

# Desenvolvimento de um sistema multiusuário para edifícios inteligentes

ALUNO: Victor Hugo de Almeida Alicino  
ORIENTADOR: Antonio Marcos Massao Hachisuca

# CONTEÚDO

1. Introdução ao Tema
  2. Objetivos
  3. O Sistema
  4. Testes e Resultados
  5. Conclusão
-

# INTRODUÇÃO

## Edifício Inteligente

- O que é um Edifício Inteligente?
- Existem no mínimo 30 definições diferentes para esse conceito.<sup>[1]</sup>

# INTRODUÇÃO

## Edifício Inteligente

Um edifício inteligente é aquele edifício capaz de:

- Monitorar o ambiente em que se encontra;
- Controlar o ambiente em que se encontra.

# INTRODUÇÃO

## Edifício Inteligente

**Monitora** o ambiente em que se encontra através de **sensores**, e  
**Controla** o ambiente em que se encontra através de **atuadores**.

---

# INTRODUÇÃO

## Edifício Inteligente

Exemplo de Sensor/Atuador de mercado:

Página inicial / HVAC and Building Controls / Building Controls - Automated Logic / Automated Logic OF253A-E2 OptiFlex Advanced Application Equipment Controller [New] [2]

### Automated Logic OF253A-E2 OptiFlex Advanced Application Equipment Controller [New]

Automated Logic

**\$1,112.64**

[Write a Review](#)

SKU: B2E-TRO-19047  
Condition: New  
Availability: In Stock - Ships Immediately  
Shipping: Calculated at Checkout

Current Stock: 6



Quantity:

[ADD TO QUOTE](#)

[ADD TO CART](#)

[f](#) [✉](#) [🖨](#) [🐦](#) [in](#) [p](#)

[Recommend 0](#)



Overview [Other Details](#)

FONTE: [www.b2esurplus.com](http://www.b2esurplus.com)

# INTRODUÇÃO

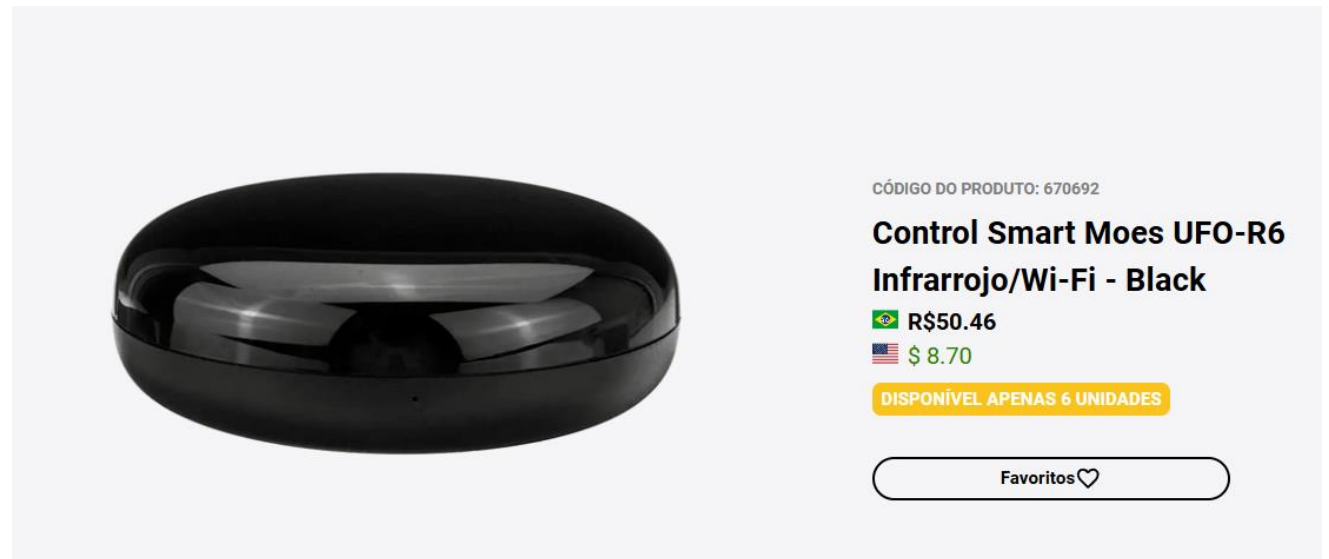
## Edifício Inteligente

- Dispositivos de Internet das Coisas para Casas Inteligentes são 60% de todos os dispositivos conectados ao redor do mundo; <sup>[3]</sup>
  - Sensores estão ficando mais acessível economicamente nas últimas duas décadas. <sup>[4]</sup>
-

# INTRODUÇÃO

## Edifício Inteligente

Exemplo de Sensor/Atuador IoT para Casas Inteligentes:



[5]

FONTE: [www.mobilezone.com.br](http://www.mobilezone.com.br)



# OBJETIVOS

## Objetivo Geral

Este trabalho visa desenvolver um sistema básico que demonstre a viabilidade da operação simultânea de múltiplos usuários em um edifício inteligente. A proposta é criar uma solução funcional que atenda a múltiplos usuários em um ambiente compartilhado.

# OBJETIVOS

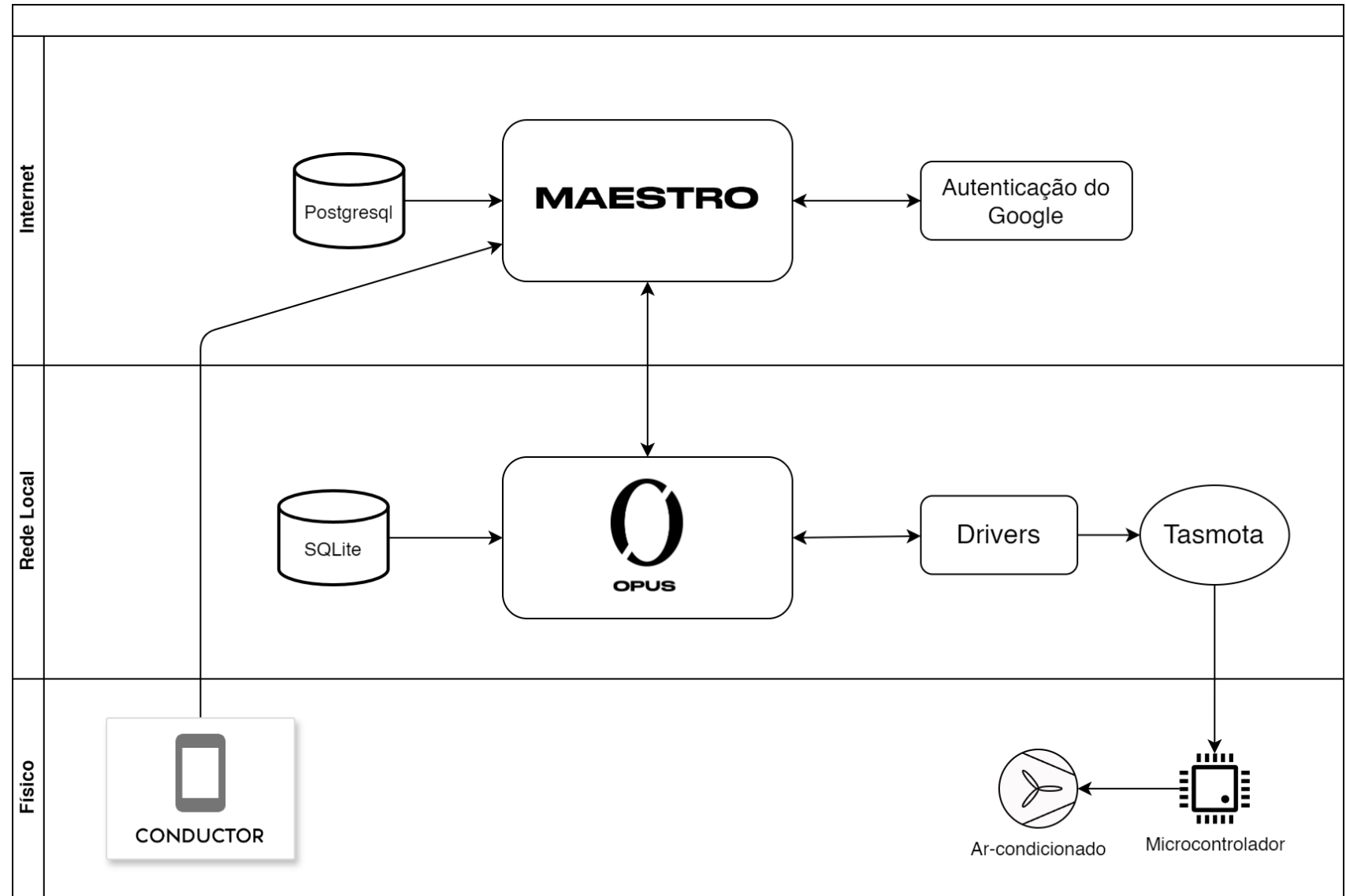
## Objetivos Específicos

Para alcançar o objetivo do trabalho, é proposto uma aplicação simplificada que permita entender a viabilidade do uso de um sistema de edifício inteligente por diversos usuários, focando no controle das unidades de climatização.

- Definir requisitos para um sistema de gerenciamento de um edifício inteligente;
  - Propor e implementar um sistema de gerenciamento para edifícios inteligentes;
  - Tornar este sistema acessível a múltiplos usuários como visitantes;
-

# O SISTEMA

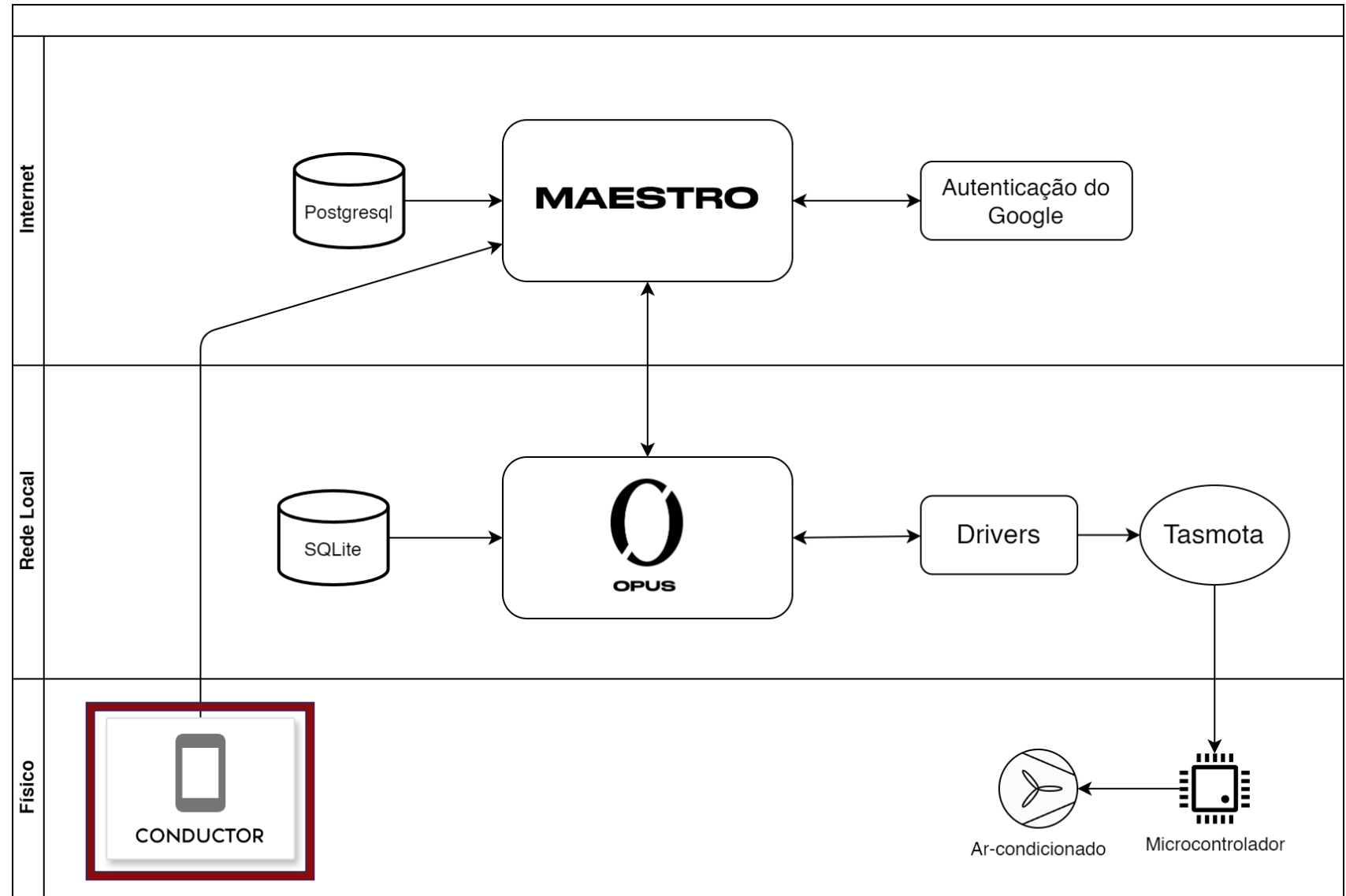
- Arquitetura do sistema Opus



FONTE: Autor

# O SISTEMA

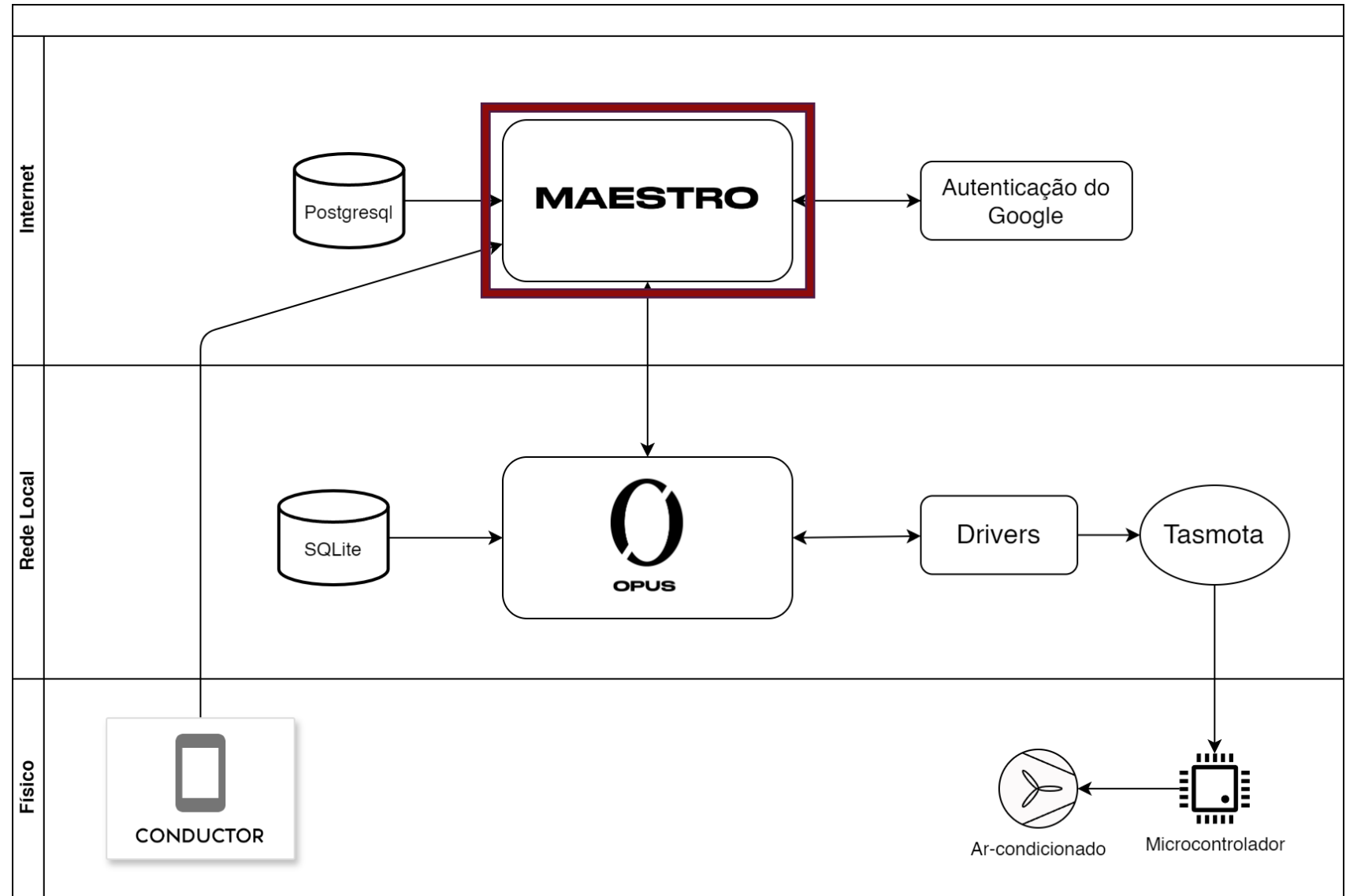
- Arquitetura do sistema Opus



FONTE: Autor

# O SISTEMA

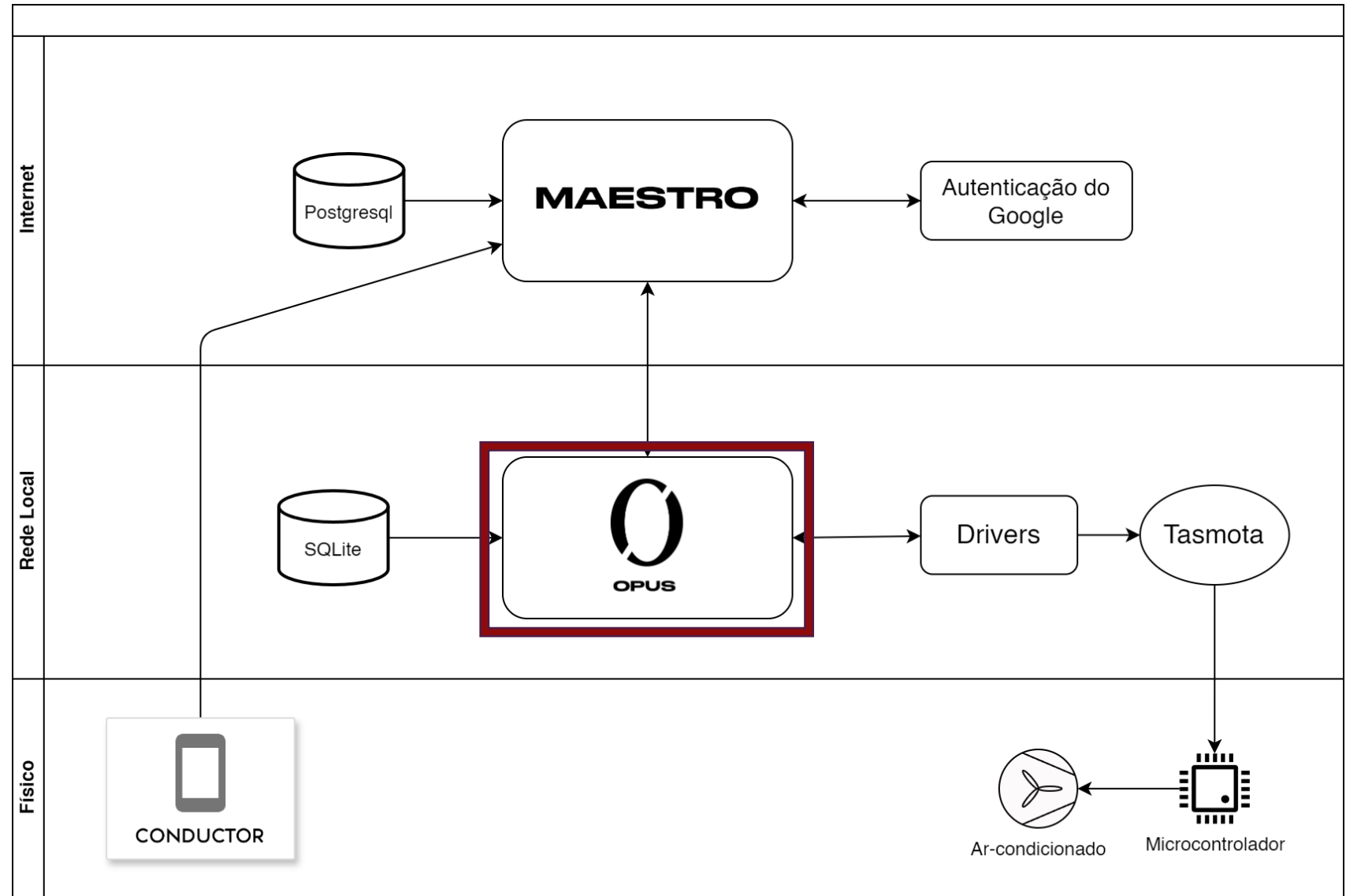
- Arquitetura do sistema Opus



FONTE: Autor

# O SISTEMA

- Arquitetura do sistema Opus

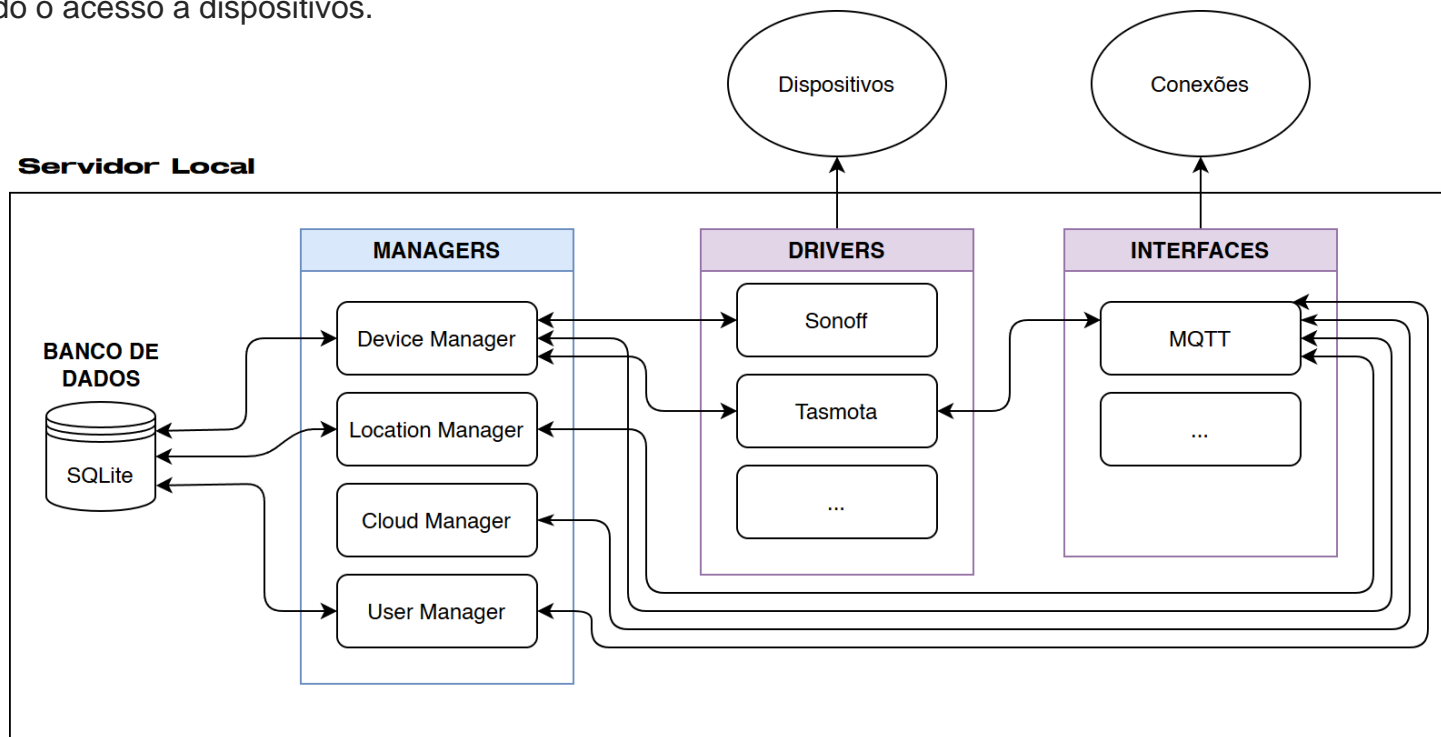


FONTE: Autor

# O SISTEMA

- **O Servidor Local**

Escrito em Python, o **Servidor Local** é responsável por gerenciar todo o acesso a dispositivos.



## LEGENDA:

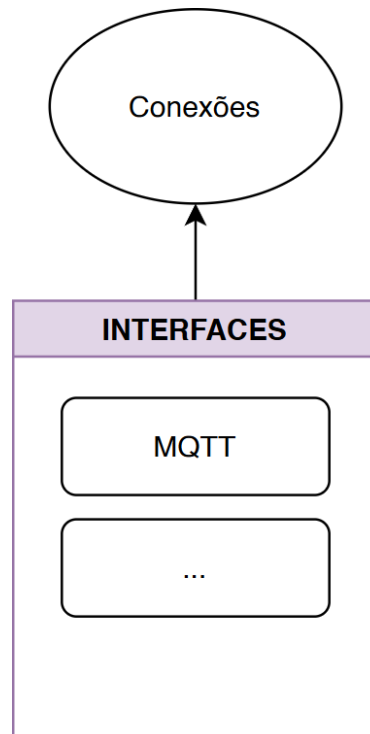
Podem ser adicionados novos módulos

Módulos são fixos e não devem ser alterados

FONTE: Autor

# O SISTEMA

- O Servidor Local - Interfaces



FONTE: Autor

```
class Interface1():
    """Classe exemplo de Interface"""
    def __init__(self):
        """
        O construtor da classe não
        deve iniciar a comunicação, apenas
        definir parâmetros fixos para a comunicação
        como por exemplo versão do protocolo
        """
        self.client = foreign_protocol.Client(
            protocol_version="1.0"
        )
        log.info("Interface1 intialized")

    def begin(self, config:dict) -> bool:
        """Método de início da comunicação"""
        """
        O método begin deve iniciar a comunicação
        utilizando os parâmetros passados pelo dicionário
        de configuração
        """
        try:
            self.client.connect(
                address=config["address"],
                port=config["port"]
            )
            log.info("Connection established")
            return True
        except SpecificException as e:
            log.error("Connection failed")
            return False

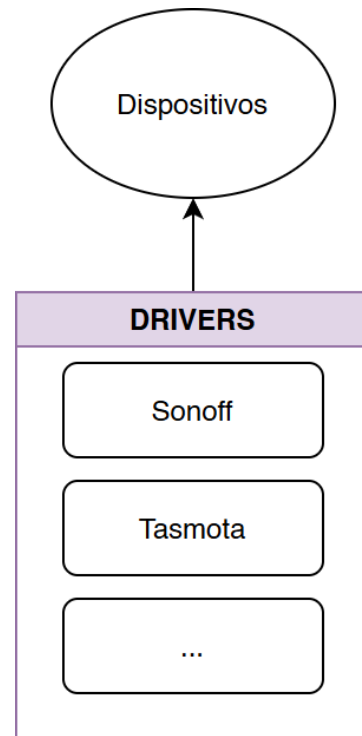
# A função initialize será a primeira a ser chamada
# no módulo
def initialize() -> Interface1:
    """Método de inicialização da interface"""
    return Interface1()
```

FONTE: Autor



# O SISTEMA

- O Servidor Local - Drivers



FONTE: Autor

```
from core.devices._generic import OpusDevice
from core.device_manager import DeviceManager
from core.devices.light import OpusLight

from typing import override
from uuid import UUID, uuid1

# O Opus irá preencher o valor da chave com a Interface
# solicitada neste dicionário
interfaces = dict(luz, str) = {
    "Interface1": None,
    "Interface2": None
}

class DriverDevice(OpusDevice):
    """Dispositivo genérico exemplo para o Driver!"""
    def __init__(self):
        super().__init__()
        self.specific_address = str = ""
        self.id = uuid1()

class DriverLight(OpusLight):
    """Dispositivo do tipo luz para o Driver!"""
    def __init__(self, name: str, driver_device: DriverDevice):
        super().__init__(
            name=name,
            uuid=uuid1(str(driver_device.id)),
            driver="Driver!"
        )
        self.specific_address = str = driver_device.specific_address

    @override
    def on(self):
        """
        Reimplementação do método Ligar, de acordo
        com os requisitos do Driver
        """
        interfaces["Interface1"].enviar_mensagem(
            {"address": self.specific_address, "ação": "ligar_luz"}
        )

class Driver:
    """Driver de exemplo para o Opus"""
    def __init__(self, device_manager: DeviceManager):
        self.opus_device_manager = DeviceManager + device_manager

    def translate_message_to_devices(self,
                                     message: str) -> list[DriverDevice]:
        """
        Método exemplo para receber a mensagem da Interface
        e retornar quais novos dispositivos foram descobertos
        """
        # Implementar a lógica de acordo com o protocolo
        # de comunicação entre o Driver e a Interface
        pass

    def find_devices(self):
        """
        Drivers podem encontrar e reportar novos
        dispositivos para o Opus
        """
        global interfaces
        # Exemplo de uso de uma interface em um Driver
        interfaces["Interface2"].enviar_mensagem("POC:USAR_DISPOSITIVOS")
        msg = interfaces["Interface2"].receber_mensagem()
        new_devices: list[DriverDevice] = self.translate_message_to_devices(msg)
        # Adiciona dispositivos descobertos a lista de dispositivos disponíveis
        for device in new_devices:
            self.opus_device_manager.new_device(device)

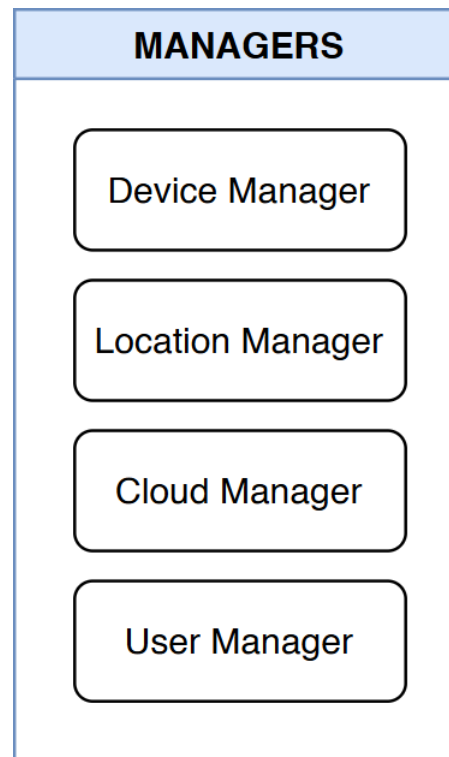
    def new_light(name: str, base_device: DriverDevice) -> DriverLight:
        """
        Ao registrar um dispositivo, o Opus irá chamar a função
        de acordo com o Driver que o dispositivo pertence
        """
        return DriverLight(name, base_device)

    def start(dirs: dict,
              config: dict,
              drivers: dict,
              interfaces: dict,
              managers: dict) -> None:
        """Função chamada pelo Opus para iniciar o Driver"""
        finder = Driver(managers["devices"])
```

FONTE: Autor

# O SISTEMA

- **O Servidor Local - Gerenciadores**

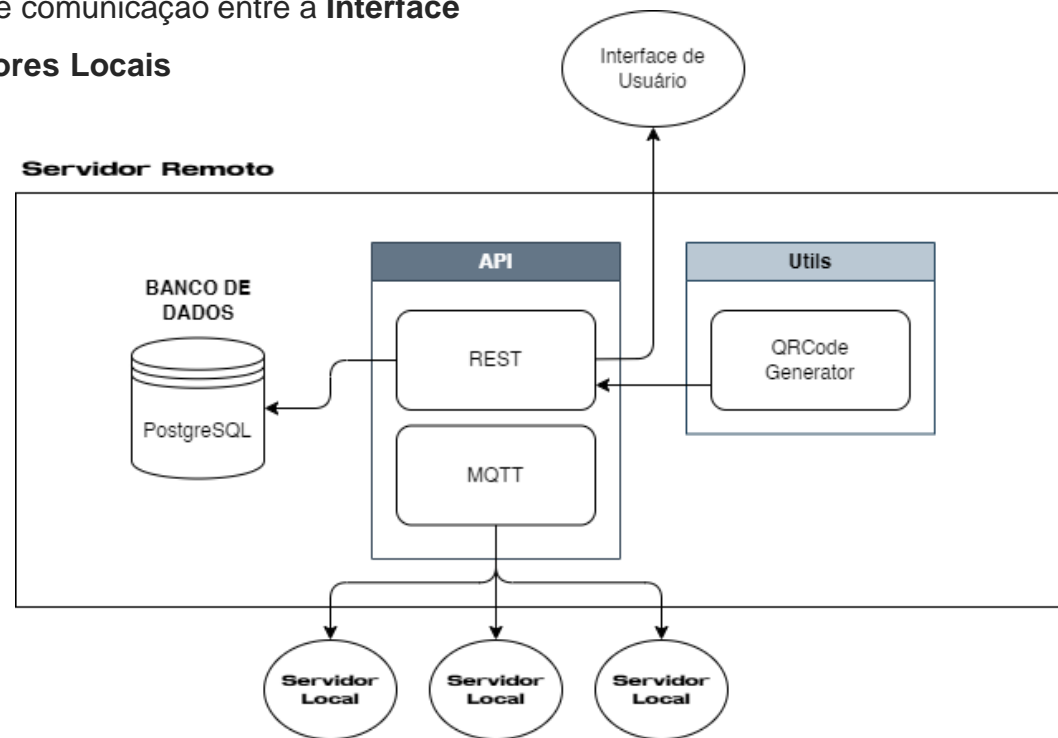


FONTE: Autor

# O SISTEMA

- **O Servidor Remoto**

Escrito em Python, o **Servidor Remoto** é responsável pela autenticação de usuários e comunicação entre a **Interface de Usuários** e os **Servidores Locais**



FONTE: Autor

# O SISTEMA

- **O Servidor Remoto**

- Afim de garantir a autenticidade dos usuários, o Servidor Remoto utiliza o Google como mecanismo de Login.

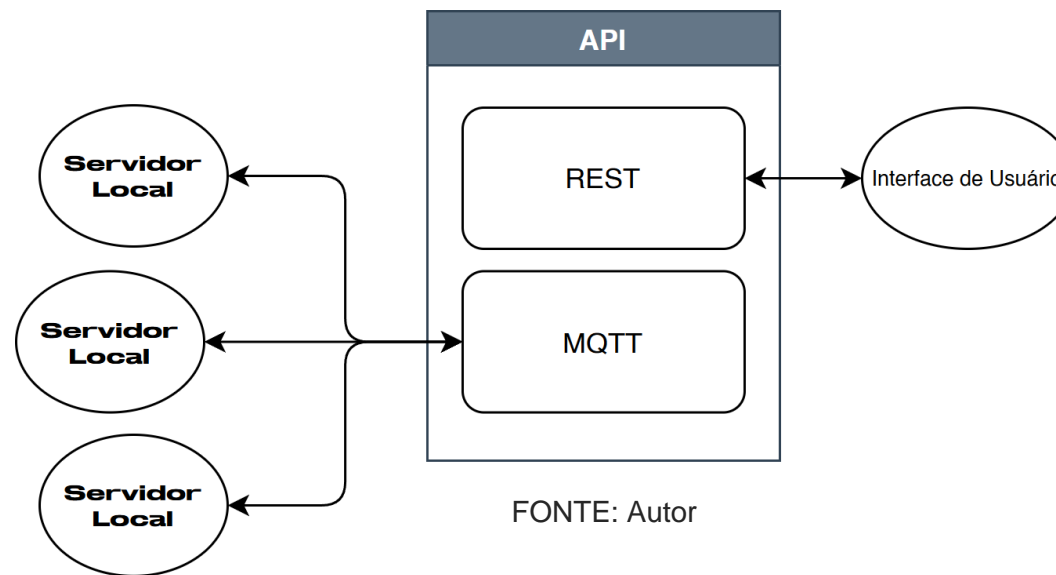
```
{  
  "email": "victor.alicino@gmail.com",  
  "familyName": "Alicino",  
  "givenName": "Victor",  
  "id": "00000000000000000000",  
  "name": "Victor Alicino",  
  "photo": "https://url-exemplo.com/photo.jpg"  
}
```

FONTE: Autor

# O SISTEMA

- **O Servidor Remoto**

- Possui duas API, uma em REST utilizando FastAPI para comunicação da Interface de Usuário e uma em MQTT para comunicação com os Servidores Locais

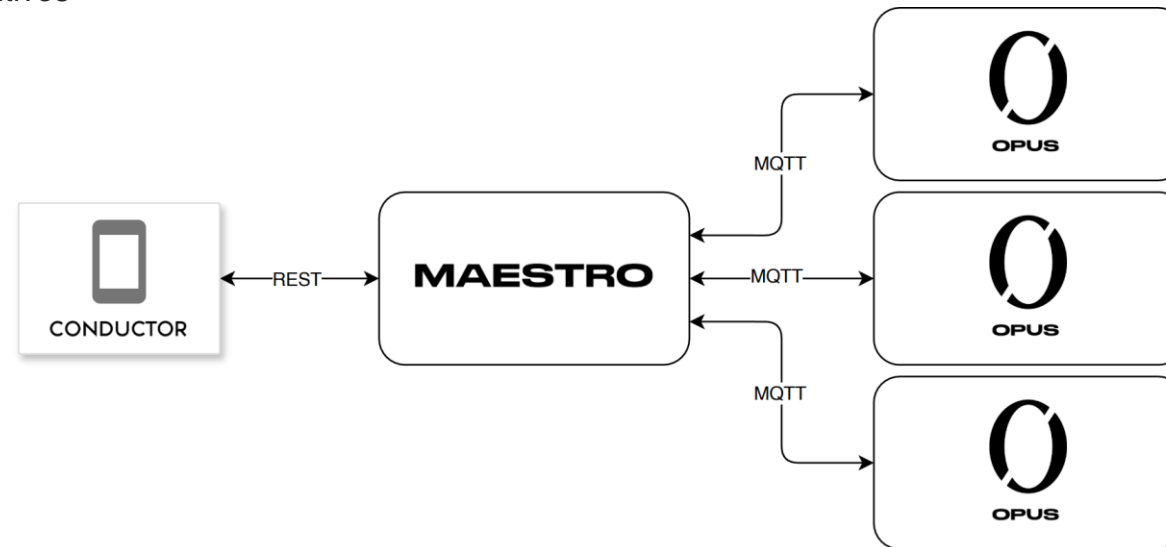


# O SISTEMA

- **Interface de Usuário**

A Interface de Usuário foi escrita em JavaScript com o framework React e suas três principais funções são:

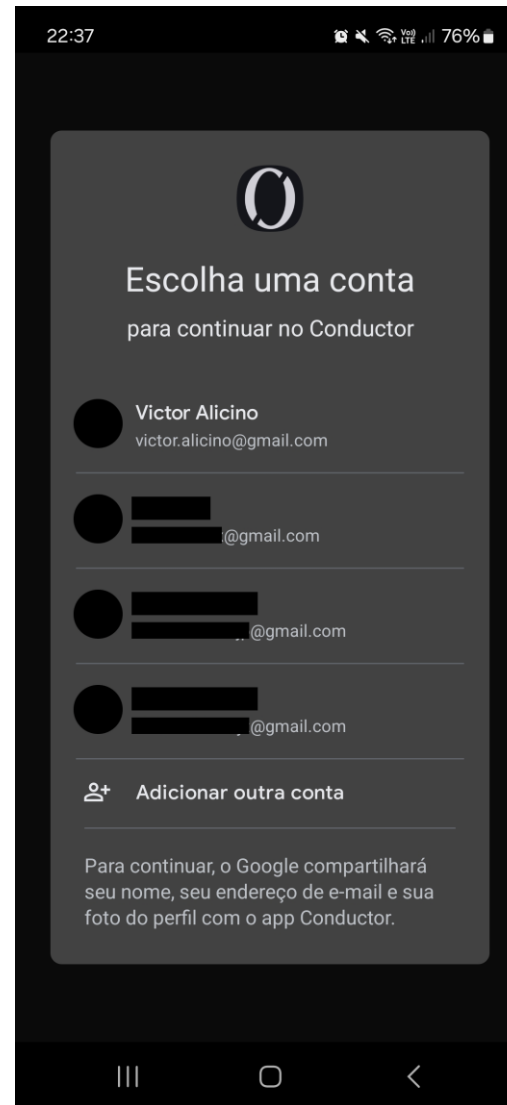
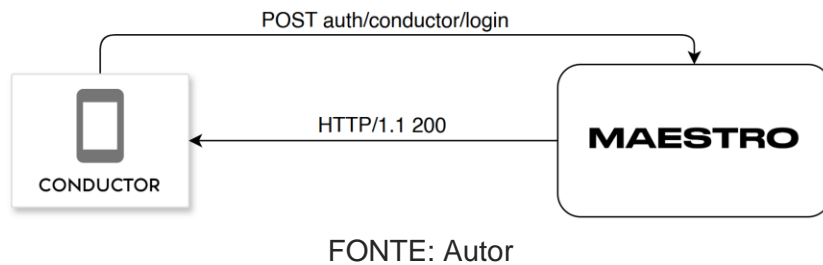
- Capturar as informações do usuário para autenticação
- Exibir os dispositivos disponíveis para o usuário
- Permitir o controle dos dispositivos



FONTE: Autor

# O SISTEMA

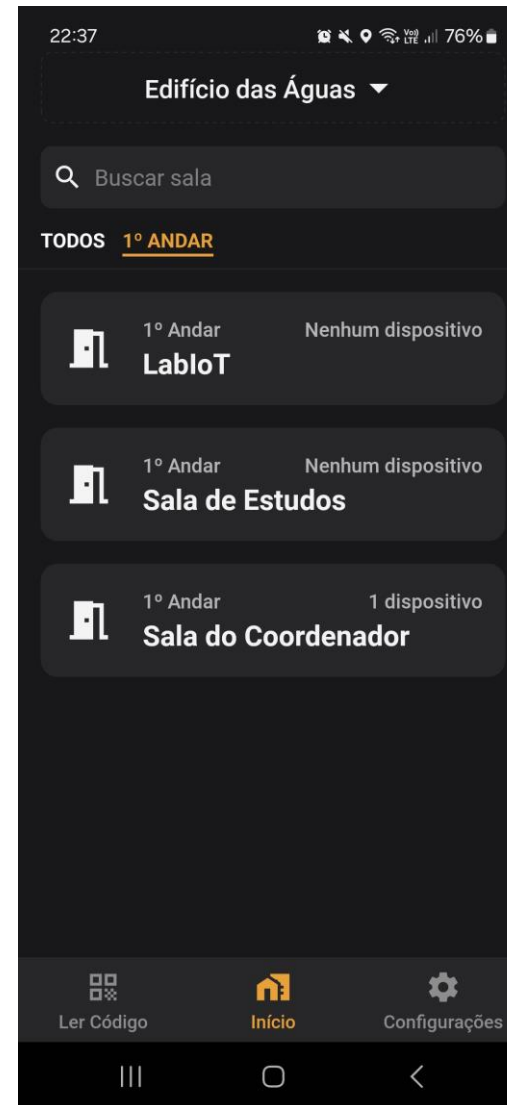
- **Interface de Usuário**
  - Capturar as informações do usuário para autenticação



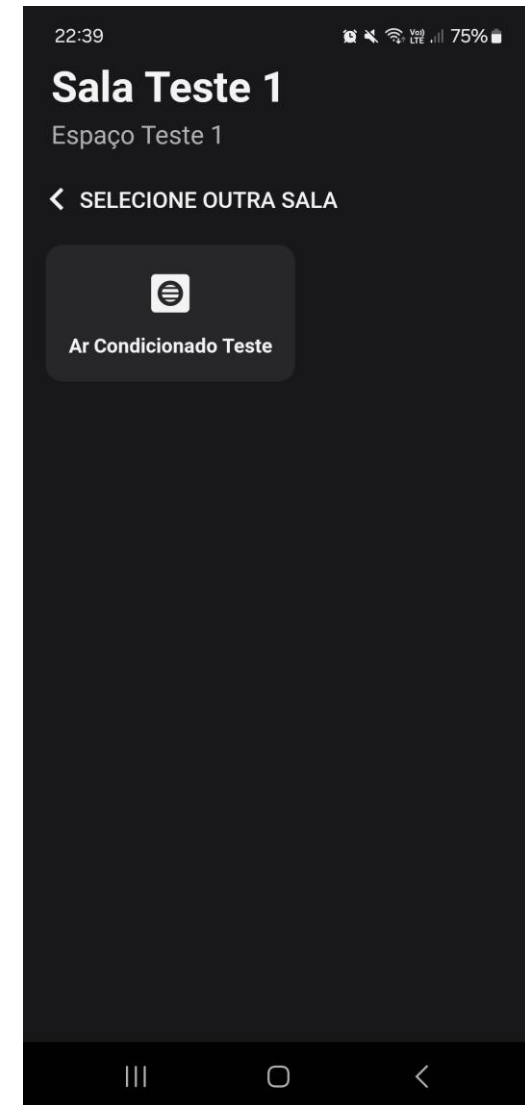
FONTE: Autor

# O SISTEMA

- **Interface de Usuário**
  - Exibir os dispositivos disponíveis para o usuário



FONTE: Autor

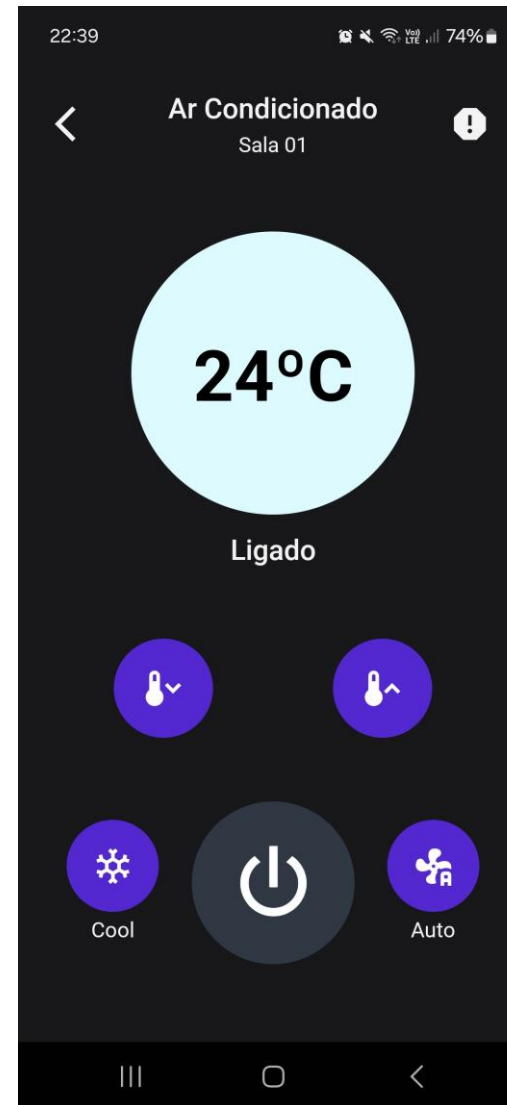


FONTE: Autor



# O SISTEMA

- **Interface de Usuário**
  - Permitir o controle dos dispositivos



FONTE: Autor

# O SISTEMA

## Sistema Multiusuário

# O SISTEMA

## Sistema Multiusuário

- Usuários podem requisitar o controle de um dispositivo através de um QR Code

# O SISTEMA

## SISTEMA MULTIUSUÁRIO

- Usuários podem requisitar o controle de um dispositivo através de um QR Code
  - Ler o QR Code concede acesso por tempo limitado
  - Apenas um visitante pode controlar o dispositivo por vez

# O SISTEMA

## Sistema Multiusuário

- Usuários podem requisitar o controle de um dispositivo através de um QR Code



FONTE: Autor

# TESTES E RESULTADOS

Para cumprir com os objetivos específicos:

- Propor e implementar um sistema de gerenciamento para edifícios inteligentes;
- Tornar este sistema acessível a múltiplos usuários como visitantes;

foram feitos **3 testes**, com **3 objetivos**:

1. Verificar se o sistema é capaz de manter mais de um usuário;
  2. Verificar se o sistema é capaz de gerar a chave (QR Code) para um dispositivo acessível para múltiplos usuários;
  3. Verificar se o sistema é capaz de permitir o controle do dispositivo para múltiplos usuários seguindo os limites especificados
-

# TESTES E RESULTADOS

## MATERIAIS

- Servidor Local:

Para executar o software Opus foi utilizado um computador de mesa com o sistema operacional Windows 10;

Para executar o broker MQTT local, foi utilizado um Raspberry Pi 3 executando o software Mosquitto;

Para o Dispositivo foi utilizado um emissor de infravermelho inteligente modificado para aceitar o firmware Tasmota.

# TESTES E RESULTADOS

## MATERIAIS

- Servidor Remoto:

Para executar o software Maestro foi necessário um servidor na nuvem, foi utilizado uma máquina virtual na Oracle Cloud Infrastructure (OCI) com o “shape” VM.Standard.E2.1.Micro e sistema operacional Ubuntu 22.04.5 LTS

Para executar o broker remoto foi adicionado na máquina o broker Mosquitto;



# TESTES E RESULTADOS

## MATERIAIS

- Interface de Usuário:

Para interface foi utilizado 3 smartphones com uma conta Google diferente em cada, com software Android em todos.

# TESTES E RESULTADOS

## TESTE 1

Verificar se o sistema é capaz de manter mais de um usuário;

# TESTES E RESULTADOS

## TESTE 1

Verificar se o sistema é capaz de manter mais de um usuário;

```
DEBUG: Victor Alicino has logged in via Conductor
INFO: 189.46.241.65:0 - "POST /auth/conductor/login HTTP/1.1" 200
INFO: 189.46.241.65:0 - "GET /users/opus_server/dump HTTP/1.1" 200
DEBUG: Meyre Alicino has logged in via Conductor
INFO: 189.46.241.65:0 - "POST /auth/conductor/login HTTP/1.1" 200
INFO: 189.46.241.65:0 - "GET /users/opus_server/dump HTTP/1.1" 200
DEBUG: Sergio Dias Alicino has logged in via Conductor
INFO: 189.46.241.65:0 - "POST /auth/conductor/login HTTP/1.1" 200
INFO: 189.46.241.65:0 - "GET /users/opus_server/dump HTTP/1.1" 200
```

FONTE: Autor

# TESTES E RESULTADOS

## TESTE 1

Verificar se o sistema é capaz de manter mais de um usuário;

```
DEBUG: Victor Alicino has logged in via Conductor
INFO: 189.46.241.65:0 - "POST /auth/conductor/login HTTP/1.1" 200
INFO: 189.46.241.65:0 - "GET /users/opus_server/dump HTTP/1.1" 200
DEBUG: Meyre Alicino has logged in via Conductor
INFO: 189.46.241.65:0 - "POST /auth/conductor/login HTTP/1.1" 200
INFO: 189.46.241.65:0 - "GET /users/opus_server/dump HTTP/1.1" 200
DEBUG: Sergio Dias Alicino has logged in via Conductor
INFO: 189.46.241.65:0 - "POST /auth/conductor/login HTTP/1.1" 200
INFO: 189.46.241.65:0 - "GET /users/opus_server/dump HTTP/1.1" 200
```

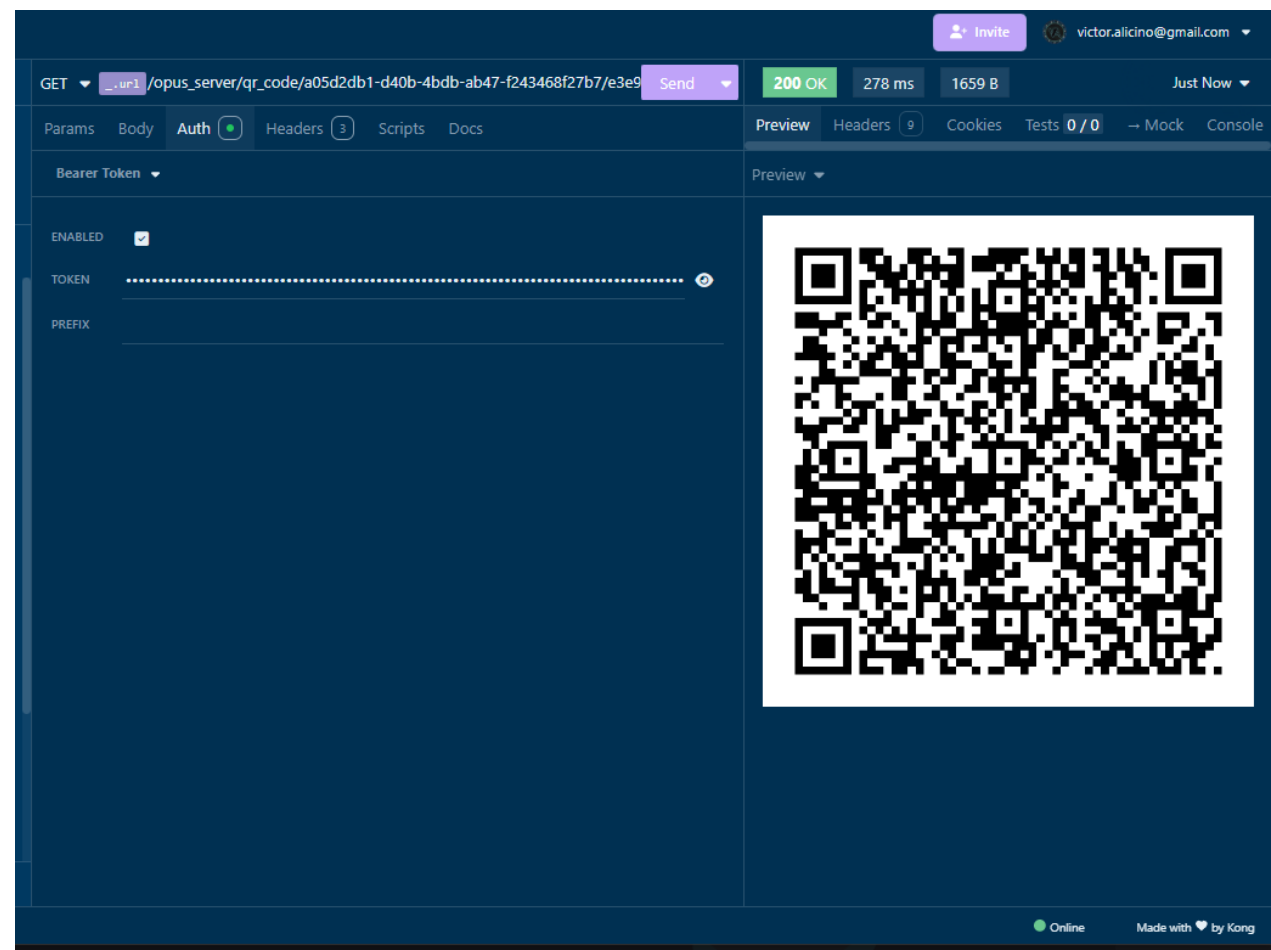
FONTE: Autor

## Sucesso

# TESTES E RESULTADOS

## TESTE 2

Verificar se o sistema é capaz de gerar a chave (QR Code) para um dispositivo acessível para múltiplos usuários;



FONTE: Autor

# TESTES E RESULTADOS

## TESTE 2

Verificar se o sistema é capaz de gerar a chave (QR Code) para um dispositivo acessível para múltiplos usuários;

```
{
  "server_id": "a05d2db1-d40b-4bdb-ab47-f243468f27b7",
  "grant_until": "2025-03-17 06:45:05",
  "device": {
    "device_name": "Ar Condicionado Teste",
    "device_pk": "e3e9906f-fcb0-11ef-8ded-001a7dda710a",
    "device_type": "HVAC"
  },
  "state": {
    "power_state": "Off",
    "vendor": "TCL112AC",
    "model": null,
    "mode": "Cool",
    "fan_speed": "Auto",
    "swing_vertical": null,
    "swing_horizontal": null,
    "is_celsius": null,
    "temperature": 24,
    "quiet": null,
    "turbo": null,
    "economy": null,
    "light": null,
    "filter": null,
    "clean": null,
    "beep": null,
    "sleep": null,
    "state_mode": null
  }
}
```

FONTE: Autor

# TESTES E RESULTADOS

## TESTE 2

Verificar se o sistema é capaz de gerar a chave (QR Code) para um dispositivo acessível para múltiplos usuários;

**Sucesso**

```
{
  "server_id": "a05d2db1-d40b-4bdb-ab47-f243468f27b7",
  "grant_until": "2025-03-17 06:45:05",
  "device": {
    "device_name": "Ar Condicionado Teste",
    "device_pk": "e3e9906f-fcb0-11ef-8ded-001a7dda710a",
    "device_type": "HVAC"
  },
  "state": {
    "power_state": "Off",
    "vendor": "TCL112AC",
    "model": null,
    "mode": "Cool",
    "fan_speed": "Auto",
    "swing_vertical": null,
    "swing_horizontal": null,
    "is_celsius": null,
    "temperature": 24,
    "quiet": null,
    "turbo": null,
    "economy": null,
    "light": null,
    "filter": null,
    "clean": null,
    "beep": null,
    "sleep": null,
    "state_mode": null
  }
}
```

FONTE: Autor

# TESTES E RESULTADOS

## TESTE 3

Verificar se o sistema é capaz de permitir o controle do dispositivo para múltiplos usuários seguindo os limites especificados

- Ler o QR Code concede acesso por tempo limitado
- Apenas um visitante pode controlar o dispositivo por vez



# TESTES E RESULTADOS

## TESTE 3

Este teste possui um roteiro:

1. Usuário 1 lê o QR Code gerado no teste 2.
  2. Usuário 1 controla o dispositivo.
  3. Usuário 2 lê o QR Code.
  4. Usuário 2 controla o dispositivo.
  5. Usuário 3 lê o QR Code.
  6. Usuário 2 tenta controlar o dispositivo novamente.
  7. Usuário 3 controla o dispositivo.
-

# TESTES E RESULTADOS

## TESTE 3

Usuário 1 lê o QR Code gerado no teste 2.

```
Mar 17 06:15:05 maestro api-alicino[1798403]: [2025-03-17 06:15:05] DEBUG: User 42afc86c-a126-4c7f-87b5-95875df7166d has been granted GUEST access to device e3e9906f-fcb0-11ef-8ded-001a7dda710a on server a05d2db1-d40b-4bdb-ab47-f243468f27b7 until 2025-03-17 06:45:05
Mar 17 06:15:05 maestro api-alicino[1798403]: [2025-03-17 06:15:05] INFO: 189.46.241.65:0 - "GET /opus_server/guest_access/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzZXJ2ZXJfaWQiOiJhMDVhMmRiMS1kNDBiLTRiZGItYWl0Ny1mMjQzNDY4ZjI3VjciLCJkZXZpY2VfaWQiOiJLM2U5OTA2Zi1mY2IwLTExZWYtOGRLZC0wMDFhN2RkYTcxMGEifQ.TmXgA6H7rrVV3PIim_FJezCzjtCF4jFpKa-b2uY8Tpg HTTP/1.1" 200
Mar 17 06:15:05 maestro api-alicino[1798403]: [2025-03-17 06:15:05] INFO: 189.46.241.65:0 - "GET /opus_server/a05d2db1-d40b-4bdb-ab47-f243468f27b7/devices/e3e9906f-fcb0-11ef-8ded-001a7dda710a HTTP/1.1" 200
Mar 17 06:15:07 maestro api-alicino[1798403]: [2025-03-17 06:15:07] INFO: 189.46.241.65:0 - "PUT /opus_server/a05d2db1-d40b-4bdb-ab47-f243468f27b7/devices/e3e9906f-fcb0-11ef-8ded-001a7dda710a/set_state HTTP/1.1" 200
Mar 17 06:15:09 maestro api-alicino[1798403]: [2025-03-17 06:15:09] INFO: 189.46.241.65:0 - "PUT /opus_server/a05d2db1-d40b-4bdb-ab47-f243468f27b7/devices/e3e9906f-fcb0-11ef-8ded-001a7dda710a/set_state HTTP/1.1" 200
Mar 17 06:15:11 maestro api-alicino[1798403]: [2025-03-17 06:15:11] INFO: 189.46.241.65:0 - "PUT /opus_server/a05d2db1-d40b-4bdb-ab47-f243468f27b7/devices/e3e9906f-fcb0-11ef-8ded-001a7dda710a/set_state HTTP/1.1" 200
```

FONTE: Autor

# TESTES E RESULTADOS

## TESTE 3

Usuário 1 controla o dispositivo.

```
[2025-03-17 06:15:00][805] INFO: User Victor Alicino [Administrator] is accessing device Ar Condicionado Teste
[2025-03-17 06:15:00][805] DEBUG: State of e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:15:00][806] DEBUG: Name: Ar Condicionado Teste
[2025-03-17 06:15:00][807] DEBUG:   | Power State: Off
[2025-03-17 06:15:00][808] DEBUG:   | Mode: Cool
[2025-03-17 06:15:00][808] DEBUG:   | Fan Speed: Auto
[2025-03-17 06:15:00][809] DEBUG:   | Temperature: 24.0
[2025-03-17 06:15:03][532] DEBUG: MQTT Message Received: opus-server-5be2/devices/set_state/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:15:03][540] INFO: User Victor Alicino [Administrator] is accessing device Ar Condicionado Teste
[2025-03-17 06:15:03][541] DEBUG: State of e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:15:03][542] DEBUG: Name: Ar Condicionado Teste
[2025-03-17 06:15:03][544] DEBUG:   | Power State: On
[2025-03-17 06:15:03][545] DEBUG:   | Mode: Cool
[2025-03-17 06:15:03][546] DEBUG:   | Fan Speed: Auto
[2025-03-17 06:15:03][547] DEBUG:   | Temperature: 24.0
[2025-03-17 06:15:05][266] DEBUG: MQTT Message Received: opus-server-5be2/devices/set_state/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:15:05][271] INFO: User Victor Alicino [Administrator] is accessing device Ar Condicionado Teste
[2025-03-17 06:15:05][272] DEBUG: State of e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:15:05][273] DEBUG: Name: Ar Condicionado Teste
[2025-03-17 06:15:05][274] DEBUG:   | Power State: Off
[2025-03-17 06:15:05][274] DEBUG:   | Mode: Cool
[2025-03-17 06:15:05][275] DEBUG:   | Fan Speed: Auto
[2025-03-17 06:15:05][276] DEBUG:   | Temperature: 24.0
```

FONTE: Autor

# TESTES E RESULTADOS

# TESTE 3

## Usuário 2 lê o QR Code

```
Mar 17 06:22:16 maestro api-alicino[1798403]: [2025-03-17 06:22:16] DEBUG: User 14673966-9950-477d-bc87-be1a1ecbfbfb has been granted GUEST access to device e3e9906f-fcb0-11ef-8ded-001a7dda710a on server a05d2db1-d40b-4b4b-f243468f27b7 until 2025-03-17 06:52:16
Mar 17 06:22:16 maestro api-alicino[1798403]: [2025-03-17 06:22:16] INFO: 177.173.221.161:0 - "GET /opus_server/guest_access/eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzZXZlZXJfaWQiOiJhMDVhMmRlMS1kNDBiLTZGItYWI0Ny1mMjQzNDY4ZjI3YjciLCJkZXZpY2VfaWQiOiJlMjU5OTAzZi1mY2IwLTExZWY0ZGRlZC0wMDFhMjRkY2YtcXMGElFQ.TmKgAGH7rrVV3PIIm_FJezCzjtCF4jFpKa-b2uVBTpg HTTP/1.1" 200
Mar 17 06:22:16 maestro api-alicino[1798403]: [2025-03-17 06:22:16] INFO: 177.173.221.161:0 - "GET /opus_server/a05d2db1-d40b-4b4b-ab47-f243468f27b7/devices/e3e9906f-fcb0-11ef-8ded-001a7dda710a HTTP/1.1" 200
Mar 17 06:22:20 maestro api-alicino[1798403]: [2025-03-17 06:22:20] INFO: 177.173.221.161:0 - "PUT /opus_server/a05d2db1-d40b-4b4b-ab47-f243468f27b7/devices/e3e9906f-fcb0-11ef-8ded-001a7dda710a/set_state HTTP/1.1" 200
Mar 17 06:22:21 maestro api-alicino[1798403]: [2025-03-17 06:22:21] INFO: 177.173.221.161:0 - "PUT /opus_server/a05d2db1-d40b-4b4b-ab47-f243468f27b7/devices/e3e9906f-fcb0-11ef-8ded-001a7dda710a/set_state HTTP/1.1" 200
```

FONTE: Autor

# TESTES E RESULTADOS

## TESTE 3

Usuário 2 controla o dispositivo

```
[2025-03-17 06:22:09][703] DEBUG: MQTT Message Received: opus-server-5be2/devices/get/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:09][796] DEBUG: MQTT Message Received: opus-server-5be2/devices/get_state/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:09][904] DEBUG: MQTT Message Received: opus-server-5be2/devices/get_type/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:10][48 ] DEBUG: MQTT Message Received: opus-server-5be2/devices/get_state/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:14][205] DEBUG: MQTT Message Received: opus-server-5be2/devices/set_state/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:14][217] INFO: User Sergio [Guest] is accessing device Ar Condicionado Teste
[2025-03-17 06:22:14][219] DEBUG: State of e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:14][220] DEBUG: Name: Ar Condicionado Teste
[2025-03-17 06:22:14][221] DEBUG:   └─ Power State: On
[2025-03-17 06:22:14][222] DEBUG:   └─ Mode: Cool
[2025-03-17 06:22:14][224] DEBUG:   └─ Fan Speed: Auto
[2025-03-17 06:22:14][225] DEBUG:   └─ Temperature: 24.0
[2025-03-17 06:22:15][296] DEBUG: MQTT Message Received: opus-server-5be2/devices/set_state/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:15][304] INFO: User Sergio [Guest] is accessing device Ar Condicionado Teste
[2025-03-17 06:22:15][306] DEBUG: State of e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:22:15][307] DEBUG: Name: Ar Condicionado Teste
[2025-03-17 06:22:15][308] DEBUG:   └─ Power State: On
[2025-03-17 06:22:15][309] DEBUG:   └─ Mode: Cool
[2025-03-17 06:22:15][310] DEBUG:   └─ Fan Speed: Auto
[2025-03-17 06:22:15][311] DEBUG:   └─ Temperature: 25.0
```

FONTE: Autor

# TESTES E RESULTADOS

## TESTE 3

Usuário 3 lê o QR Code

Usuário 2 tenta controlar o dispositivo novamente



FONTE: Autor

# TESTES E RESULTADOS

## TESTE 3

Usuário 3 controla o dispositivo

```
[2025-03-17 06:28:07][306] INFO: User Meyre [Guest] is accessing device Ar Condicionado Teste
[2025-03-17 06:28:07][307] DEBUG: State of e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:28:07][309] DEBUG: Name: Ar Condicionado Teste
[2025-03-17 06:28:07][310] DEBUG:   | Power State: On
[2025-03-17 06:28:07][311] DEBUG:   | Mode: Cool
[2025-03-17 06:28:07][312] DEBUG:   | Fan Speed: Auto
[2025-03-17 06:28:07][313] DEBUG:   | Temperature: 26.0
[2025-03-17 06:28:08][605] DEBUG: MQTT Message Received: opus-server-5be2/devices/set_state/e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:28:08][614] INFO: User Meyre [Guest] is accessing device Ar Condicionado Teste
[2025-03-17 06:28:08][615] DEBUG: State of e3e9906f-fcb0-11ef-8ded-001a7dda710a
[2025-03-17 06:28:08][617] DEBUG: Name: Ar Condicionado Teste
[2025-03-17 06:28:08][618] DEBUG:   | Power State: Off
[2025-03-17 06:28:08][619] DEBUG:   | Mode: Cool
[2025-03-17 06:28:08][620] DEBUG:   | Fan Speed: Auto
[2025-03-17 06:28:08][621] DEBUG:   | Temperature: 26.0
```

FONTE: Autor

## Sucesso

# CONCLUSÃO

---



# TRABALHOS FUTUROS

- Integração com mais dispositivos;
  - Integração com mais protocolos;
  - Implementação de uma hierarquia a definir pelo administrador do dado Servidor Local
-

---

# REFERÊNCIAS

- [1] WIGGINTON, M.; HARRIS, J. Intelligent skins. [S.l.]: Butterworth-Heinemann, 2002. 176 p. ISBN 0750648473
- [2] Automated Logic OF253A-E2 OptiFlex Advanced Application Equipment Controller. Disponível em: <<https://www.b2esurplus.com/automated-logic-of253a-e2-optiflex-advanced-application-equipment-controller-new/?srsltid=AfmBOoqV3V7dGW3HHcneGtbFiGmoV0h5WwJ5XY2wbBvHix5PBSvNB9Tn>>. Acesso em 27 de março de 2025.
- [3] STATISTA. Number of Internet of Things (IoT) connections worldwide from 2022 to 2023, with forecasts from 2024 to 2033. 2024. Disponível em: (<<https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/>>). Acesso em 27 de março de 2025.
- [4] MICROSOFT. 2019 manufacturing trends report. 2019. Disponível em: (<<https://info.microsoft.com/rs/157-GQE-382/images/EN-US-CNTNT-Report-2019-Manufacturing-Trends.pdf>>). Acesso em 21 de março de 2025.
- [5] Control Smart Moes UFO-R6 Infrarrojo/Wi-Fi - Black. Disponível em: <<https://www.mobilezone.com.br/product/670692/>>. Acesso em 27 de março de 2025.

STUBBINGS, M. Intelligent buildings. Contracts, 1986.

---

**OBRIGADO!**