

# **Manual de Instruções**

**Mirage IoT - Prototype  
Atech**

## Controle do Documento

### Histórico de revisões

Data	Autor	Versão	Resumo da atividade
08/11/2022	Kathlyn Diwan	1.0	Atualização da seção 1.1 e 2.1
12/11/2022	Caio Martins	1.1	Atualização da seção 1.2
18/11/2022	Kathlyn Diwan	1.2	Preenchimento da seção 3.0
19/11/2022	Kathlyn Diwan	1.3	Revisão do documento
20/11/2022	Caio Martins	1.4	Atualização da seção 2.3
04/12/2022	Kathlyn Diwan	1.5	Revisão e atualização do documento.
15/12/2022	Giovana Thomé	1.6	Atualização das seções 1, 2, 3, 4 e 5



# Índice

<b>1. Introdução</b>	<b>3</b>
1.1. Solução	3
1.2. Arquitetura da Solução	4
<b>2. Componentes e Recursos</b>	<b>5</b>
2.1. Componentes de hardware	5
2.2. Componentes externos	6
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>
<b>7. Troubleshooting</b>	<b>17</b>
<b>8. Créditos</b>	<b>17</b>



# 1. Introdução

## 1.1. Solução

O IoT Prototype é uma solução proposta pela empresa Atech, do Grupo Embraer, e trazida e desenvolvida pelo Grupo 4 - Mirage, da turma 3 do Instituto de Tecnologia e Liderança – Inteli, que tem como principal objetivo a localização de ativos em um ambiente indoor.

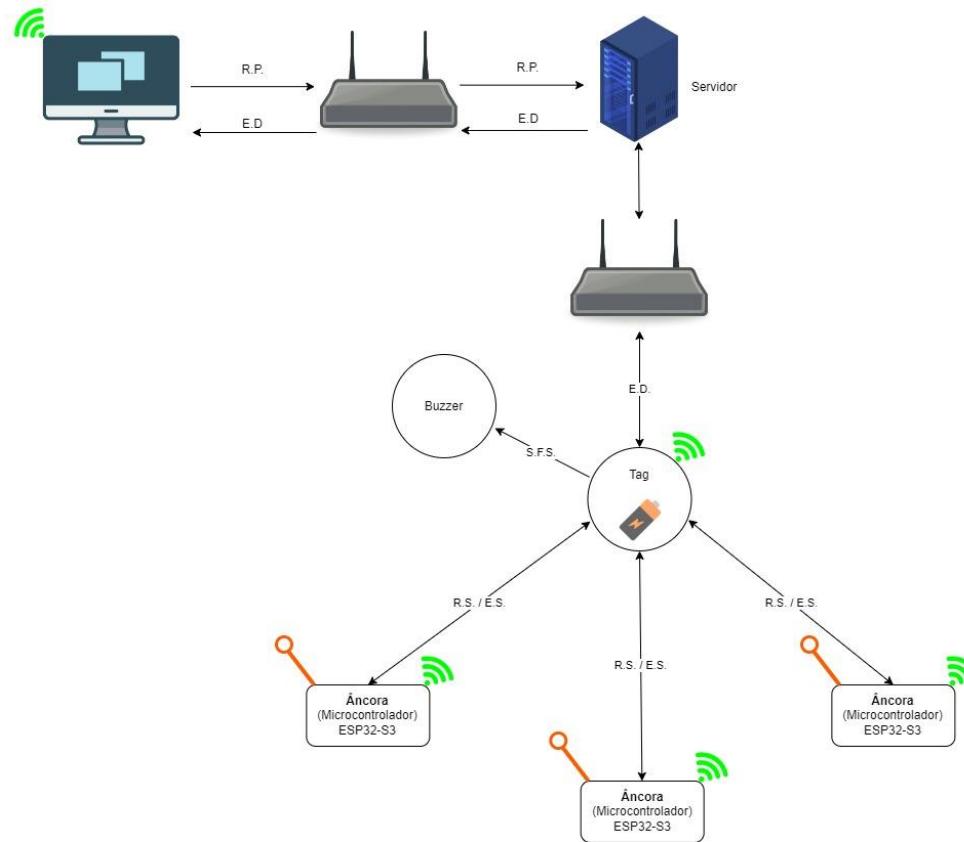
A solução não é destinada à empresa Atech, mas sim para seus clientes que desejam fazer controle de estoque e/ou outras utilizações possíveis para o sistema. O MVP apresentado consta em microcontroladores, utilizados como âncoras (beacons) e ativos (tags), que realizam troca de informações entre si via wi-fi e calculam a distância relativa do ativo, que é enviada para um servidor em nuvem que exibe as informações coletadas em uma interface web.

O presente documento tem como objetivo situar o usuário sobre os componentes envolvidos, montagem, instalação e configuração do sistema, assim como utilização do software dedicado.

O sistema de Internet of Things (IoT) desenvolvido não é embarcado, trazendo conectividade e sincronia entre software e hardware para alcançar maior potencial e entrega de valor ao usuário final.



## 1.2. Arquitetura da Solução



**Figura 1:** esquema lógico da arquitetura da solução

**E.D: Envio de dados** – Esta reta simboliza o envio dos dados processados pelos microcontroladores para o servidor e deste servidor para uma unidade de processamento.

**R.S / E.S: Recebimento de sinal e emissão de sinal** – Estas retas bidirecionais representam um caminho de dupla via entre emissor e sensor, ou seja, é possível fazer emissão de sinal de ambas as fontes e receber este sinal em ambas. Respectivamente, um será o emissor e outro receptor ou vice e versa.

**R.P: Requisição posicional** – Esta seta representa a requisição de dados do servidor para saber onde o objeto estava em um determinado instante.

**S.F.S. Sinal para frequência sonora** – Emissão de energia da tag para o Buzzer de modo a produzir uma frequência sonora pré-determinada, no momento em que a checkbox for marcada na aplicação web.



O símbolo de Wi-Fi representa a conexão que os equipamentos utilizados terão com a rede local para que seja feita a leitura de dados.



O símbolo da chave laranja representa uma conexão com a rede elétrica do local em que a âncora está localizada.





O símbolo de bateria estará contido nos sensores e representa o funcionamento destes com esta fonte de energia, uma bateria.



O roteador indica o meio de comunicação entre os microcontroladores e o servidor.



Utilizaremos microcontroladores ESP32-S3 para fazer envios e recebimentos de dados via wi-fi. Beacons serão utilizados para criação de redes wi-fi embarcadas e envio de dados para tags, que por sua vez têm papel de localizador de ativos, recebendo dados dos beacons e enviando para o servidor.



Unidade de interação com usuário: computador ou máquina em que o software de monitoramento está sendo utilizado.

## 2. Componentes e Recursos

### 2.1. Componentes de hardware

- Três (3) ou mais microcontroladores ESP32-S3-WROOM-1 Espressif ou microcontrolador equivalente compatível com o código que servirão como *beacon*.
- Um (1) ou mais microcontroladores ESP32-S3-WROOM-1 Espressif ou microcontrolador equivalente compatível com o código que servirão como *tag* a serem rastreados
- Quatro (4) ou mais cabos de conexão USB, com uma extremidade USB e outra USB tipo C. Marca e especificações de escolha do usuário. A quantidade de cabos depende da quantidade total de microcontroladores ESP32 utilizados.
- Três (3) ou mais fontes de alimentação que aceitam conexão USB dos cabos e recebem energia da tomada. Marca e especificações da escolha do usuário. A quantidade de



fontes depende da quantidade de microcontroladores utilizados como *beacons*.

- Um (1) ou mais carregadores portáteis ou qualquer tipo de alimentação portátil que alimente via USB-C. Marca e especificações da escolha do usuário. A quantidade de carregadores portáteis depende da quantidade de microcontroladores utilizados como *tags*.
- Dois (2) ou mais jumpers. Se a montagem for feita via solda, utilizar cabos de condução normais. Senão, utilizar jumpers macho-macho para utilização na protoboard. A quantidade de jumpers depende da quantidade de microcontroladores utilizados como *tags*.
- Um (1) ou mais buzzers (emissores de frequências sonoras). Marca e especificações da escolha do usuário. A quantidade de buzzers depende da quantidade de microcontroladores utilizados como *tags*.

## 2.2. Componentes externos

- Serviço de hospedagem de servidor e API CodeSandbox.
- Computador para edição e configuração dos códigos dos *beacons* e *tags*.
- Editor de código Arduino IDE ou outro compilador equivalente.

## 2.3. Requisitos de conectividade

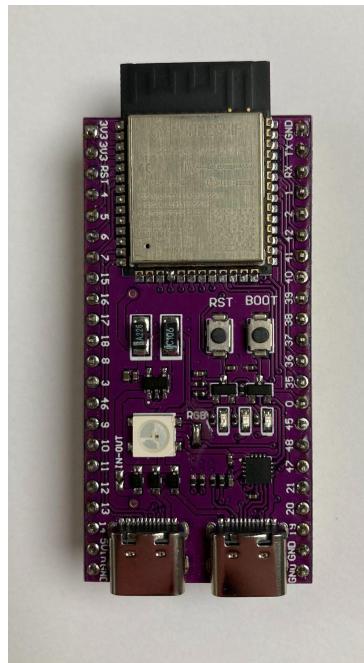
O protocolo de conexão entre beacon e tag utilizado é Transmission Control Protocol, TCP, responsável por assegurar o envio de pacotes de dados entre os microcontroladores, beacon e tag.

Para a interface web será utilizado o protocolo de comunicação HTTP para realização das requisições e respostas para o servidor.



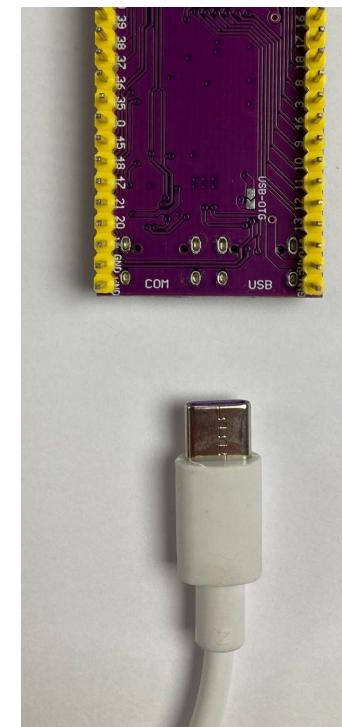
### 3. Guia de Montagem

1. Para realizar a montagem, inicialmente bastam quatro microcontroladores ESP 32 (figura 2) servindo como *beacons*



**Figura 2:** microcontrolador ESP 32-S3

2. Separar cabos USB tipo C para cada placa (figura 3)



**Figura 3:** cabo USB tipo C



3. Conectar os cabos nos microcontroladores nas portas USB (figuras 4 e 5)

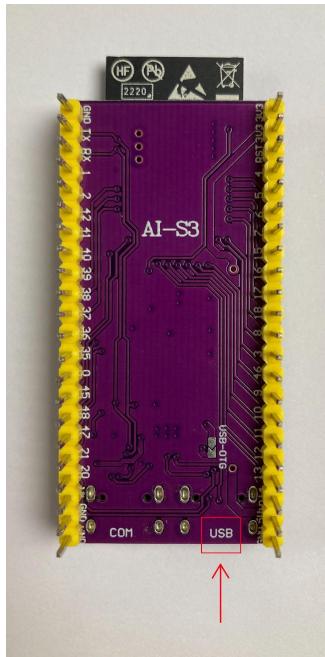


Figura 4: indicação da porta USB



Figura 5: cabo conectado à porta

4. As três placas que servirão como beacons devem estar conectadas em uma fonte de alimentação estática (tomadas) (figura 6)

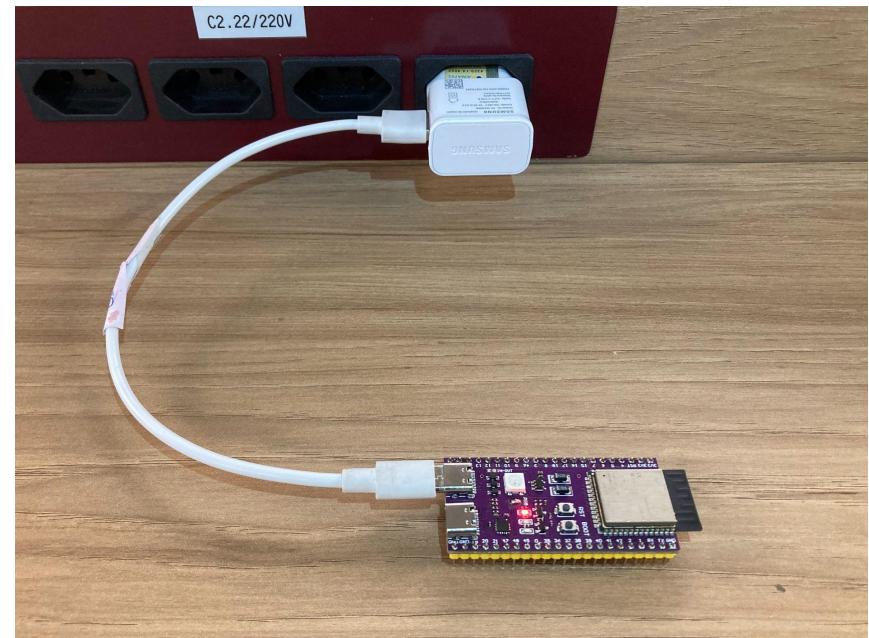
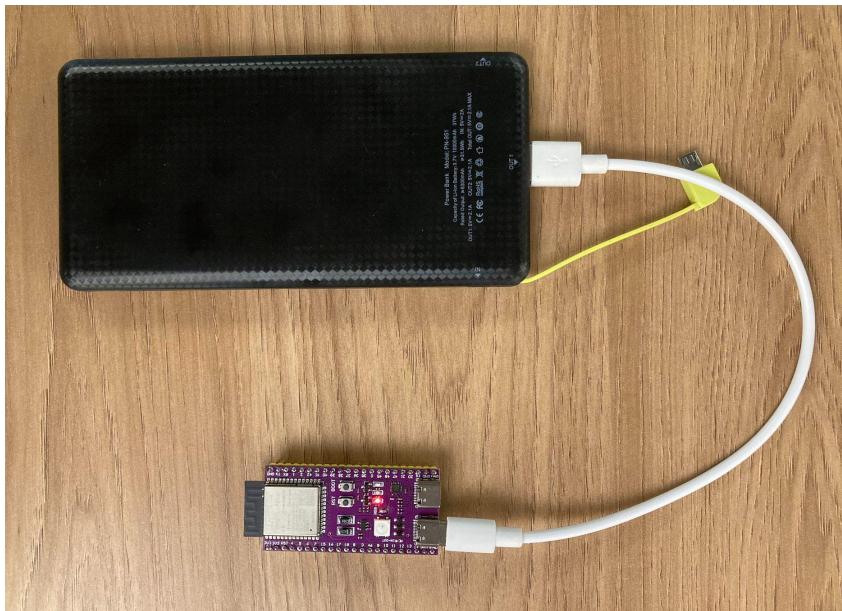


Figura 6: microcontrolador beacon conectado à tomada

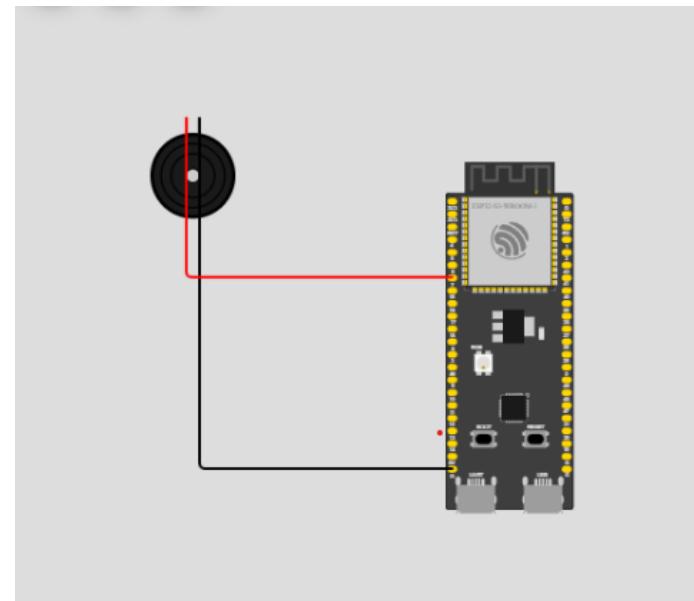


5. A(s) placa(s) que servirão como *tags* deverão estar conectadas em uma fonte de alimentação portátil de sua escolha (figura 7)



**Figura 7:** microcontrolador *tag* conectado à fonte de alimentação portátil

6. O buzzer deve ser conectado à tag como ilustrado no diagrama a seguir, seguindo as ligações positivas e negativas (figura 8):



**Figura 8:** buzzer conectado ao microcontrolador com função de *tag*

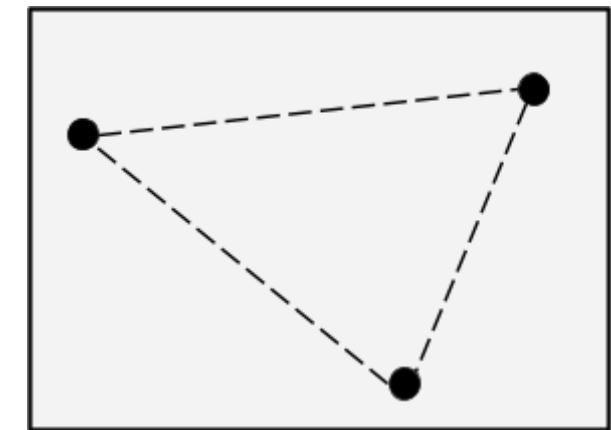
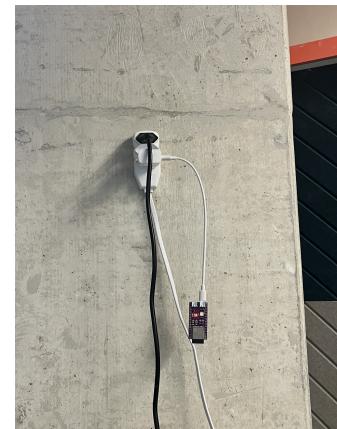


## 4. Guia de Instalação

Para instalar os dispositivos IoT no espaço físico adequado, siga as seguintes instruções:

1. Utilizando a fórmula de triangulação, posicione seus ESPs ligados a uma fonte de alimentação com uma distância de no máximo 20 metros;
2. Após posicionar os Beacons, verifique se a distância esteja correta, e certifique-se de que os Beacons, Tag e componentes externos(computador) estejam ligados à mesma rede wi-fi;
3. Os beacons serão utilizados para identificar a localização da Tag em um ambiente indoor;
4. A figura 9, mostra o Beacon conectado em uma rede/fonte de alimentação;
5. A figura 10 foi feita como exemplo para demonstrar como a triangulação foi feita no ambiente testado.
6. Depois disso, você deve acessar o código configurado já no Arduino IDE, para que você possa ver o resultado da compilação.
7. O software será utilizado com o intuito de configurar e compilar as respectivas funções e distâncias do Beacon em relação à Tag.

8. Depois dessa etapa, pode ser realizado um teste de movimento, movendo a Tag para poder analisar os resultados nas mudanças e compilação.



**Figuras 9 e 10:** Beacons conectados e demonstrados na triangulação.



## 5. Guia de Configuração

**Configuração dos beacons:** para mudar as credenciais das redes criadas pelos beacons, basta mudar as variáveis `WIFI_FTM_SSID` e `WIFI_FTM_PASS`, indicados no trecho abaixo dos seus respectivos scripts correspondentes aos microcontroladores:

```
const char *WIFI_FTM_SSID = "BEACON2G4";
const char *WIFI_FTM_PASS = "GRUPO4";
```

**Figura 11:** bloco de código de nomeação da rede criada pelos ESP32, dos scripts dos beacons `sketch_beacon_x`

Não são recomendados SSIDs ou senhas com número pequeno de caracteres ou utilização de hífen ou *underline* a fim de melhor funcionamento das redes.

**Configuração das tags:** ao alterar as credenciais dos beacons, o código da tag deve ser atualizado com as respectivas alterações nos vetores `SSIDS` e `PWD`, como indicado no trecho de código a seguir:

```
const char* SSIDS[4]={"suaRedeLocal","BEACON1-G4",
"BEACON2-G4","BEACON3-G4"};
const char* PWD[4]={ "suaSenhaLocal", "GRUPO4", "GRUPO4",
"GRUPO4"};
```

**Figura 12:** bloco de código de especificação das redes que a *tag* se conectará, do script `sketch_tag`

Além disso, deve-se alterar a rede wi-fi local que as tags estarão conectadas, também através dos vetores mencionados acima. A ordem das redes do vetor é: wi-fi local, rede do beacon 1, rede do beacon 2, rede do beacon 3 e assim por diante.

Para alterar a porta em que o buzzer será configurado e conectado fisicamente na placa da *tag*, o script `sketch_tag` deve ser alterado no seguinte trecho:

```
// codigo buzzer
const int buzzer = 2;
```

**Figura 13:** trecho de código de especificação da porta em que o buzzer está conectado na ESP32 *tag*, no script `sketch_tag`.

O número atrelado à constante `buzzer` deve corresponder ao número da porta de output da placa ESP32.



## 6.0 Guia de Operação

Figura 14: operação da página inicial do software

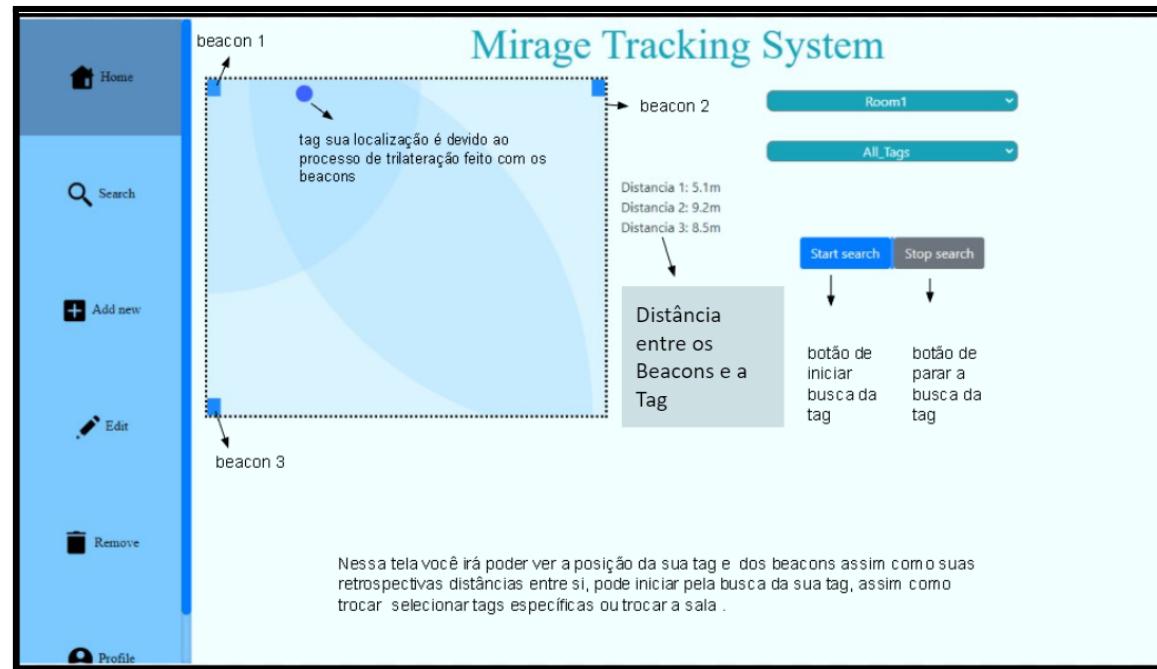


Figura 15: operação da página 'search' do software

Identificação do objeto que será rastreado, a partir dos dados já cadastrados

De acordo com as informações lidas do dispositivo IoT, o Status de sua tag, estará presente nessa coluna

**Mirage Tracking System**

Room	Tag	Status	Refresh Rate (s)
Room1	Tag1	Online	60
Room1	Tag2	Offline	1200
Room1	Tag3	Online	60000
Room2	Tag4	Online	30
Room3	Tag5	Online	30
Room4	Tag1	Online	5

O dispositivo IoT permite com que você faça uma requisição da taxa de atualização, e a informação concebida será presente nessa coluna.

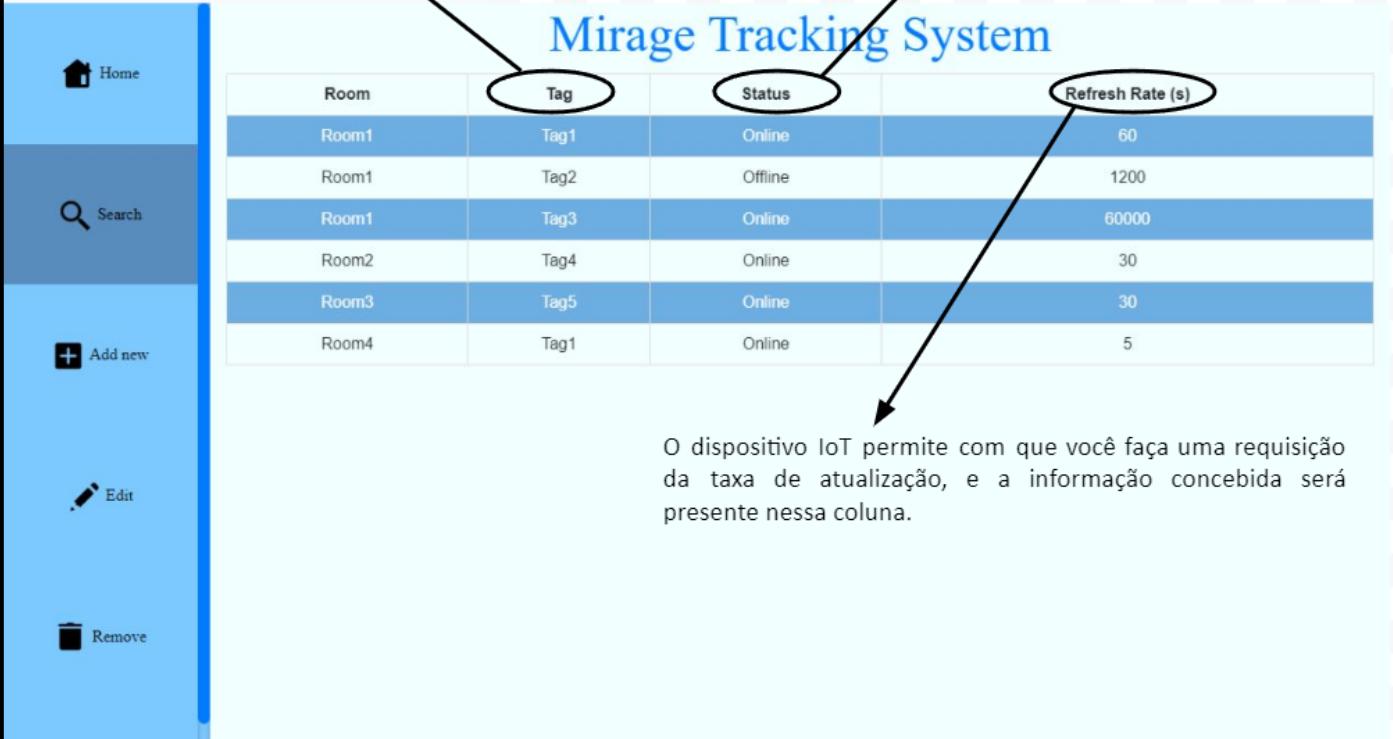



Figura 16: operação da página 'add new' do software

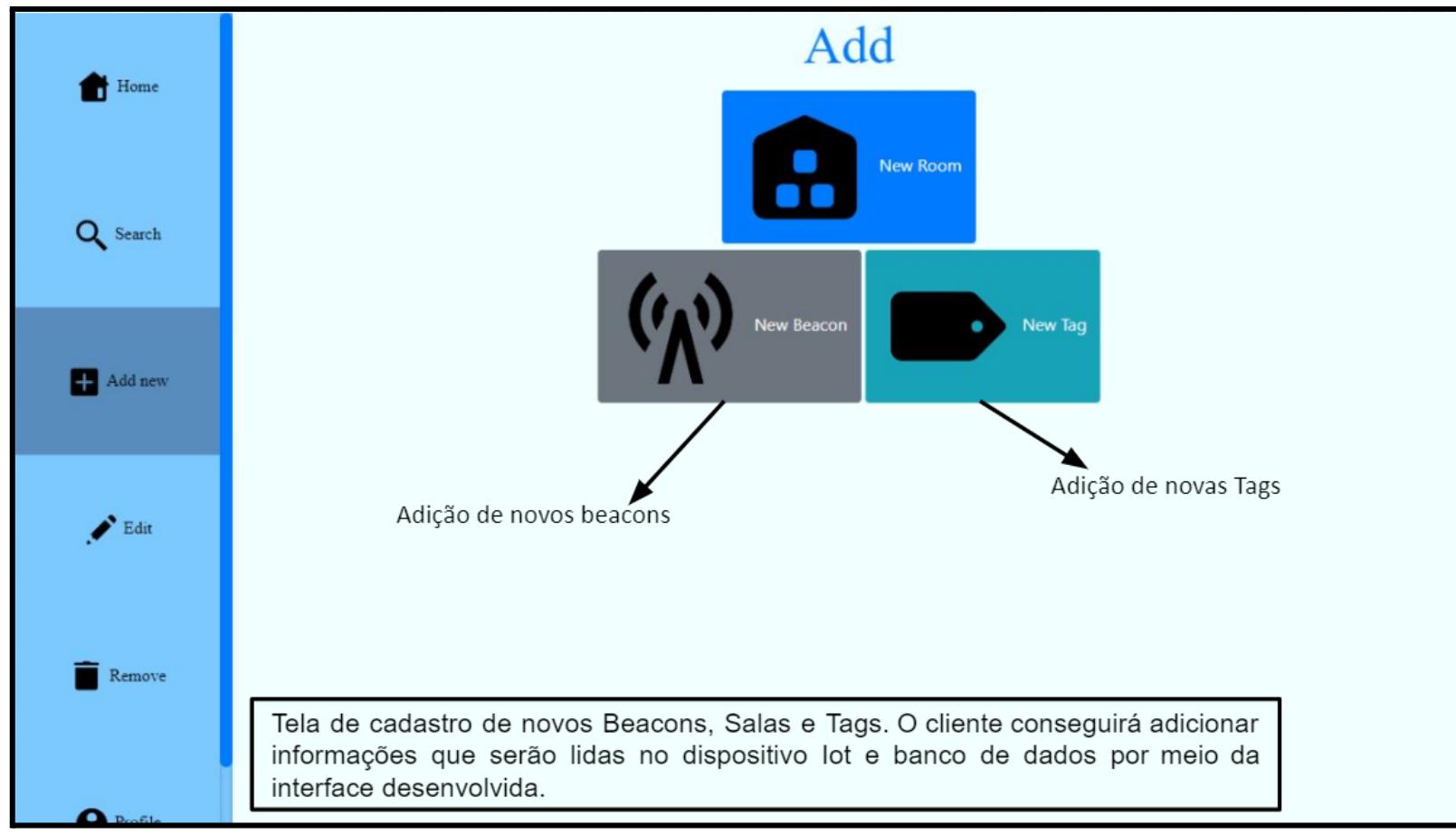


Figura 17: operação da página 'edit' do software

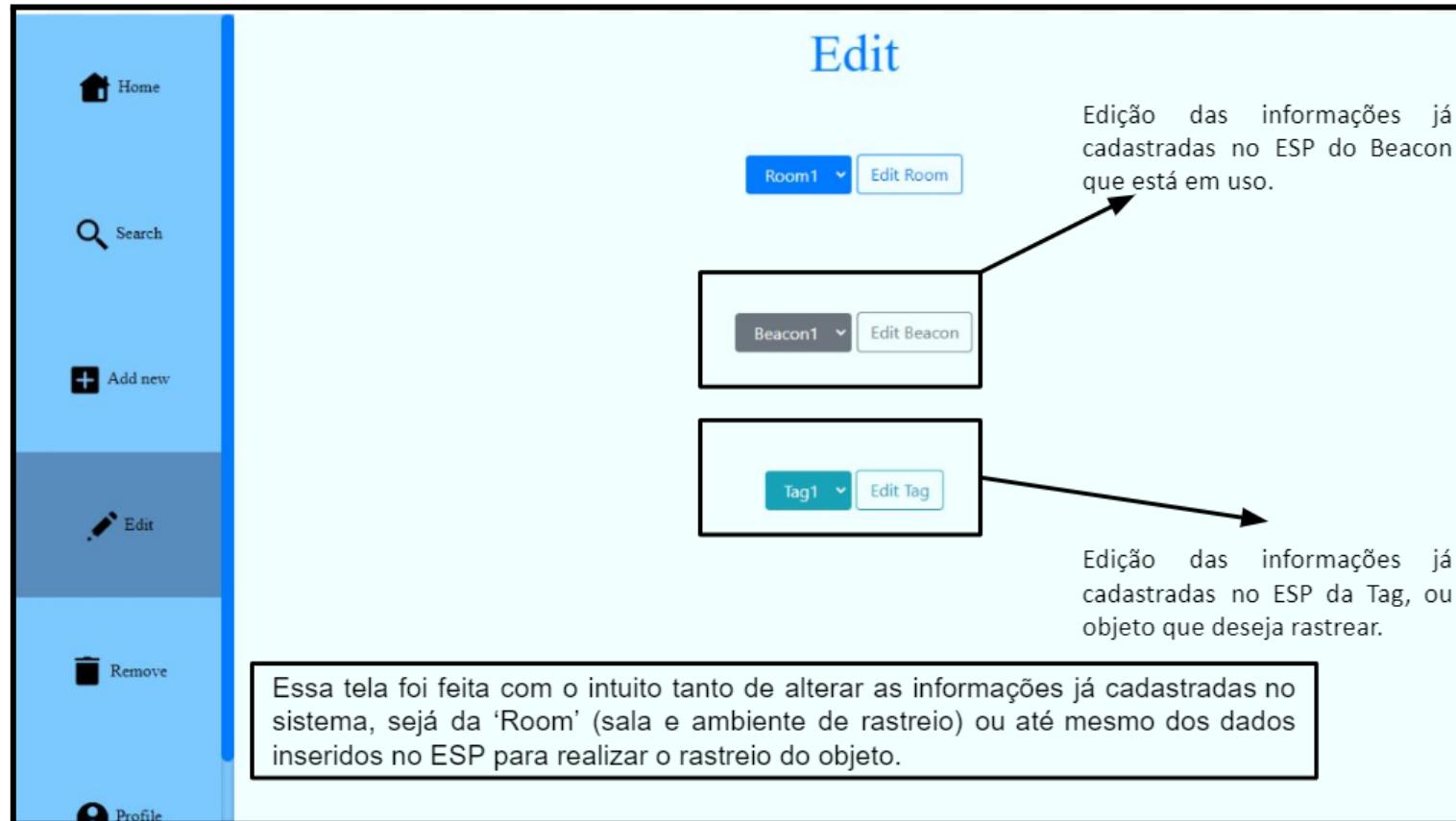
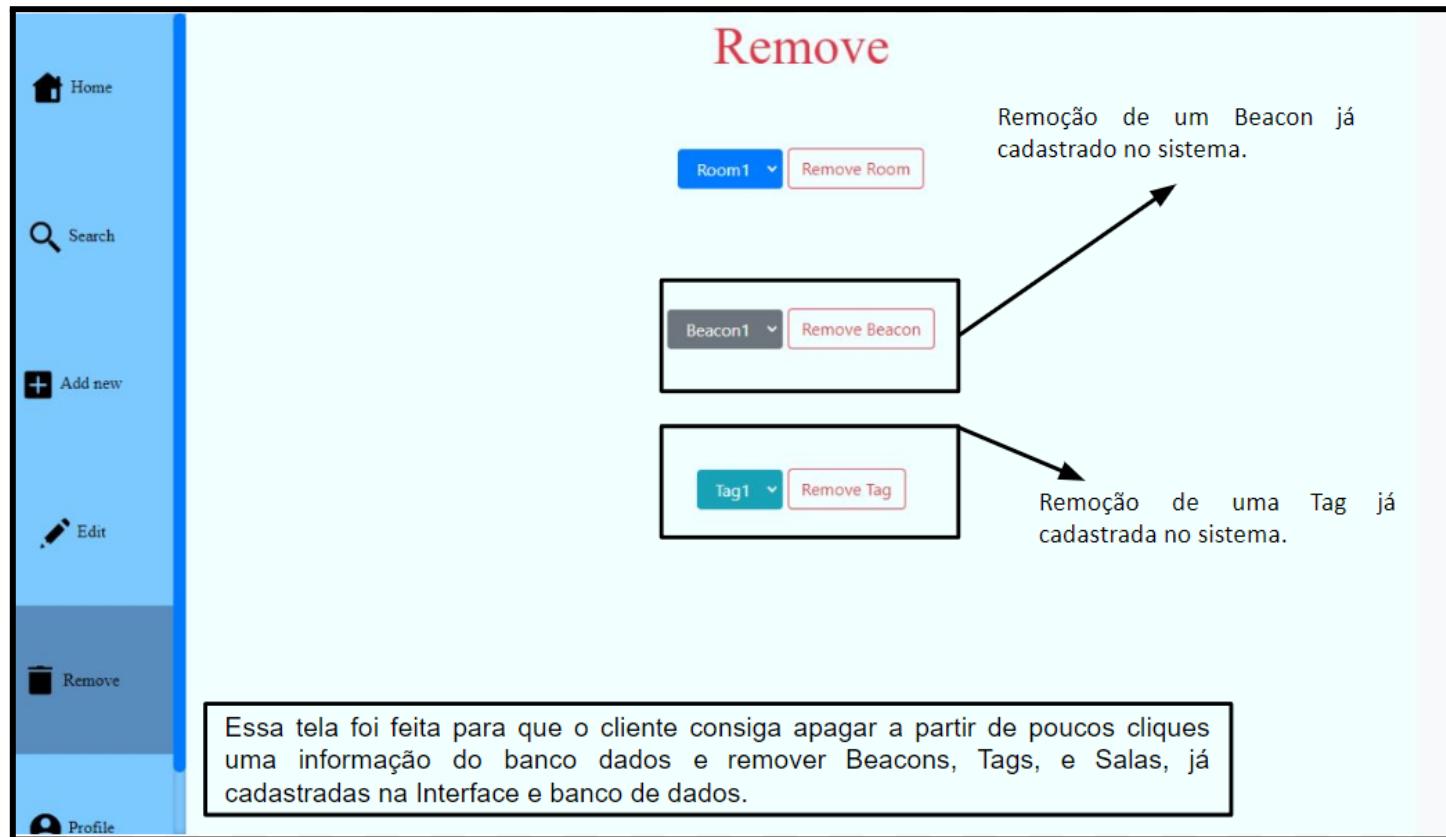


Figura 18: operação da página 'remove' do software



## 7. Troubleshooting

#	Problema	Possível solução
1	interferência por objetos	<ul style="list-style-type: none"> <li>- Remover objetos que possam estar alterando o alcance do dispositivo</li> <li>- Reiniciar o modem, caso o problema esteja na rede Wi-fi</li> </ul>
2	Distância dos Beacons e ESP no ambiente Indoor	<ul style="list-style-type: none"> <li>- Recalcular a triangulação para evitar erros</li> <li>- Certificar de que a distância dos beacons é uma distância adequada e de alcance para o dispositivo desenvolvido.</li> <li>- </li> </ul>
3	Falta de conectividade	<ul style="list-style-type: none"> <li>- Se conectar a uma rede que o “ssid” não tenha espaços; renomear a rede.</li> </ul>
4	Falha no sistema operacional do esp32	trocar o equipamento (esp 32), por um que esteja com o funcionamento correto

## 8. Créditos

O grupo Mirage é composto pelos seguintes integrantes: Caio Martins; Gabriel Pascoli; Giovanna Thomé; Israel Carvalho; Kathlyn Diwan e Pablo Ruan.

Os desenvolvedores do grupo ficaram responsáveis pela montagem e escrita deste Manual, pela construção do bloco central, desenvolvimento do Front e Back-End, entre outras funções.

Copyright © 2022 Instituto de Tecnologia e Liderança (INTELI).

