



*Serviço Nacional de Aprendizagem Industrial*  
**PELO FUTURO DO TRABALHO**

# **Situação de Aprendizagem**

## **Montagem Smart 4.0**

Novembro/2023  
**Florianópolis/SC**


**Alunos:** Alessandro Bento, Cesar Glufke, Daniel Barros, Jefferson Cunha

## SUMÁRIO

1. INTRODUÇÃO .....	3
2. REDES .....	4
2.1. MQTT.....	5
2.2. ENDEREÇO IP .....	11
2.3. DASHBOARD.....	10
3. DESCRIÇÃO GERAL DO SISTEMA DESENVOLVIDO .....	11
3.1. SENSOR DE GÁS MQ.....	13
3.2. ATUADORES .....	19
4. REQUISITOS DO SISTEMA .....	21
4.1. REQUISITOS FUNCIONAIS.....	21
4.2. REQUISITOS NÃO FUNCIONAIS .....	21
4.3. ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA.....	22
4.4. DIAGRAMAS.....	24
5. RESULTADOS.....	26
5.1. CÓDIGO COMENTADO.....	26
5.2. CIRCUITO.....	30
6. REFERÊNCIAS.....	31

## 1. INTRODUÇÃO

Este documento apresenta o funcionamento dos sistemas de sensores de análise da etapa de montagem da Smart 4.0. Um protótipo com fins de aprendizagem na matéria Modelagem de Sistemas do curso de Análise e Desenvolvimento de Sistemas.

Composto por 4 sensores módulos sensor Lm393 de obstáculo infravermelho e um módulo ESP32 e uma protoboard, que detectam a presença da base no ponto de montagem e as possíveis lâminas nas laterais. As informações serão mostradas ainda em uma dashboard do sistema Taigo.io, ao lado dos dados provenientes do sistema  coletados com o Node-red.

## 2. REDES

### O que é TagoIO?

TagoIO é uma plataforma web, 100% cloud e de alto nível para monitoramento de ambientes via dispositivos IoT conectados à sua rede. Ela oferece diversas funcionalidades para que uma solução de IoT pode ser agilmente desenvolvida e implementada no mercado. Tais aplicações podem acontecer em segmentos como: automação industrial, irrigação inteligente, localização interna de depósitos, composição, refrigeração e telemática, entre outros.

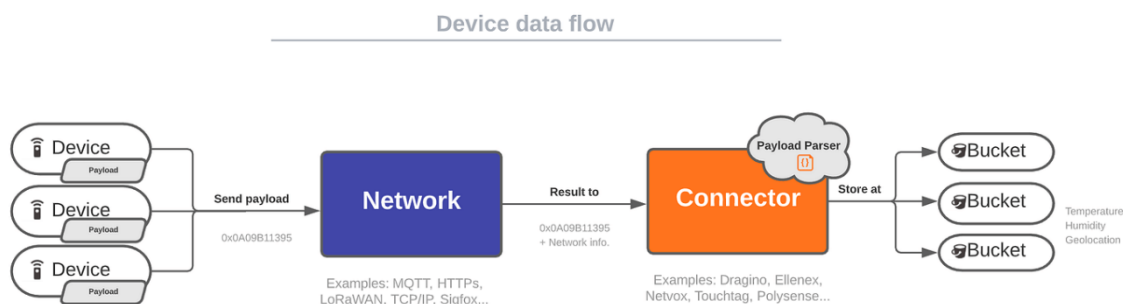
Por meio da plataforma, o usuário pode observar os dados coletados em um dashboard prático para construir suas aplicações. Tendo um painel simples e dinâmico, é possível compartilhar e rastrear o uso da aplicação, além de criar níveis de acesso para diversos usuários, definindo o que cada um poderá visualizar e editar.

### Como funciona a plataforma TagoIO?

O processo de criação da solução IoT acontece em quatro etapas. A primeira envolve conexão de um dispositivo por Wi-Fi, LoRa, Sigfox, GPRS, LTE, BLE, ZigBee, entre outras, que realize a leitura dos ambientes e envio dos dados para a plataforma.

Os dispositivos podem se conectar diretamente à API do TagoIO utilizando os protocolos HTTPS ou MQTT, embora também seja possível utilizar diferentes protocolos para bancos de dados e serviços web a fim de realizar essa conexão.

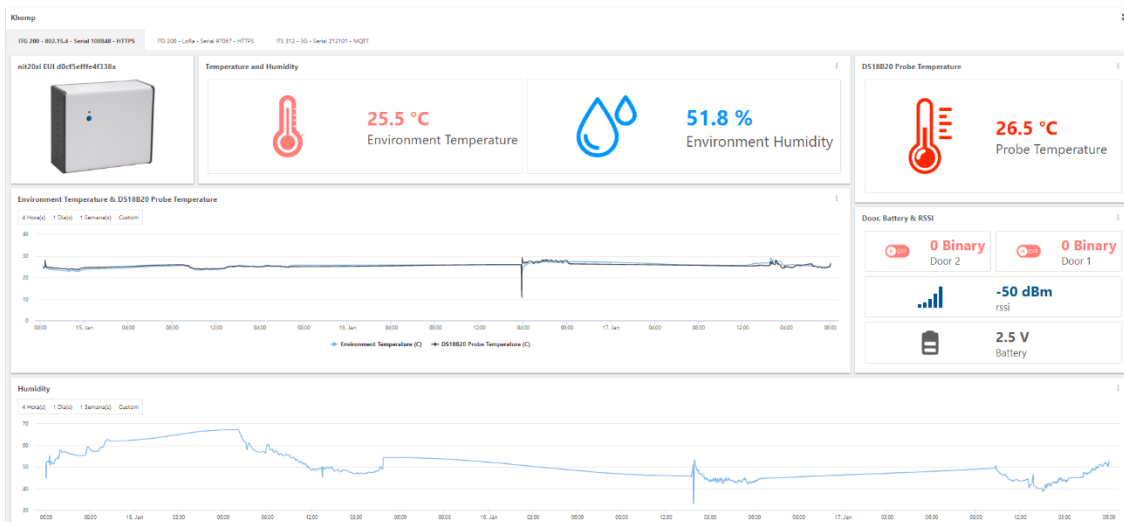
Abaixo, uma representação do fluxo de dados (*data flow*), partindo do dispositivo:



Depois, será necessário escolher um template para iniciar a construção da aplicação e integrar as soluções, combinando os dados com sistemas externos. Para finalizar o processo, é preciso criar e gerenciar os usuários e dispositivos, escalando a aplicação de forma rápida.

Após tal configuração, o dashboard no sistema já estará pronto para o monitoramento.

O usuário poderá criar diversos painéis por meio dos recursos para mapas e marcadores, além da possibilidade de adicionar links, imagens, formulários, gráficos, vídeos e outros widgets que melhor se adequem à solução, em tempo real.



O usuário poderá criar diversos painéis por meio dos recursos para mapas e marcadores, além da possibilidade de adicionar links, imagens, formulários, gráficos, vídeos e outros widgets que melhor se adequem à solução, em tempo real.

## 2.1. MQTT

Desenvolvido na década de 90 pela IBM e Eurotech, o MQTT (Message Queuing Telemetry Transport, e tendo tradução para português sob o nome de Transporte de Filas de Mensagem de Telemetria) é um protocolo de mensagens que 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.

Por esse motivo e por ter fundamentos na pilha TCP/IP ou em outros protocolos de rede, o MQTT tem extrema utilidade dentro da área de desenvolvimento de projetos de comunicação entre máquinas, também conhecido pelo termo M2M (Machine to Machine). Outra área também onde se torna muito presente é para conectividade de IoT (Internet of Things).

O protocolo MQTT funciona conforme os princípios do modelo de publicação/assinatura. Na comunicação em rede tradicional, clientes e servidores se comunicam diretamente entre si. Os clientes solicitam recursos ou dados do servidor, e o servidor processa e envia uma resposta. Porém, o MQTT usa um padrão de publicação/assinatura para desacoplar o remetente da mensagem (publicador) do destinatário da mensagem (assinante). Em vez disso, um terceiro componente, chamado agente de mensagens ou *Broker*, soluciona a comunicação entre publicadores e assinantes. O trabalho do Broker é filtrar todas as mensagens recebidas dos publicadores e distribuí-las corretamente aos assinantes.

Abaixo, uma visão geral de como o MQTT funciona:

1. Um cliente MQTT estabelece uma conexão com o agente MQTT (Broker).
2. Depois de conectado, o cliente pode publicar mensagens, assinar mensagens específicas ou fazer as duas coisas.
3. Ao receber uma mensagem, o Broker a encaminha aos assinantes interessados.

Vamos esmiuçar os detalhes para melhorar a compreensão.

### **Tópico do MQTT**

O termo “tópico” refere-se a palavras-chave que o Broker usa para filtrar mensagens para os clientes MQTT. Os tópicos são organizados de maneira hierárquica, semelhante a um diretório de arquivos ou pastas. Por exemplo, imagine um sistema de casa inteligente que está em funcionamento em uma casa de vários andares que tem diferentes dispositivos inteligentes em cada andar.

### **Publicação de MQTT**

Os clientes MQTT publicam mensagens contendo o tópico e os dados em formato de bytes. O cliente determina o formato de dados, como dados de texto, dados binários, arquivos XML ou JSON. Por exemplo, uma lâmpada no sistema de casa inteligente pode publicar uma mensagem “on” em um tópico intitulado “livingroom/light”.

### **Assinatura de MQTT**

Os clientes MQTT enviam uma mensagem SUBSCRIBE (ASSINAR) ao agente MQTT para receber mensagens sobre tópicos de interesse. Esta mensagem contém um identificador exclusivo e uma lista de assinaturas. Por exemplo, o aplicativo de casa inteligente de seu telefone deseja exibir quantas luzes estão acesas em sua casa. Ele assinará o tópico *light* e aumentará o contador para todas as mensagens *on*.

No protocolo MQTT nós temos 3 qualidades de serviço (ou “QoS” - Quality of Service) e cada conexão com o Broker pode especificar qual será utilizada., sendo estas: "no máximo uma vez", "no mínimo uma vez" e "exatamente uma vez".

#### **QoS 0 - No máximo uma vez:**

Conhecido como *fire and forgot* (atirar e esquecer), nesse QoS a mensagem é enviada apenas uma vez e não haverá passos seguintes, dessa forma a mensagem não será armazenada, nem haverá um feedback para saber se ela chegou ao destinatário.

Esse modo de transferência é o mais rápido, porém o menos seguro já que a mensagem será perdida caso o envio falhe ou o cliente esteja desconectado.

#### **QoS 1 - Pelo menos uma vez:**

Nesse modo de transferência, a mensagem é entregue pelo menos uma vez, havendo uma espera da recepção de feedback da entrega da mensagem, o chamado PUBACK. Não recebendo o PUBACK, a mensagem continuará sendo enviada até que haja o feedback. Nesse QoS pode acontecer de a mensagem ser enviada diversas vezes e ser processada diversas vezes.

Para que haja o envio da mensagem mais de uma vez, a mensagem precisa ser armazenada. Ela será excluída do receptor após ter recebido o feedback de confirmação do envio.

### **QoS 2 - Exatamente uma vez:**

Nesse modo de transferência, a mensagem é entregue exatamente uma vez, necessitando que a mensagem seja armazenada localmente no emissor e no receptor até que seja processada. Para garantir a segurança desse QoS é necessário o envio de 2 pares de request-response (chamado de four-part handshake), onde temos o envio da mensagem (PUBLISH), a resposta de recepção (PUBREC), o aviso do recebimento do PUBREC (PUBREL) e a confirmação de que o processo foi concluído e pode ser feita a exclusão (PUBCOMP). Após o recebimento do PUBREL, o *receiver* pode excluir a mensagem e, quando o sender receber o PUBCOMP, ele poderá excluir a mensagem.

## **2.2. ENDEREÇO IP**

O IP, em linhas gerais, é um código numérico atribuído a cada dispositivo conectado em uma rede. Como esse número é único para cada aparelho, ele pode ser encarado como um endereço desse equipamento. Por exemplo, se a sua rede doméstica tem cinco dispositivos conectados, cada um deles terá um IP diferente para essa rede.

É muito importante deixar claro que o computador tem dois IPs diferentes e independentes entre si: um é o código da sua rede interna, e o outro, vinculado ao seu modem - fixo ou não - é atribuído pela sua operadora e enxergado pela Internet como endereço da sua rede.

### **Quais são as diferenças entre IP dinâmico e IP estático/fixo?**

O **IP dinâmico** é o mais comum e se refere, principalmente, a um endereço que muda sempre, normalmente quando você liga o modem, ou em intervalos de tempo definidos pelo provedor. É o padrão ideal para uso doméstico, já que não requer equipamentos de melhor performance e não depende de conhecimentos um pouco mais avançados para configuração e manutenção.

O **IP estático** ou fixo, é mais raro e, em alguns casos, sua oferta pelo provedor está vinculada a taxas adicionais. Como é possível deduzir a partir da explicação sobre o dinâmico, o código fixo é um endereço de IP imutável. Ou seja, seu computador sempre terá o mesmo endereço, desde que conectado à rede com o fixo (se você levar o laptop para uma viagem e conectar de outro lugar, ele terá um IP diferente).

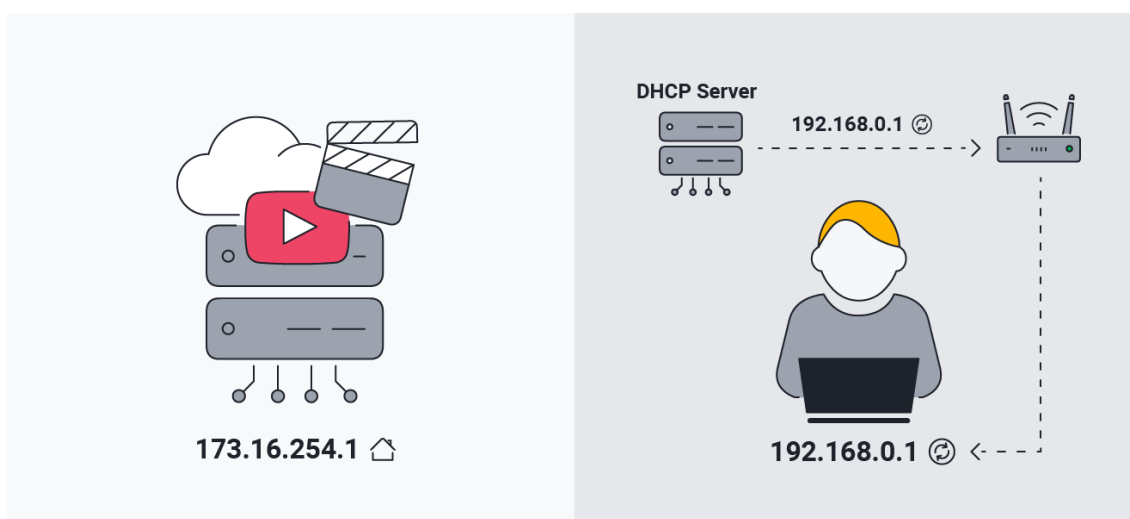


A principal diferença entre um endereço IP dinâmico e um estático é a conexão uniforme que os IPs estáticos oferecem. As mudanças do IP dinâmico do seu dispositivo pessoal acontecem quando você fica online em diferentes redes. Mas sites comerciais, como Netflix, CNN ou Facebook, precisam de endereços IP estáticos para ajudar os clientes a se conectarem sem problemas a eles.

IPs estáticos garantem que a velocidade e a qualidade da conexão permaneçam as mesmas e, para sites de streaming, que nenhum vídeo seja interrompido. Mas, embora os endereços IP estáticos geralmente ofereçam melhores conexões e velocidades, eles geralmente custam mais caro.

Por que os endereços IP mudam? A resposta breve é que não há endereços IP suficientes para todos usarem a internet. Os endereços IP estáticos exigem uma configuração manual complexa, enquanto os dinâmicos são configurados e atribuídos automaticamente, como a conexão doméstica que você provavelmente está usando agora.

Endereços IP estáticos são úteis para grandes servidores que hospedam uma quantidade enorme de tráfego, enquanto os dinâmicos são melhores para dispositivos domésticos. Os endereços IP estáticos mantêm uma conexão permanente, enquanto os dinâmicos podem mudar a qualquer momento.



## 2.3. DASHBOARD

O dashboard é um painel de informações que facilita a interpretação e o acompanhamento de indicadores importantes.

Há alguns benefícios na utilização de uma dashboard, como:

- Atualização e acompanhamento dos indicadores
- Centralização das informações
- Tomada de decisão mais rápida
- Identificação de padrões
- Facilidade de comunicação
- Otimização

O Widget **Display** é uma tela que exibe informações de modo visual, como uma TV ou uma tela de um celular.

O Widget **Line (gráficos em linha)** é como o nome diz, um gráfico em linha que serve para visualizar uma data ou hora que foi recebido tal informação, e o valor desta informação, ficando gravada como uma forma de histórico.

O Widget **VuMeter**, medidor de volume de áudio, mais conhecido como VU (unidade de volume) é um dispositivo de medição de áudio. Para este trabalho utilizaremos como um medidor de gás, já que na plataforma Tago.io não a este widget disponível, então adaptaremos outro para o nosso uso.

O Widget **PushButton** é basicamente um botão de liga e desliga que ao clicar envia informações para outra plataforma.

### 3. DESCRIÇÃO GERAL DO SISTEMA DESENVOLVIDO

Tendo como area de trabalho o modulo de montagem do sistema Smart 4.0 o objetivo foi atraveis da implementação de um sistema de sensores externa, analisar o coreto posicionamento das laminas na base, e possibilitando a visualização atraveis das dashboard, via web.

- **Uma placa ESP32**, O com código desenvolvido no IDE do Arduino, liguagem C#. Resebendo as informações dos sensores de presença infra vermelho, e tendo como saída o envio da leitura para o sstema Taigo.IO. A placa ESP32 é alimentação via USB. E é um microcontrolador usado em diversos projetos de IoT (Internet das Coisas), robótica, automação residencial e outros projetos que envolvem conexão com a internet. Ele é composto basicamente por um processador Xtensa Dual-Core de 32 bits, uma porta micro-USB para alimentação e programação, e um conversor USB serial integrado, além de já possuir WiFi nativo.



- **Módulo sensor Lm393 de o obstáculo infravermelho**

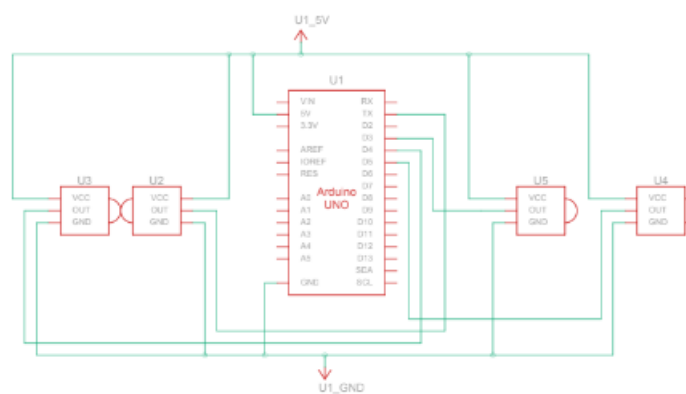


- **Uma protoboard** de 400 pontos interligadas, que servirá como matriz de contatos para a construção dos circuitos;
- **Um cabo de alimentação USB**, para alimentação do sistema e comunicação do NodeMCU com o computador;
- **Cabos jumper** macho-macho para fazer as conexões entre os

componentes eletrônicos na protoboard;

- **Dois resistores**, que servirão para ajudar na corrente que ligará os LED's;

### 3.1. ESQUEMA ELETRICO ARDUINO



Na figura acima podemos ver o esquema elétrico dos sensores e arduino;

- **VCC** fornece energia para o módulo. Deve ser conectado no pino 5V da placa de desenvolvimento utilizada;
- **GND** é o pino de aterramento / neutro, e precisa ser conectado ao pino GND da placa;
- **D23 D26 D32 D33** pinos que recebem a leitura digital dos sensores infravermelhos

## 4. REQUISITOS DO SISTEMA

## 4.1. REQUISITOS FUNCIONAIS

### 4,1,1, [RF001] Registrar a concentração do gás CO2 no ambiente

**Prioridade:**    ☒Essencial            ☐Importante            ☐Desejável

O sistema deve permitir a coleta e o registro dos dados referentes à concentração do gás CO2 no ambiente.

### 4,1,2, [RF002] Registrar a presença do gás GLP no ambiente

**Prioridade:**    ☒Essencial            ☐Importante            ☐Desejável

O sistema deve permitir a coleta e o registro dos dados referentes à presença do gás GLP no ambiente.

### 4,1,3, [RF003] Sinalizar níveis de segurança dos gases ao usuário

**Prioridade:**    ☒Essencial            ☐Importante            ☐Desejável

O sistema deve emitir sinais visuais e sonoros para alertar o usuário sobre os níveis de segurança dos gases no ar.

## 4.2. REQUISITOS NÃO FUNCIONAIS

### 4,2,1, [NF001] Dashboard Digital

O sistema deve permitir, em uma única tela, o registro dos dados coletados através de displays e gráficos. Esta dashboard é gerada e administrada por meio da plataforma Tago.IO.

### 4,2,2, [NF002] Comunicação Wi-Fi com Tago.IO

O sistema deve permitir a comunicação com a plataforma Tago.IO via wi-fi.

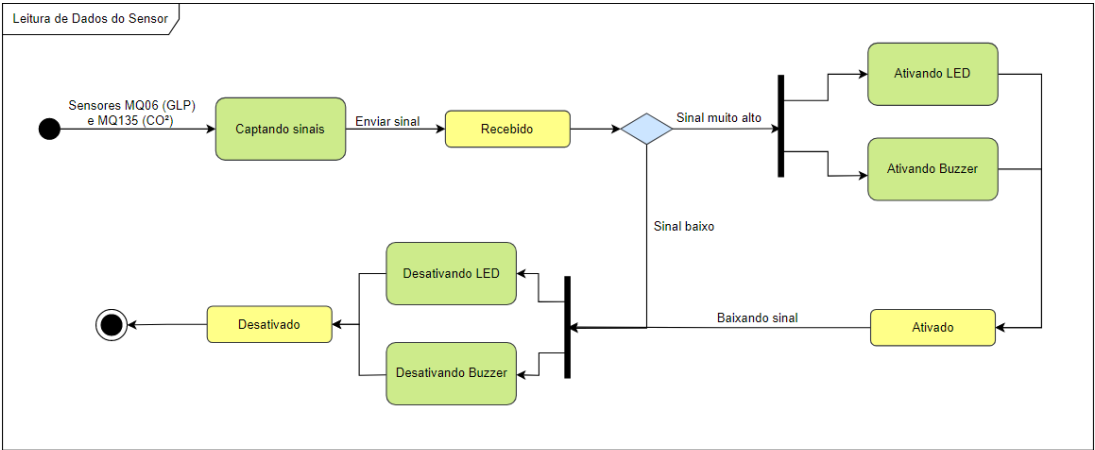
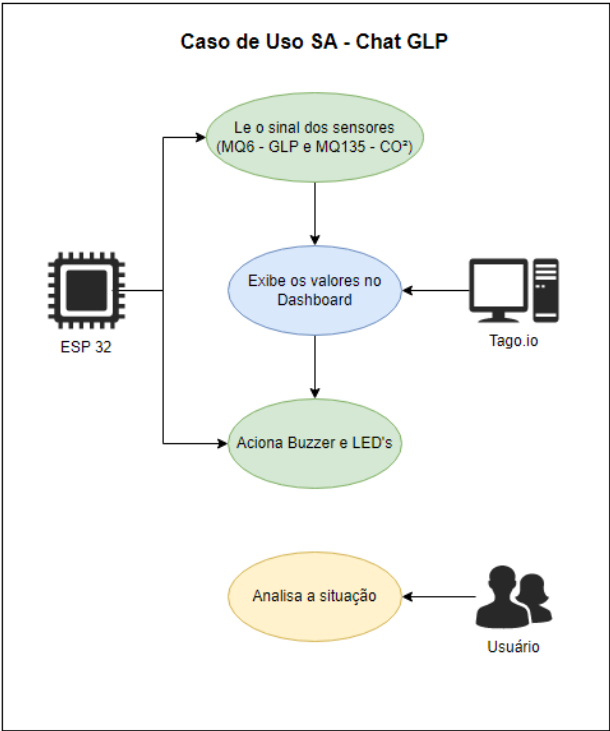
### 4.3. ESPECIFICAÇÃO DE REQUISITOS DO SISTEMA

<b>RS001</b>	<b>Registrar a concentração do gás CO2 no ambiente</b>
<i>Referência</i>	[Registrar a concentração do gás CO2 no ambiente. RF001]
<i>Sumário</i>	O caso de uso é responsável por coletar e registrar a concentração do gás CO2 no ambiente.
<i>Pré-condições</i>	O sistema deve estar ligado à alimentação via USB e logado na plataforma Tago.IO com as devidas configurações do protocolo MQTT. [NF002]
<i>Atores</i>	Sensor MQ-135, Placa ESP32 e Tago.IO
<i>Descrição</i>	<ol style="list-style-type: none"> <li>1. O sensor MQ-135 coleta dados sobre a presença do gás CO2 (dióxido de carbono) no ambiente e envia ao pino analógico da placa ESP32.</li> <li>2. O programa da placa ESP32 processa os dados coletados e os envia à plataforma Tago.IO para registro.</li> <li>3. O Tago.IO recebe e registra os dados, atualizando sua dashboard com as devidas informações por meio de displays e gráficos [NF002].</li> </ol>
<i>Alternativas</i>	--
<i>Exceção</i>	--

<b>RS002</b>	<b>Registrar a presença do gás GLP no ambiente</b>
<i>Referência</i>	[Registrar a presença do gás GLP no ambiente. RF002]
<i>Sumário</i>	O caso de uso é responsável por coletar e registrar a presença do gás GLP no ambiente.
<i>Pré-condições</i>	O sistema deve estar ligado à alimentação via USB e logado na plataforma Tago.IO com as devidas configurações do protocolo MQTT. [NF002]
<i>Atores</i>	Sensor MQ-6, Placa ESP32 e Tago.IO
<i>Descrição</i>	<ol style="list-style-type: none"> <li>1. O sensor MQ-6 coleta a resposta digital sobre a presença do gás GLP (gás de cozinha) no ambiente e envia ao pino digital da placa ESP32.</li> <li>2. O programa da placa ESP32 processa o dado coletado (0 ou 1) e o envia à plataforma Tago.IO para registro.</li> <li>3. O Tago.IO recebe e registra os dados, atualizando sua dashboard com as devidas informações por meio de displays e gráficos [NF002].</li> </ol>
<i>Alternativas</i>	--
<i>Exceção</i>	--

<b>RS003 Sinalizar níveis de segurança dos gases ao usuário</b>	
<i>Referência</i>	[Sinalizar níveis de segurança dos gases ao usuário. RF003]
<i>Sumário</i>	O caso de uso é responsável por registrar o estoque de ingredientes.
<i>Pré-condições</i>	O sistema deve estar ligado à alimentação via USB e logado na plataforma Tago.IO com as devidas configurações do protocolo MQTT. [NF002]
<i>Atores</i>	LED's (1 vermelho e 1 verde) e Buzzer
<i>Descrição</i>	<ol style="list-style-type: none"> <li>1. Os sensores MQ-6 e MQ-135 coletam dados sobre a presença dos gases GLP (gás de cozinha) e CO2 (dióxido de carbono), respectivamente, e enviam à placa ESP32.</li> <li>2. O programa da placa ESP32 processa os dados coletados e confere se o CO2 excede o limite da concentração de gás permitida no ar e se existe a presença de GLP.</li> <li>3. Em caso afirmativo para qualquer uma das condições do item 2, o sistema aciona um buzzer com alarme sonoro, e o LED vermelho.</li> <li>4. Em caso negativo, o buzzer é desativado e o LED verde é aceso (o que é considerado o estado padrão ideal do sistema.)</li> </ol>
<i>Alternativas</i>	--
<i>Exceção</i>	--

4.4. DIAGRAMAS





## Detalhamento de caso de uso

### Cenário típico:

- 1 - Começa quando os sensores leem os sinais;
- 2 - O ESP32 capta esses sinais e envia para uma plataforma;
- 3 - A plataforma Tago.io exibe os valores em uma dashboard;
- 4 - O ESP32 verifica se o sinal é mais alto do que o esperado;

### Fluxo alternativo:

- 1 - No passo 4, caso o sinal esteja dentro do esperado, um LED verde é acionado e o Buzzer permanece desligado.

### Requisitos especiais:

- 1 - O LED verde sempre fica acionado se o sinal estiver dentro do esperado, o LED vermelho e o Buzzer só são acionados caso o sinal seja muito alto.

## Historico de alterações

Data	Versão	Descrição	Autor
16/05/2023	0.1	Criação do protótipo físico do sistema com sensores MQ-6, MQ-135, placa ESP32 e buzzer.	João Carlos Becker Jr. Arthur G. Becker
23/05/2023	0.2	Criação e configuração do sistema para comunicação com a plataforma Tago.IO.	João Carlos Becker Jr. Arthur G. Becker
30/05/2023	0.3	Adição dos LEDs verde e vermelho ao sistema, e sua condicional para funcionamento no código.	João Carlos Becker Jr. Arthur G. Becker
01/06/2023	0.4	Implementação de código de alarme com toque personalizado para o buzzer.	João Carlos Becker Jr. Arthur G. Becker
12/06/2023	0.5	Criação da marca e personalização estética do dashboard no Tago.IO.	João Carlos Becker Jr. Arthur G. Becker
15/06/2023	1.0	Revisão e ajustes finais aos comentários do código.	João Carlos Becker Jr. Arthur G. Becker

## 5. RESULTADOS

### 5.1. CÓDIGO COMENTADO

```
/*
 * Dióxido de Carbono PPM (Parts Per Million)
 *
 * Equipe Chat GLP:
 * Arthur Guilherme Becker
 * João Carlos Becker Junior
 *
 * JUNHO 2023
 */

/*
 * Nível de CO2 na Atmosfera.....400ppm
 * Média CO2 ambientes internos.....350-450ppm
 * Nível máximo de CO2 aceitável.....1000ppm
 * Níveis perigosos de CO2.....>2000ppm
 *
 * Referências: ScottyD www.youtube.com/c/learnelectronics
 */

//-----
//                                LIBRARIES
//-----
#include <ArduinoJson.h>           // inclui a biblioteca ArduinoJson
#include "EspMQTTClient.h"         // inclui a biblioteca EspMQTTClient
#include "pitches.h"               // inclui a biblioteca EspMQTTClient

//-----
//                                DEFINES
//-----
#define MQ135          33          // pino analógico do sensor MQ135
#define MQ6             34          // pino digital do sensor MQ6
#define Buzzer         25          // pino analógico do buzzer
#define led_vermelho   26          // pino analógico do led vermelho
#define led_verde       27          // pino analógico do led verde

//-----
//                                VARIÁVEIS
//-----
char json_MQ135[100];             // formatação da mensagem para envio ao MQTT
char json_MQ6[100];               // formatação da mensagem para envio ao MQTT
int limiteSensor = 1000;          // define um valor máx. para a leitura do gás
int valor_MQ135;                  // inicia a variável valor_MQ135
int valor_MQ6;                   // inicia a variável valor_MQ6
int calibragem = 0;               // inicia uma variável para ajuste manual do valor lido
bool buzzerBoolean;              // inicia uma var booleana destinada a ligar/desligar o buzzer
```

```
//-----  
//                                     ALARME  
//-----  
int alarme[] = {                                // Notas para a melodia do alarme  
    0, NOTE_A7, NOTE_C8, NOTE_DS8, 0, NOTE_A7, NOTE_C8, NOTE_DS8, 0, NOTE_A7,  
NOTE_C8, NOTE_DS8  
};  
int noteDurations[] = {                        // Duração das notas 4 = um quarto de nota, etc.  
    2, 4, 4, 1, 2, 4, 4, 1, 2, 4, 4, 1  
};  
  
//-----  
//                                     CONFIGURAÇÕES MQTT  
//-----  
EspMQTTClient client  
(  
    "FIESC_IOT",                                // nome da rede Wi-Fi  
    "C6qnM4ag81",                              // senha da rede Wi-Fi  
    "mqtt.tago.io",                            // MQTT Broker server ip padrão da tago  
    "GLP",                                     // username  
    "353a4997-c744-469d-a5c7-7ab6ef2eee5a",    // Código do Token  
    "TestClient",                             // Client name that uniquely identify your device  
    1883                                         // The MQTT port, default to 1883. this line can be omitted  
);  
  
//-----  
//                                     SETUP  
//-----  
void setup()  
{  
    pinMode(MQ135, INPUT);                      // configura o pino do sensor MQ135 como entrada  
    pinMode(MQ6, INPUT);                       // configura o pino do sensor MQ6 como entrada  
    pinMode(Buzzer, OUTPUT);                   // configura o pino do buzzer como saída  
    pinMode(led_verde, OUTPUT);                 // configura o pino do led verde como saída  
    pinMode(led_vermelho, OUTPUT);              // configura o pino do led vermelho como saída  
    Serial.begin(9600);                        // código do serial  
}  
  
void converte_json()                          // conversão json para envio de dados  
{  
    StaticJsonDocument<300> sjson_MQ135;  
    StaticJsonDocument<300> sjson_MQ6;  
  
    int ppm = (valor_MQ135 / 4) - calibragem; // ajuste do valor recebido pelo sensor  
mq135 para envio  
    int mq6read = valor_MQ6;  
  
    sjson_MQ135["variable"] = "mq135";         // atribui o valor de mq135 à "variable"  
    sjson_MQ135["value"] = ppm;                // atribui o valor de ppm ao "value"  
    serializeJson(sjson_MQ135, json_MQ135);   // empacotamento de dados para envio
```

```
sjson_MQ6["variable"] = "mq6";           // atribui o valor de mq6 à "value"
sjson_MQ6["value"] = mq6read;           // atribui o valor de mq6read ao "value"
serializeJson(sjson_MQ6, json_MQ6);     // empacotamento de dados para envio
}

void envia_msg()
{
  client.publish("node/mq135", json_MQ135); // envio para o topic mqtt "node/mq135"
  client.publish("node/mq6", json_MQ6);     // envio para o topic mqtt "node/mq6"
}

void tocaAlarme() {                      // Função do alarme

  if (buzzerBoolean){                    // Condicional: se a variável for verdadeira...
    int wholeNoteDuration = 1000 / 4;    // Calcula a duração de uma nota inteira
    // Passa pelas notas musicais do alarme
    for (int i = 0; i < sizeof(alarme) / sizeof(alarme[0]); i++) {
      // Calcula a duração da nota
      int noteDuration = wholeNoteDuration / noteDurations[i];
      tone(Buzzer, alarme[i], noteDuration); // Toca a nota no buzzer
      delay(noteDuration + 50);              // Delay p/ a duração da nota + pausa
      noTone(Buzzer);                       // Para de tocar o buzzer
    }
  }
}

//-----
//                               MAIN LOOP
//-----

void loop()
{
  valor_MQ6 = digitalRead(MQ6);           // recebe a leitura do sensor na var 'valor_MQ6'
  valor_MQ135 = analogRead(MQ135);        // recebe a leitura na variável 'valor_MQ135'
  int ppm = (valor_MQ135 / 4) - calibragem; // divide o valor recebido por 4, sub-
  trai uma calibragem (se necessário) e guarda na variável 'ppm'

  converte_json();                       // chama a função 'converte_json' definida no setup
  envia_msg();                           // chama a função 'envia_msg' definida no setup

  if(ppm > limiteSensor || valor_MQ6 == 1){ // condicional: se os valores lidos fo-
  rem maior do que o permitido
    tocaAlarme();                        // chama a função "tocaAlarme()"
    digitalWrite(led_verde, LOW);        // desativa o led verde
    digitalWrite(led_vermelho, HIGH);    // ativa o led vermelho
  }else{                                // caso contrário
    digitalWrite(led_verde, HIGH);       // ativa o led verde
    digitalWrite(led_vermelho, LOW);     // desativa o led vermelho
  }

  Serial.print("Sensor CO2 | MQ135 valor: "); // imprime a string no Serial Monitor
```

```
Serial.println(ppm); // imprime o valor de ppm no Serial Monitor
Serial.print("Sensor GLP | MQ6 valor: "); // imprime a string no Serial Monitor
Serial.println(valor_MQ6); // imprime o valor de ppm no Serial Monitor
delay(1000); // delay de leitura em milisegundos

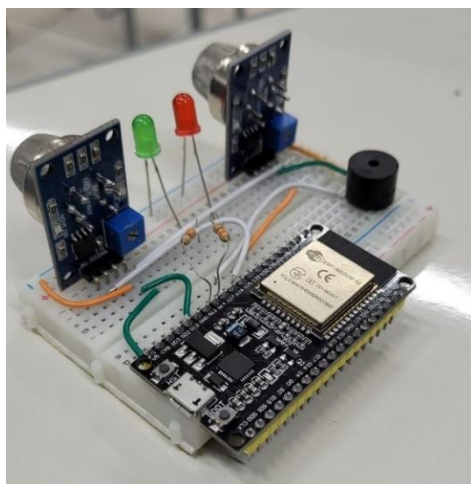
client.loop();
}

//-----
// MQTT
//-----
// processamento do payload do TagoIO
void processa_msg(const String payload)
{
    StaticJsonDocument<300> msg;

    DeserializationError err = deserializeJson(msg, payload);
    if (err) {
        Serial.print(F("deserializeJson() failed with code "));
        Serial.println(err.f_str());
    }
    String var = msg["variable"]; // recebe o nome da variável enviado pelo Ta-
    goIO dentro de var
    if(var == "buzzer") // se a variável lida for "buzzer"...
    {
        String val = msg["value"]; // recebe o valor da variável enviado pelo Ta-
        goIO dentro de val
        if(val == "ligado"){ // caso o valor de val seja igual a "ligado"...
            buzzerBoolean = true; // recebe o valor "true" dentro da var buzzerBoolean
            Serial.println("Alarme LIGADO"); // imprime "Alarme LIGADO" no s. monitor
        }else{ // caso contrário...
            buzzerBoolean = false; // recebe o valor "false" dentro da var buzzerBoolean
            Serial.println("Alarme DESLIGADO"); // imprime "Alarme DESLIGADO" no s. moni-
            tor
        }
    }
}

// Função chamada assim que tudo estiver conectado (Wifi e MQTT)
void onConnectionEstablished()
{
    client.subscribe("node/status", [] (const String &payload) {
        Serial.println(payload);
        processa_msg(payload);
    });
}
```

## 5.2. CIRCUITO



## 6. REFERÊNCIAS

- CONSIGAZ. Consigaz, GLP - GÁS LIQUEFEITO DE PETRÓLEO, Disponível em: <https://www.consigaz.com.br/gas-glp/>. Acesso em: 09, maio 2023.
- FERREIRA, Victor Ricardo. "Monóxido de carbono". Brasil Escola. Disponível em: <https://brasilecola.uol.com.br/quimica/monoxido-carbono.htm>. Acesso em: 15 de junho de 2023
- GUSE, Rosana. Como funciona o sensor de gás MQ-135? Make Hero, 2022, Disponível em: <https://www.makehero.com/blog/como-funciona-o-sensor-de-gas-mq-135/>. Acesso em: 15 de junho de 2023

### NodeMCU:

- OLIVEIRA, Greici. NodeMCU - Uma plataforma com características singulares para o seu projeto IoT. Blog Master Walker Shop. Disponível em: <https://blogmasterwalkershop.com.br/embarcados/nodemcu/nodemcu-uma-plataforma-com-caracteristicas-singulares-para-o-seu-projeto-iot>. Acesso em: 15 de junho de 2023
- Wikipedia, NodeMCU. Disponível em: <https://en.wikipedia.org/wiki/NodeMCU>. Acesso em: 15 de junho de 2023
- Electronic Wings. Introduction to NodeMCU. Disponível em: <https://www.electronicwings.com/nodemcu/introduction-to-nodemcu>. Acesso em: 15 de junho de 2023
- FAORO, Igor Wilian, Utilização de NodeMCU em projetos IoT, Micreiros.com, 2020, Disponível em: <https://micreiros.com/utilizacao-de-nodemcu-em-projetos-iot/>. Acesso em: 15 de junho de 2023

### Desenvolvimento

- MENEZES, Ana Luiza, Saiba quais são os sintomas da intoxicação por gás, Plano.news, 2019. Disponível em: <https://pleno.news/saude/saiba-quais-sao-os-sintomas-da-intoxicacao-por-gas.html>. Acesso em: 15 de junho de 2023
- COPAGAZ, Gás de cozinha: tem cheiro? É líquido? É tóxico? Veja aqui! Copgaz, 2021 Disponível em: <https://www.copagaz.com.br/blog/gas-de-cozinha-tem-cheiro/>. Acesso em: 15 de junho de 2023
- Blog Multcomercial, Protoboard: o que é e como usar. IPElab. Disponível em: <https://ipelab.ufg.br/n/156373-protoboard-o-que-e-e-como-usar>. Acesso em: 15 de junho de 2023
- Curso Baroni. Os diferentes tipos de jumpers para utilizar no protoboard - Industrializado ou caseiro, com dicas, Curso Baroni, 2021. Disponível em: <https://cursobaroni.com.br/2021/09/14/os-diferentes-tipos-de-jumpers-para-utilizar-no-protoboard-industrializado-ou-caseiro-com-dicas/>. Acesso em: 15 de junho de 2023
- MOTA, Allan, Protoboard - O que é e como usar? Vida de Silício, 2018. Disponível em: <https://portal.vidadesilicio.com.br/protoboard/>. Acesso em: 15 de junho de 2023.

## MQ-6

- VIDADESILICIO, MQ-6 - Sensor de Gás Propano e GLP. Vida de Silício. Disponível em: <https://www.vidadesilicio.com.br/produto/mq-6-sensor-de-gas-propano-glp/>. Acesso em: 15 de junho de 2023
- USINAINFO, Detector de Gás / Sensor de Gás MQ-6- GLP (Gás de Cozinha), Propano, Isobutano e Gás Natural Liquefeito. UsinaInfo. Disponível em: [https://www.usinainfo.com.br/sensor-de-gas-arduino/detector-de-gas-sensor-de-gas-mq-6-glp-gas-de-cozinha-propano-isobutano-e-gas-natural-liquefeito-2963.html?search\\_query=mq6&results=1](https://www.usinainfo.com.br/sensor-de-gas-arduino/detector-de-gas-sensor-de-gas-mq-6-glp-gas-de-cozinha-propano-isobutano-e-gas-natural-liquefeito-2963.html?search_query=mq6&results=1). Acesso em: 15 de junho de 2023

## MQ-135

- USINAINFO, Detector de Gás / Sensor de Gás MQ-135 - Amônia, Óxido Nítrico, Álcool, Benzeno, Dióxido de Carbono e Fumaça. UsinaInfo. Disponível em: [https://www.usinainfo.com.br/sensor-de-gas-arduino/detector-de-gas-sensor-de-gas-mq-135-amonia-oxido-nitrico-alcool-benzeno-dioxido-de-carbono-e-fumaca-2964.html?search\\_query=mq135&results=1](https://www.usinainfo.com.br/sensor-de-gas-arduino/detector-de-gas-sensor-de-gas-mq-135-amonia-oxido-nitrico-alcool-benzeno-dioxido-de-carbono-e-fumaca-2964.html?search_query=mq135&results=1). Acesso em: 15 de junho de 2023
- CANDIDO, Grandimilo. Sensor de Gás MQ-135 e a família MQ de detectores de Gás. Vida de Silício, 2017. Disponível em: <https://portal.vidadesilicio.com.br/sensor-de-gas-mq-135/>. Acesso em: 15 de junho de 2023

## Buzzer

- MOTA, Allan, Usando o buzzer com Arduino - Transdutor piezo elétrico. Vida de Silício, 2017. Disponível em: <https://portal.vidadesilicio.com.br/usando-o-buzzer-com-arduino-transdutor-piezo-eletrico/>. Acesso em: 15 de junho de 2023
- USINAINFO, Buzzer Ativo 5V Bip Contínuo PCI 12mm. UsinaInfo. Disponível em: [https://www.usinainfo.com.br/buzzer/buzzer-ativo-5v-bip-continuo-pci-12mm-2988.html?search\\_query=buzzer+ativo&results=28](https://www.usinainfo.com.br/buzzer/buzzer-ativo-5v-bip-continuo-pci-12mm-2988.html?search_query=buzzer+ativo&results=28). Acesso em: 15 de junho de 2023

## LED

- USINAINFO, LED Vermelho 10mm Difuso. UsinaInfo. Disponível em: [https://www.usinainfo.com.br/led-difuso/led-vermelho-10mm-difuso-3068.html?search\\_query=LED&results=439](https://www.usinainfo.com.br/led-difuso/led-vermelho-10mm-difuso-3068.html?search_query=LED&results=439). Acesso em: 16 de junho de 2023



## Redes

- ALCTEL. Alctel, 2023. O que é protocolo IOT e como funciona na prática? Disponível em: <https://www.alctel.com.br/o-que-e-protocolo-iot-e-como-funciona-na-pratica/>. Acesso em: 09, maio 2023.
- SANTOS, Guilherme. Protocolo MQTT: O Que é, Como Funciona e Vantagens. Automação Industrial, 2023. Disponível em: <https://www.automacaoindustrial.info/mqtt/>. Acesso em: 09, maio de 2023.
- AWS. AWS, 2023. O que é MQTT? Disponível em: <https://aws.amazon.com/pt/what-is/mqtt/>. Acesso em: 09, maio 2023
- NERI, Renan; LOMBA, Matheus; BULHÕES, Gabriel. MQTT. Rio de Janeiro: Escola Politécnica UFRJ - Departamento de Eletrônica, 2019. Disponível em: <https://www.gta.ufrj.br/ensino/eel878/redes1-2019-1/vf/mqtt/>. Acesso em: 09, maio de 2023.
- FREDÁ, Anthony. Endereços IP estáticos vs. dinâmicos: Qual IP é o melhor? AVG, 2022. Disponível em: <https://www.avg.com/pt/signal/static-vs-dynamic-ip-addresses>. Acesso em: 09, maio de 2023.
- GARRETT, Filipe. IP dinâmico, IP estático ou IP fixo? Saiba as diferenças, prós e contras. TechTudo, 2022. Disponível em: <https://www.techtudo.com.br/noticias/2022/11/ip-dinamico-ip-estatico-ou-ip-fixo-saiba-as-diferencas-pros-e-contras.ghtml>. Acesso em: 09, maio de 2023.
- XAVIER, Raquel Cruz. Conheça a TagoIO, a primeira ferramenta cloud para desenvolvimento de solução IoT homologada Khomp. Khomp, 2019. Disponível em: <https://www.khomp.com/pt/tagoio-solucao-iot/>. Acesso em: 11, maio de 2023.
- Redação Opentech. Importância do uso de dashboard logístico. Opentech, 2022. Disponível em: <https://opentechgr.com.br/blog/importancia-do-uso-de-dashboard-logistico/#:~:text=O%20dashboard%20%C3%A9%20um%20painel,meio%20de%20tabelas%20e%20gr%C3%A1ficos>. Acesso em: 11, maio de 2023.
- FREITAS, Carlos. VU Meter - O que é isso? Audio Reporter, 2014. Disponível em: <https://www.audioreporter.com.br/resenha/vu-meter-o-que-e-isso/>. Acesso em: 11, maio de 2023.
- Dashboard de vendas: 7 benefícios do uso da ferramenta. Reportei. Disponível em: <https://reportei.com/dashboard-de-vendas/>. Acesso em: 11, maio de 2023.