um som **1.000 vezes mais intenso**.

Exemplo:

- **40 dB** (conversa em tom baixo)
- **50 dB** (ruído de escritório) → 10 vezes mais intenso que 40 dB
- **60 dB** (conversa normal) → 100 vezes mais intenso que 40 dB

Por isso, um aumento de apenas alguns decibéis pode significar uma grande diferença na percepção do som.

Níveis Sonoros

Tipo de área	Diurno (7h-22h)	Noturno (22h-7h)
Área estritamente residencial	50 dB	45 dB
Área predominantemente residencial	55 dB	50 dB
Área mista (residencial/comercial)	60 dB	55 dB
Área comercial/administrativa	65 dB	60 dB
Área industrial	70 dB	65 dB

ABNT NBR 10.151/2019

Níveis Sonoros

Nível (dB)	Fonte do Som	Efeito no Ouvido Humano
10 dB	Respiração, estúdio de gravação silencioso	Quase imperceptível
20 dB	Sussurro, biblioteca silenciosa	Muito baixo
30 dB	Quarto silencioso à noite	Tranquilo
40 dB	Sala de estar, conversação baixa	Som ambiente confortável
50 dB	Escritório silencioso, chuva fraca	Pouco incômodo
60 dB	Conversa normal, ar-condicionado	Nível moderado
70 dB	Trânsito leve, aspirador de pó	Límite confortável por longos períodos
80 dB	Trânsito intenso, liquidificador	Pode ser incômodo
90 dB	Moto, secador de cabelo	Prolongado pode causar danos
100 dB	Show de rock, buzina	Exposição prolongada pode causar perda auditiva
110 dB	Serra elétrica, boate	Muito alto e prejudicial
120 dB	Motor de avião a 30m, trovão	Dores no ouvido
130 dB	Tiro de arma de fogo, fogos de artifício	Risco imediato de dano auditivo
140+ dB	Explosão, motor de jato próximo	Limiar da dor

Dano Auditivo e Segurança

A exposição prolongada a níveis sonoros elevados pode causar **danos auditivos permanentes**.

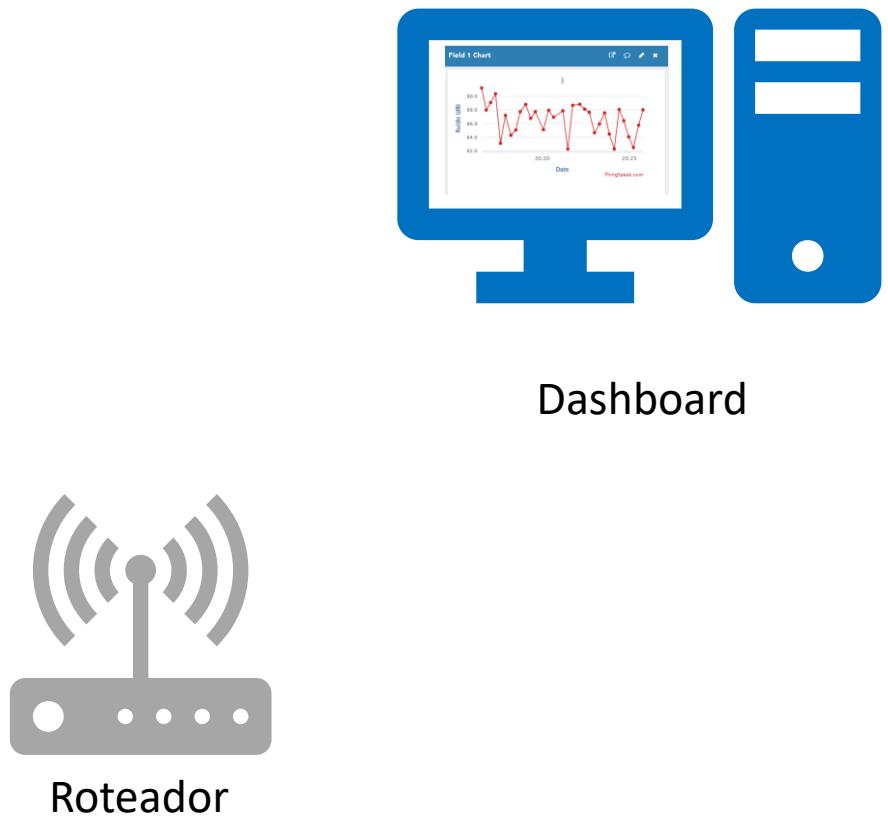
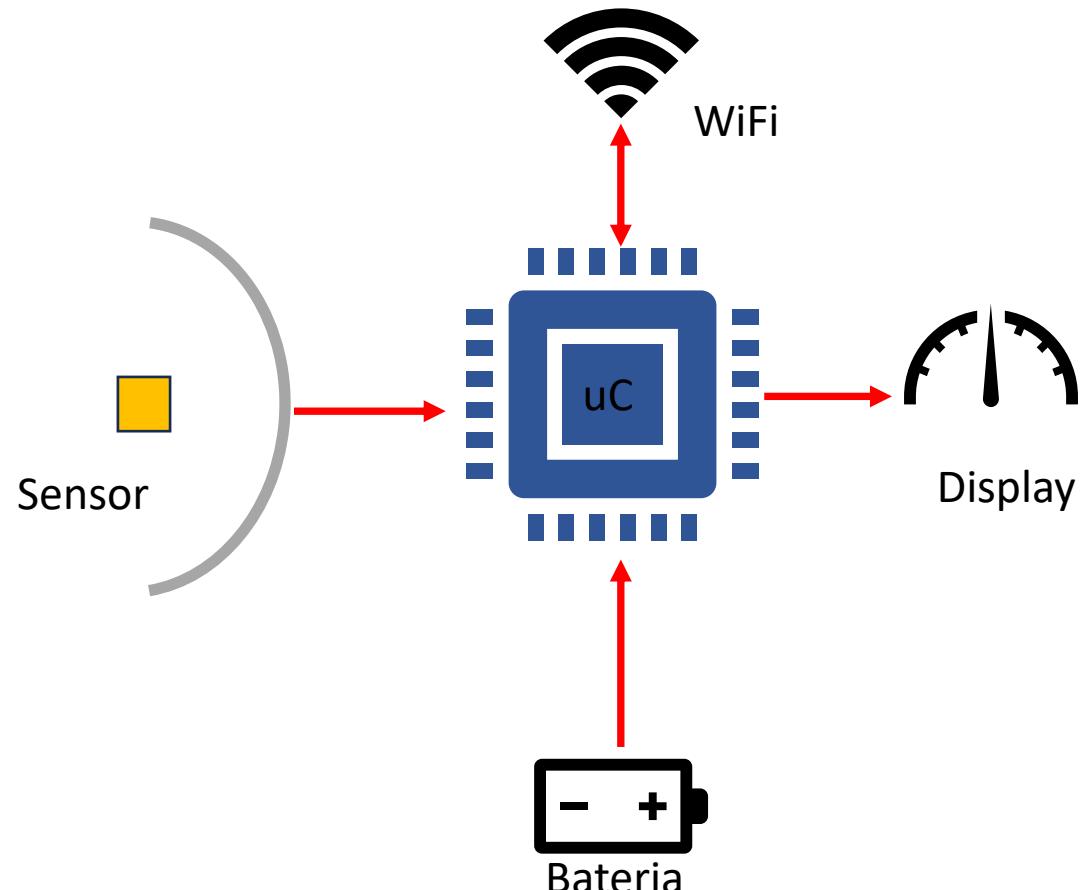
Segundo normas de segurança (como a **NR-15** no Brasil):

- **Acima de 85 dB** → Pode causar danos auditivos após exposição prolongada.
- **Acima de 120 dB** → Pode causar dor e danos imediatos.

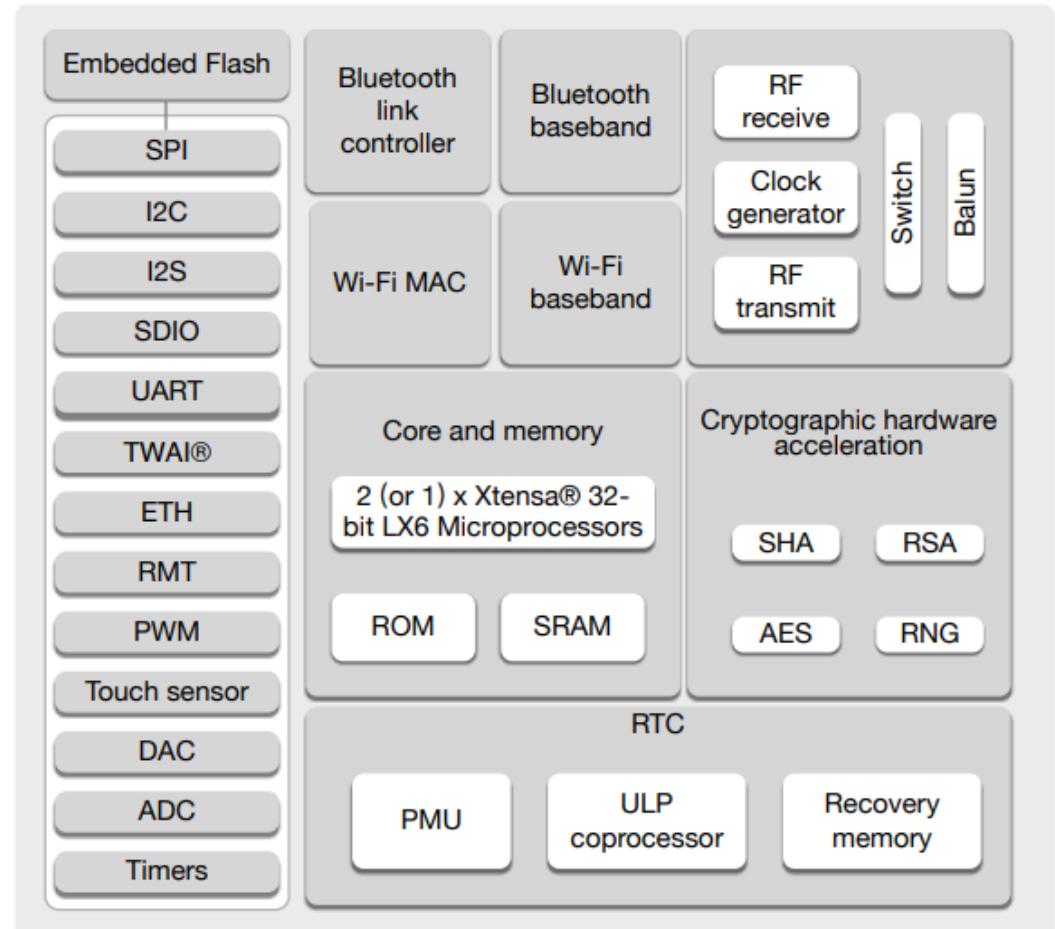
Projeto

- Criar sistema de tempo real para monitoramento sonoro gerando alarmes e alertas.

Diagrama de Blocos do Projeto



uC - ESP32



https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

ESP32 DOIT DEV KIT V1 - HW Básico



DOIT ESP32 DEVKIT V1 PINOUT

	Pin	Function	ESP Pin
POWER	pin1	VIN	VDD 3V3
GND	pin2	GND	GND
Serial Pin	pin3	HS2_CLK	GPIO15
Analog Pin	pin4	SD_CLK	GPIO14
Control	pin5	HSPI_CLK	GPIO13
Physical Pin	pin6	MTMS	GPIO12
Port Pin	pin7	HS2_DATA2	SD_DATA2
Touch Pin	pin8	HSPI_MISO	RTC_GPIO15
DAC Pin	pin9	MTDI	ADC2_CH5
	pin10	HS2_DATA3	SD_DATA3
	pin11	HSPI_MOSI	RTC_GPIO14
	pin12	MTCK	ADC2_CH4
	pin13		
	pin14		
	pin15		
	pin16		
	pin17		
	pin18		
	pin19		
	pin20		
	pin21		
	pin22		
	pin23		



playelek.com



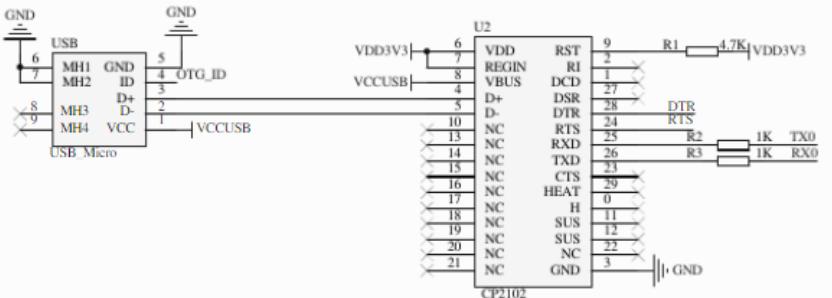
22-1116-2016

VER 1

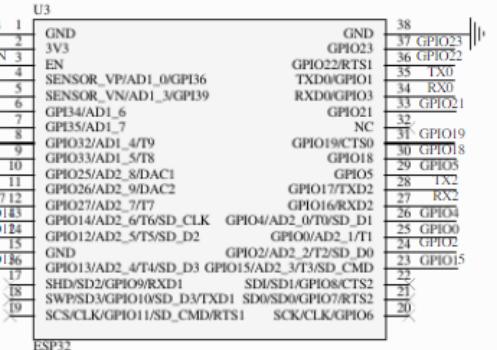
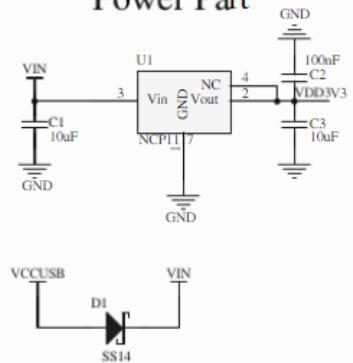


ESP32 DOIT DEV KIT V1 - HW Básico

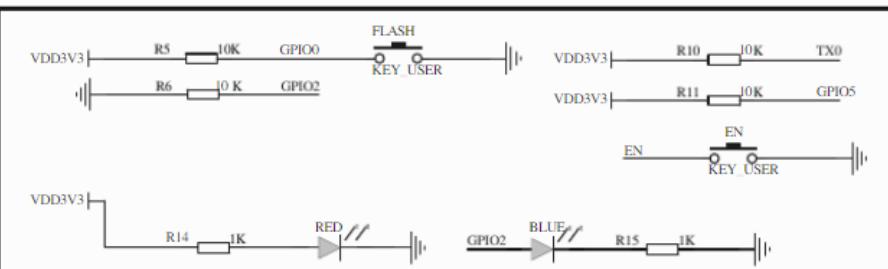
USB Part



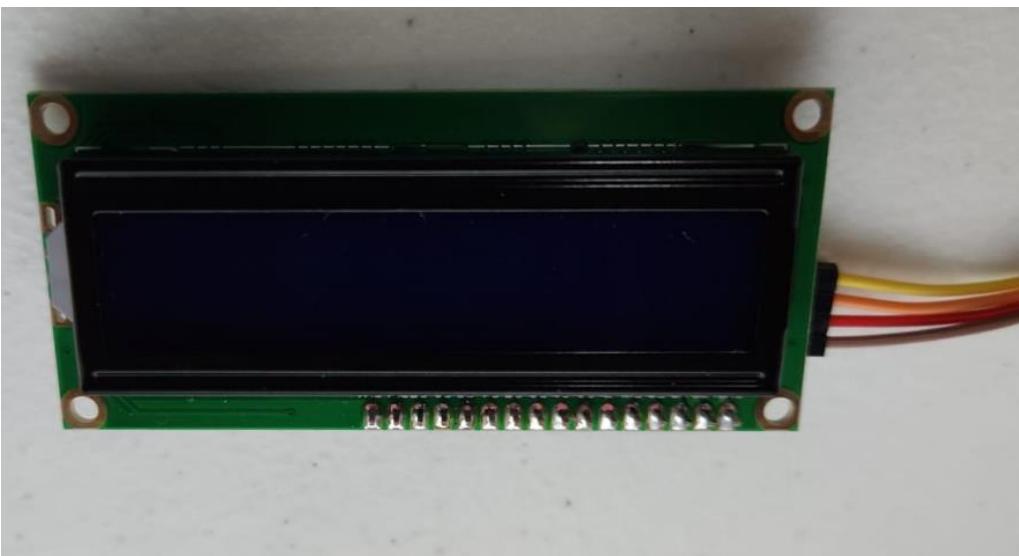
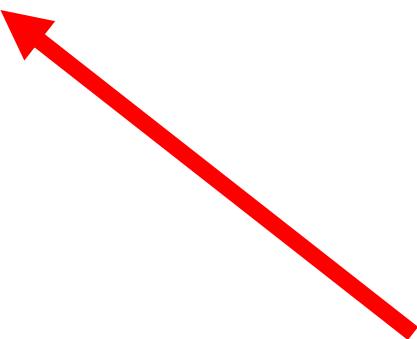
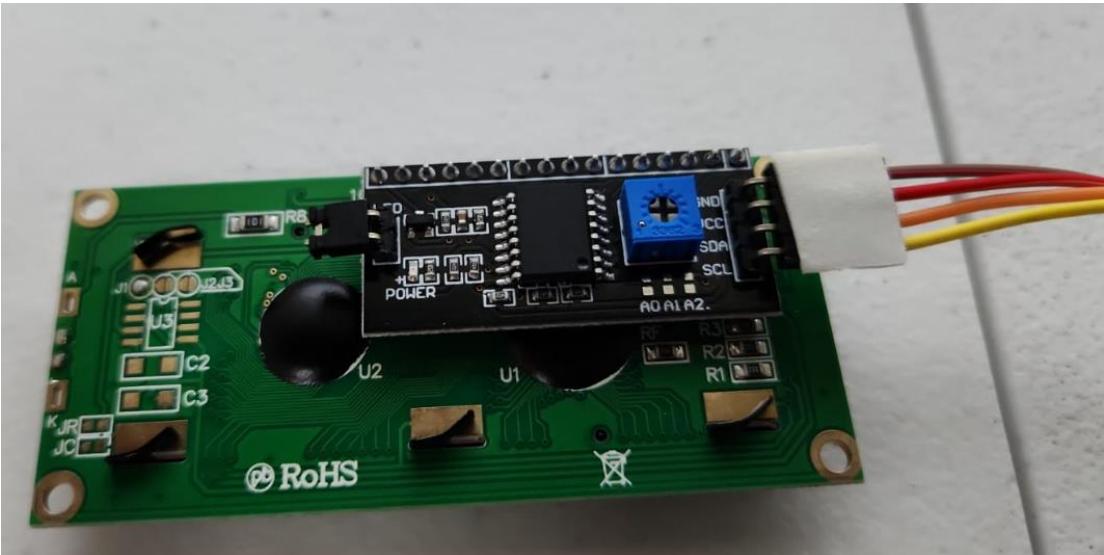
Power Part



J1	PIN1	PIN2	PIN3	PIN4	PIN5	PIN6	PIN7	PIN8	PIN9	PIN10	PIN11	PIN12	PIN13	PIN14	PIN15
J2	VDD3V3	GND	GPIO1	GPIO2	GPIO4	GPIO5	TX2	GPIO5	GPIO18	GPIO19	GPIO21	GPIO2	TX0	GPIO22	GPIO23
THT_Male_P_1x15	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

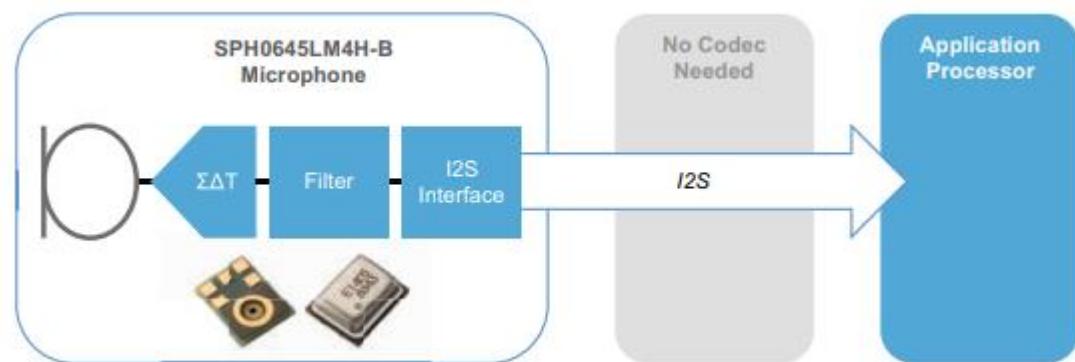
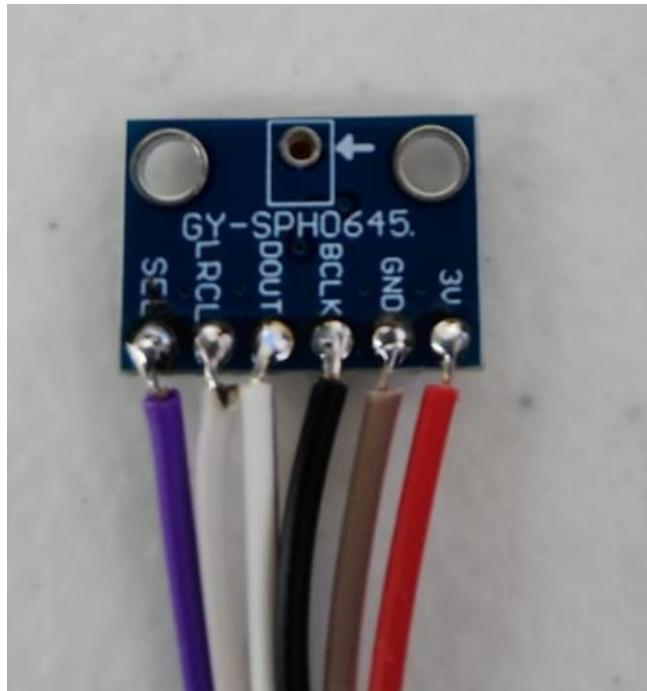
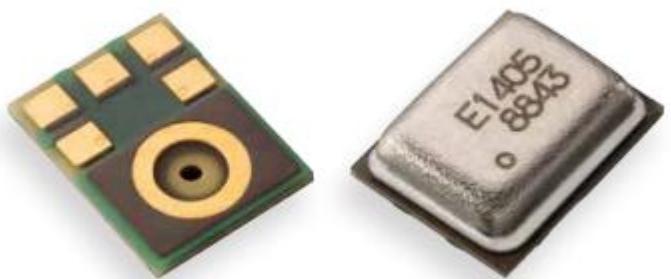
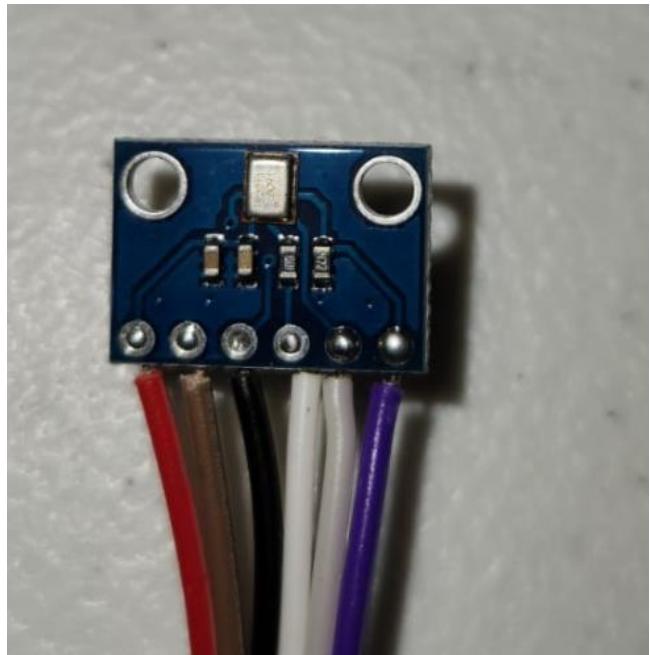


Display

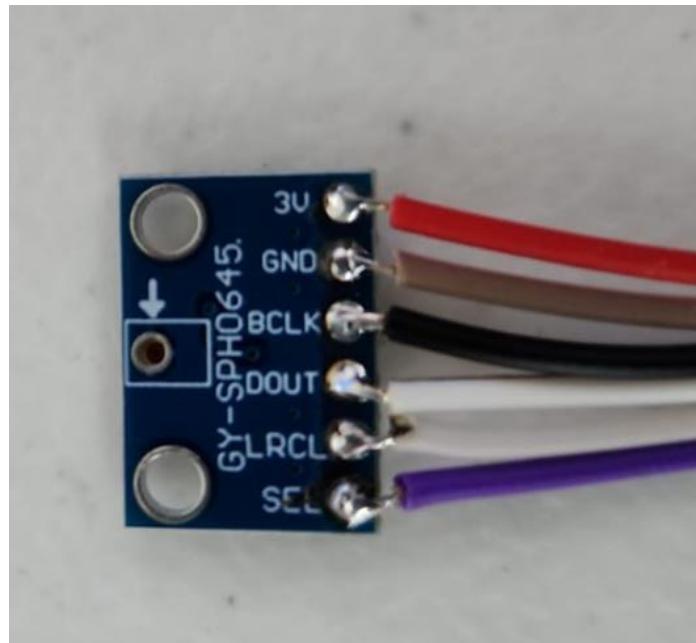


Vermelho → Vin (5V)
Marrom → GND
Laranja → D21 (ESP32)
Amarelo → D22 (ESP32)

Sensor SPH0645

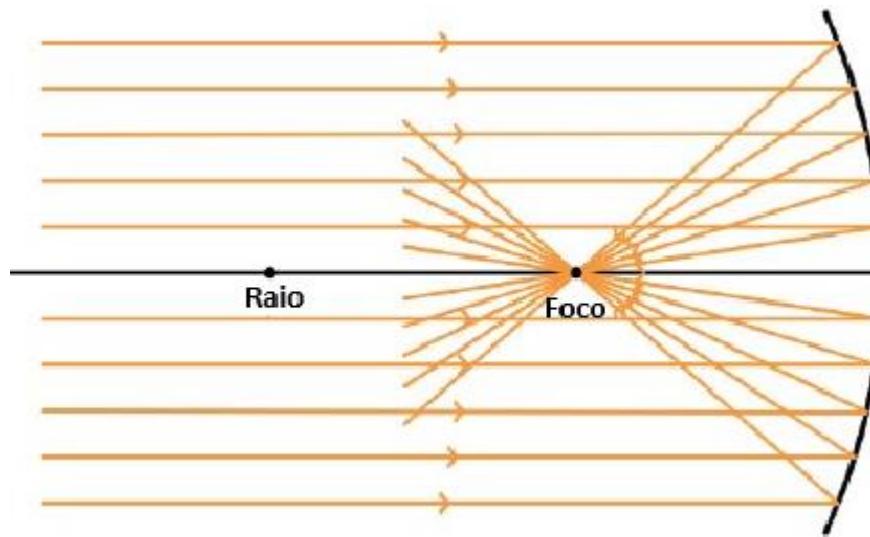


Sensor SPH0645



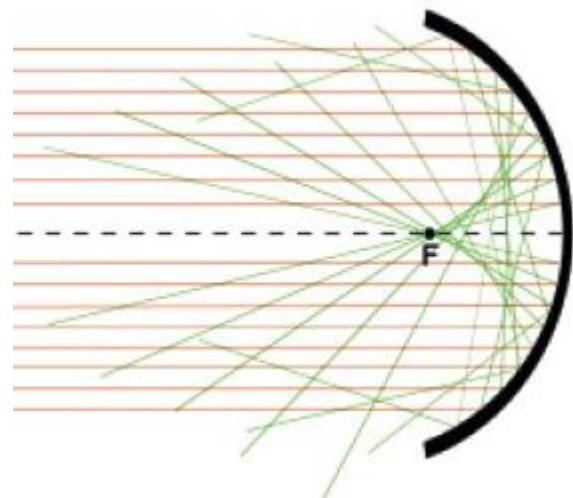
Vermelho	→ 3V3
Marrom	→ GND
Preto	→ D23 (ESP32)
Branco	→ D32 (ESP32)
Cinza	→ D33 (ESP32)
Roxo	→ GND

Refletor Esférico Gaussiano

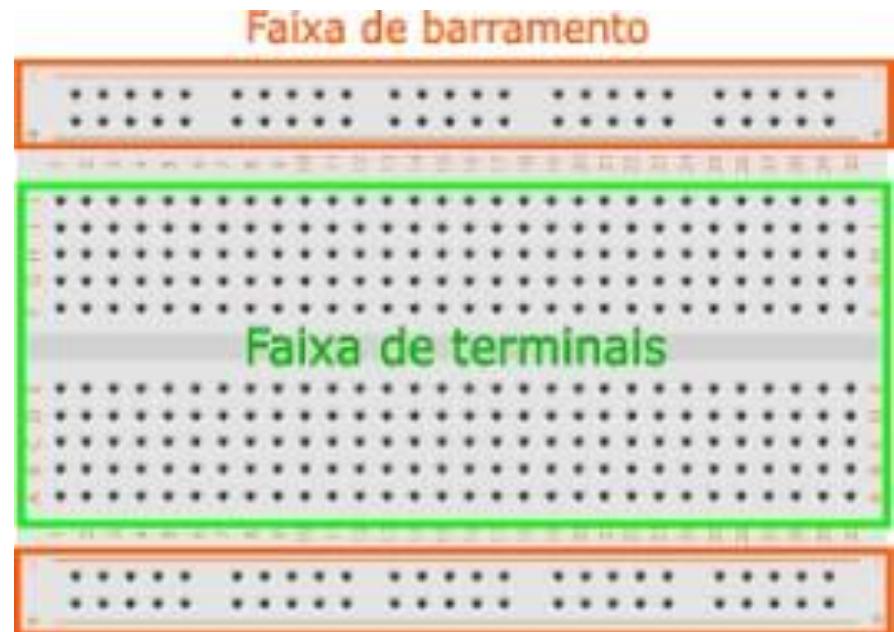
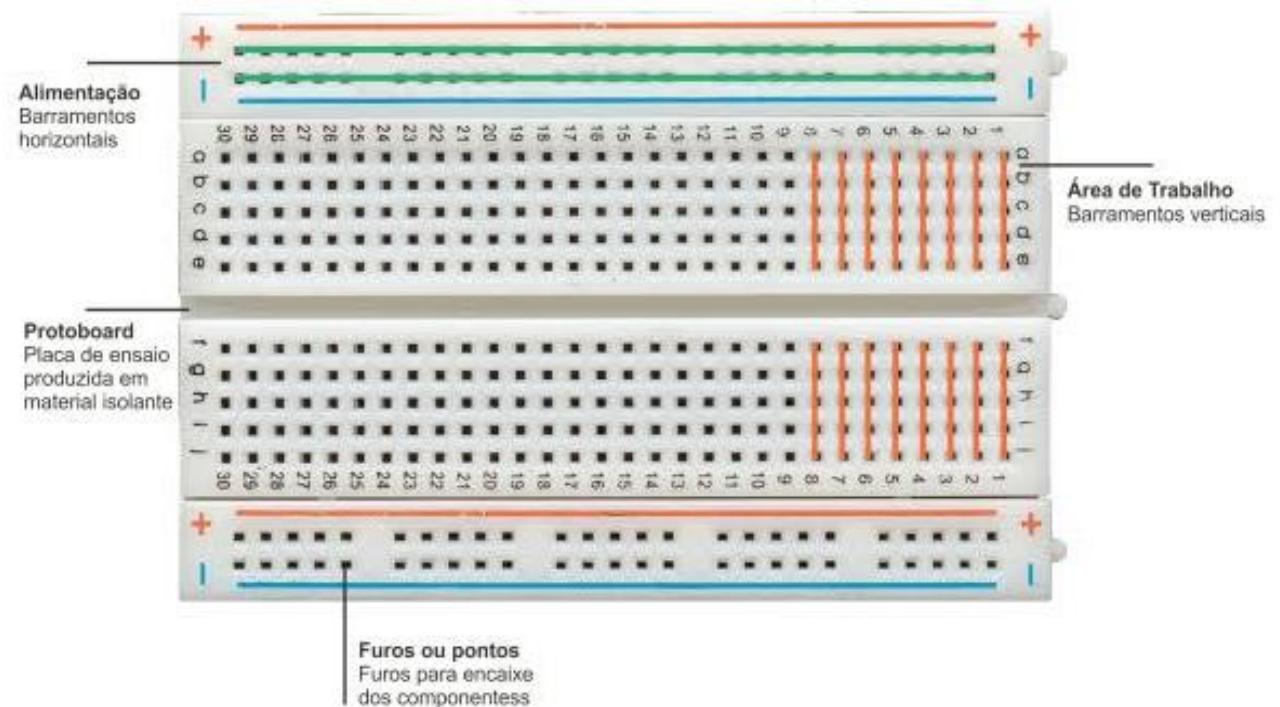


$$\text{Foco} = \text{Raio} / 2$$

Refletor Esférico Gaussiano



Protoboard





Arduino Uno

Desenvolvedor	<ul style="list-style-type: none"> Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis. Baseado no <i>Processing</i>, de Casey Reas e Ben Fry. Comunidade Código aberto.
Plataforma	C/C++
Lançamento	2005
Versão estável	1.8.11 (27 de janeiro de 2020; há 21 meses) [1]
Versão em teste	1.9.0
Idioma(s)	67 idiomas ^[2]
Escrito em	Java
Sistema operacional	Microsoft Windows, Linux, Mac OS X ^{[3][4]}
Gênero(s)	Ambiente de desenvolvimento integrado
Licença	<ul style="list-style-type: none"> Software em LGPL ou GPL Hardware em Creative Commons
Estado do desenvolvimento	Ativo
Página oficial	http://www.arduino.cc/en/ (em inglês)

Ambiente de Desenvolvimento (IDE)



Arduino é uma plataforma de prototipagem eletrônica de hardware livre e de placa única, projetada com um microcontrolador Atmel AVR com suporte de entrada/saída embarcados, uma linguagem de programação padrão, a qual tem origem em Wiring, e é essencialmente C/C++.

O objetivo do projeto é criar ferramentas que são acessíveis, com baixo custo, flexíveis e fáceis de se usar por principiantes e profissionais. Principalmente para aqueles que não teriam alcance aos controladores mais sofisticados e ferramentas mais complicadas.

Pode ser usado para o desenvolvimento de objetos interativos independentes, ou ainda para ser conectado a um computador.

Uma típica placa Arduino é composta por um controlador, alguns pinos de E/S digital e analógica, além de uma interface serial ou USB, para interligar-se ao computador, que é usado para programá-la e interagir em tempo real. A placa em si não possui qualquer recurso de rede, porém é comum combinar um ou mais Arduinos deste modo, usando extensões apropriadas chamadas de shields.

A interface do computador é simples, podendo ser escrita em várias linguagens. A mais popular é a Processing, mas outras que podem comunicar-se com a conexão serial são: Max/MSP, Pure Data, SuperCollider, ActionScript e Java.

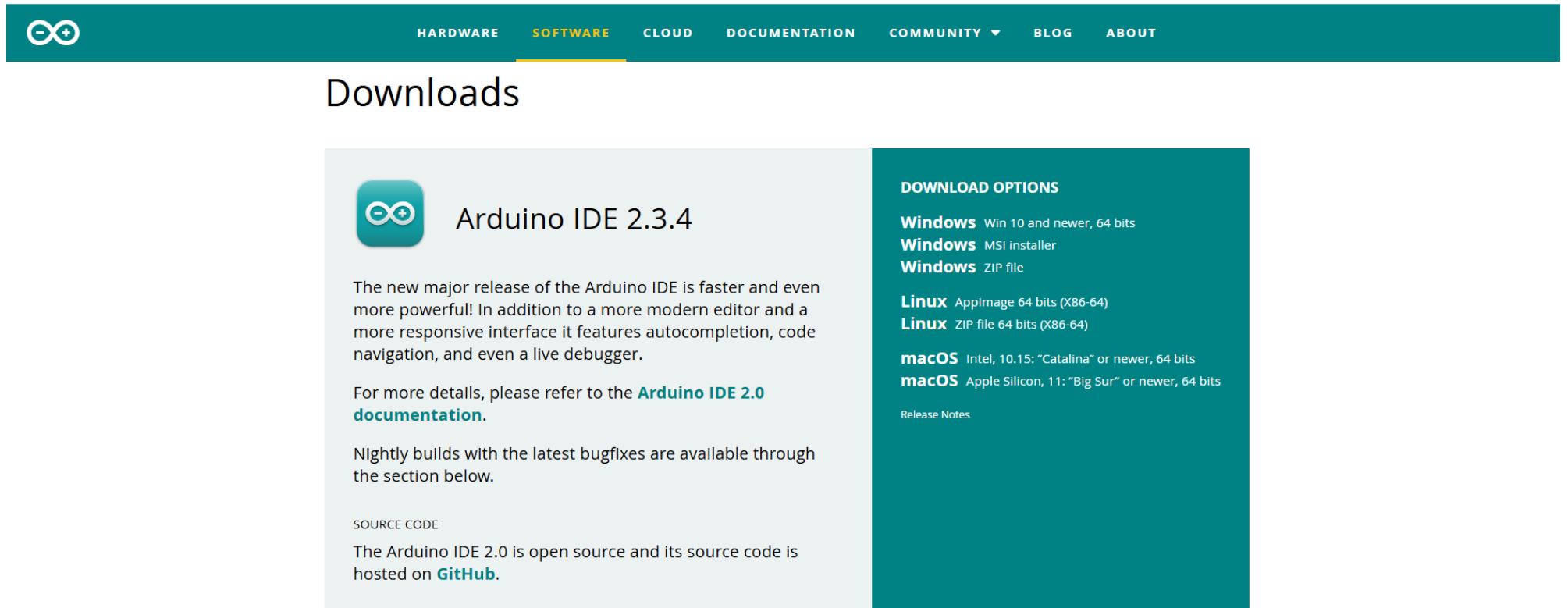
O nome Arduino vem de um bar em Ivrea, Itália, onde alguns dos fundadores do projeto costumavam se reunir.

Fonte: <https://pt.wikipedia.org/wiki/Arduino> (17/11/2021)

Ambiente de Desenvolvimento (IDE)

Fazer Download e instalar Arduino IDE (versão mais recente)

<https://www.arduino.cc/en/Main/Software>



The screenshot shows the Arduino Software (IDE) Downloads page. At the top, there's a navigation bar with links for HARDWARE, SOFTWARE (which is highlighted in yellow), CLOUD, DOCUMENTATION, COMMUNITY ▾, BLOG, and ABOUT. Below the navigation bar, the word "Downloads" is centered. On the left side of the main content area, there's a large image of the Arduino logo (two interlocking circles) and the text "Arduino IDE 2.3.4". A paragraph describes the features of this release, mentioning faster performance, a modern editor, a responsive interface, autocomplete, code navigation, and a live debugger. It also links to the "Arduino IDE 2.0 documentation". Below this, there's a "SOURCE CODE" link and a note about the open-source nature of the IDE, linking to GitHub. On the right side, under the heading "DOWNLOAD OPTIONS", there are links for Windows (Win 10 and newer, 64 bits; MSI installer; ZIP file), Linux (AppImage 64 bits (X86-64); ZIP file 64 bits (X86-64)), and macOS (Intel, 10.15: "Catalina" or newer, 64 bits; Apple Silicon, 11: "Big Sur" or newer, 64 bits). A "Release Notes" link is also present.

Bibliotecas Instaladas

ThingSpeak

Type: All

Topic: All

ThingSpeak by MathWorks...

ThingSpeak Communication Library for Arduino, ESP8266 & ESP32 ThingSpeak (...)

[More info](#)

2.0.1

INSTALL

Arduino ESP32 Boards
by Arduino

Boards included in this package:
Arduino Nano ESP32

[More info](#)

2.0.18-i

INSTALL

LiquidCrystal_I2C frank

Type: All

Topic: All

LiquidCrystal I2C by Frank de Brabander

1.1.2 installed

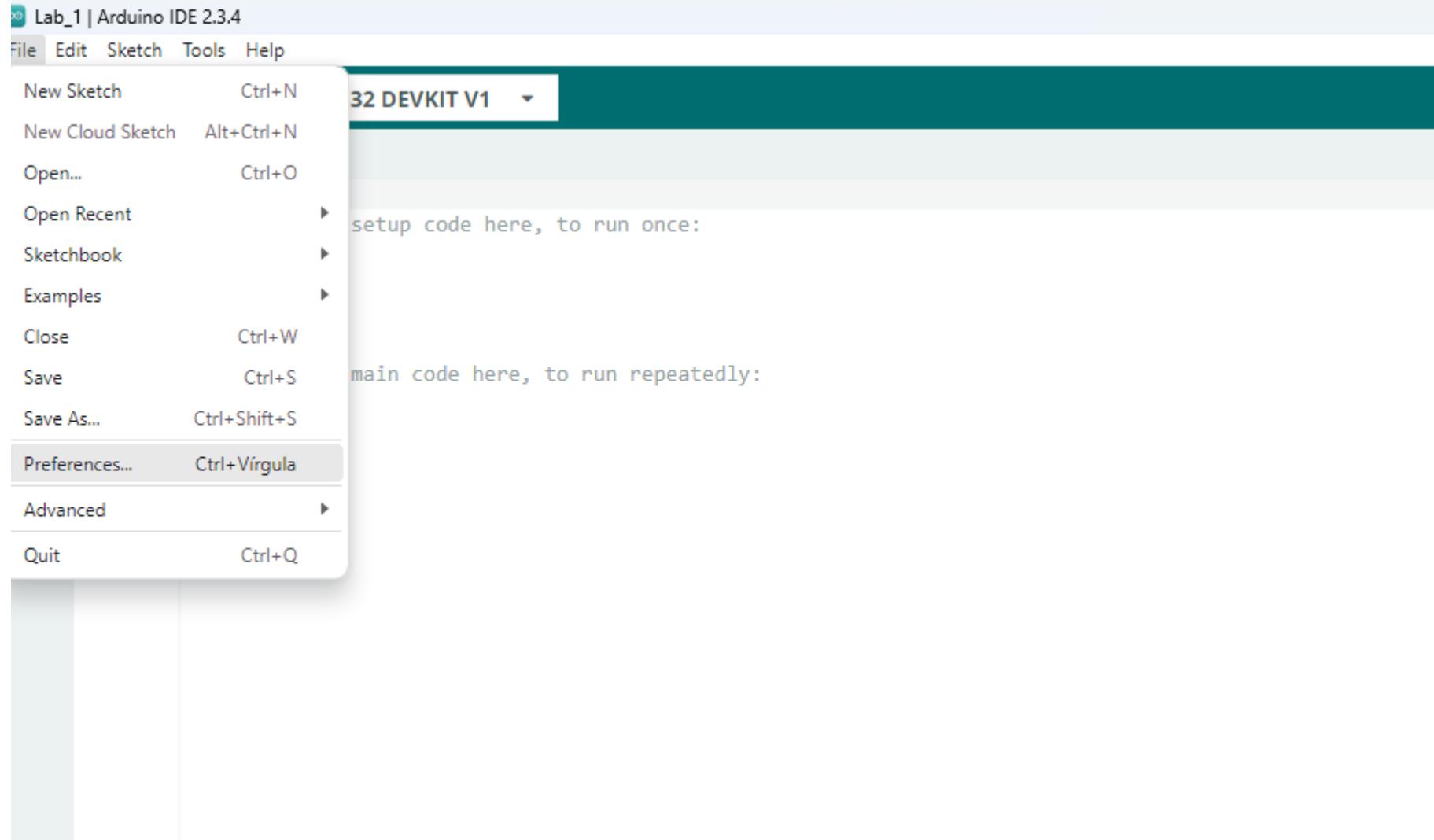
A library for I2C LCD displays. The library allows to control I2C displays with functions extremel...

[More info](#)

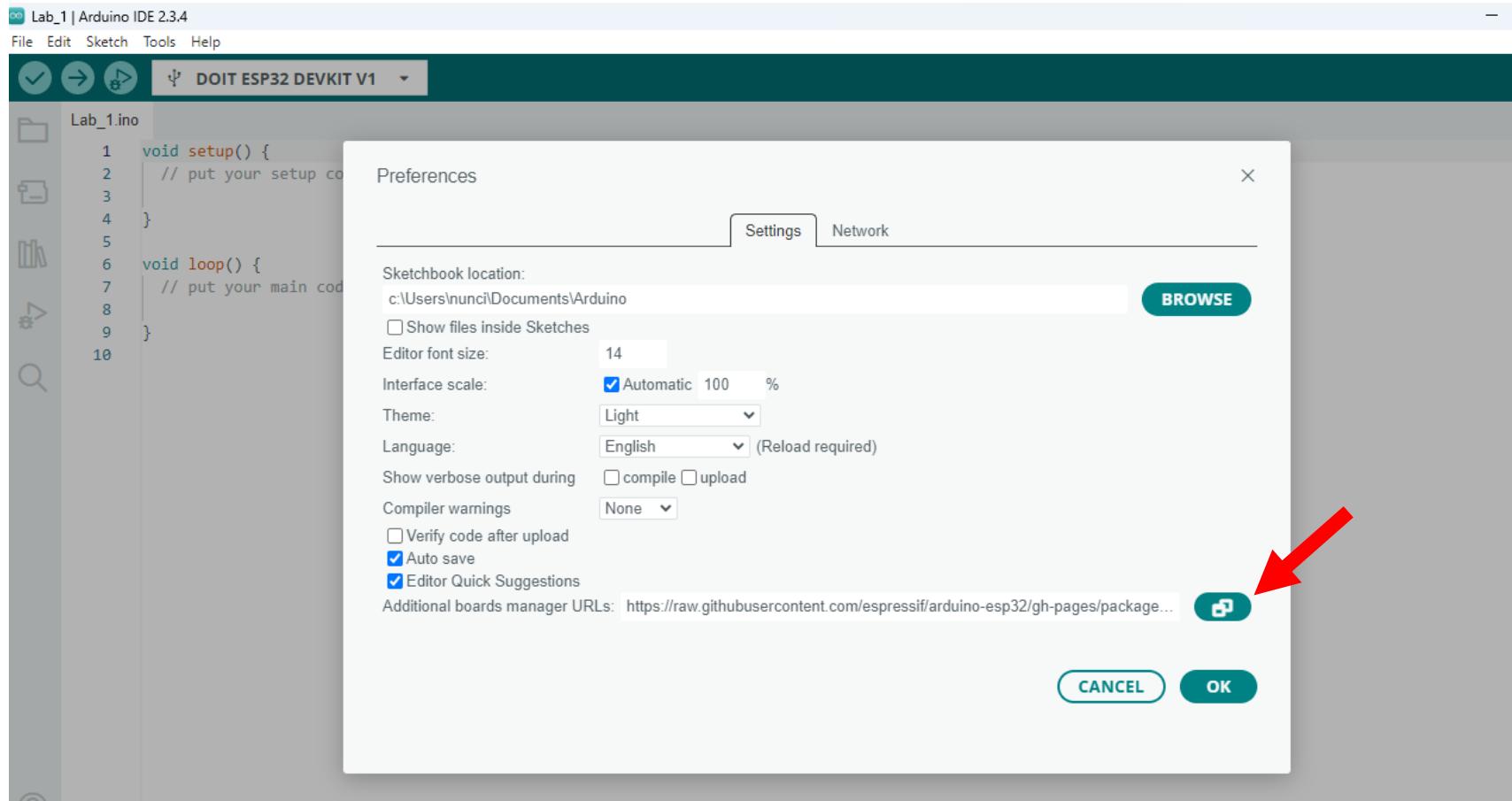
1.1.2

REMOVE

Configurando IDE

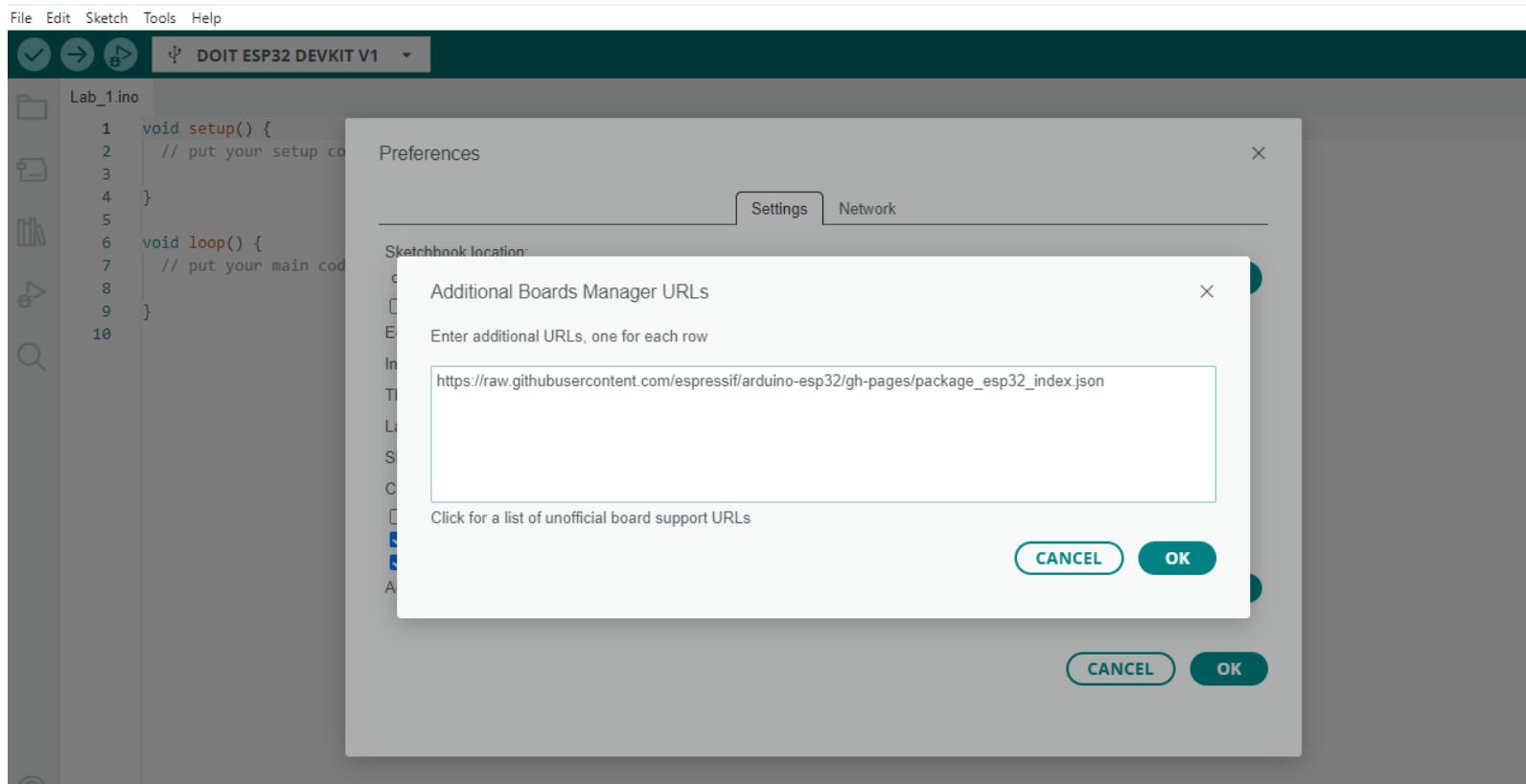


Configurando IDE

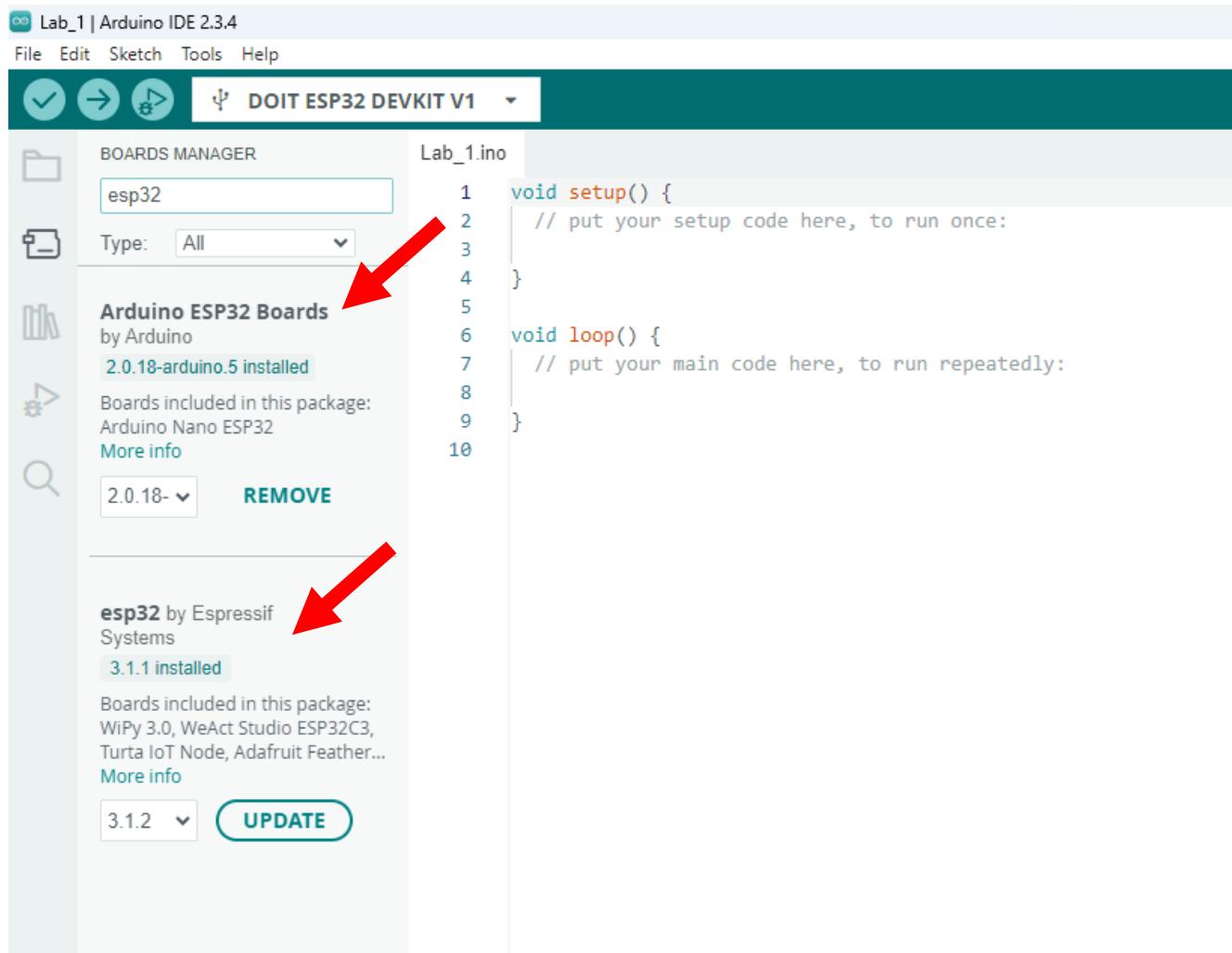


Configurando IDE

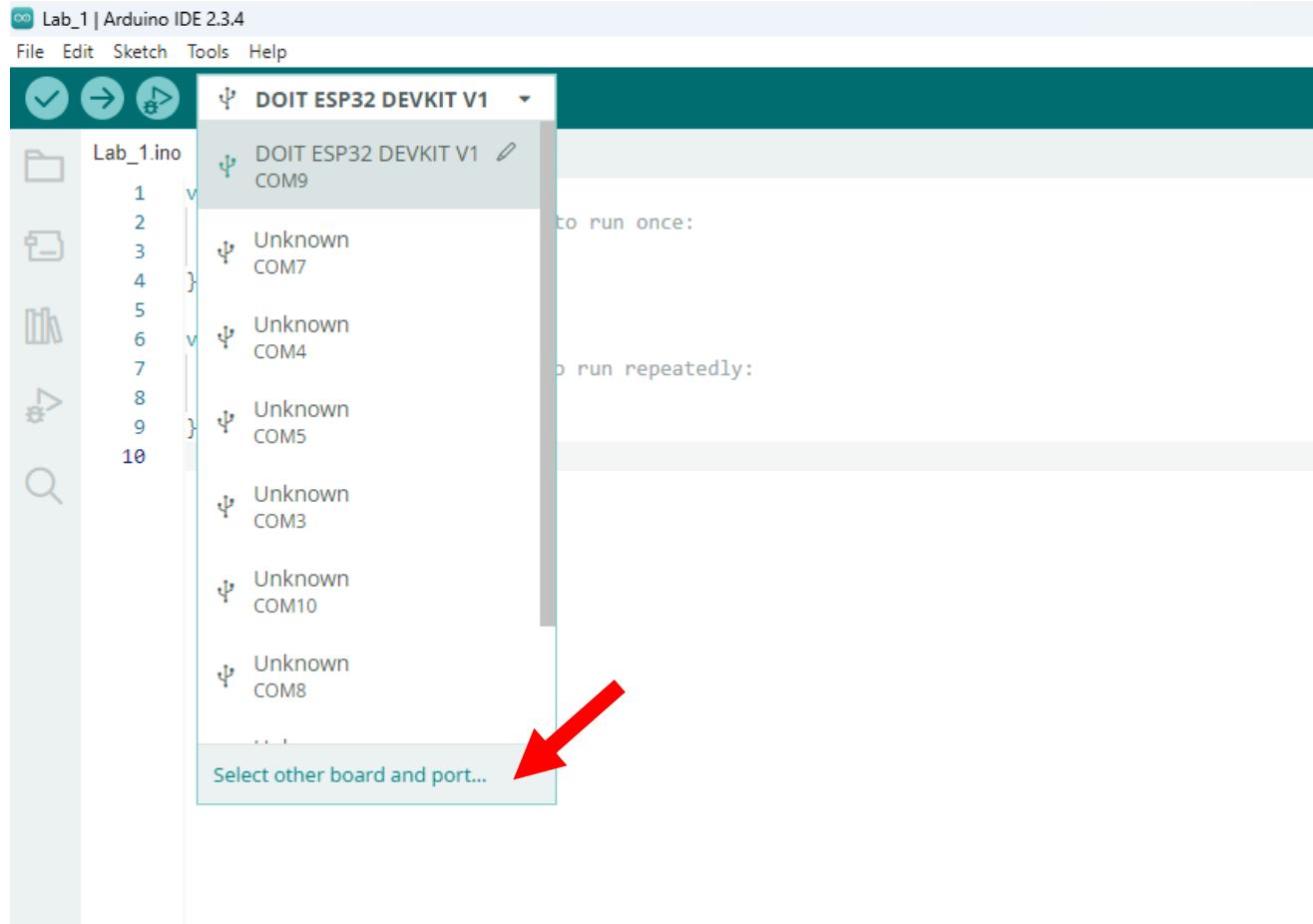
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json



Configurando IDE – Procurar ESP32

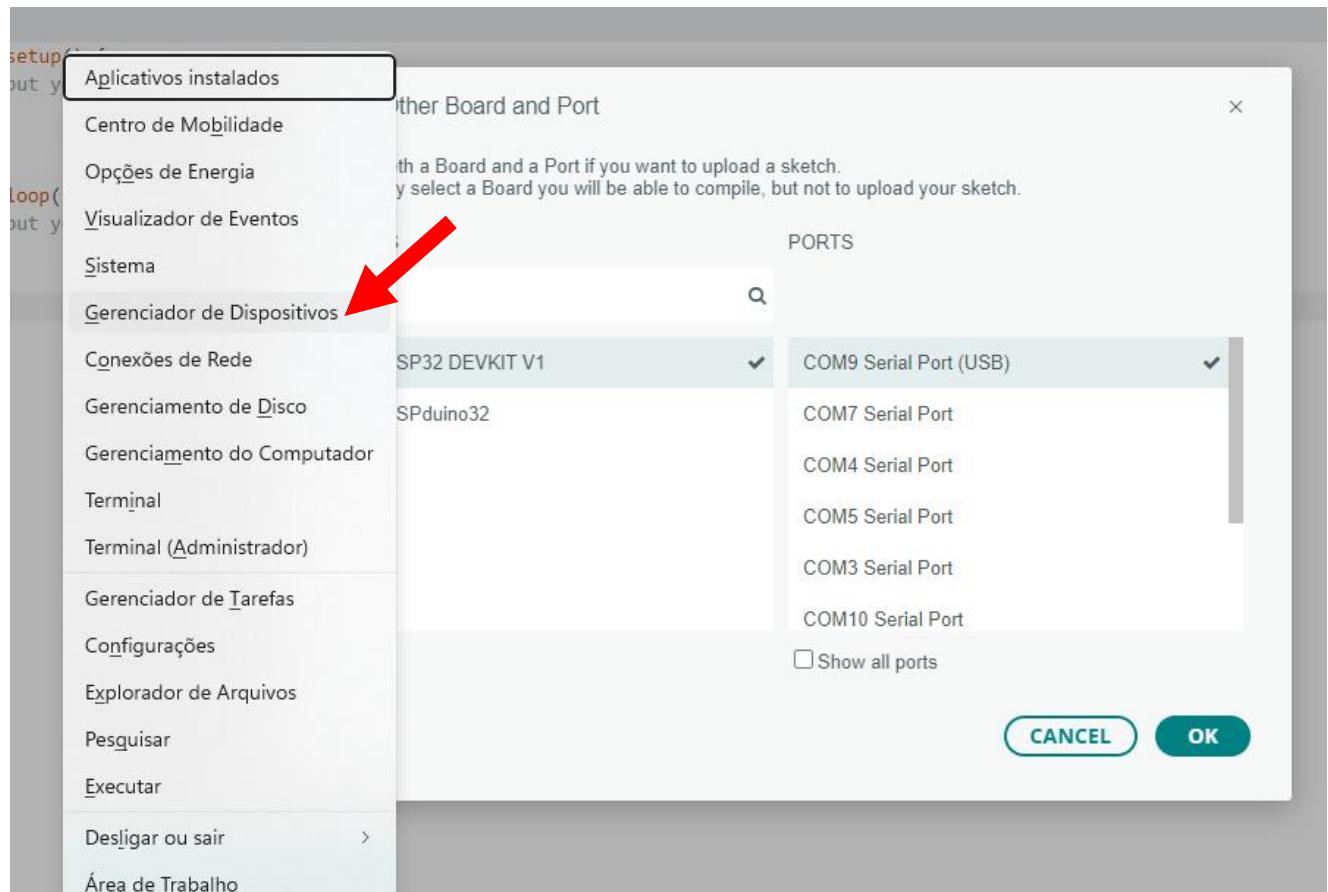


Configurando IDE – ESP32 DOIT DEVKIT V1



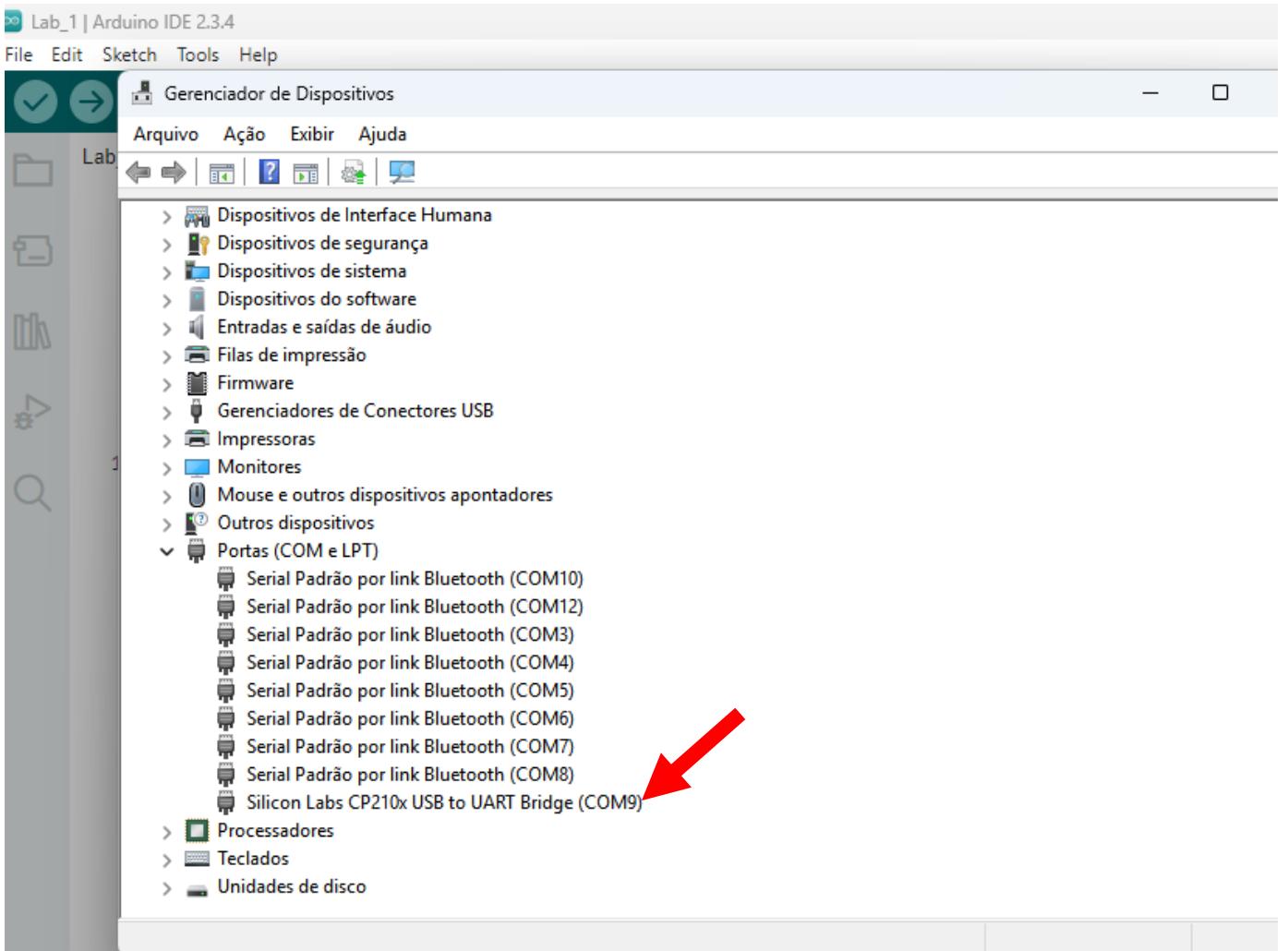


Configurando IDE – ESP32

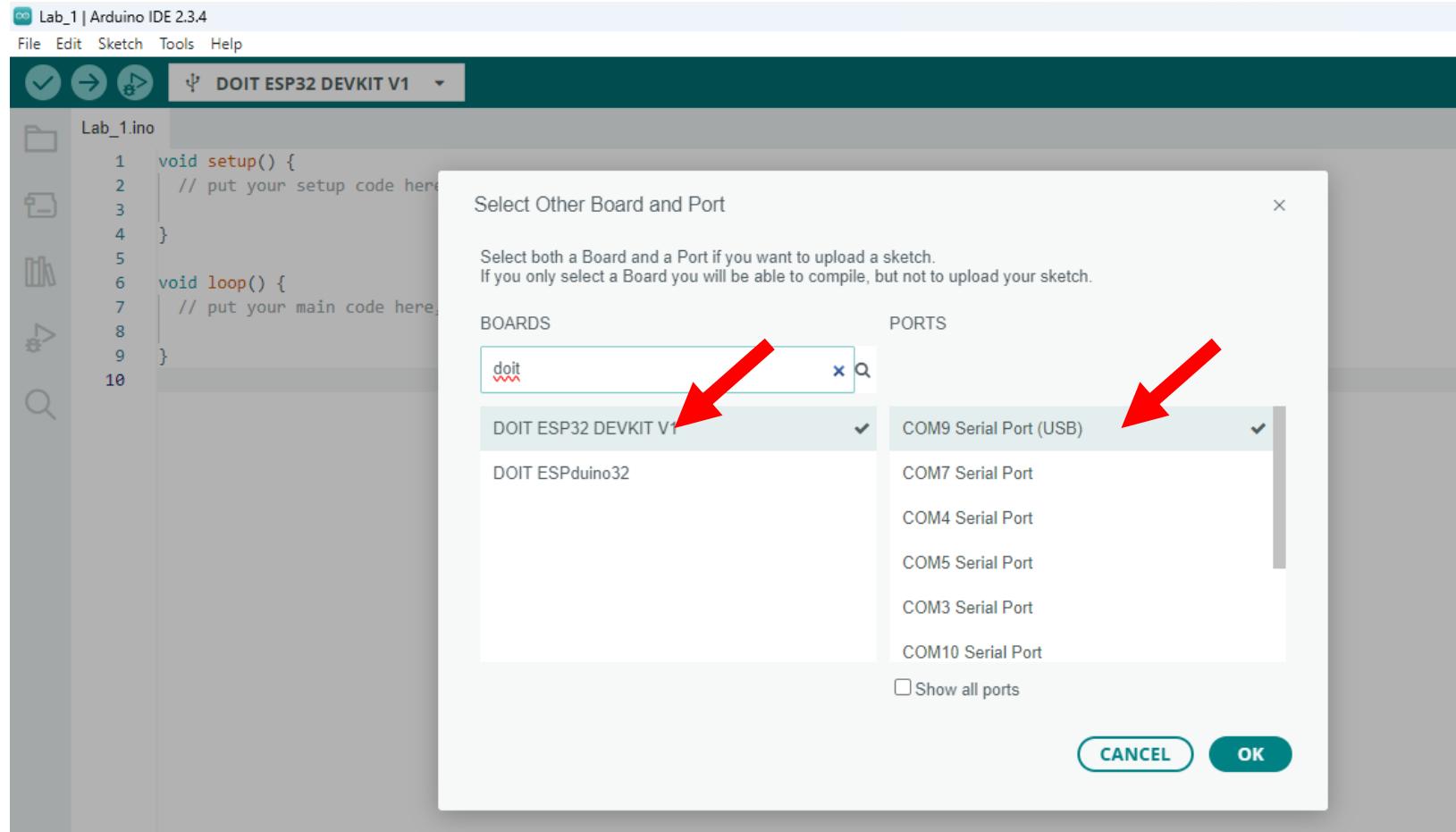


Teclas: Windows + x

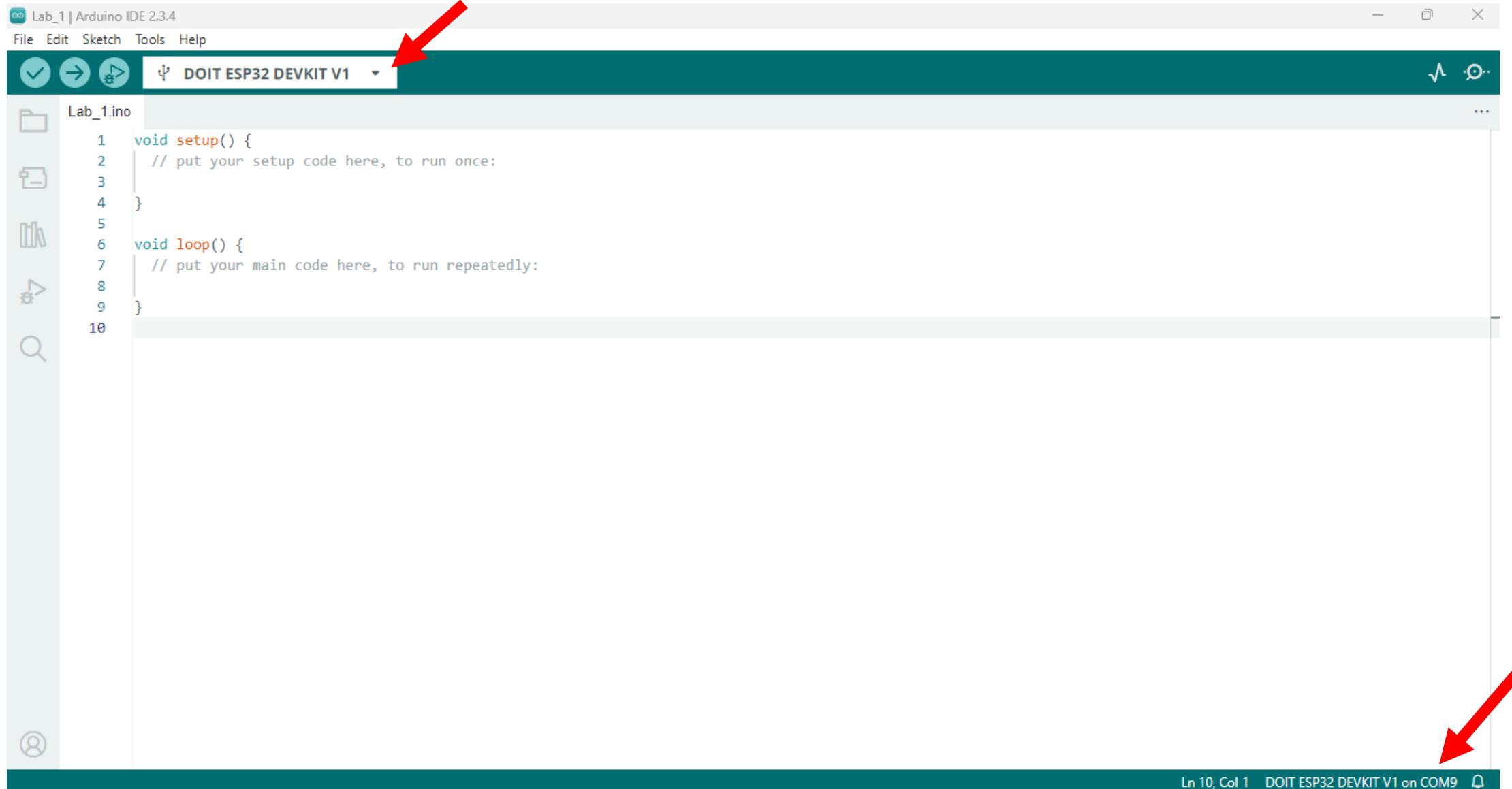
Configurando IDE – ESP32



Configurando IDE – ESP32 DOIT DEVKIT V1



Configurando IDE – ESP32 DOIT DEVKIT V1



Ajuda Arduino

Lab_1 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

Lab_1.ino

```

1 void //: 
2
3
4 }
5
6 void //: 
7
8 }
9
10

```

Getting Started
Environment
Troubleshooting
Reference
Find in Reference Ctrl+Shift+F
Frequently Asked Questions
Visit Arduino.cc
Privacy Policy
Check for Arduino IDE Updates
About Arduino IDE

eatedly:



https://docs.arduino.cc/language-reference/

DOCS

Programming En Language Reference

Functions Variables Structure

Language Reference

Arduino programming language can be divided in three main parts: functions, values (variables and constants), and structure.

For controlling the Arduino board and performing computations.

Digital I/O	Math	Bits and Bytes
digitalRead() digitalWrite() pinMode()	abs() constrain() map() max() min() pow() sqrt()	bit() bitClear() bitRead() bitSet() bitWrite() highByte() lowByte()

Analog I/O	Trigonometry	External Interrupts
analogRead() analogReadResolution() analogReference() analogWrite() analogWriteResolution()	cos() sin() tan()	attachInterrupt() detachInterrupt() digitalPinToInterrupt()

Ajuda

Acesso aos Códigos (Software)



Product ▾ Solutions ▾ Resources ▾ Open Source ▾ Enterprise ▾ Pricing



Search or jump to...



Sign in



Sign up

Build and ship software on a single, collaborative platform

Join the world's most widely adopted AI-powered developer platform.

Enter your email

Sign up for GitHub

Try GitHub Copilot



Acesso aos Códigos (Software)



Product ▾ Solutions ▾ Resources ▾ Open Source ▾ Enterprise ▾ Pricing

Search or jump to...



Sign in

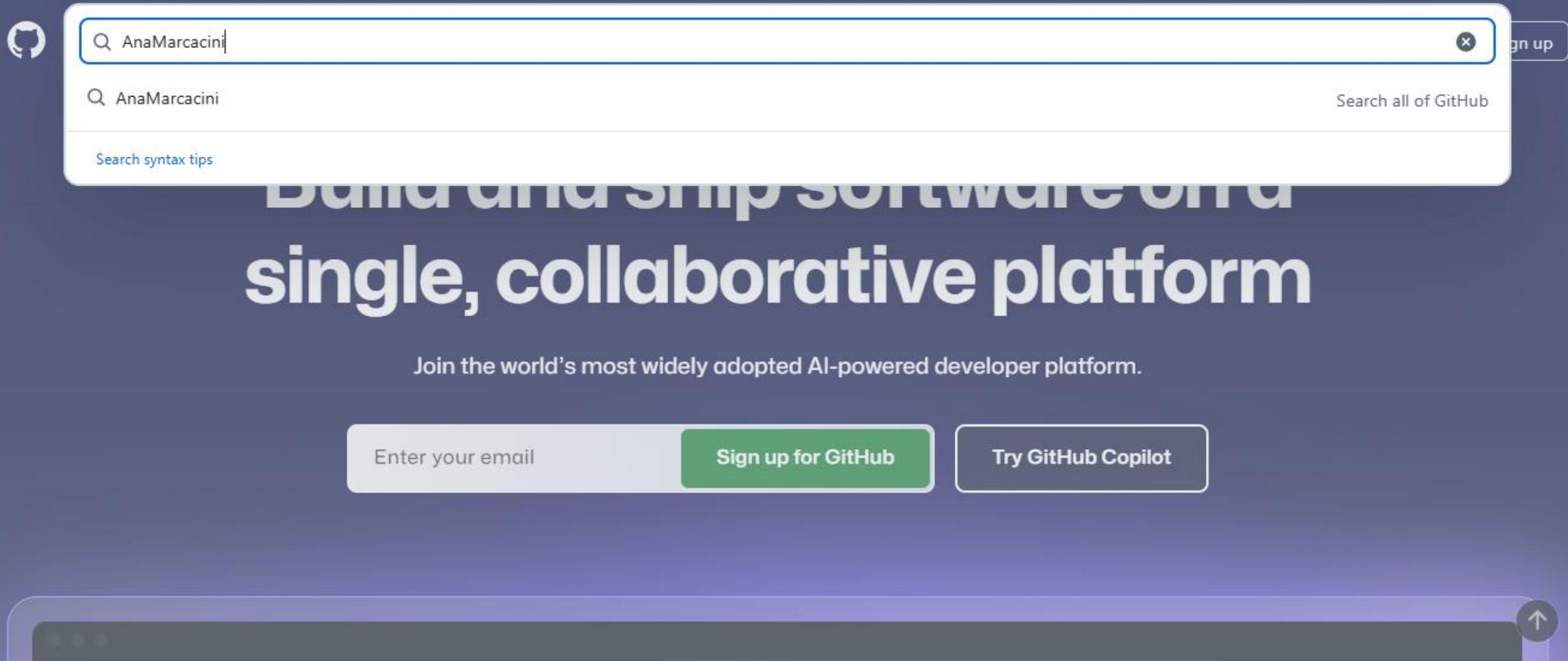
Sign up

B
si



<https://github.com/AnaMarcacini/SensorSonoro>

Acesso aos Códigos (Software)



The image shows a screenshot of the GitHub homepage. At the top, there is a search bar with the placeholder "AnaMarcacini". Below the search bar, there is a list of results, with the first result being "AnaMarcacini". To the right of the search bar, there is a "Search all of GitHub" button. Below the search area, there is a large, bold, white text that reads "Build and ship software on a single, collaborative platform". Underneath this text, there is a smaller white text that reads "Join the world's most widely adopted AI-powered developer platform." At the bottom of the screenshot, there are three buttons: "Enter your email", "Sign up for GitHub" (which is highlighted in green), and "Try GitHub Copilot".

Acesso aos Códigos (Software)

Filter by

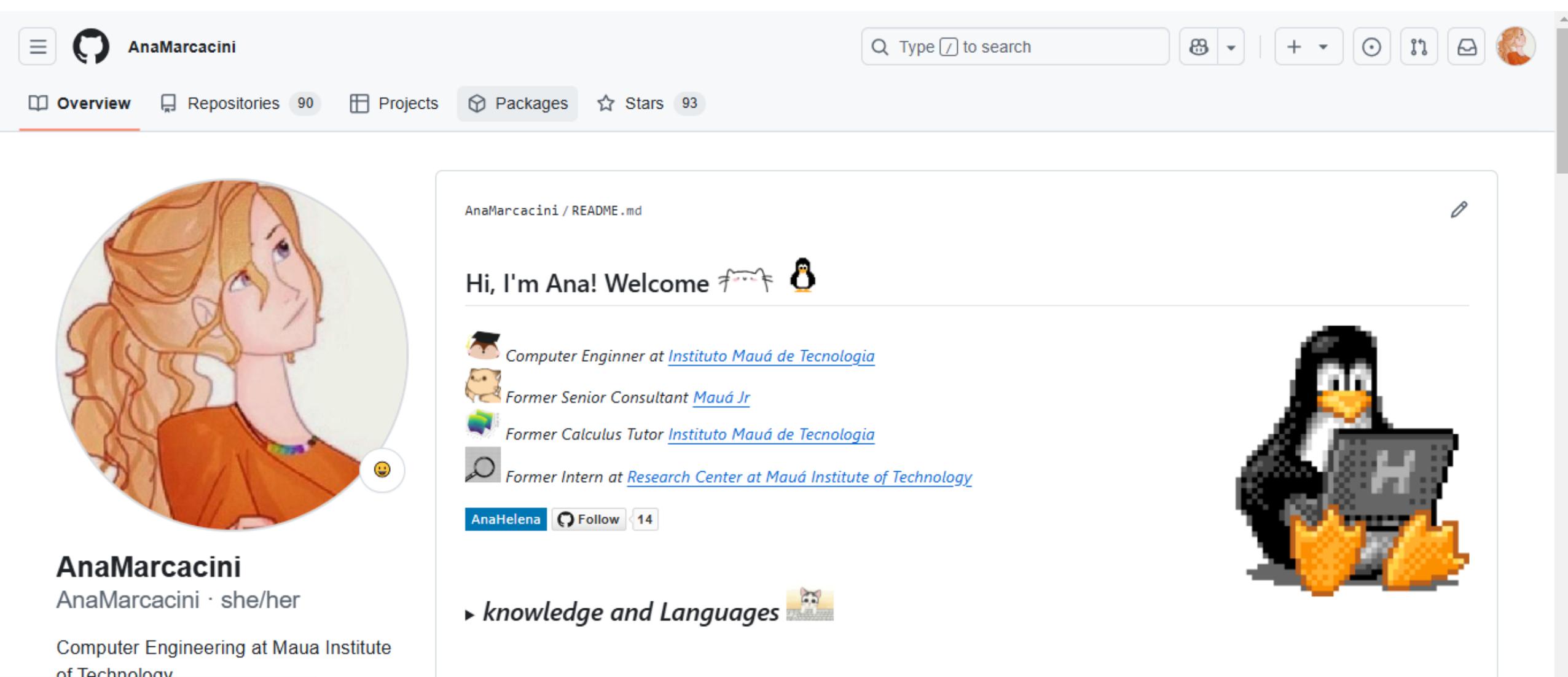
- <> Code 28
- Repositories 2
- Issues 0
- Pull requests 2
- Discussions 0
- Users 1**
- More

1 result (101 ms) Sort by: Best match ▾ Save ...



AnaMarcacini AnaMarcacini
Computer Engineering at Maua Institute of Technology

Acesso aos Códigos (Software)



The screenshot shows AnaMarcacini's GitHub profile page. The top navigation bar includes icons for three, a user profile (AnaMarcacini), search, and various account settings. Below the bar, tabs for Overview, Repositories (90), Projects, Packages, and Stars (93) are visible, with Overview selected.

Overview:

- User Picture:** A circular portrait of AnaMarcacini with orange hair and a white background.
- Name:** AnaMarcacini
- Location:** AnaMarcacini · she/her
- Education:** Computer Engineering at Mauá Institute of Technology
- Profile Summary:** Hi, I'm Ana! Welcome  
- Experience:**
 - Computer Enginner at [Instituto Mauá de Tecnologia](#)
 - Former Senior Consultant [Mauá Jr](#)
 - Former Calculus Tutor [Instituto Mauá de Tecnologia](#)
 - Former Intern at [Research Center at Mauá Institute of Technology](#)
- Follow:** AnaHelena  Follow  14
- Skills:** knowledge and Languages 

Packages: A section showing AnaMarcacini's packages, with a note that there are 93 packages available.

Acesso aos Códigos (Software)



AnaMarcacini

Overview **Repositories** 90 Projects Packages Stars 93

Find a repository... Type ▾ Language ▾ Sort ▾ New

SensorSonoro Public

C++ Updated 3 hours ago

AnaMarcacini Public

Config files for my GitHub profile.

config github-config

Python Updated 16 hours ago

a Private

Updated last week



AnaMarcacini
AnaMarcacini · she/her
Computer Engineering at Mauá Institute of Technology

Acesso aos Códigos (Software)

AnaMarcacini / SensorSonoro

Type / to search

Code Issues Pull requests Actions Projects Security Insights Settings

SensorSonoro Public

Pin Unwatch 1 Fork 0 Star 0

main 1 Branch 0 Tags

Go to file Add file Code

About

No description, website, or topics provided.

Matheuscoe07 Esquema eletrico v2 623a1c4 · 3 hours ago 5 Commits

Hw Esquema eletrico v2 3 hours ago

Modelo_3D Código para a IDE do VSCode 2 weeks ago

Sw Todos os Códigos 6 hours ago

README.md Código para a IDE do VSCode 2 weeks ago

README

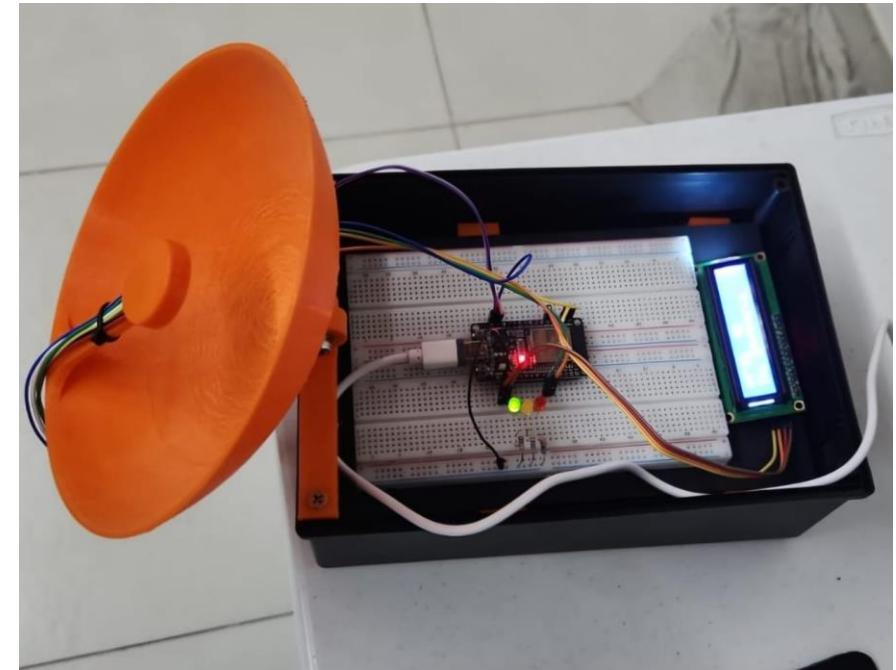
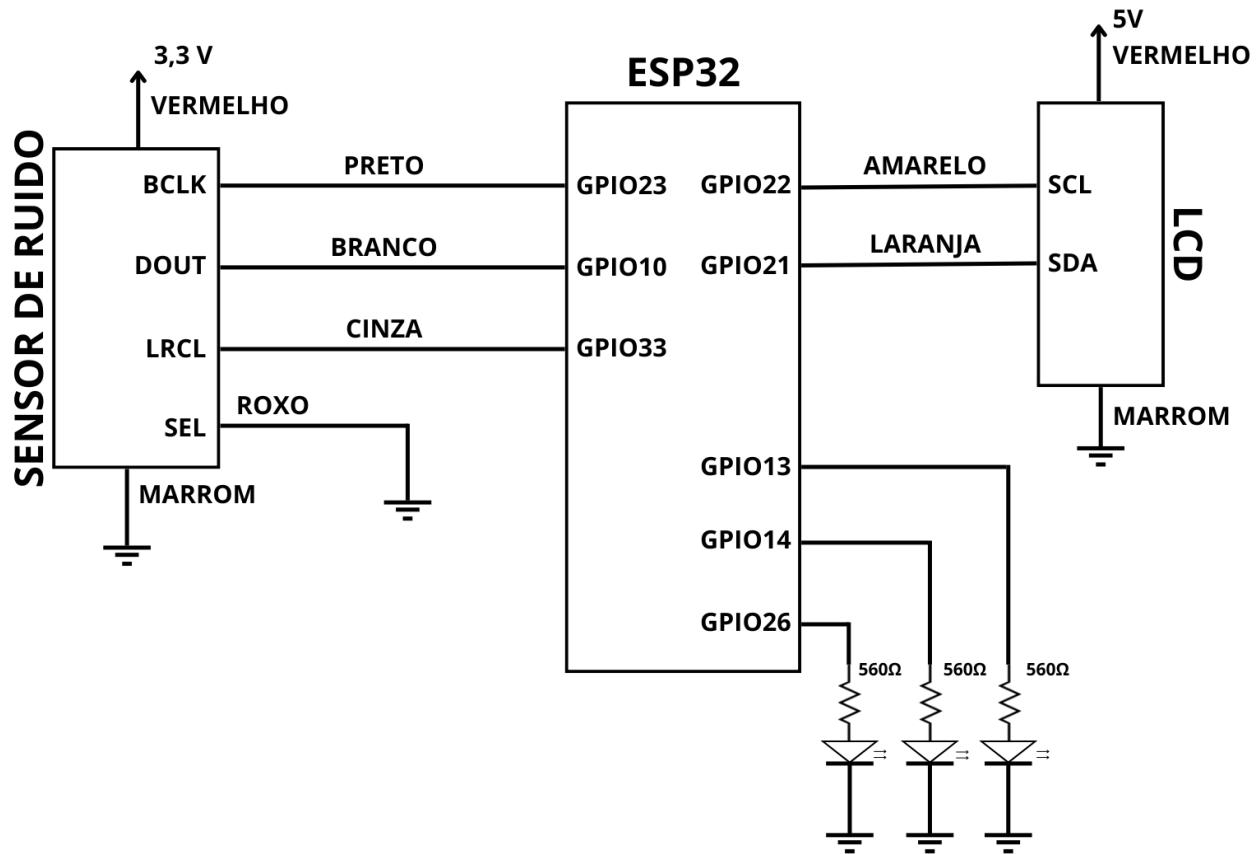
Releases

No releases published

Create a new release

SensorSonoro

Projeto Desenvolvido

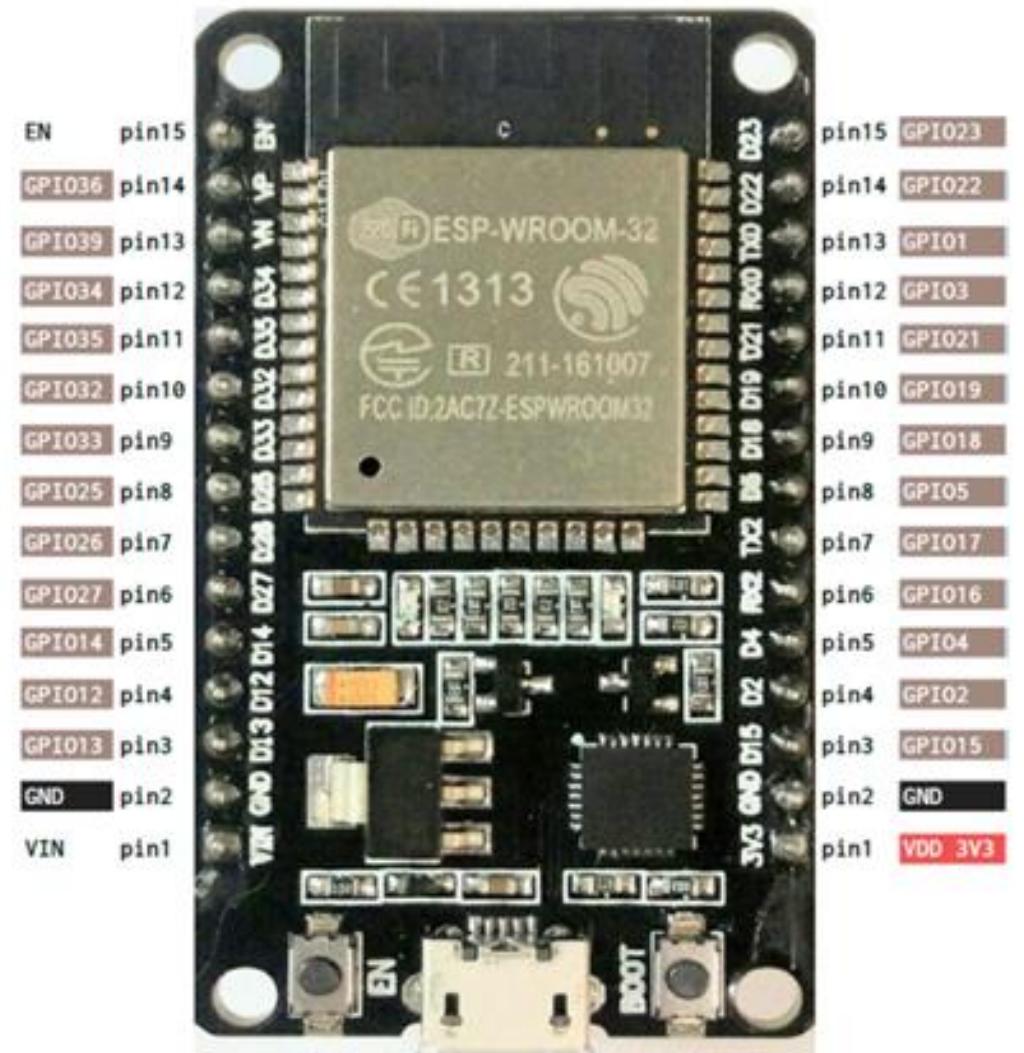


LAB 1

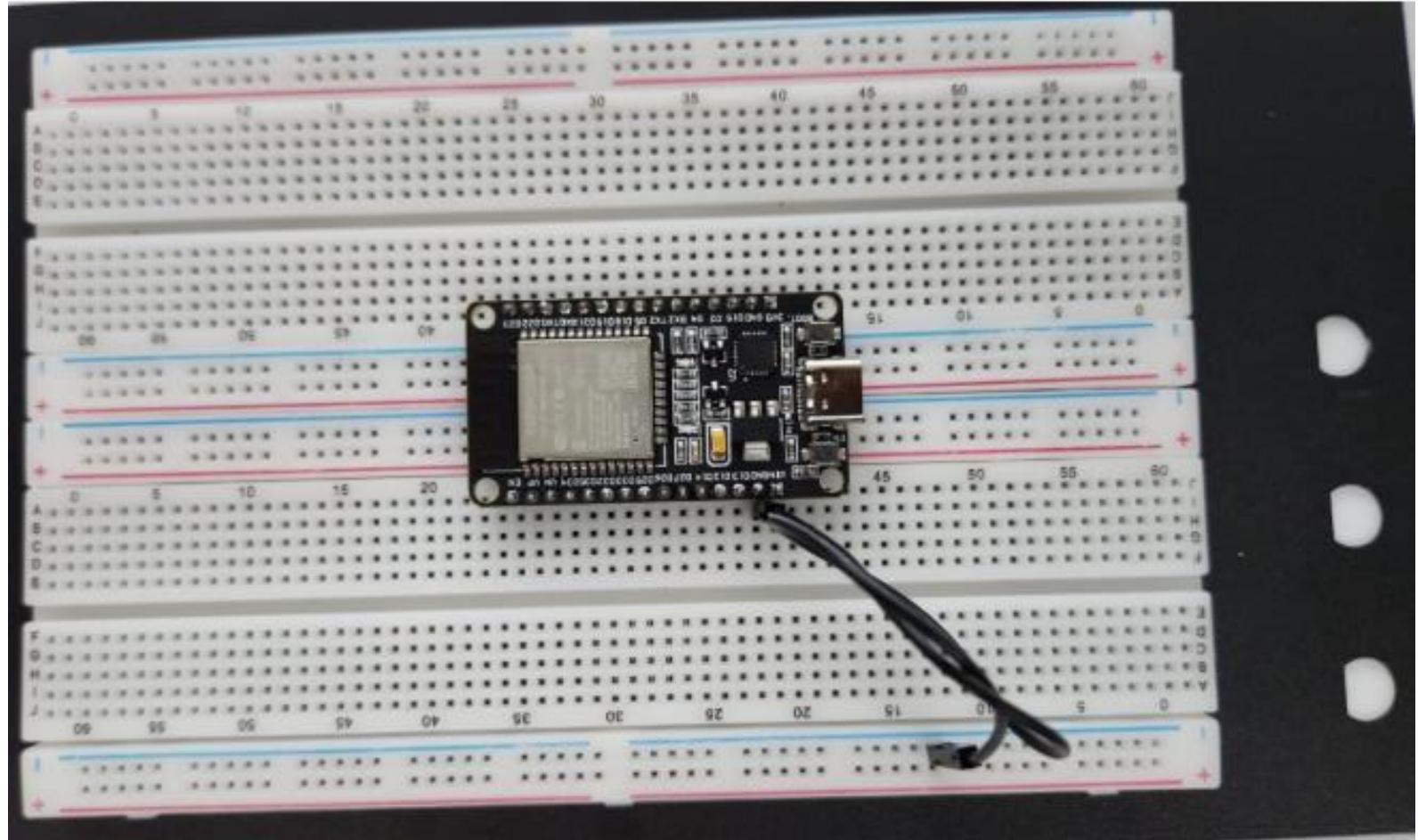


MAUÁ

Lab 1 – Montagem (HW)

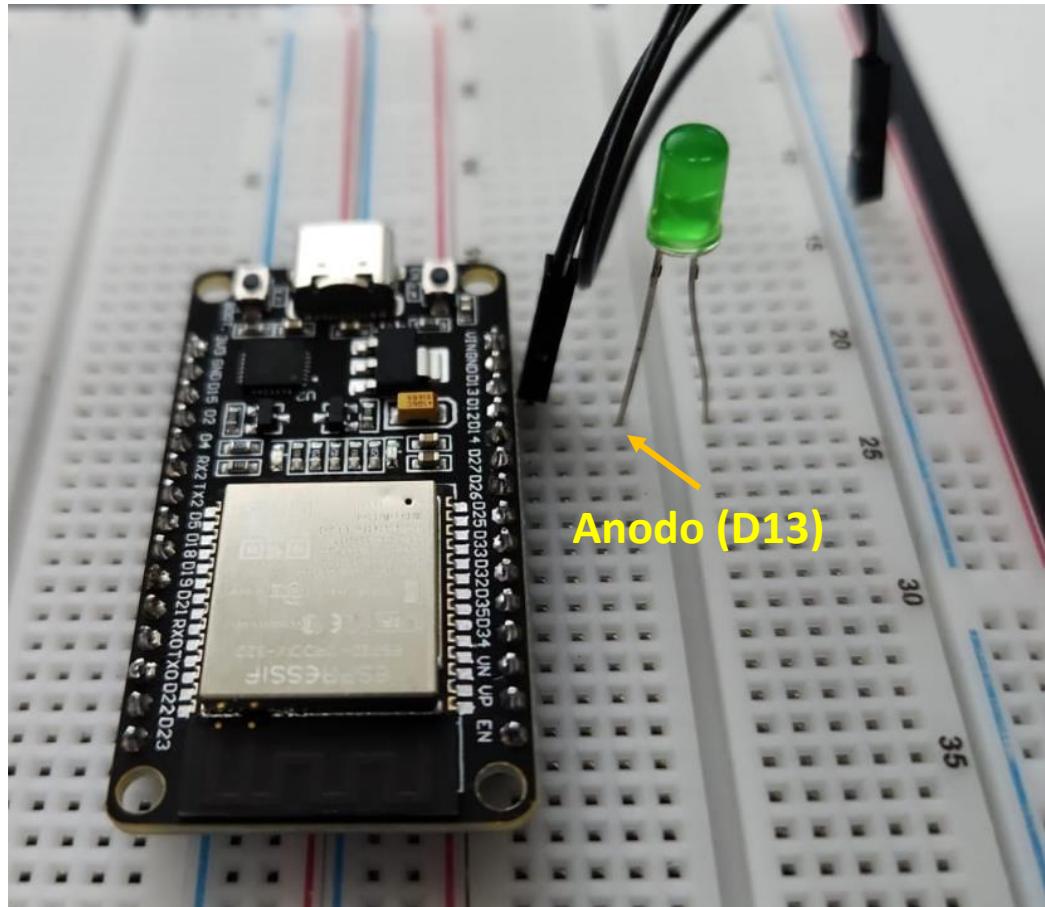


Lab 1 – Montagem (HW)

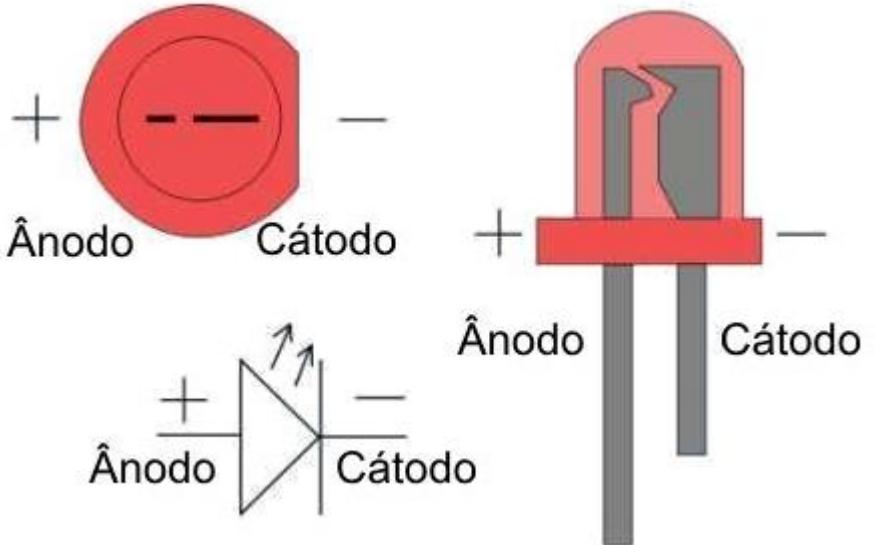


1 – Ligar Fio Preto Linha Azul – GND ESP32

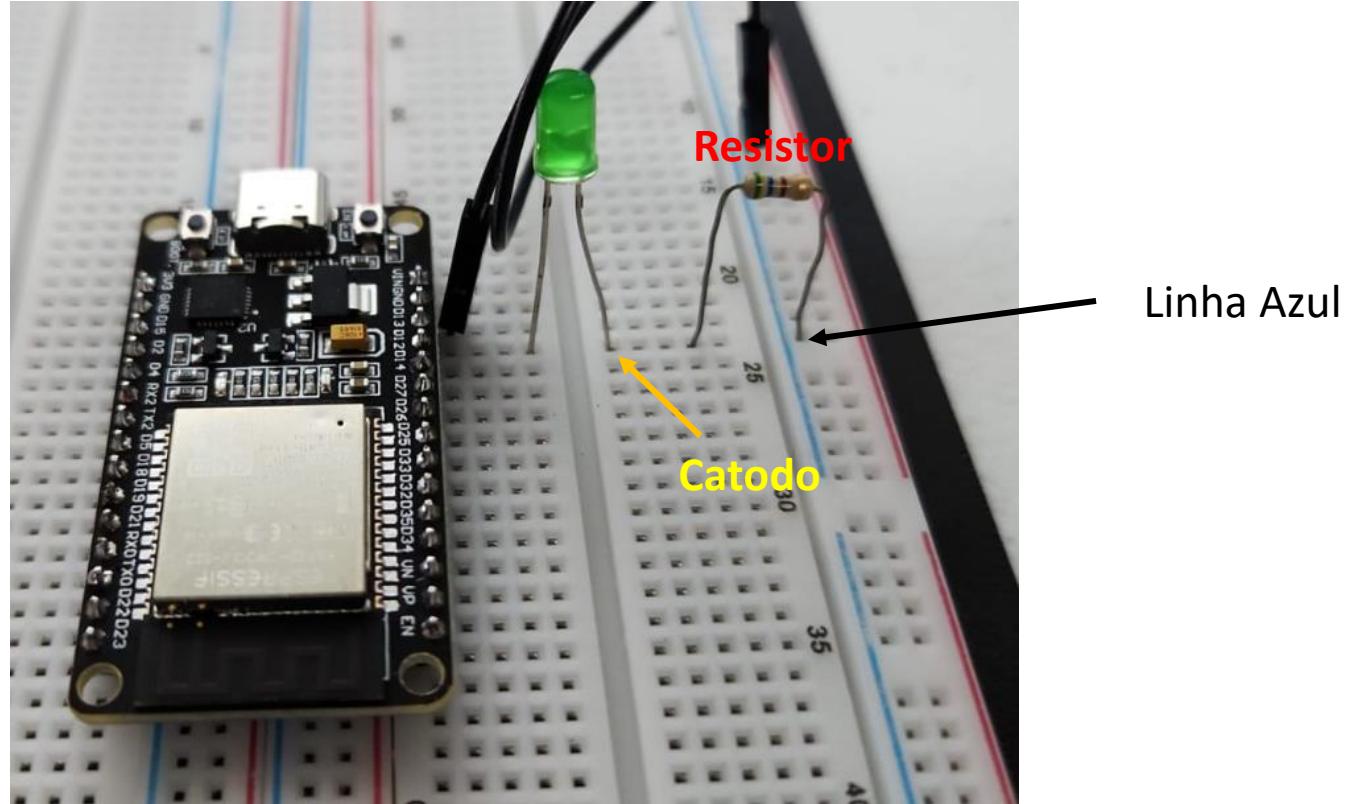
Lab 1 – Montagem (HW)



2 – Ligar Anodo do Led Verde no pino D13 ESP32

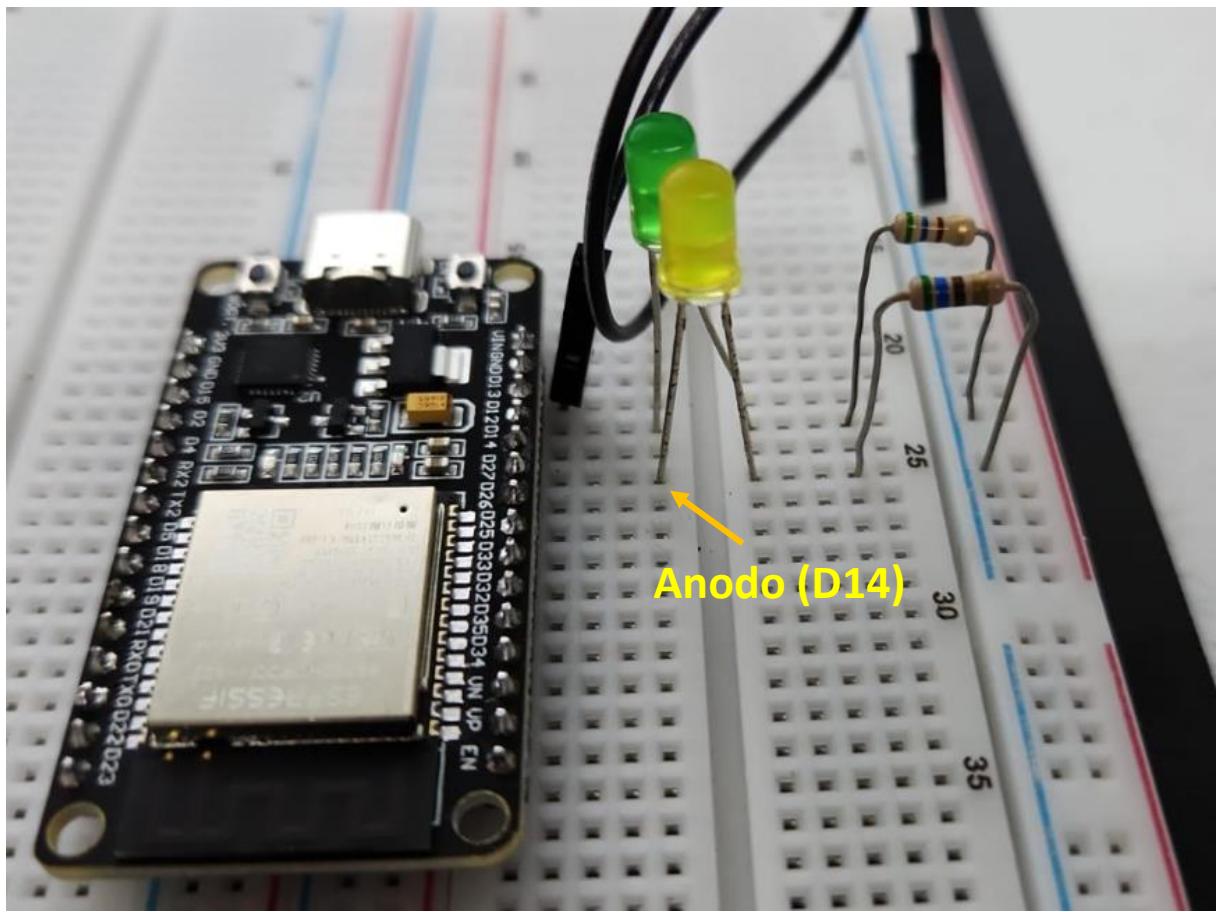


Lab 1 – Montagem (HW)



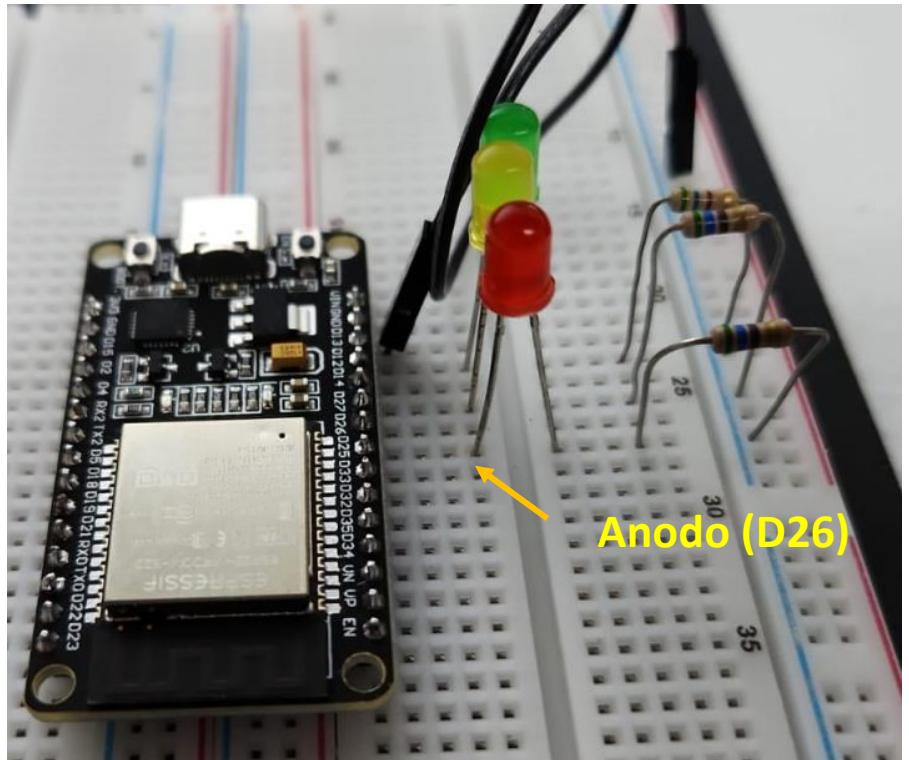
3 – Ligar terminal de resistor Catodo do do Led Verde outro terminal ligar na linha Azul

Lab 1 – Montagem (HW)



4 – Ligar Anodo do Led Amarelo no pino D14 ESP32, ligar o resistor conforme ilustra a figura

Lab 1 – Montagem (HW)

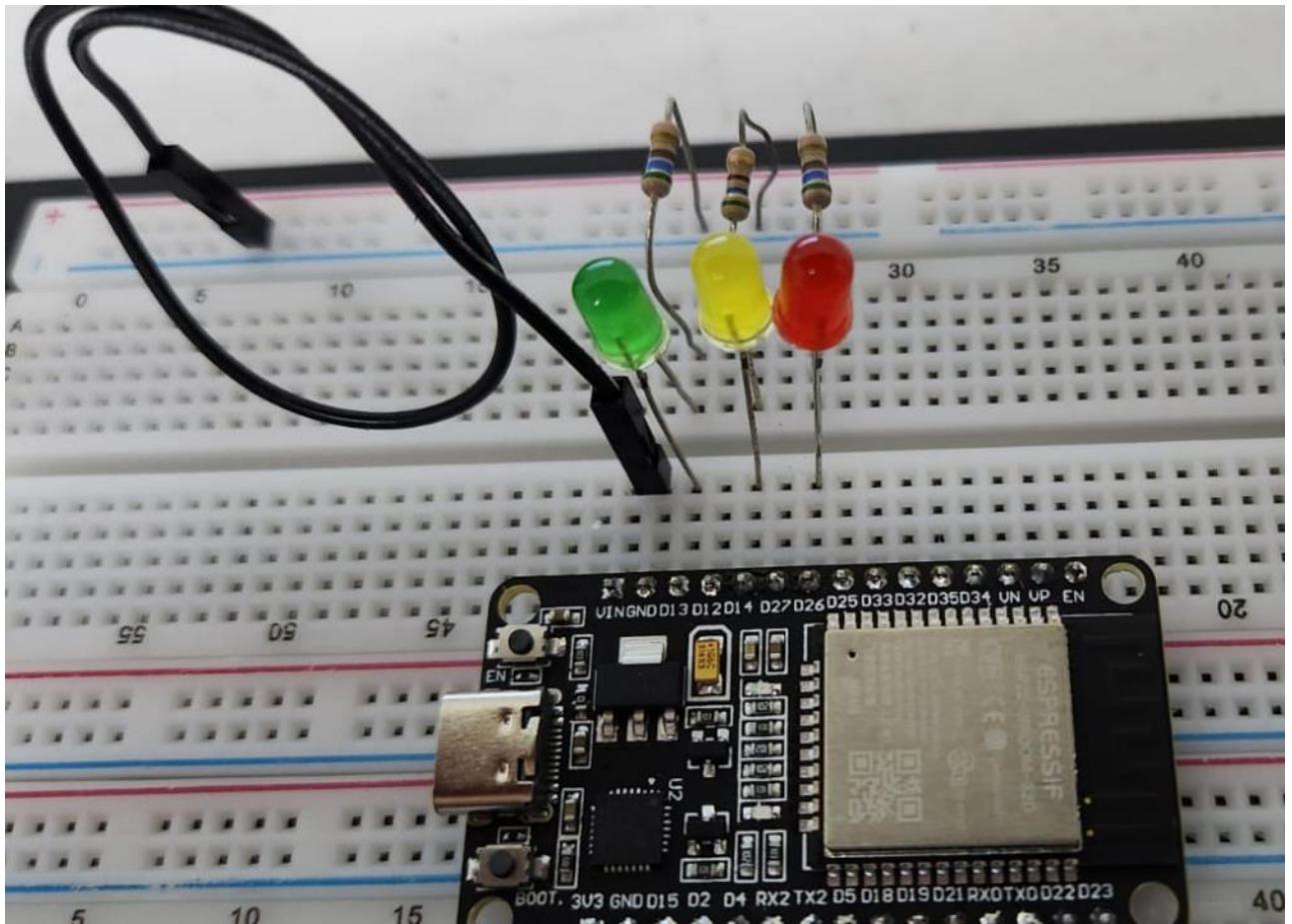


5 – Ligar Anodo do Led Vermelho no pino D26 ESP32, ligar o resistor conforme ilustra a figura

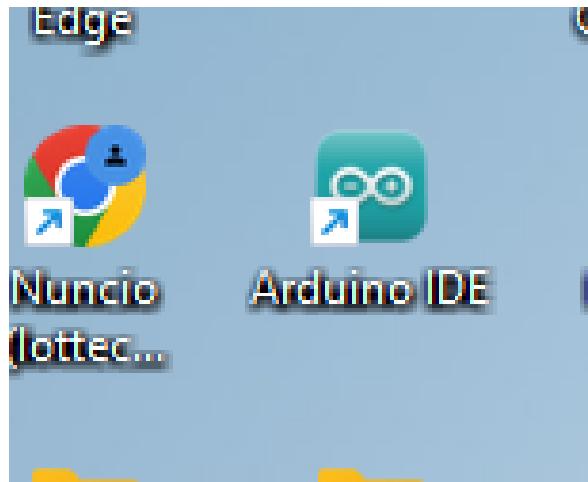


MAUÁ

Lab 1 – Montagem (HW)



Lab 1 – SW



- 1- Abrir o IDE Arduino
- 2- Carregar Sketch: Lab_1.ino

Lab 1 – SW

Lab_1 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

```

Lab_1.ino
1 //***** *****
2 //
3 // Instituto Mauá de Tecnologia
4 // Engenharia de Computação - 2025
5 //
6 // Coordenação:
7 // Prof. Dr. Ângelo Sebastião Zanini
8 //
9 // Professores:
10 // Prof. MSc Nuncio Perrella
11 // Prof. MSc Everson Denis
12 //
13 // Corpo Técnico:
14 // Eng. Gabriel Godoy
15 // Enga. Ana Helena
16 // Eng. Matheus Coelho
17 //
18 //*****
19
20 int LED1 = 13;           // Led Verde
21 int LED2 = 14;           // Led Amarelo
22 int LED3 = 26;           // Led Vermelho
23 int LED4 = 2;            // Led Azul placa
24
25
26 void setup() {
27
28     Serial.begin(115200);
29     Serial.println("IMT - Engenharia da Computação - 2025");
30
31     pinMode(LED1, OUTPUT);
32     pinMode(LED2, OUTPUT);
33     pinMode(LED3, OUTPUT);
34     pinMode(LED4, OUTPUT);
35
36     digitalWrite(LED1, LOW);
37     digitalWrite(LED2, LOW);
38     digitalWrite(LED3, LOW);
39     digitalWrite(LED4, LOW);
40
41 }
42
43 void loop() {
44
45     digitalWrite(LED1, HIGH);    // Acende Led Verde
46

```

Compilar e carregar

Lab 1 – SW

Lab_1 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

```

Lab_1.ino

1 //*****
2 //
3 // Instituto Mauá de Tecnologia
4 // Engenharia de Computação - 2025
5 //
6 // Coordenação:
7 // Prof. Dr. Ângelo Sebastião Zanini
8 //
9 // Professores:
10 // Prof. MSc Nuncio Perrella
11 // Prof. MSc Everson Denis
12 //
13 // Corpo Técnico:
14 // Eng. Gabriel Godoy
15 // Engr. Ana Helena
16 // Eng. Matheus Coelho
17 //
18 //*****
19
20
21 int LED1 = 13;      // Led Verde
22 int LED2 = 14;      // Led Amarelo
23 int LED3 = 26;      // Led Vermelho
24 int LED4 = 2;       // Led Azul placa
25
26
27 void setup() {
28
29     Serial.begin(115200);
30     Serial.println("IMT - Engenharia da Computação - 2025");
31 }
```

Output

```

Writing at 0x0000c019... (20 %)
Writing at 0x00028bd8... (30 %)
Writing at 0x0002e214... (40 %)
Writing at 0x00033f5f... (50 %)
Writing at 0x0003942c... (60 %)
Writing at 0x0003e98f... (70 %)
Writing at 0x00044066... (80 %)
Writing at 0x0004b06c... (90 %)
Writing at 0x00054cd6... (100 %)
Wrote 300896 bytes (160347 compressed) at 0x00010000 in 3.0 seconds (effective 790.4 kbit/s)...
Hash of data verified.

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

Lab 1 – SW

Lab_1 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

Lab_1.ino

```

1 //***** Instituto Mauá de Tecnologia *****
2 //
3 // Instituto de Mauá de Tecnologia
4 // Engenharia de Computação - 2025
5 //
6 // Coordenação:
7 // Prof. Dr. Ângelo Sebastião Zanini
8 //
9 // Professores:
10 // Prof. MSc Nuncio Perrella
11 // Prof. MSc Everson Denis
12 //
13 // Corpo Técnico:
14 // Eng. Gabriel Godoy
15 // Eng. Ana Helena
16 // Eng. Matheus Coelho
17 //
18 //*****
19
20
21 int LED1 = 13;      // Led Verde
22 int LED2 = 14;      // Led Amarelo
23 int LED3 = 26;      // Led Vermelho
24 int LED4 = 2;       // Led Azul placa
25
26
27 void setup() {
28
29     Serial.begin(115200);
30     Serial.println("IMT - Engenharia da Computação - 2025");
31 }
```

Output Serial Monitor ×

Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM9')

Both NL & CR 115200 baud

```

21:59:28.327 -> ho 0 tail 12 room 4
21:59:28.327 -> load:0x40080400,len:4
21:59:28.327 -> load:0x40080404,len:3196
21:59:28.327 -> entry 0x400805a4
21:59:28.442 -> IMT - Engenharia da Computação - 2025
21:59:33.456 -> Grupo - X
21:59:38.436 -> Grupo - X
21:59:43.434 -> Grupo - X
21:59:48.458 -> Grupo - X
21:59:53.454 -> Grupo - X
21:59:58.443 -> Grupo - X
22:00:03.436 -> Grupo - X
```

Ln 60, Col 1 DOIT ESP32 DEVKIT V1 on COM9 22:00
14/02/2025

4 22°C Parc. nublado

Pesquisar

Lab 1 – Conclusões

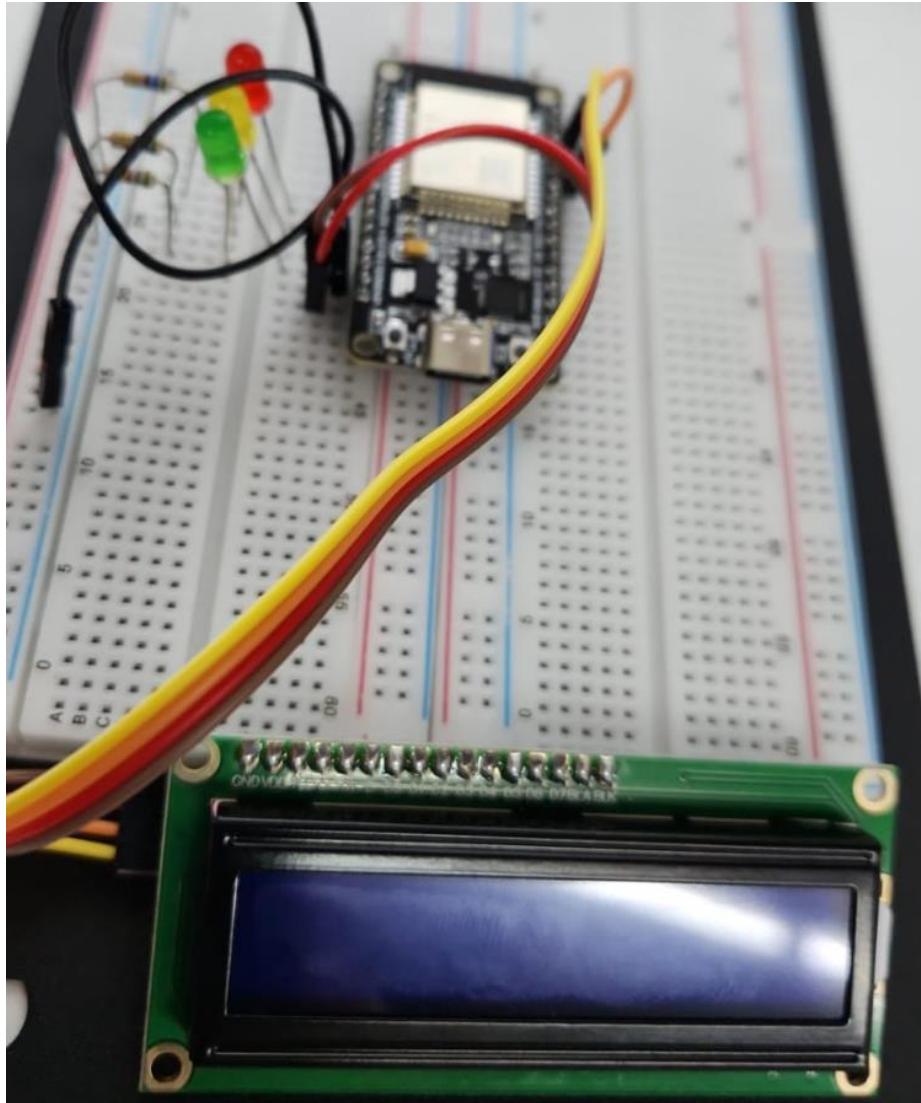
Listar as observações e conclusões do laboratório

Lab 1 – Conclusões

Parabéns !!! Se você consegue acender um LED você
Pode fazer qualquer coisa em sistemas embarcados!!!

LAB 2

Lab 2 – Montagem (HW)



Ligar os fios dos display

- Vermelho → Vin (5V)
- Marrom → GND
- Laranja → D21 (ESP32)
- Amarelo → D22 (ESP32)

Lab 2 – SW

Lab_2 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

Lab_2.ino

```

1 // Prof. MSC Nuncio Perrella
2 // Prof. MSC Everson Denis
3 //
4 // Corpo Técnico:
5 // Eng. Gabriel Godoy
6 // Enga. Ana Helena
7 // Eng. Matheus Coelho
8 //
9 ****
10 //include <Wire.h>
11 #include <LiquidCrystal_I2C.h>
12
13 #define I2C_SDA 21
14 #define I2C_SCL 22
15
16 LiquidCrystal_I2C lcd(0x27, 16, 2); // Endereço I2C do LCD: 0x27, 16 colunas, 2 linhas
17
18 int LED1 = 13; // Led Verde
19 int LED2 = 14; // Led Amarelo
20 int LED3 = 26; // Led Vermelho
21 int LED4 = 2; // Led Azul placa
22
23 void setup() {
24
25
26
27
28
29
30
31
32
33
34
35
36
37

```

Output

```

Writing at 0x00010000... (9 %)
Writing at 0x0001c2c2... (18 %)
Writing at 0x00029739... (27 %)
Writing at 0x0002ecce... (36 %)
Writing at 0x00034810... (45 %)
Writing at 0x0003a141... (54 %)
Writing at 0x0003f6fb... (63 %)
Writing at 0x00044d65... (72 %)
Writing at 0x0004a5b4... (81 %)
Writing at 0x000556cc... (90 %)
Writing at 0x0005b40a... (100 %)
Wrote 322400 bytes (173164 compressed) at 0x00010000 in 3.4 seconds (effective 754.0 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

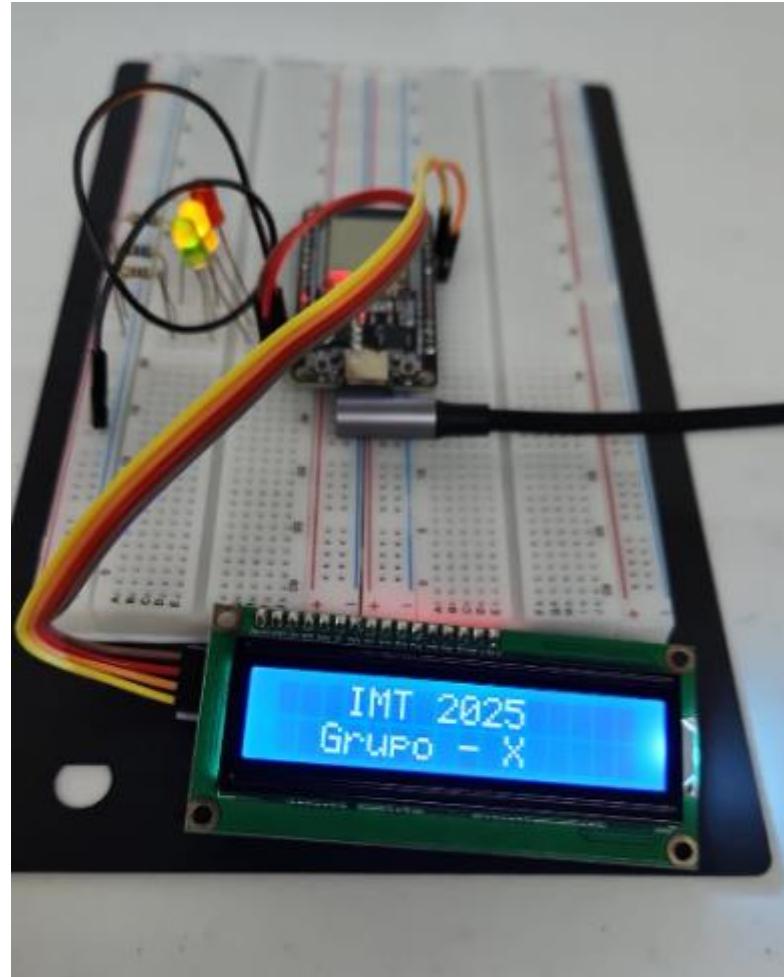
```

- 1- Abrir o IDE Arduino
- 2- Carregar Sketch: Lab_2.ino
- 3- Compilar e carregar

Lab 2 – Conclusões

Listar as observações e conclusões do laboratório

Lab 2 – Conclusões



LAB 3

Lab 3 – SW

Lab_3 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

```

Lab_3.ino
23
24
25 #define I2C_SDA 21
26 #define I2C_SCL 22
27
28
29 LiquidCrystal_I2C lcd(0x27, 16, 2); // Endereço I2C do LCD: 0x27, 16 colunas, 2 linhas
30
31
32 int LED1 = 13; // Led Verde
33 int LED2 = 14; // Led Amarelo
34 int LED3 = 26; // Led Vermelho
35 int LED4 = 2; // Led Azul placa
36 int cont = 0;
37 boolean pisca = false;
38
39
40 const char* ssid      = "ZTE_E58644"; // Roteador celular
41 const char* password = "08032114"; // Senha
42
43 WiFiClient client; // Wifi simples
44
45
46 void setup() {
47
48   Serial.begin(115200);
49   Serial.println("IMT - Engenharia da Computação - 2025");
50

```

Output Serial Monitor ×

Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM9')

```

23:01:40.722 -> WiFi SSID: ZTE_E58644
23:01:40.722 -> IP Modulo: 192.168.1.154
23:01:40.722 ->
23:03:17.949 -> Setup - Iniciando a conexão WiFiSetup - Tentando Conexão Wifi: 1
23:03:18.966 ->
23:03:18.966 -> WiFi conectado.
23:03:18.966 -> WiFi SSID: ZTE_E58644
23:03:18.966 -> IP Modulo: 192.168.1.154
23:03:18.966 ->
23:05:29.161 -> Setup - Iniciando a conexão WiFiSetup - Tentando Conexão Wifi: 1
23:05:30.192 ->
23:05:30.192 -> WiFi conectado.
23:05:30.192 -> WiFi SSID: ZTE_E58644
23:05:30.231 -> IP Modulo: 192.168.1.154
23:05:30.231 ->

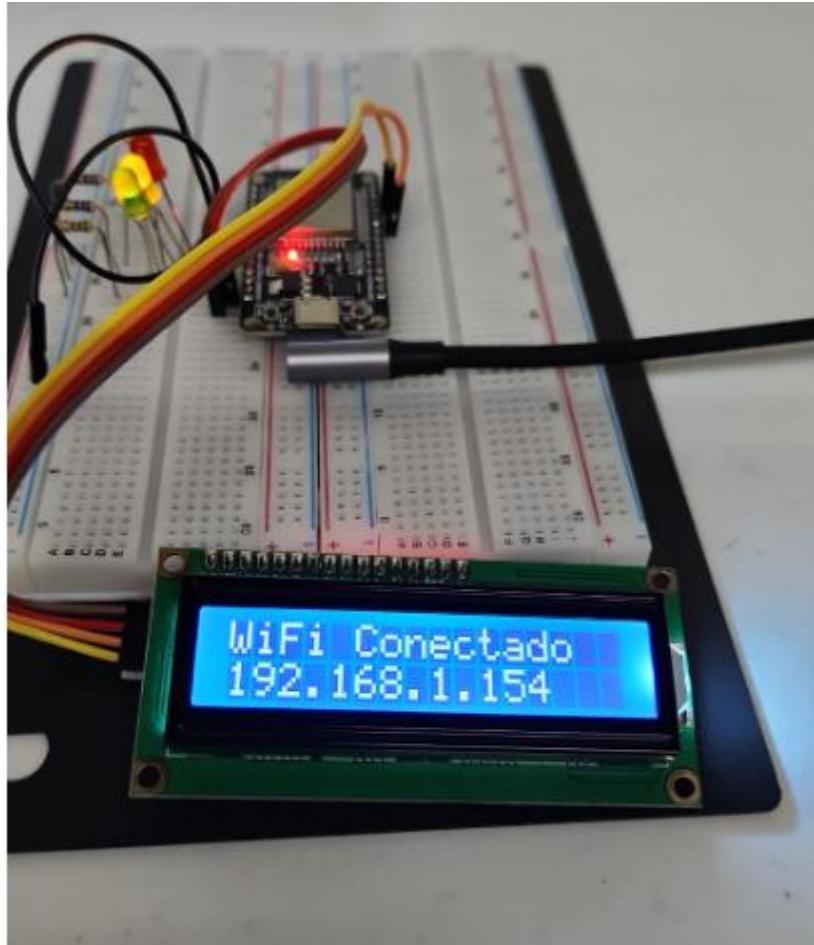
```

- 1 - Abrir o IDE Arduino
- 2 - Carregar Sketch: Lab_3.ino
- 3 - Colocar celular modo Roteador
- 4 - Linha 40 carregar SSID do roteador do seu celular
- 5 – Linha 41 carregar a senha do seu roteador
- 6- Compilar e carregar

Lab 3 – Conclusões

Listar as observações e conclusões do laboratório

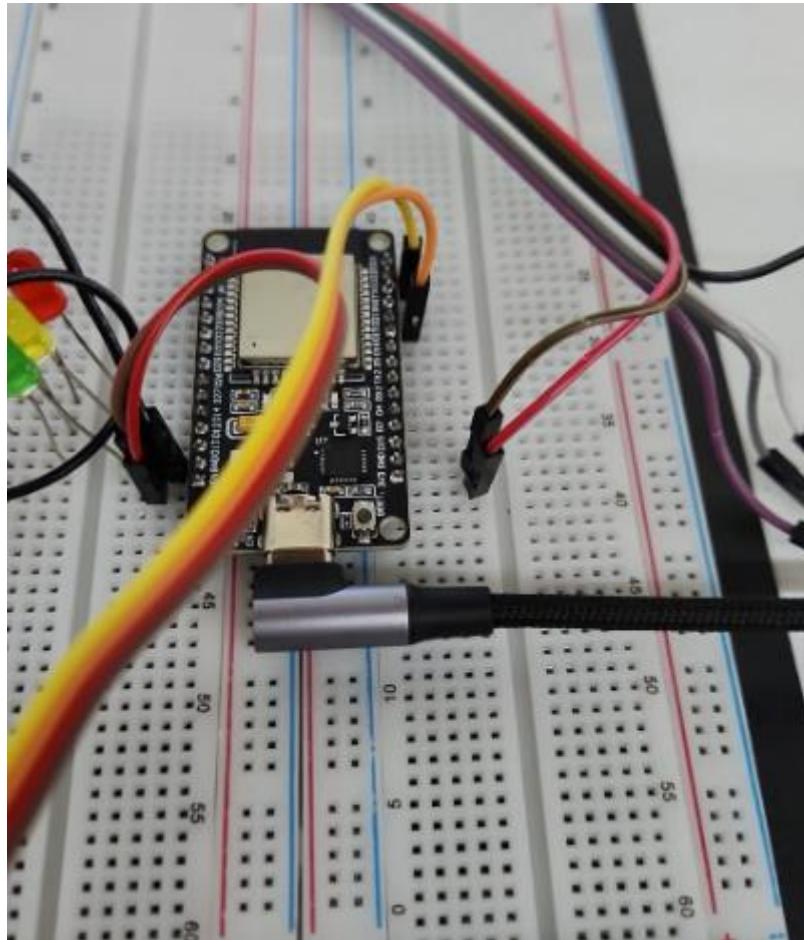
Lab 3 – Conclusões



LAB 4



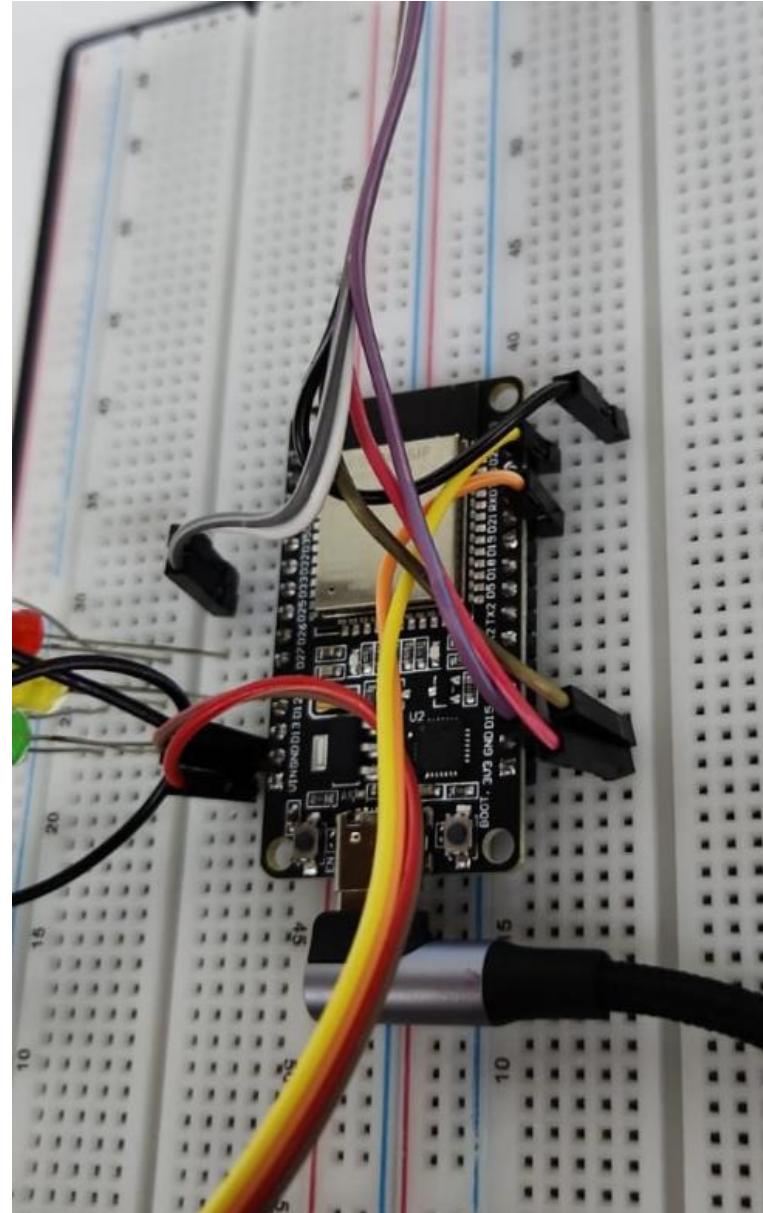
Lab 4 – Montagem (HW)



- 1 – Ligar Fio Vermelho do Sensor – 3V3 ESP32
- 2 – Ligar Fio Marrom do Sensor – GND ESP32



Lab 4 – Montagem (HW)



- 1 – Ligar Fio Preto do Sensor – D23 ESP32
- 2 – Ligar Fio Branco do Sensor – D32 ESP32
- 3 – Ligar Fio Cinza do Sensor – D33 ESP32
- 4 – Ligar Fio Roxo do Sensor – GND ESP32

Lab 4 – SW

Lab_3 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

```

Lab_3.ino
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25 #define I2C_SDA 21
26 #define I2C_SCL 22
27
28
29 LiquidCrystal_I2C lcd(0x27, 16, 2); // Endereço I2C do LCD: 0x27, 16 colunas, 2 linhas
30
31
32 int LED1 = 13; // Led Verde
33 int LED2 = 14; // Led Amarelo
34 int LED3 = 26; // Led Vermelho
35 int LED4 = 2; // Led Azul placa
36 int cont = 0;
37 boolean pisca = false;
38
39
40 const char* ssid      = "ZTE_E58644"; // Roteador celular
41 const char* password = "08032114"; // Senha
42
43 WiFiClient client; // Wifi simples
44
45
46 void setup() {
47
48   Serial.begin(115200);
49   Serial.println("IMT - Engenharia da Computação - 2025");
50

```

Output Serial Monitor ×

Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM9')

```

23:01:40.722 -> WiFi SSID: ZTE_E58644
23:01:40.722 -> IP Modulo: 192.168.1.154
23:01:40.722 ->
23:03:17.949 -> Setup - Iniciando a conexão WiFiSetup - Tentando Conexão Wifi: 1
23:03:18.966 ->
23:03:18.966 -> WiFi conectado.
23:03:18.966 -> WiFi SSID: ZTE_E58644
23:03:18.966 -> IP Modulo: 192.168.1.154
23:03:18.966 ->
23:05:29.161 -> Setup - Iniciando a conexão WiFiSetup - Tentando Conexão Wifi: 1
23:05:30.192 ->
23:05:30.192 -> WiFi conectado.
23:05:30.192 -> WiFi SSID: ZTE_E58644
23:05:30.231 -> IP Modulo: 192.168.1.154
23:05:30.231 ->

```

1 - Abrir o IDE Arduino

2 - Carregar Sketch: Lab_4.ino

3 - Colocar celular modo Roteador

4 - Linha 40 carregar SSID do roteador do seu celular

5 – Linha 41 carregar a senha do seu roteador

6- Compilar e carregar



Lab 4 – SW

Lab 4 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

```
Lab_4.ino
12
73
74 void setup() {
75
76     Serial.begin(115200);
77     Serial.println("IMT - Engenharia da Computação - 2025");
78
79     pinMode(LED1, OUTPUT);
80     pinMode(LED2, OUTPUT);
81     pinMode(LED3, OUTPUT);
82     pinMode(LED4, OUTPUT);
83
84     digitalWrite(LED1, LOW);
85     digitalWrite(LED2, LOW);
86     digitalWrite(LED3, LOW);
87     digitalWrite(LED4, LOW);
88
89     Wire.begin(I2C_SDA, I2C_SCL); // Inicializa a comunicação I2C com os pinos definidos
90     lcd.begin(16, 2); // Inicializa o LCD com 16 colunas e 2 linhas
91     lcd.backlight(); // Liga a luz de fundo
92     lcd.setCursor(3, 0); // Posiciona cursor na linha 0, terceira coluna
93     lcd.print("IMT 2025");
94     lcd.setCursor(2, 1);
95     lcd.print("Prof. Nuncio");
96     delay(3000);
97     lcd.clear();
98
99
100    i2s_driver_install(i2s_num, &i2s_config, 0, NULL); // install and start i2s driver
```

Output Serial Monitor X

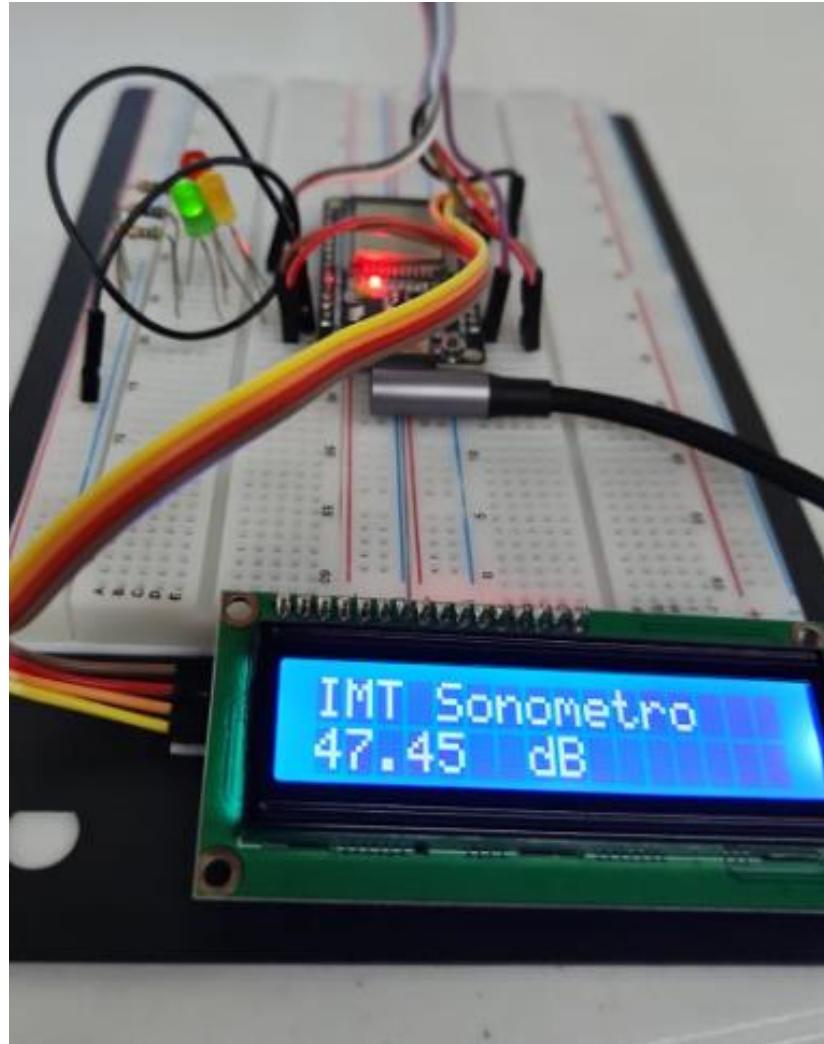
Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM9')

```
23:41:12.116 -> load:0x400080404,len:3196
23:41:12.116 -> entry 0x4000805a4
23:41:12.431 -> IMT - Engenharia da Computação - 2025
23:41:16.657 -> Setup - Iniciando a conexão WiFiSetup - Tentando Conexão Wifi: 1
23:41:17.689 ->
23:41:17.689 -> WiFi conectado.
23:41:17.689 -> WiFi SSID: ZTE_E58644
23:41:17.689 -> IP Modulo: 192.168.1.154
23:41:17.689 ->
23:41:20.752 -> 43.94
23:41:20.796 -> Menor que 60
23:41:21.288 -> 46.44
23:41:21.333 -> Menor que 60
23:41:21.820 -> 46.91
23:41:21.861 -> Menor que 60
```

Lab 4 – Conclusões

Listar as observações e conclusões do laboratório

Lab 4 – Conclusões

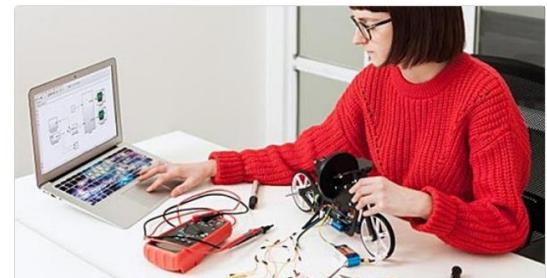
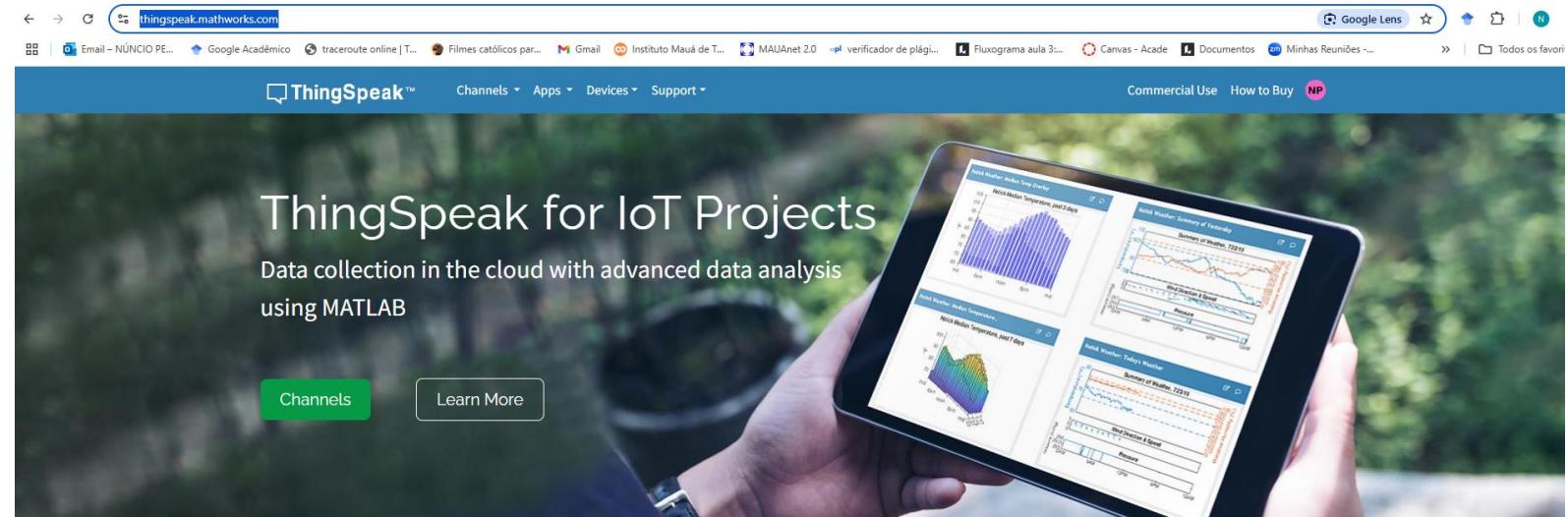


LAB 5

ThingSpeak

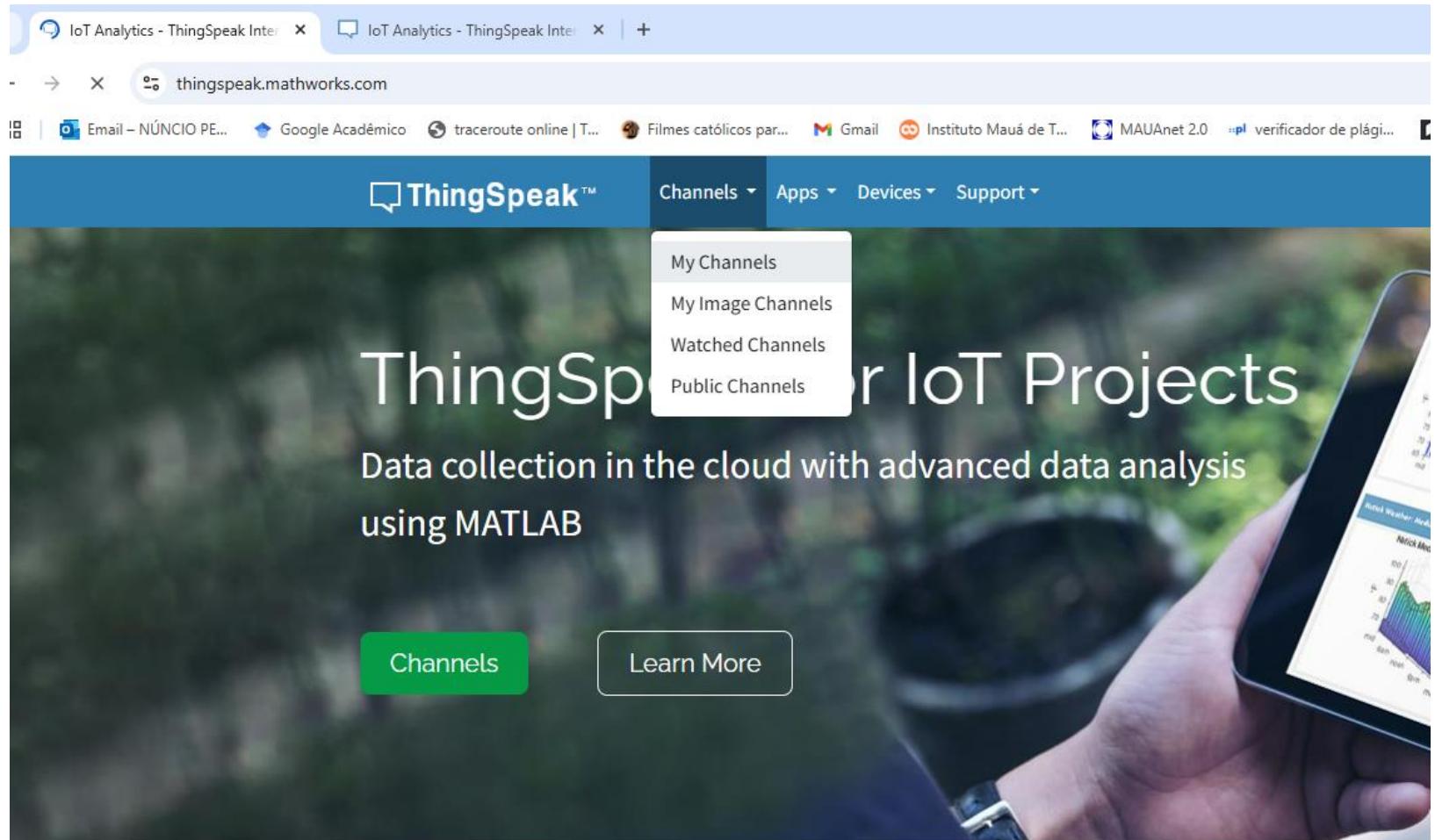
Acessar: <https://thingspeak.mathworks.com/>

Criar conta

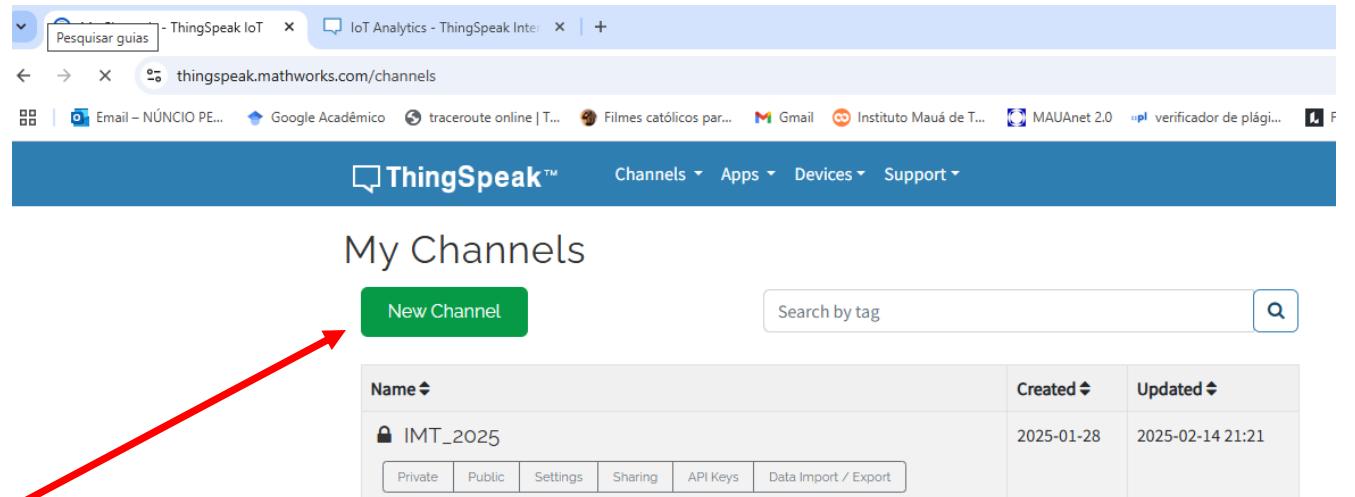


ThingSpeak

→ My Channels



ThingSpeak



The screenshot shows a web browser window for ThingSpeak. The address bar displays 'thingspeak.mathworks.com/channels'. The main content area is titled 'My Channels' and shows a table with one row. The table has columns for 'Name', 'Created', and 'Updated'. The single entry is 'IMT_2025' (with a lock icon), created on 2025-01-28 and updated on 2025-02-14 21:21. Below the table are buttons for 'Private', 'Public', 'Settings', 'Sharing', 'API Keys', and 'Data Import / Export'. At the top of the page, there is a navigation bar with links for 'Channels', 'Apps', 'Devices', and 'Support'. A green 'New Channel' button is located at the top left of the channel list area. A red arrow points from the text 'New Channel' below the screenshot to this button.

Name	Created	Updated
IMT_2025	2025-01-28	2025-02-14 21:21

New Channel

ThingSpeak

 **ThingSpeak™**

[Channels](#) ▾ [Apps](#) ▾ [Devices](#) ▾ [Support](#) ▾

[Commercial Use](#) [How to Buy](#) [NP](#)

New Channel

Name	<input type="text" value="IMT_2025_Lab_5"/>		
Description	<input type="text"/>		
Field 1	<input type="text" value="Ruido (dB)"/>	<input checked="" type="checkbox"/>	
Field 2	<input type="text"/>	<input type="checkbox"/>	
Field 3	<input type="text"/>	<input type="checkbox"/>	
Field 4	<input type="text"/>	<input type="checkbox"/>	
Field 5	<input type="text"/>	<input type="checkbox"/>	
Field 6	<input type="text"/>	<input type="checkbox"/>	
Field 7	<input type="text"/>	<input type="checkbox"/>	
Field 8	<input type="text"/>	<input type="checkbox"/>	
Metadata	<input type="text"/>		
Tags	<input type="text"/> <small>(Tags are comma separated)</small>		
Link to External Site	<input type="text" value="http://"/>		

Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- **Percentage complete:** Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Link to External Site:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- **Show Channel Location:**
 - **Latitude:** Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.
 - **Longitude:** Specify the longitude position in decimal degrees. For example, the longitude of the city of London is -0.1275.
 - **Elevation:** Specify the elevation position meters. For example, the elevation of the city of London is 35.052.
- **Video URL:** If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.
- **Link to GitHub:** If you store your ThingSpeak code on GitHub®, specify the GitHub repository URL.

Using the Channel

You can get data into a channel from a device, website, or another ThingsSpeak channel. You can then

ThingSpeak

ThingSpeak™
Channels ▾ Apps ▾ Devices ▾ Support ▾
Commercial Use How to Buy NP

Metadata

Tags

(Tags are comma separated)

Link to External Site

Link to GitHub

Elevation

Show Channel Location

Latitude

Longitude

Show Video

YouTube

Vimeo

Video URL

Show Status

- **Longitude:** Specify the longitude position in decimal degrees. For example, the longitude of the city of London is -0.1275.
- **Elevation:** Specify the elevation position meters. For example, the elevation of the city of London is 35.052.
- **Video URL:** If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.
- **Link to GitHub:** If you store your ThingSpeak code on GitHub®, specify the GitHub repository URL.

Using the Channel

You can get data into a channel from a device, website, or another ThingsSpeak channel. You can then visualize data and transform it using ThingSpeak [Apps](#).

See [Get Started with ThingSpeak](#) for an example of measuring dew point from a weather station that acquires data from an Arduino® device.

[Learn More](#)


Save Channel

ThingSpeak



Channels ▾ Apps ▾ Devices ▾ Support ▾

Commercial Use How to Buy 

IMT_2025_Lab_5

Channel ID: 2841645

Author: mwa0000035623603

Access: Private

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

 Add Visualizations

 Add Widgets

 Export recent data

 MATLAB Analysis

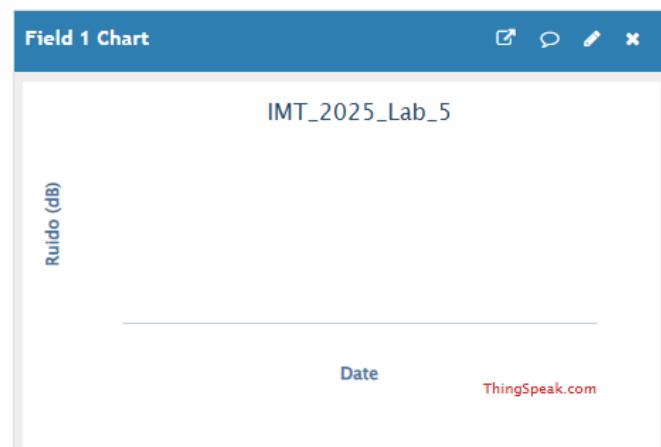
 MATLAB Visualization

Channel 2 of 2 < >

Channel Stats

Created: less than a minute ago

Entries: 0



ThingSpeak

IMT_2025_Lab_5

Channel ID: 2841645 ← **Copiar Channel Id e Write API Key**

Author: mwa0000035623603

Access: Private

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Write API Key

Key: GSA6IINXZV96XRNM →

Generate New Write API Key

Read API Keys

Key: 5NTD7WQYTTD6MJKP

Note:

Save Note Delete API Key

Add New Read API Key

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- **Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Write a Channel Feed

```
GET https://api.thingspeak.com/update?api_key=GSA6IINXZV96XRNM&field1=0
```

Read a Channel Feed

```
GET https://api.thingspeak.com/channels/2841645/feeds.json?api_key=5NTD7WQYTTD6MJK
```

Read a Channel Field

```
GET https://api.thingspeak.com/channels/2841645/fields/1.json?api_key=5NTD7WQYTTD6
```

ThingSpeak

ThingSpeak™

Channels ▾ Apps ▾ Devices ▾ Support ▾

Commercial Use How to Buy NP

IMT_2025_Lab_5

Channel ID: 2841645

Author: mwa0000035623603

Access: Private

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

+ Add Visualizations

+ Add Widgets

Export recent data

MATLAB Analysis

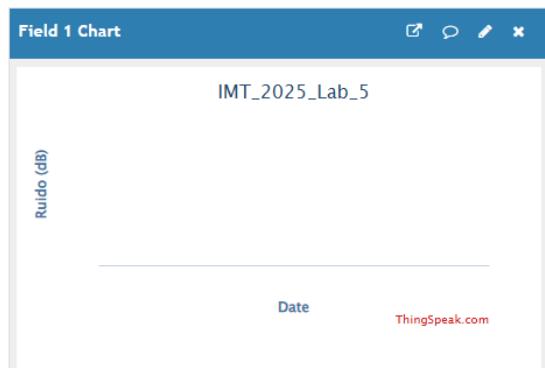
MATLAB Visualization

Channel 2 of 2 < >

Channel Stats

Created: 6 minutes ago

Entries: 0



Lab 5 – SW

Lab_5 | Arduino IDE 2.3.4

File Edit Sketch Tools Help

DOIT ESP32 DEVKIT V1

```

Lab_5.ino
25 #include <LiquidCrystal.h>
26 #include "ThingSpeak.h"
27
28 #define BUFSIZE 256
29 #define I2C_SDA 21
30 #define I2C_SCL 22
31
32
33 LiquidCrystal_I2C lcd(0x27, 16, 2); // Endereço I2C do LCD: 0x27, 16 colunas, 2 linhas
34
35
36 int LED1 = 13;           // Led Verde
37 int LED2 = 14;           // Led Amarelo
38 int LED3 = 26;           // Led Vermelho
39 int LED4 = 2;            // Led Azul placa
40 int cont = 0;
41 boolean pisca = false;
42
43 const char* ssid      = "ZTE_E58644";           // Roteador celular
44 const char* password  = "08032114";             // Senha
45
46                                         // Configurações do ThingSpeak
47 const char* apiKey   = "GSA6IINXZV96XRNM";       // Substitua pela API Key do seu canal ThingSpeak
48 const long channelID = 2841645;                  // Substitua pelo ID do seu canal ThingSpeak
49
50                                         // Variáveis para controle do tempo
51 unsigned long lastTime = 0;                      // Tempo da última execução
52 const unsigned long interval = 30000;            // Intervalo de 30 segundos (em milissegundos)
53
54 WiFiClient client;                            // Wifi simples
55
56
57
58

```

- 1 - Abrir o IDE Arduino
- 2 - Carregar Sketch: Lab_5.ino
- 3 - Colocar celular modo Roteador
- 4 - Linha 44 carregar SSID do roteador do seu celular
- 5 - Linha 45 carregar a senha do seu roteador
- 6 – Linha 48 colar API Key ThingSpeak
- 7 – Linha 49 colar Channel ID ThingSpeak
- 9- Compilar e carregar

ThingSpeak

 ThingSpeak™ Channels ▾ Apps ▾ Devices ▾ Support ▾

IMT_2025_Lab_5

Channel ID: **2841645**
Author: [mwa0000035623603](#)
Access: Private

Private View Public View Channel Settings Sharing API Keys Data Import / Export

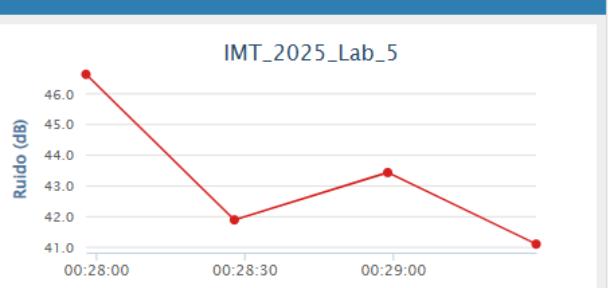
 Add Visualizations  Add Widgets  Export recent data

Channel Stats

Created: [12 minutes ago](#)
Entries: 4

Field 1 Chart

IMT_2025_Lab_5



Date	Ruido (dB)
00:28:00	46.0
00:28:30	42.0
00:29:00	43.5
00:29:30	41.0

ThingSpeak.com

Lab 5 – Conclusões

Listar as observações e conclusões do laboratório

LAB 6

Lab 6 – Trabalho de Campo

1. Desligar o Cabo USB do Computador.
2. Montar sistema com antena esférica
3. Ligar sistema Power Bank e verificar se os dados estão subindo para nuvem
4. Desligar o sistema.
5. Cada grupo irá para uma localidade da Mauá, definida
6. Nesta localidade ligar o sensor e aguardar 10 minutos de aquisição de dados
7. Retornar para laboratório para análise de dados.

Lab 6 – Trabalho de Campo

Localidades de Inspeção

1. Biblioteca
2. Laboratório de Motores
3. Cantina da Biblioteca
4. Centro Acadêmico
5. Recepção Reitoria
6. Laboratório de Eletrônica W300
7. Sala U22
8. Fablab
9. Sala Professores
10. Secretaria

Lab 6 – Análise de Dados / Dashboard.

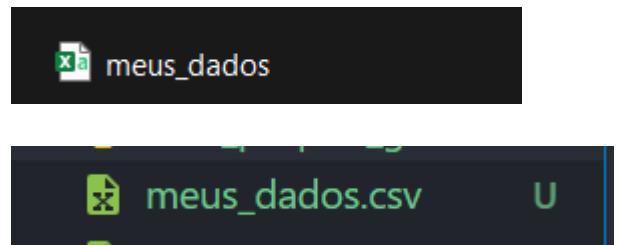
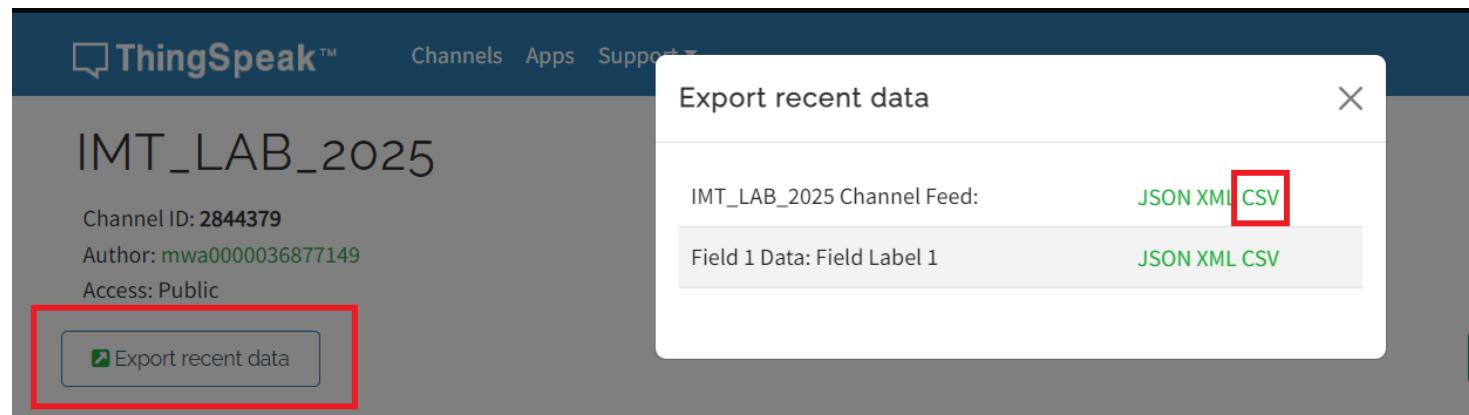
Preparando o ambiente para análise de dados:

1. Acessar Github: github.com/GabrielGodoy01/mapa_maua
2. Clonar (copiar) o repositório em sua máquina:
 - Clique em "Code" > "Download Zip" > Extraia o conteúdo
3. Abra o aplicativo "VS Code" ou "Visual Studio Code"
4. Clique em "File" > "Open Folder" > Selecione a pasta extraída
5. Abra um terminal no VS Code: "Terminal" > "New Terminal"
 - Use o seguinte comando no terminal para instalar as bibliotecas do projeto:
 - **pip install pandas numpy matplotlib folium**

Lab 6 – Análise de Dados / Dashboard.

Faça download dos dados coletados no ThingSpeak:

- Clique em "Export Recent Data"
- Depois em "CSV" (nome da extensão em formato "excel")
- Salve o arquivo como "meus_dados" dentro da pasta extraída do Github



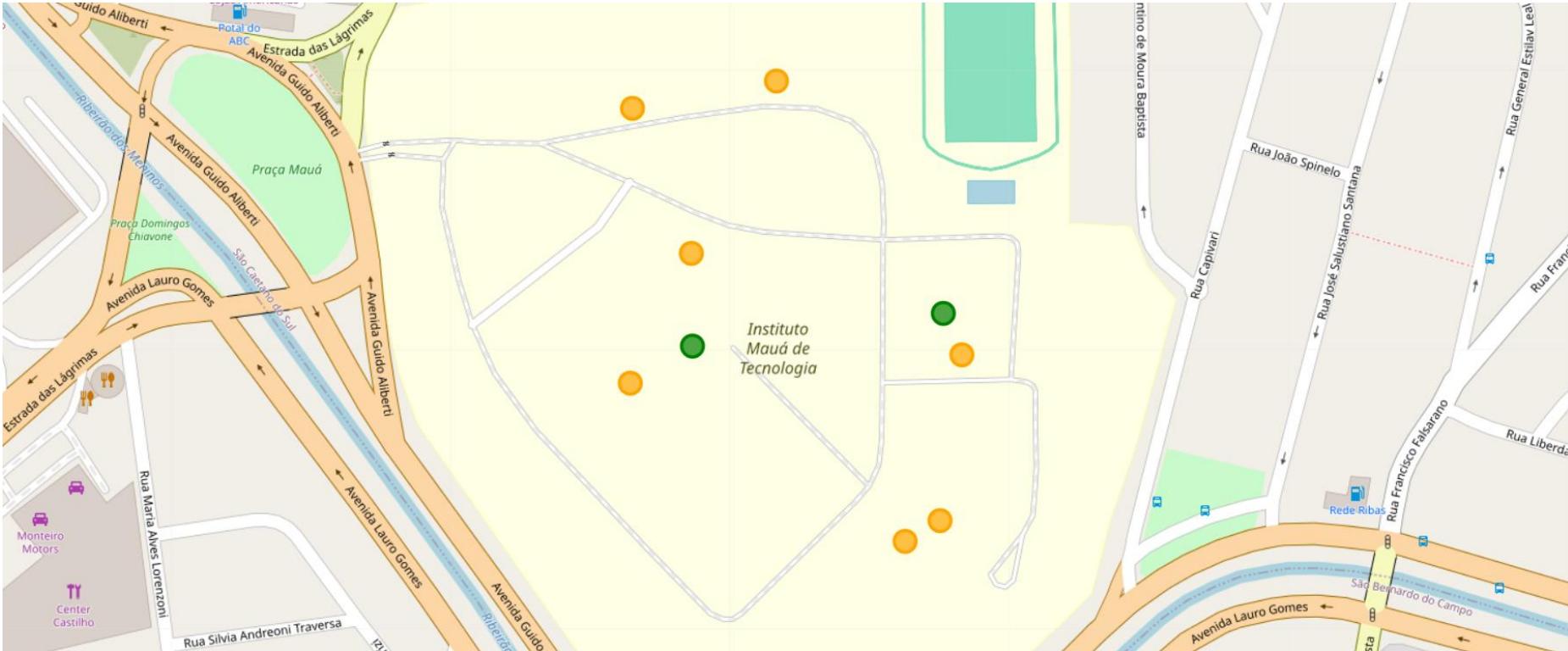
Lab 6 – Análise de Dados / Dashboard.

Analizando seus próprios dados:

1. No arquivo "meu_proprio_grafico.py" podemos usar o Python para fazer qualquer tipo de gráfico sem depender do ThingSpeak, por exemplo.
2. Abra novamente o terminal e utilize o comando para abrir o gráfico:
 - **python meu_proprio_grafico.py**
3. Aguarde, um gráfico personalizado será aberto

Lab 6 – Análise de Dados / Dashboard.

Demonstração de análise dos dados de todos os grupos:



Lab 6 – Conclusões

Listar as observações e conclusões do laboratório

Coordenador Prof. Dr. Angelo Sebastião Zanini
angelo.zanini@maua.br

INSTITUTO MAUÁ DE TECNOLOGIA



Campus São Caetano do Sul
Praça Mauá, 01 - São Caetano do Sul - SP

Prof. Nuncio Perrella
nuncio.perrella@maua.br