

Manual de Instruções

# LOCALIZADOR ONLINE DE VEÍCULOS

Estapar

## Controle do Documento

### Histórico de revisões

| Data       | Autor  | Versão | Resumo da atividade       |
|------------|--|--------|---------------------------|
| 21/11/2022 | Moises Cazé,<br>João Gonzalez,<br>Felipe Silberberg,<br>Ueliton Rocha. | 1.0    | Criação do documento.     |
| 02/12/2022 | Elias Biondo, João<br>Gonzalez, Ueliton<br>Rocha.                      | 2.0    | Criação das seções 4 e 5. |

# Índice

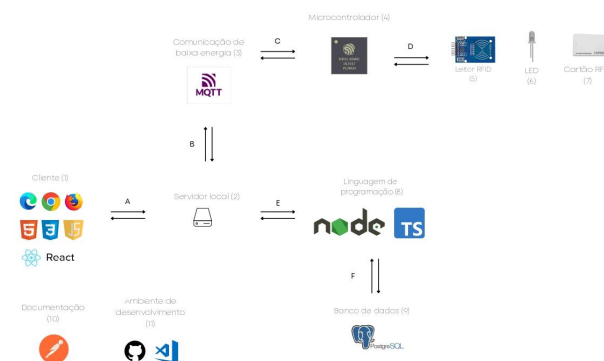
|                                  |           |
|----------------------------------|-----------|
| <b>1. Introdução</b>             | <b>3</b>  |
| 1.1. Solução                     | 3         |
| 1.2. Arquitetura da Solução      | 3         |
| <b>2. Componentes e Recursos</b> | <b>5</b>  |
| 2.1. Componentes de hardware     | 5         |
| 2.2. Componentes externos        | 5         |
| 2.3. Requisitos de conectividade | 6         |
| <b>3. Guia de Montagem</b>       | <b>7</b>  |
| <b>4. Guia de Instalação</b>     | <b>10</b> |
| <b>5. Guia de Configuração</b>   | <b>11</b> |

# 1. Introdução

## 1.1. Solução

Na ausência de um sistema de controle e monitoramento de processos, o cliente não consegue mapear o percurso feito pelo manobrista desde o recebimento do veículo até o estacionamento e vice-versa, bem como o tempo estimado para a realização dos trajetos de ida e volta. O gerente, por não ter um controle de produtividade, não possui uma maneira de gerenciar os manobristas e controlá-los. A solução é um sistema de controle que vincula o veículo ao manobrista que o conduz até a vaga na qual foi estacionado. Visando aumentar a qualidade e eficiência dos serviços oferecidos, permite aos gerentes controlar e monitorar os manobristas de acordo com sua produtividade, e aos clientes ter uma fila de espera dos veículos, bem como uma estimativa de tempo até o recebimento destes veículos.

## 1.2. Arquitetura da Solução



| Tabela de itens |                  |                |         |  |
|-----------------|------------------|----------------|---------|--|
| ID              | Objeto           | Sub-itens      | Versão  | Motivo de utilização   |
| 1               | Cliente          | Navegadores    | -       | Programas que permitem seus usuários acessarem sites na Internet.  |
|                 |                  | HTML           | 5       | Linguagem de marcação.   |
|                 |                  | CSS            | 3       | Linguagem de estilização.  |
|                 |                  | ES (JS)        | 6       | Linguagem de programação.  |
|                 |                  | React          | 17      | Biblioteca para construção de interface do usuário.  |
| 2               | Servidor         | Servidor Local | -       | Servidor local hospedado para a realização de serviços de maior performance, como processamento de imagens, tratamento de dados etc. |
| 3               | CBE              | MQTT           | 5       | Protocolo de comunicação de baixa energia.   |
| 4               | Microcontrolador | ESP32          | S3      | Módulo de alta performance e de baixa energia para aplicações.   |
| 5               | Periférico       | Leitor RFID    | -       | Sensor de reconhecimento do cartão RFID  |
| 7               |                  | LED            | -       | Diodo emissor de luz.  |
| 8               |                  | Cartão RFID    | -       | Identificação por radiofrequência (método de identificação automática através de sinais de rádio).                                   |
| 10              | LP               | NodeJS         | 16.06   | Executor de código Javascript.   |
|                 |                  | TypeScript     | 4.8.4   | Linguagem de programação.  |
| 11              | Banco de Dados   | PostgreSQL     | 12      | Sistema de gerenciamento de banco de dados relacional.   |
| 12              | Documentação     | Postman        | 10.0.33 | Plataforma de API para desenvolvedores.  |
| 13              | AD               | Git            | -       | Plataforma para versionamento de código.   |
|                 |                  | VSCode         | 1.72    | Ambiente de desenvolvimento integrado.   |

| Tabela de relacionamentos |           |   |   |
|---------------------------|-----------|---|---|
| Identificação             | Relação   | Descrição   | Exemplo de Uso  |
| A                         | 1-2       | Agentes de modificação do sistema. Envio, recebimento e consulta de dados na nuvem. | A notificação de uma nova ordem de serviço será enviada pelo servidor para o cliente.                             |
| B                         | 2-3       | Transmissão de dados tratados para os periféricos da aplicação.                     | Serão acionados, no dispositivo, os atuadores de som e luminosidade.  |
| C                         | 3-4       | Transmissão de dados com baixo custo de energia. Envio e recebimento de ordens.     | Toda a comunicação e comandos que serão executados no microcontrolador.   |
| D                         | 4-5 e 4-7 | Leitor RFID integrado no microcontrolador recebendo dados do cartão RFID.           | O manobrista aproximará sua tag RFID ao leitor e estará vinculado.  |
| D                         | 4-6       | Visibilidade do sistema, exibição de status etc.                                    | O led estará aceso continuamente na cor verde quando não associado à algum manobrista, e vermelho quando estiver. |
| E                         | 2-8       | Transmissão das ordens recebidas para as estruturas de controles responsáveis.      | Envio dos dados do manobristas para o servidor, onde serão tratados de acordo.                                    |
| F                         | 8-9       | Criação, leitura, atualização e deleção de dados. Controle do banco como um todo.   | Adição de novas ordens de serviço, manobristas, tudo que for registrado.  |

## 2. Componentes e Recursos

### 2.1. Componentes de hardware

| Componentes       | Utilidade  | Quantidade |
|-------------------|--|------------|
| ESP32-S3-WR OOM-1 | Serve para controlar sistemas eletrônicos não digitais   | 1          |
| ESP32-S3-WR OOM-1 | Serve como ponto de acesso   | 1          |
| RFID-RC522        | Permite que os usuários identifiquem e rastreiem de maneira automática e exclusiva os ativos                                       | 1          |
| Cabos Jumper      | Transmitir sinais analógicos e digitais  | 9          |
| Resistores        | Limita o fluxo da corrente elétrica  | 6          |
| Leds LDR          | Controle de iluminação   | 2          |
| Buzzer            | Geração de sinais sonoros  | 2          |
| Protoboard        | É uma placa com diversos furos e conexões condutoras verticais e horizontais para a montagem de circuitos elétricos experimentais. | 1          |
| Cabo USB          | Comunicação entre os equipamentos  | 1          |

### 2.2. Componentes externos

- Painel para que os manobristas verifiquem os veículos a estacionar e outro para o gestor captar os dados do cliente e enviar para os manobristas que precisam atender o cliente, seja estacionando ou retirando o veículo;
- Vscodex: O Visual Studio Code, que é um editor de código fonte que auxilia na criação do código de software, inclusive nas fases de testes;
- Servidor com Node.js: Node.js é um ambiente de execução Javascript, linguagem padrão de manipulação de páginas HTML. Sua escolha é embasada em uma característica muito peculiar: sua alta escalabilidade, uma vez que a execução single-thread permite criar um Event Loop com requisições que não demandam output;
- Banco de dados PostgreSQL: O Postgresql é um gerenciador de Bancos de Dados muito conhecido e usado no mundo do Desenvolvimento Web. Seus Bancos de Dados são relacionais, muito similar ao que vemos no MySQL, por exemplo;
- Documentação com Postman: O Postman é uma ferramenta que dá suporte à documentação das requisições feitas pela API. Ele possui ambiente para a documentação, execução de testes de APIs e requisições em geral;

## 2.3. Requisitos de conectividade

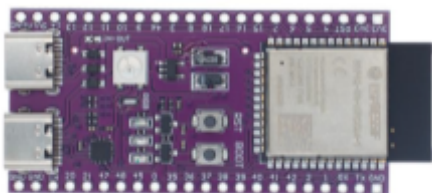
- **Protocolo I2C:** O modo de funcionamento do protocolo I2C é baseado na interação entre elementos seguindo a hierarquia mestre/escravo, ou seja, quando temos vários dispositivos se comunicando segundo esta premissa, pelo menos um destes deve atuar como mestre e os demais serão escravos. A função do mestre consiste em realizar a coordenação de toda a comunicação, pois, ele tem a capacidade de enviar e requisitar informações aos escravos existentes na estrutura de comunicação, os quais devem responder às requisições citadas.
- 
- **Rede WiFi:** O WiFi funciona através de ondas de rádio, assim como as TVs, aparelhos de rádio e celulares. A antena do roteador é a responsável por captar e emitir os sinais, bem como decodificá-los. E é assim que os aparelhos conseguem trocar informações.
- 
- **FTM:** protocolo de comunicação baseado na estimativa da distância entre dois aparelhos, através do tempo que um sinal demora para propagar de um aparelho para o outro.
- **API:** consiste em um conjunto de normas que possibilita a comunicação entre plataformas através de uma série de padrões e protocolos. Por meio desta, softwares e aplicativos, capazes de se comunicar com outras plataformas, podem ser criados.
- **Protocolo MQTT:** foi criado com o objetivo de oferecer um baixo consumo de rede, banda e também dos demais recursos de software. O formato utilizado no MQTT é de Cliente/Servidor. Para funcionar, o Protocolo MQTT utiliza um modelo de Publish/Subscribe onde permite que o cliente faça postagens e/ou capte informações enquanto o servidor irá administrar esse envio e o recebimento dos respectivos dados. Ou seja, em um MQTT haverá um publicador onde será responsável por publicar as mensagens em um determinado tópico onde um assinante irá inscrever-se neste tópico para poder acessar a mensagem. Como não há uma conexão direta entre o assinante e o publicador, para que essas mensagens aconteçam, o protocolo MQTT irá precisar de um gerenciador de mensagens chamado de Broker.
- **Protocolo SPI:** é uma interface de comunicação simples de 4 fios, outra característica do SPI é que não existe o conceito de transferência de propriedade da "barra de dados" ou seja não existe um endereço específico para o master e ou slave. O SPI opera em modo full duplex, isto significa que os dados são transferidos em ambas as direções e ao mesmo tempo e isso faz com que sua velocidade de troca de dados seja bem mais rápida, superior a 10 MHz em comparação com outros sistemas.

### 3. Guia de Montagem

#### Lista de Componentes:

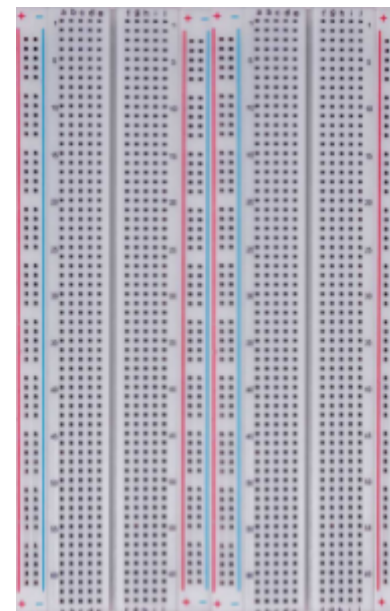
- 1 microcontrolador ESP32-S3;
- 2 leds RGB - cátodos comuns;
- 6 resistores 220r (vermelho, vermelho, marrom, dourado);
- 2 buzzers;
- 1 leitor RFID; e
- 11 jumpers Dupont macho x macho.

1. O protagonista deste fluxo é o ESP32-S3, microcontrolador de baixa potência e baixo custo. Destina-se especialmente a fornecer versatilidade, robustez e confiabilidade em um grande número de aplicações.



Fonte: [l1nq.com/esp32-image](https://l1nq.com/esp32-image)

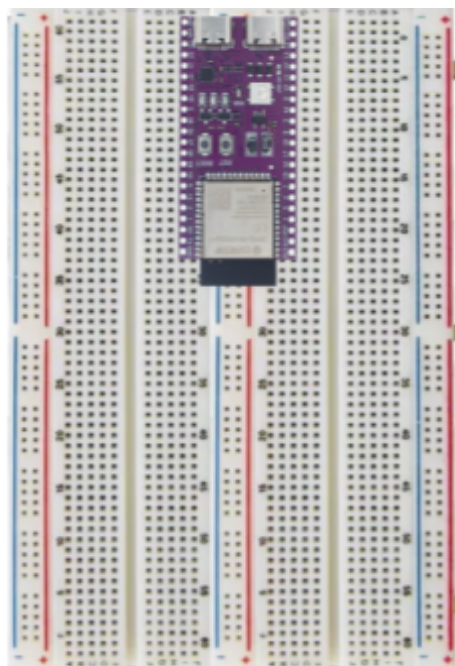
2. O suporte usado para a montagem dos circuitos deste protótipo é chamado protoboard. Uma placa na qual o microcontrolador (neste caso, o ESP) é integrado para que seja feita dentre os periféricos uma conexão que é possível através da transmissão de correntes elétricas provenientes do próprio microcontrolador. Segue o anexo abaixo:



Fonte: [l1nq.com/protoboard-image](https://l1nq.com/protoboard-image)

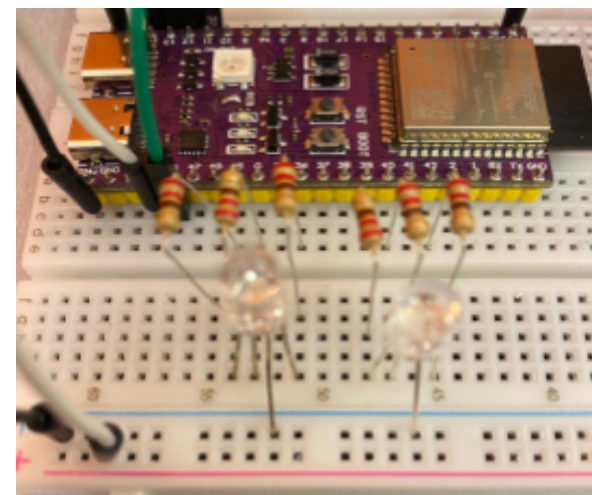


3. O ESP32 deve ser inserido com as entradas USB próximas ao final da placa, com a primeira linha horizontal que se encontra mais abaixo (onde se encontram os 2 pinos GND) posicionada na fila de número 61 do protoboard. A linha do topo, que, por sua vez, possui os pinos GND e 3U3, deve estar posicionada na linha de número 40. Segue abaixo uma demonstração:



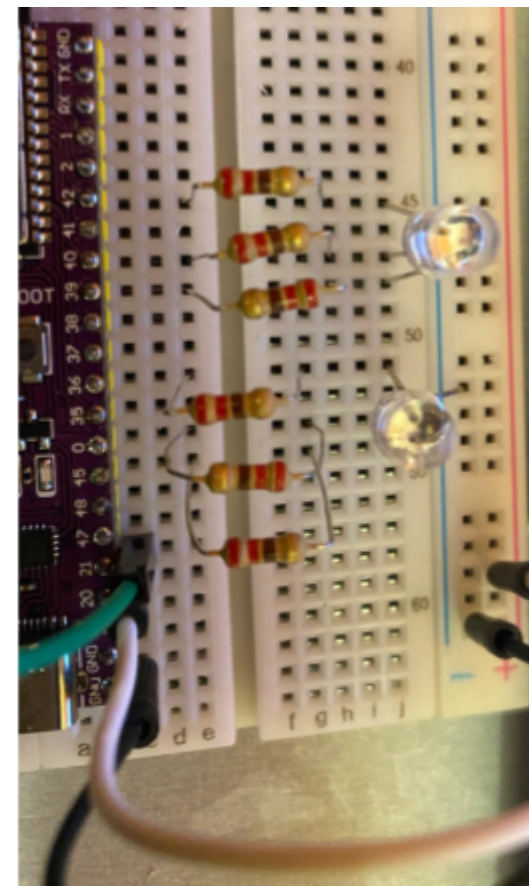
ESP32 conectada à protoboard

4. Antes de mostrar como os leds RGB devem estar posicionados, vamos a uma breve explicação sobre a estrutura deles: um cátodo comum possui 4 pinos, onde 3 são respectivos às cores vermelho, verde e azul e um, à corrente negativa de energia. Da esquerda à direita, são os pinos: vermelho, negativo (o maior de todos), verde e, por último, o da cor azul. A fila vertical posicionada à direita do sinal de negativo e a linha vertical azul, uma vez conectada a um pino GND do ESP, pode receber pinos de vários terminais negativos, o mesmo para o positivo. portanto, vamos conectar os 2 terminais negativos dos leds (pernas maiores) na última fila negativa à direita da protoboard, nos pontos 46 e 52, conforme ilustrado na imagem abaixo:



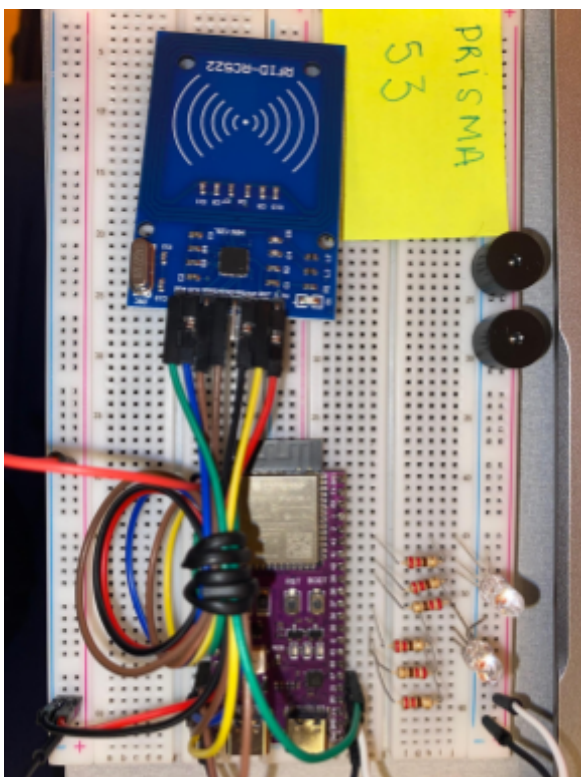
Leds RGB conectados

5. Todos os 6 resistores estão conectados aos pinos positivos (que representam as cores) dos leds. Segue a ordem: O led RGB que já possui seu pino no ponto de número 52 da fila negativa se encontrará no seguinte estado: pino azul no ponto 54, verde no 53 e vermelho no 52. Deve-se inserir um resistor na linha horizontal onde se encontra o pino azul (ponto 54) que conecta à porta 45 do ESP, outro na linha onde fica o pino verde (53) que conecta à porta 0, e, por último, um no vermelho (52) que conecta na porta 36. Quanto ao segundo led, devem ser feitas as seguintes conexões com resistores: vermelho (45) na porta 42 do ESP, verde (47) na porta 40 e azul (48) na porta 39. Segue anexo desta parte do circuito já conectada.
6. Por fim, as filas positivas e negativas da protoboard que não estejam conectadas a porta nenhuma devem ser conectadas a portas 3V3 e GNDs, respectivamente, para que a energia seja passada do microcontrolador aos periféricos conectados nelas.



Resistores nos pontos positivos do led RGB

Uma vez montado todo o circuito, a protoboard se encontrará conforme ilustrado na figura abaixo:



Circuito completamente montado

## 4. Guia de Instalação

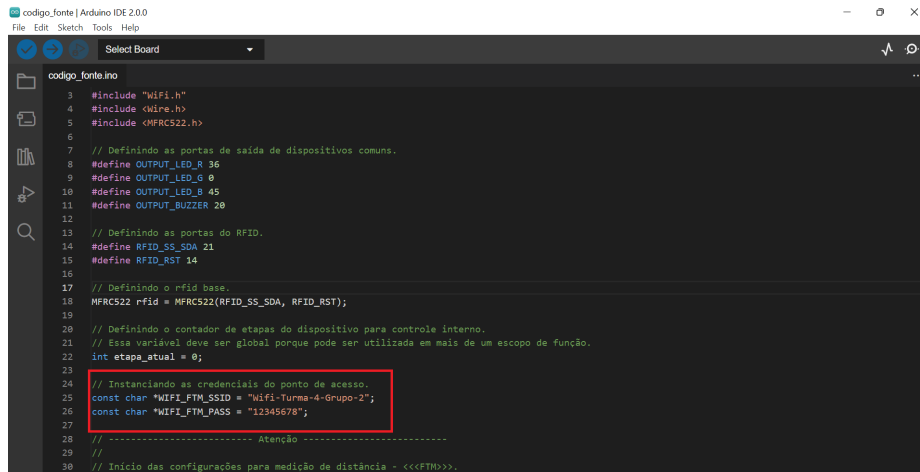
O dispositivo ficará na parte interior do prisma, e deverá ser conectado à rede WiFi do estacionamento ou em um ESP-32. Ambos podem servir como roteadores de sinal e têm o objetivo de medir a distância do local de entrega do veículo até o local onde o carro está estacionado. Assim, para que o cálculo do tempo estimado seja o mais próximo do tempo real de entrega do veículo, é indicado instalar o roteador o mais próximo possível do totem, no qual o cliente verá o tempo estimado da entrega.

Em testes realizados pelo grupo, constatamos uma distância máxima de transmissão de sinal do ESP-32 de aproximadamente quinze metros de distância com uma margem de erro de dois metros, tanto para cima quanto para baixo.

Para configurar o dispositivo e o ESP 32 que poderá servir como roteador, é necessário instalar o [Arduino IDE](#).

## 5. Guia de Configuração

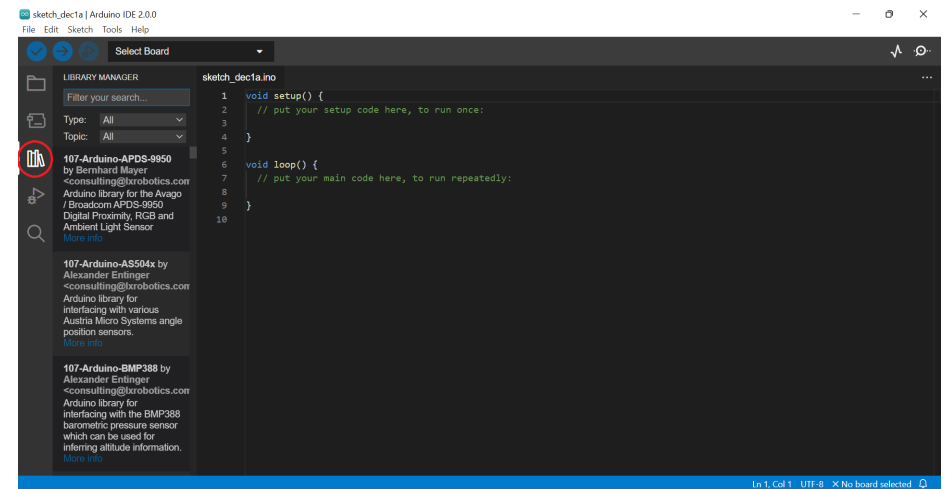
Para executar o código "codigo\_fonte.ino" é necessário configurar a rede WiFi local e instalar algumas bibliotecas. Com o fim de configurar o WiFi basta inserir o nome da rede e a senha na parte do código que está destacada na imagem a seguir:



```

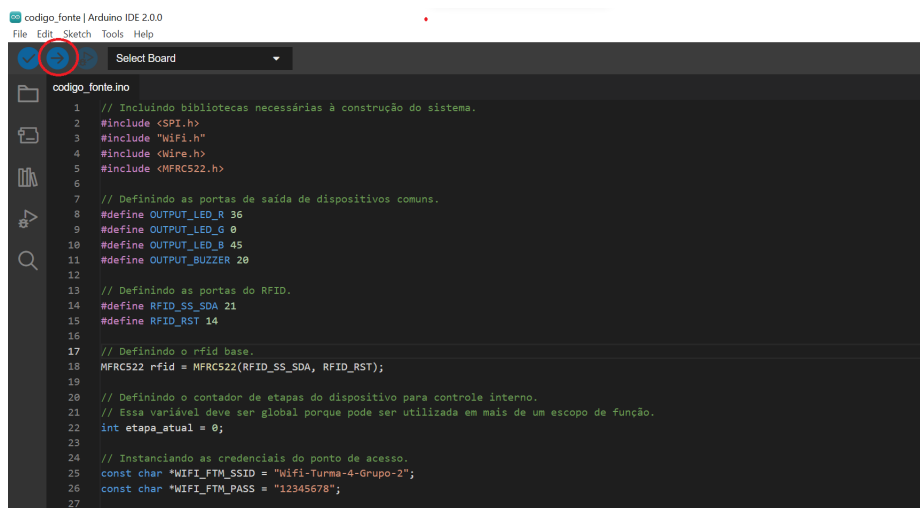
1 #include "WiFi.h"
2 #include <Wire.h>
3 #include <MFRC522.h>
4
5 // Definindo as portas de saída de dispositivos comuns.
6 #define OUTPUT_LED_R 36
7 #define OUTPUT_LED_G 0
8 #define OUTPUT_LED_B 45
9 #define OUTPUT_BUZZER 20
10
11 // Definindo as portas do RFID.
12 #define RFID_SS_SDA 21
13 #define RFID_RST 14
14
15 // Definindo o rfid base.
16 MFRC522 rfid = MFRC522(RFID_SS_SDA, RFID_RST);
17
18 // Definindo o contador de etapas do dispositivo para controle interno.
19 // Essa variável deve ser global porque pode ser utilizada em mais de um escopo de função.
20 int etapa_atual = 0;
21
22 // Instanciando as credenciais do ponto de acesso.
23 const char *WIFI_FTM_SSID = "WIFI-Turma-4-Grupo-2";
24 const char *WIFI_FTM_PASS = "12345678";
25
26 // ----- Atensão -----
27 //
28 // Início das configurações para medição de distância - <<<FTM>>>.
  
```

Para instalar as bibliotecas, clique no ícone destacado conforme a indicado na figura abaixo:



Após clicar no ícone basta pesquisar e instalar as bibliotecas a seguir: SPI, WiFi, Wire e MFRC522M.

Em seguida, é preciso compilar o código para o ESP-32. Para isso, basta conectar o dispositivo ao computador por meio de um cabo USB-C e clicar na opção de upload que está circulada em vermelho conforme a imagem a seguir:



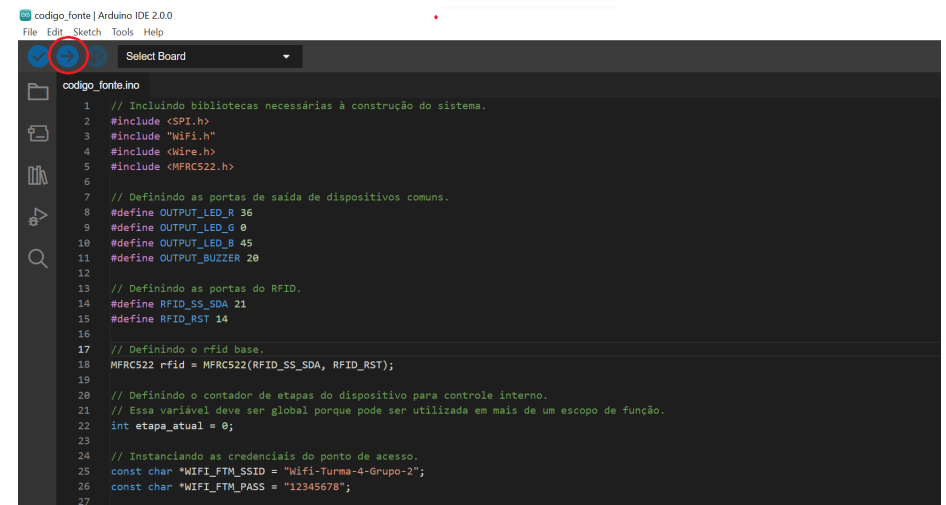
The screenshot shows the Arduino IDE 2.0.0 interface. The 'Upload' button, represented by a blue arrow pointing right, is circled in red. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The 'Select Board' dropdown menu is open, showing a list of boards. The main editor area displays the code for 'codigo\_fonte.ino'.

```

1 // Incluindo bibliotecas necessárias à construção do sistema.
2 #include <SPI.h>
3 #include "Wifi.h"
4 #include <Wire.h>
5 #include <MFRC522.h>
6
7 // Definindo as portas de saída de dispositivos comuns.
8 #define OUTPUT_LED_R 36
9 #define OUTPUT_LED_G 0
10 #define OUTPUT_LED_B 45
11 #define OUTPUT_BUZZER 20
12
13 // Definindo as portas do RFID.
14 #define RFID_SS_SDA 21
15 #define RFID_RST 14
16
17 // Definindo o rfid base.
18 MFRC522 rfid = MFRC522(RFID_SS_SDA, RFID_RST);
19
20 // Definindo o contador de etapas do dispositivo para controle interno.
21 // Essa variável deve ser global porque pode ser utilizada em mais de um escopo de função.
22 int etapa_atual = 0;
23
24 // Instanciando as credenciais do ponto de acesso.
25 const char *WIFI_FTM_SSID = "Wifi-Turma-4-Grupo-2";
26 const char *WIFI_FTM_PASS = "12345678";
27

```

Para executar o código "roteador\_wifi" basta compilar o código clicando no ícone destacado a seguir. Para compilar, basta conectar o ESP 32 ao computador por meio de um cabo USB C e clicar na opção de upload que está circulada em vermelho na imagem a seguir.



The screenshot shows the Arduino IDE 2.0.0 interface. The 'Upload' button, represented by a blue arrow pointing right, is circled in red. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The 'Select Board' dropdown menu is open, showing a list of boards. The main editor area displays the code for 'codigo\_fonte.ino'.

```

1 // Incluindo bibliotecas necessárias à construção do sistema.
2 #include <SPI.h>
3 #include "Wifi.h"
4 #include <Wire.h>
5 #include <MFRC522.h>
6
7 // Definindo as portas de saída de dispositivos comuns.
8 #define OUTPUT_LED_R 36
9 #define OUTPUT_LED_G 0
10 #define OUTPUT_LED_B 45
11 #define OUTPUT_BUZZER 20
12
13 // Definindo as portas do RFID.
14 #define RFID_SS_SDA 21
15 #define RFID_RST 14
16
17 // Definindo o rfid base.
18 MFRC522 rfid = MFRC522(RFID_SS_SDA, RFID_RST);
19
20 // Definindo o contador de etapas do dispositivo para controle interno.
21 // Essa variável deve ser global porque pode ser utilizada em mais de um escopo de função.
22 int etapa_atual = 0;
23
24 // Instanciando as credenciais do ponto de acesso.
25 const char *WIFI_FTM_SSID = "Wifi-Turma-4-Grupo-2";
26 const char *WIFI_FTM_PASS = "12345678";
27

```