INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE RONDÔNIA CAMPUS ARIQUEMES
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

JOSIAS DUTRA DA SILVA

SISTEMA DE AUTOMAÇÃO DE CLIMATIZAÇÃO BASEADO EM IOT PARA EFICIÊNCIA ENERGÉTICA EM AMBIENTES ACADÊMICOS DO INSTITUTO FEDERAL DE RONDÔNIA – CAMPUS ARIQUEMES

Ariquemes
2025

JOSIAS DUTRA DA SILVA

SISTEMA DE AUTOMAÇÃO DE CLIMATIZAÇÃO BASEADO EM IOT PARA EFICIÊNCIA ENERGÉTICA EM AMBIENTES ACADÊMICOS DO INSTITUTO FEDERAL DE RONDÔNIA – CAMPUS ARIQUEMES

Relatório técnico entregue como Trabalho Conclusão de Curso ao Instituto Federal de Educação, Ciência e Tecnologia de Rondônia – (IFRO), Campus Ariquemes, como requisito parcial para obtenção do grau de tecnólogo junto ao curso de Análise e Desenvolvimento de Sistemas.
Orientador: Andrey Alencar Quadros

Ariquemes
2025

Ficha catalográfica elaborada pelo Sistema Gerador de Ficha Catalográfica do IFRO.

S586s

Silva, Josias Dutra da.

Sistema de automação de climatização baseado em IOT para eficiência energética em ambientes acadêmicos do Instituto Federal de Rondônia - *Campus* Ariquemes / Josias Dutra da Silva. - Ariquemes, 2025.

39 f. : il.

Orientador(a): Prof. Me. Andrey Alencar Quadros.

Trabalho de Conclusão de Curso (Superior de Tecnologia em Análise e Desenvolvimento de Sistemas) – Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO, Ariquemes, 2025.

1. internet das Coisas. 2. automação. 3. eficiência energética. 4. IFRO. 5. climatização. I. Quadros, Andrey Alencar (orient.). II. Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO. III. Título.

Bibliotecário(a) Responsável: Renilce Silva Morais, CRB-11/906

JOSIAS DUTRA DA SILVA

SISTEMA DE AUTOMAÇÃO DE CLIMATIZAÇÃO BASEADO EM IOT PARA EFICIÊNCIA ENERGÉTICA EM AMBIENTES ACADÊMICOS DO INSTITUTO FEDERAL DE RONDÔNIA – CAMPUS ARIQUEMES

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Analista e Desenvolvedor de Sistemas" e aprovado em sua forma final pelo Curso de Tecnologia em Análise e Desenvolvimento de Sistemas.

Ariquemes, 05 de Dezembro de 2025.

Banca Examinadora:

gov.br
Documento assinado digitalmente
ANDREY ALENCAR QUADROS
Data: 19/12/2025 13:40:02-0300
Verifique em https://validar.iti.gov.br

Prof. Mestre Andrey Alencar Quadros
(orientador)
Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO - Campus Ariquemes)

gov.br
Documento assinado digitalmente
LUCIANO TOPOLNIAK
Data: 19/12/2025 15:08:54-0300
Verifique em https://validar.iti.gov.br

Prof. Mestre Luciano Topolniak
Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO - Campus Ariquemes)

gov.br
Documento assinado digitalmente
MARCOS ALVES FAINO
Data: 19/12/2025 18:27:36-0300
Verifique em https://validar.iti.gov.br

Prof. Especialista Marcos Alves Faino
Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO - Campus Ariquemes)

AGRADECIMENTOS

Em primeiro lugar, agradeço a Deus, fonte de toda sabedoria e força, por me conceder saúde, perseverança e direção durante toda esta jornada acadêmica. Sem a Sua presença constante, nada disso seria possível.

Aos meus pais, Maria e Rildo, minha eterna gratidão apoio e incentivo incondicional. Foram eles que me ensinaram o valor do esforço, da fé e da honestidade, e me sustentaram nos momentos mais desafiadores.

Ao meu orientador, professor Andrey Alencar Quadros, agradeço pela paciência, dedicação e por compartilhar seu conhecimento com clareza e comprometimento, contribuindo de forma essencial para o desenvolvimento deste trabalho.

Estendo meus agradecimentos a todos os professores do curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Rondônia – Campus Ariquemes, pelos ensinamentos, pela inspiração e pelo compromisso em formar profissionais preparados para transformar a sociedade por meio da tecnologia.

A todos que, de alguma forma, contribuíram para esta conquista, deixo aqui o meu sincero muito obrigado.

Não fui eu que ordenei a você? Seja forte e corajoso! Não se apavore nem desanime, pois o Senhor, o seu Deus, estará com você por onde você andar.(Josué 1:9)

RESUMO

O presente trabalho apresenta o desenvolvimento de um sistema de automação de climatização baseado em Internet das Coisas (IoT) voltado à eficiência energética em ambientes acadêmicos. O sistema, denominado IFRO ClimaTech, foi desenvolvido utilizando Python, FastAPI, SQLite e integração com a Tuya Cloud API, permitindo o controle remoto, agendamento e monitoramento inteligente de aparelhos de ar-condicionado. A solução visa reduzir o desperdício de energia e promover a sustentabilidade institucional no Instituto Federal de Rondônia – Campus Ariquemes.

Palavras-chave: Internet das Coisas; Automação; Eficiência Energética; IFRO; Climatização.

ABSTRACT

This work presents the development of an air conditioning automation system based on the Internet of Things (IoT) aimed at energy efficiency in academic environments. The system, named IFRO ClimaTech, was developed using Python, FastAPI, SQLite, and Tuya Cloud API integration, allowing remote control, scheduling, and intelligent monitoring of air conditioners. The solution aims to reduce energy waste and promote institutional sustainability at the Federal Institute of Rondônia – Campus Ariquemes.

Keywords: Internet of Things; Automation; Energy Efficiency; IFRO; Air Conditioning.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
CSS	Cascading Style Sheets
DOM	Document Object Model
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IFRO	Instituto Federal de Rondônia
IoT	Internet das Coisas
JSON	JavaScript Object Notation
SPA	Single Page Application

SUMÁRIO

1 INTRODUÇÃO

Nas últimas décadas, o avanço das tecnologias digitais e o surgimento da IoT (Internet das Coisas) transformaram de forma significativa a maneira como os ambientes físicos são administrados e controlados. A integração entre dispositivos inteligentes, redes de comunicação e sistemas de gestão remota permitiu o desenvolvimento de ambientes automatizados, capazes de otimizar recursos, reduzir custos e proporcionar maior conforto e sustentabilidade.

Segundo (Ideali, 2021), "a internet das coisas é um conceito que se aplica à interconexão digital de objetos cotidianos com a internet, algo novo e que está tomando um lugar no mundo digital, que, podemos dizer, está derrubando antigos paradigmas." Em outras palavras, trata-se da possibilidade de conectar objetos e equipamentos à rede mundial de computadores, de modo que possam enviar e receber informações, sendo "uma rede de objetos e dispositivos físicos capaz de coletar e transmitir dados, tudo controlado por microcontroladores e microprocessadores a distância"

No contexto das instituições de ensino, a eficiência energética tornou-se um tema de grande relevância, uma vez que o consumo de energia elétrica tem aumentado de forma expressiva com a ampliação do uso de tecnologias em salas de aula e laboratórios. Entre os principais responsáveis por esse consumo estão os sistemas de climatização, que frequentemente permanecem ligados por longos períodos, mesmo em horários de inatividade. No caso do IFRO (Instituto Federal de Rondônia) – Campus Ariquemes, observou-se que os aparelhos de ar-condicionado seguem uma rotina operacional contínua, sendo acionados no início das aulas e desligados apenas no final do expediente, o que ocasiona desperdício de energia, aumento nos custos institucionais e redução da vida útil dos equipamentos.

A aplicação da Internet das Coisas nesse contexto apresenta-se como alternativa eficiente para o monitoramento e controle automatizado dos sistemas de climatização. Por meio da integração entre software, hardware e serviços em nuvem, é possível desenvolver soluções inteligentes e acessíveis, que permitam o controle remoto dos aparelhos, o agendamento automático de funcionamento e a geração de relatórios de uso. Assim, a automação baseada em IoT contribui para a sustentabilidade institucional e para o uso racional da energia elétrica, ao mesmo tempo em que torna os ambientes mais inteligentes e eficientes, corroborando a visão de que "a internet das coisas é a coleção de objetos na internet ou na rede, em que nós confiamos um determinado controle para facilitar nossas vidas" (Ideali, 2021).

1.1 OBJETIVO GERAL

Desenvolver um sistema de automação de climatização baseado em IoT (Internet das Coisas), utilizando o *framework FastAPI*, banco de dados *SQLite* e integração com

a *Tuya Cloud API*, com o objetivo de possibilitar o controle remoto, o agendamento automático e o monitoramento inteligente dos aparelhos de ar-condicionado em ambientes acadêmicos do IFRO (Instituto Federal de Rondônia) – Campus Ariquemes, visando a redução do consumo de energia elétrica e o aumento da eficiência operacional e ambiental da instituição.

1.2 OBJETIVOS ESPECÍFICOS

Para o alcance do objetivo geral proposto, este trabalho possui os seguintes objetivos específicos:

- Implementar o *backend* do sistema utilizando o *framework FastAPI*, em linguagem *Python*, para gerenciar requisições, autenticação e comunicação com a nuvem;
- Integrar o sistema à *Tuya Cloud API*, possibilitando o envio de comandos infravermelhos para controle remoto dos aparelhos de ar-condicionado;
- Desenvolver uma interface web responsiva em HTML (HyperText Markup Language), CSS (Cascading Style Sheets) e *JavaScript*, permitindo o controle, agendamento e visualização de relatórios em tempo real;
- Criar um banco de dados *SQLite* para o armazenamento de informações de dispositivos, *logs* de uso e horários de agendamento;
- Implementar funcionalidades de agendamento automatizado, permitindo o acionamento e desligamento programado dos aparelhos de acordo com a rotina acadêmica;
- Gerar relatórios estatísticos de operação contendo indicadores como tempo médio de funcionamento, temperatura configurada e economia estimada de energia;
- Avaliar a viabilidade técnica e o impacto energético da solução desenvolvida, considerando o consumo elétrico, a sustentabilidade e o desempenho operacional do sistema.

2 METODOLOGIA

2.1 IDENTIFICAÇÃO DO PROBLEMA

Durante a análise das rotinas operacionais do IFRO (Instituto Federal de Rondônia) – Campus Ariquemes, foi identificado um problema recorrente relacionado ao uso inadequado dos aparelhos de ar-condicionado instalados nas salas de aula e nos ambientes administrativos. Observou-se que os equipamentos permanecem ligados por longos períodos, inclusive durante intervalos entre as aulas e após o encerramento das atividades, resultando em desperdício de energia elétrica e aumento dos custos institucionais.

Além do impacto financeiro, esse uso contínuo provoca sobrecarga nos sistemas elétricos e reduz a vida útil dos aparelhos, elevando a necessidade de manutenção e substituição de componentes. A ausência de um controle automatizado centralizado impede que os gestores monitorem o funcionamento dos equipamentos em tempo real, dificultando a adoção de medidas corretivas e o planejamento eficiente do consumo.

Diante dessa situação, torna-se evidente a necessidade de uma solução tecnológica capaz de automatizar o controle dos sistemas de climatização, oferecendo ferramentas de monitoramento, agendamento e desligamento remoto. A proposta é aplicar conceitos de Internet das Coisas (IoT) e automação predial, integrando hardware, software e serviços em nuvem para promover eficiência energética, sustentabilidade e gestão inteligente dos recursos.

2.2 LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos é uma das etapas mais importantes do processo de desenvolvimento de software, pois define claramente o problema a ser solucionado e orienta as decisões técnicas do projeto. Segundo (C. Sbrocco; Macedo, 2012, p. 45),

> "a engenharia de software corresponde à análise, ao projeto, à construção, à verificação e à gestão de elementos técnicos ou sociais. É importante observar que, independente do elemento a ser tratado, não podemos nos esquecer de responder às seguintes questões que devem ser levantadas inicialmente e respondidas:
>
> Qual é o problema que estamos querendo resolver?
>
> Que características do software serão usadas para resolver o problema?
>
> Como o software vai ser construído?
>
> Que abordagem será usada para descobrir erros que foram cometidos no projeto e na construção do software?
>
> Como o software será mantido a longo prazo, quando correções, adaptações e aperfeiçoamentos forem solicitados pelos usuários? "

Essas questões orientam o início de qualquer projeto de desenvolvimento e foram fundamentais para a estruturação do sistema proposto neste trabalho. No caso do Sistema

de Automação de Climatização, buscou-se compreender não apenas o problema energético identificado no IFRO (Instituto Federal de Rondônia) – Campus Ariquemes, mas também os aspectos técnicos e funcionais necessários para solucioná-lo de forma prática, eficiente e sustentável.

Durante o levantamento de requisitos, foram realizadas observações diretas nas salas de aula e nos ambientes administrativos, com o objetivo de identificar os horários de maior utilização, os equipamentos disponíveis e os padrões de uso dos aparelhos de ar-condicionado. Além disso, foram analisadas as limitações de infraestrutura de rede e energia, a fim de assegurar a viabilidade técnica da automação proposta.

A partir dessa análise, definiu-se que o sistema deveria permitir:

- O controle remoto individual ou coletivo dos aparelhos de ar-condicionado;
- A configuração de horários automáticos de funcionamento e desligamento;
- O monitoramento em tempo real do estado de cada dispositivo;
- O armazenamento de registros de uso e geração de relatórios estatísticos;
- A interface web acessível para usuários autorizados e compatível com dispositivos móveis.

Esses requisitos orientaram a concepção da arquitetura do sistema, a escolha das tecnologias empregadas e o fluxo de comunicação entre os módulos, garantindo que o produto final atendesse de forma objetiva à necessidade institucional e aos princípios de eficiência energética.

3 DESENVOLVIMENTO

O desenvolvimento do sistema de automação de climatização foi realizado em etapas progressivas, seguindo as boas práticas da engenharia de software e o princípio da prototipação incremental, que permite a evolução contínua do projeto a partir de versões funcionais testadas e aprimoradas.

A construção do sistema envolveu três frentes principais: *backend*, *frontend* e integração com a nuvem *Tuya Cloud*, cada uma com funções específicas, mas interligadas por uma arquitetura modular. A seguir, são apresentadas as fases do processo de desenvolvimento.

3.1 PLANEJAMENTO E MODELAGEM

Inicialmente, foram definidos os módulos principais e a arquitetura do sistema, com base no levantamento de requisitos e na análise do problema energético identificado no IFRO – Campus Ariquemes.

A estrutura foi concebida em três camadas:

- **Camada de Apresentação (*Frontend*):** responsável pela interação do usuário com o sistema, exibindo os dispositivos cadastrados, seus estados (ligado/desligado), temperatura atual e opções de agendamento;
- **Camada de Lógica (*Backend*):** processa as requisições da interface, realiza a autenticação, gerencia os dispositivos e registra os eventos de uso;
- **Camada de Persistência (Banco de Dados):** armazena as informações de dispositivos, horários programados, histórico de comandos e relatórios.

Durante essa etapa, também foram elaborados o diagrama de arquitetura do sistema, o modelo lógico do banco de dados e o fluxo de comunicação entre os módulos, garantindo coerência técnica e escalabilidade.

3.2 IMPLEMENTAÇÃO DO *BACKEND*

O *backend* foi desenvolvido em Python, utilizando o framework *FastAPI*, escolhido por sua alta performance, suporte assíncrono e facilidade de integração com API (Application Programming Interface) externas. Essa camada é responsável por processar as requisições vindas da interface web, autenticar os usuários, gerenciar os dispositivos de ar-condicionado e registrar os eventos no banco de dados SQLite. Além disso, o *backend* fornece *endpoints RESTful* que permitem o controle e o monitoramento dos equipamentos em tempo real.

Entre as principais funcionalidades implementadas destacam-se:

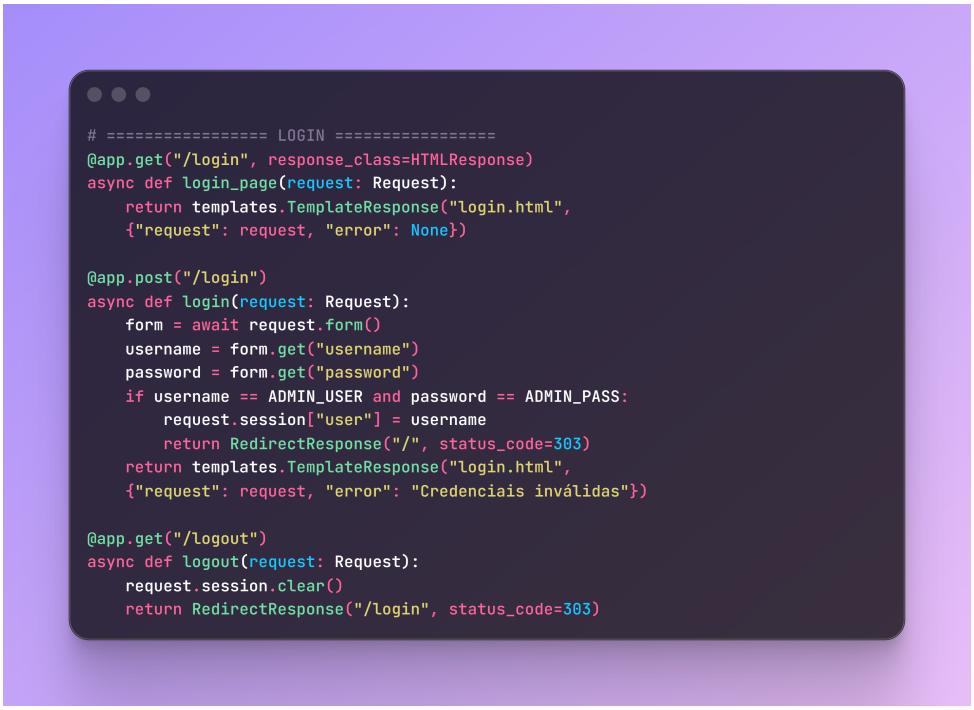
- Criação das rotas de controle (ligar, desligar, ajustar temperatura e consultar status);

- Estabelecimento de autenticação e comunicação segura com a Tuya Cloud API;
- Registro dos logs de operação e armazenamento dos eventos no banco de dados;
- Geração de relatórios e execução de rotinas automatizadas de agendamento.

3.2.1 Rotas de Autenticação e Sessão

O sistema implementa um mecanismo simples de autenticação baseado em sessões, que garante o acesso restrito às páginas e funcionalidades administrativas. As rotas responsáveis pelo processo de login e logout são apresentadas a seguir.

Figura 1 – Implementação do Login.



```python
# ================ LOGIN ================
@app.get("/login", response_class=HTMLResponse)
async def login_page(request: Request):
    return templates.TemplateResponse("login.html",
    {"request": request, "error": None})

@app.post("/login")
async def login(request: Request):
    form = await request.form()
    username = form.get("username")
    password = form.get("password")
    if username == ADMIN_USER and password == ADMIN_PASS:
        request.session["user"] = username
        return RedirectResponse("/", status_code=303)
    return templates.TemplateResponse("login.html",
    {"request": request, "error": "Credenciais inválidas"})

@app.get("/logout")
async def logout(request: Request):
    request.session.clear()
    return RedirectResponse("/login", status_code=303)
```

Fonte: Autor (2025).

A rota `/login` recebe as credenciais do usuário por meio de um formulário e, se válidas, cria uma sessão armazenada na variável `request.session`, permitindo o acesso às rotas internas. A função `must_login()` é utilizada em todas as demais rotas para verificar se o usuário está autenticado. O método `RedirectResponse` garante o redirecionamento seguro entre as páginas e o controle do fluxo de navegação.

3.2.2 Rotas de Páginas e Cadastro de Dispositivos

Após a autenticação, o usuário tem acesso às páginas internas do sistema, como o painel principal, cadastro e edição de dispositivos, relatórios e agendamentos. Essas páginas são renderizadas dinamicamente através de templates HTML (HyperText Markup Language) utilizando o mecanismo `Jinja2`.

O Jinja2 é um mecanismo de templates rápido, expressivo e extensível, amplamente utilizado em aplicações web desenvolvidas em Python. De acordo com a documentação oficial, o Jinja2 permite a inserção de marcadores especiais nos templates, possibilitando a escrita de estruturas de controle e expressões com sintaxe semelhante à linguagem Python. Após a definição do template, os dados são enviados pela aplicação backend para que o mecanismo realize a renderização do documento final de forma dinâmica (Pallets Projects, 2024).

No contexto deste trabalho, o Jinja2 foi utilizado para realizar a integração entre o backend desenvolvido em FastAPI e a interface web do sistema IFRO ClimaTech. Essa abordagem permitiu a geração dinâmica das páginas HTML, possibilitando a exibição em tempo real das informações dos dispositivos cadastrados, estados dos aparelhos de ar-condicionado e dados de controle, sem a necessidade de recarregar a página manualmente.

Figura 2 – Implementação das Rotas de Páginas e Cadastro de Dispositivos.

```python
@app.get("/", response_class=HTMLResponse)
async def index(request: Request):
    if not must_login(request):
        return RedirectResponse("/login", status_code=303)
    devices = fetch_devices()
    return templates.TemplateResponse("index.html",
    {"request": request, "devices": devices})

@app.post("/add")
async def add_device(request: Request, name: str = Form(...),
                     infrared_id: str = Form(...),
                     remote_id: str = Form(...),
                     power: int = Form(...), btu: int = Form(...)):
    if not must_login(request):
        return RedirectResponse("/login", status_code=303)
    conn = sqlite3.connect(DB_PATH)
    cur = conn.cursor()
    cur.execute("INSERT INTO devices (name,infrared_id,
    remote_id,power,btu) VALUES (?,?,?,?,?)",
                (name,infrared_id,remote_id,power,btu))
    conn.commit()
    conn.close()
    return RedirectResponse("/", status_code=303)
```

Fonte: Autor (2025).

A primeira rota (`/`) carrega a página inicial, exibindo todos os dispositivos cadastrados no banco de dados local por meio da função `fetch_devices()`. A segunda rota (`/add`) permite o cadastro de novos aparelhos de ar-condicionado, enviando os dados do formulário HTML (HyperText Markup Language) diretamente para o banco de dados

SQLite. O redirecionamento posterior retorna o usuário à página principal, refletindo as alterações na interface em tempo real.

3.2.3 Rotas de Relatórios e Agendamentos

O sistema também inclui rotas dedicadas à geração de relatórios e à execução de rotinas automáticas de agendamento. Essas rotas acessam diretamente as tabelas `events` e `schedules`, possibilitando o acompanhamento do consumo de energia, horários de funcionamento e eficiência de cada aparelho.

Figura 3 – Implementação das Rotas de Relatórios e Agendamentos.

```python
@app.get("/reports", response_class=HTMLResponse)
async def reports(request: Request, start: str = Query(None),
end: str = Query(None)):
    if not must_login(request):
        return RedirectResponse("/login", status_code=303)

    conn = sqlite3.connect(DB_PATH)
    cur = conn.cursor()
    cur.execute("SELECT COUNT(*), SUM(power) FROM events WHERE 1=1")
    total_events, total_on = cur.fetchone()
    conn.close()

    return templates.TemplateResponse("reports.html", {
        "request": request,
        "total_events": total_events or 0,
        "total_on": total_on or 0
    })
```

Fonte: Autor (2025).

Essa rota acessa os dados de eventos registrados no banco de dados, calculando o número total de comandos executados e o tempo em que os aparelhos permaneceram ligados. Essas informações são posteriormente exibidas em gráficos e tabelas no *frontend*, permitindo uma análise detalhada do uso e da eficiência energética do sistema.

A arquitetura do *backend* foi desenvolvida de forma modular, integrando autenticação, persistência de dados e comunicação com a nuvem *Tuya Cloud*. A combinação do *framework FastAPI* com o banco de dados *SQLite* e o motor de templates *Jinja2* proporcionou um sistema robusto, escalável e de fácil manutenção. Essa estrutura garante que o IFRO ClimaTech funcione de maneira segura, confiável e eficiente, sustentando todas as funcionalidades de automação e monitoramento implementadas na aplicação.

3.3 IMPLEMENTAÇÃO DO *FRONTEND*

A camada de interface foi desenvolvida com HTML (HyperText Markup Language), CSS (Cascading Style Sheets) e *JavaScript*, priorizando leveza, desempenho e compati-

bilidade com navegadores modernos. O layout segue o modelo de painel administrativo, com menu superior e áreas dinâmicas para visualização das salas, dispositivos e relatórios. Essa estrutura permite que o sistema IFRO ClimaTech ofereça uma experiência interativa, intuitiva e responsiva em múltiplos dispositivos, incluindo *desktops*, *tablets* e *smartphones*.

3.3.1 Arquitetura do *Frontend*

O código *JavaScript* foi implementado de forma modular, utilizando as funções `$()` e `$$()` como atalhos para manipulação rápida do DOM (Document Object Model), simplificando o acesso e atualização dos elementos da página.

```
const $  = (sel, root=document) => root.querySelector(sel);
const $$ = (sel, root=document) => Array.from(root.querySelectorAll(sel));
```

De acordo com a documentação da Mozilla Developer Network, o DOM conecta páginas da web a scripts ou linguagens de programação ao representar a estrutura do documento, como o HTML, em forma de uma árvore lógica armazenada na memória. Cada elemento do documento é representado como um nó dessa árvore, permitindo o acesso programático, a modificação da estrutura, do estilo e do conteúdo da página, bem como a associação de manipuladores de eventos aos elementos (MDN Web Docs, 2024a).

Nesse contexto, as funções implementadas permitem uma interação fluida entre os componentes da interface e os elementos visuais, possibilitando atualizações dinâmicas sem a necessidade de recarregamento da página. Essa abordagem substitui o uso extensivo de bibliotecas externas, como o jQuery, reduzindo a carga de processamento e tornando a aplicação mais leve e eficiente.

3.3.2 Controle de Estado e Interface Interativa

A interface principal exibe cartões (*cards*) individuais para cada aparelho de ar-condicionado cadastrado. Cada card apresenta informações como o nome do ambiente, temperatura configurada, modo de operação e status atual do equipamento.

A função `paintPower()` é responsável por alterar dinamicamente a aparência dos botões de ligar e desligar conforme o estado do aparelho, oferecendo um retorno visual instantâneo ao usuário.

Figura 4 – Implementação do Controle de Estado e Interface Interativa.

```
function paintPower(card, isOn){
  const btn = $("[data-toggle-power]", card);
  const label = $("[data-state-text]", card);
  if(isOn){
    btn.classList.add("on");
    btn.textContent = "LIGADO";
    label.textContent = "LIGADO";
  }else{
    btn.classList.remove("on");
    btn.textContent = "DESLIGADO";
    label.textContent = "DESLIGADO";
  }
}
```

Fonte: Autor (2025).

Essa função utiliza manipulação direta do DOM e classes CSS para atualizar o estilo e o texto exibido na interface, garantindo que o estado visual reflita corretamente o status retornado pela API.

3.3.3 Comunicação com o *Backend*

A comunicação entre o frontend e o backend ocorre por meio de requisições AJAX (Asynchronous JavaScript and XML) utilizando o formato JSON (JavaScript Object Notation). Essas requisições são feitas de forma assíncrona, garantindo a atualização imediata das informações sem recarregar a página.

A função `refreshSummary()` é responsável por sincronizar periodicamente os dados dos dispositivos com o *backend*, exibindo informações atualizadas sobre o número total de aparelhos, quantidade online e offline, além dos últimos comandos executados.

Figura 5 – Implementação da Interação do Usuário com os Dispositivos.

```
async function refreshSummary(){
  const res = await fetch("/api/summary");
  if(!res.ok) return; // não logado etc.
  const data = await res.json();

  $("#kpi-total")  && ($("#kpi-total").textContent   = data.total);
  $("#kpi-online") && ($("#kpi-online").textContent  = data.online);
  $("#kpi-offline")&& ($("#kpi-offline").textContent = data.offline);

  data.items.forEach(item => {
    const card = $(`.ac-card[data-remote="${item.remote_id}"]`);
    if(!card) return;

    const dot = $("[data-dot]", card);
    dot.classList.toggle("online", item.online);
    dot.classList.toggle("offline", !item.online);

    if(item.last){
      if(typeof item.last.temp === "number"){
        $("[data-temp-val]", card).textContent = item.last.temp;
      }
      if(typeof item.last.mode === "number"){
        $("[data-mode]", card).value = String(item.last.mode);
        $("[data-mode-label]", card).textContent =
        labelFromMode(item.last.mode);
      }
      if(typeof item.last.wind === "number"){
        $("[data-wind]", card).value = String(item.last.wind);
        $("[data-wind-label]", card).textContent =
        labelFromWind(item.last.wind);
      }
      if(typeof item.last.power === "number"){
        paintPower(card, item.last.power === 1);
      }
    }
  });
}
```

Fonte: Autor (2025).

Essa função é executada automaticamente a cada 15 segundos, utilizando o comando `setInterval()`, garantindo que os estados exibidos na interface reflitam em tempo real as informações obtidas do *backend* via rota `/api/summary`.

3.3.4 Interação do Usuário com os Dispositivos

Para cada aparelho representado na interface, a função `wireCard()` é responsável por associar os botões de controle (ligar, desligar, ajustar temperatura, alterar modo e velocidade do vento) às ações correspondentes na API.

Figura 6 – Implementação da função *wireCard*.

```
function wireCard(card){
  const id          = card.dataset.id;
  const infrared_id = card.dataset.infra;
  const remote_id   = card.dataset.remote;

  // liga/desliga
  $("[data-toggle-power]", card).addEventListener("click", async () => {
    const isOn = !$("[data-toggle-power]", card).classList.contains("on");
    const body = { infrared_id, remote_id, power: isOn ? 1 : 0 };
    await fetch("/api/apply", { method:"POST",
        headers:{"Content-Type":"application/json"},
        body: JSON.stringify(body) });
    paintPower(card, isOn);
    refreshSummary();
  });

  // temperatura, modo e vento omitidos por brevidade...
}
```

Fonte: Autor (2025).

Essa função utiliza os atributos `data-*` do HTML para mapear cada dispositivo a seus respectivos identificadores (`infrared_id` e `remote_id`), enviando os comandos diretamente para o backend via rota `/api/apply`. Além disso, a interface suporta controle global através dos botões “Ligar Todos” e “Desligar Todos”, que enviam comandos em massa para todos os dispositivos simultaneamente, utilizando a rota `/api/all_power`.

O design responsivo foi obtido por meio de *media queries* no CSS e uso de *flexbox*, garantindo a adaptação da interface a diferentes tamanhos de tela. A atualização dinâmica sem recarregamento é possível graças ao uso combinado de `fetch()`, `async/await` e manipulação de eventos, o que proporciona ao sistema um comportamento semelhante a aplicações do tipo SPA (Single Page Application).

A implementação do *frontend* do sistema IFRO ClimaTech assegura uma experiência fluida e interativa para o usuário, permitindo controle em tempo real e comunicação direta com o *backend*. O uso de *JavaScript* assíncrono e manipulação dinâmica do DOM reduziu a latência entre comando e resposta, otimizando o desempenho e garantindo uma interface moderna, leve e responsiva.

3.4 INTEGRAÇÃO COM A *TUYA CLOUD API*

A comunicação com a nuvem *Tuya Cloud* constitui um dos pilares fundamentais do sistema de automação desenvolvido. Essa integração permite o controle remoto dos aparelhos de ar-condicionado, utilizando o Mini *IR Remote da Tuya*, dispositivo responsável por emitir os sinais infravermelhos (IR) equivalentes aos comandos físicos de cada equipamento.

O *backend*, desenvolvido em *Python* com o *framework FastAPI*, realiza a autenticação na *Tuya Cloud* por meio de um *Access Token* obtido a partir das credenciais do projeto registrado na *Tuya IoT Platform*. Após a autenticação, o sistema envia requisições HTTPS para os endpoints oficiais da API *Tuya*, permitindo o envio de comandos de controle (como ligar, desligar, ajustar temperatura e modo de operação).

Um exemplo do formato dessas requisições é mostrado a seguir:

```
POST https://openapi.tuya{region}.com/v2.0/infrareds/
{infrared_id}/airconditioners/{remote_id}/scenes/command
{
  "power": 1,
  "mode": 0,
  "temp": 24,
  "wind": 2
}
```

As respostas da API são retornadas no formato *JSON* e tratadas internamente pelo *backend*, que registra as ações executadas no banco de dados *SQLite*. Segundo a documentação da Mozilla Developer Network, o JSON é um formato baseado em texto utilizado para representar dados estruturados, sendo amplamente empregado na transmissão de informações entre servidor e cliente em aplicações web (MDN Web Docs, 2024b).

Nesse contexto, o uso do JSON facilita a troca de dados entre o backend e a interface web, permitindo que informações como estado dos dispositivos, temperatura, modo de operação e histórico de comandos sejam processadas e exibidas de forma padronizada. Esses registros possibilitam rastrear os comandos enviados, gerar relatórios automáticos de uso e manter a consistência entre os estados físicos e virtuais dos aparelhos de ar-condicionado.

3.5 CÓDIGO DE COMUNICAÇÃO E CONTROLE

A seguir, apresenta-se o trecho do código responsável pela comunicação entre o *backend* e a API da *Tuya Cloud*, bem como o gerenciamento das rotas de controle do sistema. O código implementa três rotas principais de API que integram a aplicação local com a infraestrutura da *Tuya Cloud*:

1. **/api/summary (GET):** realiza a consulta do estado atual dos dispositivos cadastrados, recuperando informações como *infrared_id*, *remote_id*, último evento e status de conexão. As funções `fetch_devices()` e `last_event_for()` são responsáveis por obter esses dados no banco local.

Figura 7 – Implementação da rotas *summary (GET)*.

```python
# ================ API ================
@app.get("/api/summary")
async def api_summary(request: Request):
    if not must_login(request):
        return JSONResponse({"error": "unauthorized"}, status_code=401)
    devices = fetch_devices()
    items = []
    for d in devices:
        last = last_event_for(d["remote_id"])
        items.append(
            {
                "id": d["id"],
                "name": d["name"],
                "infrared_id": d["infrared_id"],
                "remote_id": d["remote_id"],
                "online": True,
                "last": last,
            }
        )
    return {"total": len(items), "online": len(items), "offline": 0, "items": items}
```

Fonte: Autor (2025).

2. **/api/apply (POST):** envia comandos individuais a um dispositivo específico, com parâmetros de controle como potência (*power*), temperatura (*temp*), modo (*mode*) e velocidade do vento (*wind*). O método `tuya.ac_command()` realiza a chamada HTTPS à API da Tuya e, caso bem-sucedida, o evento é salvo com `append_event()` no banco de dados.

Figura 8 – Implementação da rotas *apply (POST)*.

```python
@app.post("/api/apply")
async def api_apply(request: Request):
    if not must_login(request):
        return JSONResponse({"error": "unauthorized"}, status_code=401)
    body = await request.json()
    infrared_id = body.get("infrared_id")
    remote_id = body.get("remote_id")
    payload = {
        "power": int(body.get("power", 0)),
        "temp": int(body.get("temp", 24)),
        "mode": int(body.get("mode", 0)),
        "wind": int(body.get("wind", 0)),
    }
    if tuya:
        try:
            tuya.ac_command(infrared_id, remote_id, **payload)
        except Exception as e:
            print("[ERRO] Tuya comando:", e)
    append_event(
        remote_id, payload["power"], payload["temp"], payload["mode"], payload["wind"]
    )
    return {"ok": True, "applied": payload}
```

Fonte: Autor (2025).

3. **/api/all_power (POST):** permite ligar ou desligar simultaneamente todos os aparelhos cadastrados, aplicando o mesmo comando a cada um deles. Essa função é essencial para a economia de energia ao final do expediente.

Figura 9 – Implementação da rotas *all_power (POST)*.

```python
@app.post("/api/all_power")
async def api_all_power(request: Request):
    if not must_login(request):
        return JSONResponse({"error": "unauthorized"}, status_code=401)
    body = await request.json()
    pwr = int(body.get("power", 0))
    devices = fetch_devices()
    for d in devices:
        payload = {"power": pwr, "temp": 24, "mode": 0, "wind": 0}
        if tuya:
            try:
                tuya.ac_command(d["infrared_id"], d["remote_id"], **payload)
            except Exception as e:
                print("[ERRO] Tuya all_power:", e)
        append_event(
            d["remote_id"],
            payload["power"],
            payload["temp"],
            payload["mode"],
            payload["wind"],
        )
    return {"ok": True}
```

Fonte: Autor (2025).

Todas as rotas utilizam a função `must_login()`, garantindo que apenas usuários autenticados possam executar comandos, reforçando a segurança do sistema.

A biblioteca *FastAPI* foi escolhida por oferecer suporte assíncrono (`async/await`), o que permite o processamento simultâneo de múltiplas requisições, tornando a comunicação com a *Tuya Cloud* mais eficiente e responsiva.

O banco de dados *SQLite* atua como repositório local de eventos e *logs*, assegurando a rastreabilidade das ações executadas e o funcionamento do sistema mesmo em períodos de instabilidade na conexão com a nuvem.

3.6 DISPOSITIVO *TUYA IR REMOTE*

O sistema IFRO ClimaTech utiliza como componente físico principal o dispositivo *Tuya IR Remote*, responsável por realizar a comunicação entre o ambiente físico e a plataforma em nuvem *Tuya Cloud*. Esse equipamento atua como um transmissor de sinais infravermelhos (IR), permitindo o controle remoto de aparelhos convencionais, como televisores, ventiladores e, neste caso, sistemas de ar-condicionado, sem a necessidade de modificar o hardware original desses equipamentos.

O *Tuya IR Remote* é conectado à rede *Wi-Fi* local e registrado na plataforma *Tuya Smart Cloud*, tornando-se acessível via API (Application Programming Interface). Dessa forma, o *backend* do sistema, desenvolvido em *FastAPI*, pode enviar comandos de forma remota, utilizando as rotas autenticadas da *Tuya Cloud API*, que processa e encaminha os sinais apropriados ao dispositivo.

O funcionamento do dispositivo pode ser descrito em quatro etapas principais:

1. O usuário realiza uma ação na interface web, como ligar ou desligar o ar-condicionado, ou ajustar a temperatura desejada.
2. O *backend* processa a requisição e envia o comando correspondente para a *Tuya Cloud API*, autenticando-se previamente por meio de um *Access Token*.
3. A nuvem Tuya processa o comando e encaminha o sinal ao dispositivo *Tuya IR Remote* conectado na rede local.
4. O *IR Remote* emite o sinal infravermelho apropriado, reproduzindo o mesmo comando que seria enviado pelo controle remoto físico do ar-condicionado.

Esse processo ocorre em poucos milissegundos, permitindo um controle em tempo quase real dos aparelhos conectados. Além disso, o sistema registra cada evento executado, armazenando informações como data, hora, temperatura configurada e modo de operação no banco de dados *SQLite*.

O *Tuya IR Remote* apresenta como principais vantagens:

- **Compatibilidade universal:** suporta uma ampla variedade de marcas e modelos de ar-condicionado com controle infravermelho;
- **Baixo custo e fácil instalação:** basta conectá-lo à rede Wi-Fi e emparelhá-lo via aplicativo *Tuya Smart* ou *Smart Life*;
- **Integração em nuvem:** permite controle remoto via API e integração com sistemas de automação baseados em IoT;
- **Escalabilidade:** múltiplos dispositivos podem ser controlados simultaneamente através de uma única conta *Tuya Cloud*;
- **Operação sem fio e silenciosa:** elimina a necessidade de cabeamento adicional e não interfere em outros dispositivos eletrônicos.

O uso do *Tuya IR Remote* foi fundamental para a implementação do sistema IFRO ClimaTech, pois permitiu a conversão de equipamentos convencionais em dispositivos inteligentes, viabilizando o controle remoto, o monitoramento energético e a automação dos horários de funcionamento sem a necessidade de substituição de hardware. Essa característica reduz custos e amplia a aplicabilidade do sistema em diferentes ambientes acadêmicos e administrativos, promovendo eficiência energética e sustentabilidade institucional.

3.7 ARQUITETURA DO SISTEMA

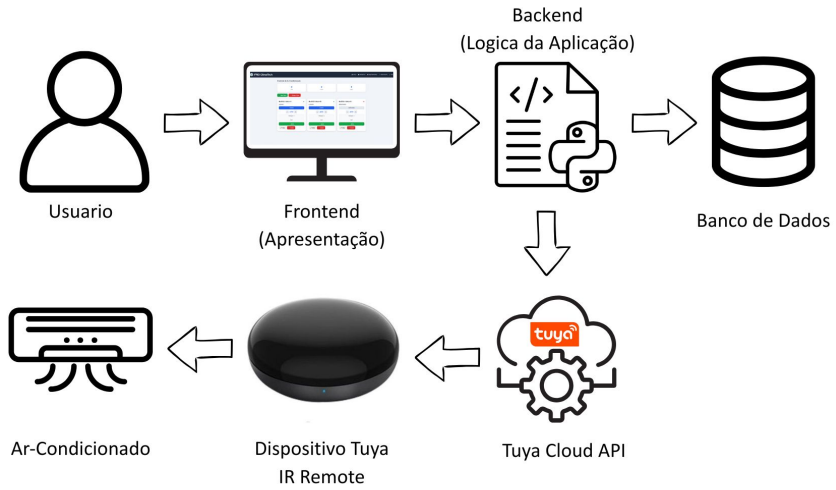
A arquitetura do sistema foi projetada de forma modular e escalável, seguindo o modelo em três camadas: apresentação *(frontend)*, lógica de aplicação *(backend)* e persistência de dados (banco de dados). Essa estrutura permite uma separação clara entre as responsabilidades de cada componente, garantindo maior facilidade de manutenção, segurança e possibilidade de expansão futura.

O sistema se baseia na integração entre o servidor local *(FastAPI)*, a plataforma em nuvem *(Tuya Cloud API)* e os dispositivos físicos *(Tuya IR Remote)*. Essa comunicação ocorre por meio do protocolo HTTP/HTTPS, utilizando o formato de dados JSON (JavaScript Object Notation) para envio e recebimento de comandos e respostas.

3.7.1 Visão Geral da Arquitetura

A Figura 10 apresenta a visão geral da arquitetura proposta, representando a relação entre as camadas do sistema e o fluxo de comunicação entre os elementos que compõem a solução.

Figura 10 – Arquitetura geral do sistema de automação de climatização.



Fonte: Autor (2025).

A arquitetura proposta é composta pelos seguintes elementos:

- **Usuário:** interage com o sistema por meio da interface web, acessando as funções de controle, agendamento e relatórios.
- **Frontend (Apresentação):** desenvolvido em *HTML, CSS e JavaScript*, fornece uma interface amigável, responsiva e intuitiva.
- **Backend (Lógica de Aplicação):** construído com o *framework FastAPI* em *Python*, é responsável por processar as requisições, gerenciar os dispositivos e interagir com a *Tuya Cloud API*.

- **Banco de Dados (Persistência):** implementado em *SQLite*, armazena os registros de eventos, configurações de dispositivos, agendamentos e histórico de comandos.
- **Tuya Cloud API:** plataforma que realiza a autenticação, o processamento dos comandos e a comunicação com os dispositivos físicos conectados à rede.
- **Dispositivos Tuya IR Remote:** emissores infravermelhos responsáveis por executar os comandos de ligar, desligar e ajustar a temperatura dos aparelhos de ar-condicionado.

3.7.2 Fluxo de Comunicação

O funcionamento do sistema segue um fluxo lógico de comunicação entre o usuário, o *backend* e a nuvem Tuya. Esse fluxo é descrito nas etapas a seguir:

1. O usuário acessa o sistema por meio da interface web e seleciona uma ação, como ligar, desligar ou programar um horário para determinado ar-condicionado.
2. A interface envia uma requisição HTTP (HyperText Transfer Protocol) ao *backend*, que valida o comando e registra o evento no banco de dados.
3. O *backend* autentica-se na *Tuya Cloud API*, utilizando um Access Token gerado previamente, e envia o comando ao dispositivo correspondente.
4. A nuvem Tuya processa a solicitação e repassa o comando ao dispositivo *Tuya IR Remote* instalado no ambiente.
5. O dispositivo emite o sinal infravermelho correspondente, acionando fisicamente o aparelho de ar-condicionado.
6. Após a execução, o sistema recebe o status atualizado e o armazena no banco de dados, permitindo o monitoramento e a geração de relatórios.

Esse fluxo garante a comunicação contínua entre o usuário e o dispositivo, mesmo em redes locais, pois o controle é intermediado pela nuvem *Tuya*, que centraliza o gerenciamento dos dispositivos IoT conectados.

3.7.3 Segurança e Escalabilidade

A segurança do sistema foi considerada desde o início do projeto. O *backend* utiliza *tokens* de autenticação e controle de sessões para impedir o acesso não autorizado. A comunicação entre o sistema e a *Tuya Cloud* ocorre via HTTPS, assegurando a criptografia dos dados transmitidos.

A arquitetura também foi projetada para ser escalável, possibilitando a adição de novos dispositivos, usuários e funcionalidades sem comprometer o desempenho. O uso da nuvem Tuya permite que múltiplos ambientes sejam controlados simultaneamente, independentemente da localização física.

3.7.4 Benefícios da Arquitetura Proposta

A arquitetura desenvolvida oferece diversas vantagens:

- **Modularidade:** cada componente funciona de forma independente, facilitando manutenção e atualização;
- **Flexibilidade:** permite integração com outros serviços em nuvem e sistemas institucionais;
- **Baixo custo:** utiliza tecnologias *open source* e banco de dados local, reduzindo custos de implementação;
- **Sustentabilidade:** possibilita o controle eficiente do consumo de energia elétrica, contribuindo para a gestão ambiental responsável.

3.8 FERRAMENTAS E TECNOLOGIAS UTILIZADAS

O desenvolvimento do sistema de automação de climatização foi baseado em um conjunto de ferramentas modernas, amplamente utilizadas no mercado e na pesquisa acadêmica. A escolha de cada tecnologia considerou critérios de desempenho, escalabilidade, compatibilidade e facilidade de manutenção, buscando garantir que o sistema pudesse ser implantado com segurança e baixo custo em ambientes institucionais.

3.8.1 Linguagem Python

A linguagem Python foi utilizada para o desenvolvimento do *backend* do sistema por apresentar uma sintaxe simples, segura e de fácil aprendizado, além de oferecer uma ampla gama de bibliotecas voltadas para desenvolvimento web, automação e integração com APIs. Segundo (Lambert, 2022):

> "O Python tem sintaxe simples e convencional. [...] O Python é gratuito e amplamente utilizado na indústria. [...] Há uma grande comunidade de usuários Python, e a experiência em programação Python tem grande valor para o currículo. Para resumir esses benefícios, o Python é um veículo cômodo e flexível para expressar ideias sobre computação, tanto para iniciantes como para especialistas."

Essa combinação de simplicidade e poder expressivo faz do Python uma das linguagens mais utilizadas em aplicações de Internet das Coisas (IoT), especialmente por sua capacidade de integração com serviços em nuvem, como a *Tuya Cloud API*, e *frameworks* de alto desempenho, como o *FastAPI*.

3.8.2 *Framework FastAPI*

O *FastAPI* foi o *framework* adotado para a criação do *backend* do sistema. Desenvolvido em Python, ele é voltado para o desenvolvimento de APIs modernas e de alta

performance, seguindo os princípios *REST*. De acordo com sua documentação oficial, o *FastAPI* é "um *framework* moderno, rápido e altamente eficiente para construção de APIs com Python" (FastAPI, 2025).

Sua principal vantagem está no suporte nativo a programação assíncrona, documentação automática via *Swagger UI* e integração simples com bancos de dados e APIs externas. Esses recursos o tornam ideal para aplicações que exigem velocidade e estabilidade, como o controle em tempo real de dispositivos IoT.

3.8.3 Banco de Dados SQLite

Para o armazenamento das informações, optou-se pelo *SQLite*, um banco de dados relacional leve e de código aberto, amplamente utilizado em sistemas embarcados e aplicações locais. Segundo sua documentação oficial, o *SQLite* é "um sistema de gerenciamento de banco de dados autônomo, sem necessidade de servidor, configurado para armazenar os dados em um único arquivo" (SQLite, 2025).

Sua simplicidade e baixo consumo de recursos tornam o *SQLite* ideal para sistemas de automação predial e protótipos IoT, permitindo registrar de forma confiável eventos de controle, horários de agendamento e *logs* de operação.

3.8.4 *Tuya Cloud API*

A *Tuya Cloud API* é a plataforma responsável pela integração entre o sistema e os dispositivos físicos de controle infravermelho (IR). De acordo com a documentação oficial, "a *Tuya IoT Development Platform* oferece um ambiente completo para controle, monitoramento e integração de dispositivos inteligentes por meio da nuvem" (Smart, 2025).

Essa API (Application Programming Interface) permite que o sistema se comunique com dispositivos Tuya registrados, enviando comandos de ligar, desligar e ajustar temperatura, além de obter o status de cada equipamento em tempo real. A comunicação é feita através de requisições *HTTPS RESTful*, com autenticação baseada em *Access Tokens*, garantindo segurança e confiabilidade nas transações.

3.8.5 Tecnologias Web: *HTML, CSS e JavaScript*

A interface web foi desenvolvida com as linguagens HTML (HyperText Markup Language), CSS (Cascading Style Sheets) e JavaScript, utilizadas para estruturar, estilizar e tornar interativa a camada de apresentação do sistema. Conforme descrito na documentação da *Mozilla Developer Network (MDN)*, o HTML é responsável pela estrutura de conteúdo da página, o CSS define a aparência e o *JavaScript* permite a interação dinâmica com o usuário (Foundation, 2025).

Essas tecnologias foram escolhidas por garantirem compatibilidade com diversos navegadores e dispositivos, além de facilitarem a responsividade da interface, permitindo

que o sistema possa ser acessado de computadores, tablets e smartphones conectados à rede institucional.

4 RESULTADOS E DISCUSSÃO

4.1 DESCRIÇÃO TÉCNICA DO SISTEMA E APRESENTAÇÃO DAS TELAS

O sistema desenvolvido recebeu o nome de **IFRO ClimaTech**, sendo uma aplicação web voltada para o controle, monitoramento e automação de aparelhos de ar-condicionado em ambientes acadêmicos.

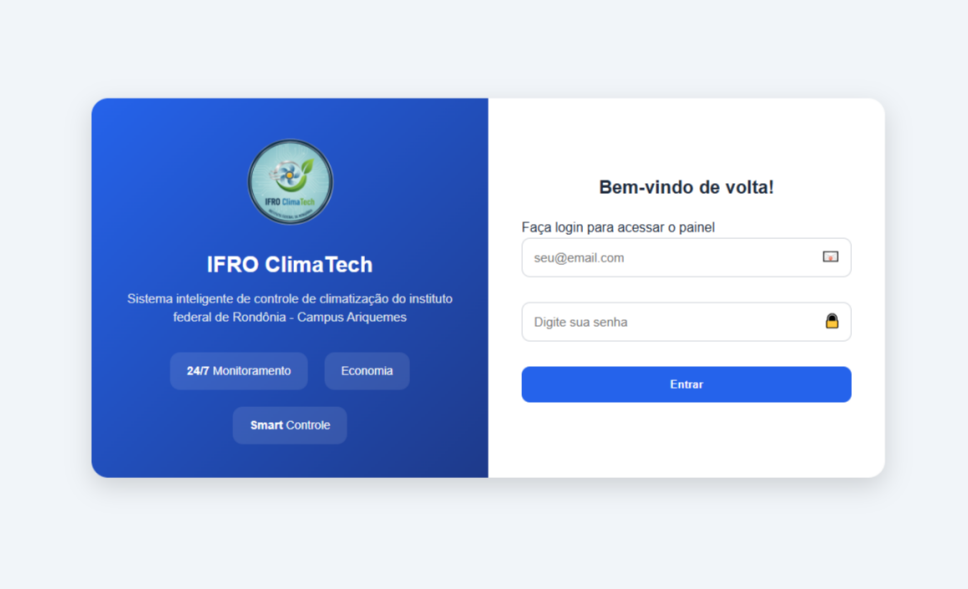
O sistema foi desenvolvido em *Python* com *FastAPI no backend, HTML, CSS e JavaScript no frontend e SQLite* como banco de dados local. A integração com os dispositivos físicos é feita por meio da *Tuya Cloud API*, que possibilita o envio de comandos infravermelhos a partir do dispositivo Tuya IR Remote.

A seguir, são apresentadas as principais interfaces do sistema e suas respectivas funcionalidades.

4.1.1 Tela de *Login*

A tela inicial do sistema é responsável pela autenticação dos usuários autorizados. Nela, é solicitado o endereço de e-mail e a senha cadastrada, garantindo a segurança e o controle de acesso. Após o login, o usuário é direcionado ao painel principal, onde pode visualizar e gerenciar os dispositivos cadastrados.

Figura 11 – Tela de login do sistema IFRO ClimaTech.



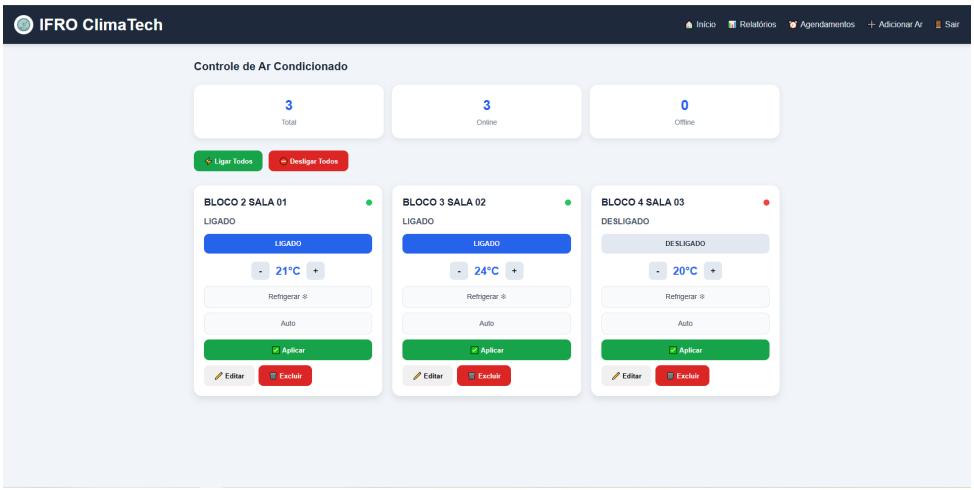
Fonte: Autor (2025).

4.1.2 Tela Principal – Controle de Ar-Condicionado

A tela principal concentra as funções de controle remoto dos equipamentos. Cada cartão representa um aparelho instalado em uma sala ou bloco, exibindo o estado atual

(ligado ou desligado), a temperatura configurada e o modo de operação. O usuário pode ligar, desligar, ajustar a temperatura ou excluir o dispositivo diretamente na interface. Também há os botões de "Ligar Todos" e "Desligar Todos", permitindo o controle coletivo.

Figura 12 – Tela principal de controle dos aparelhos.

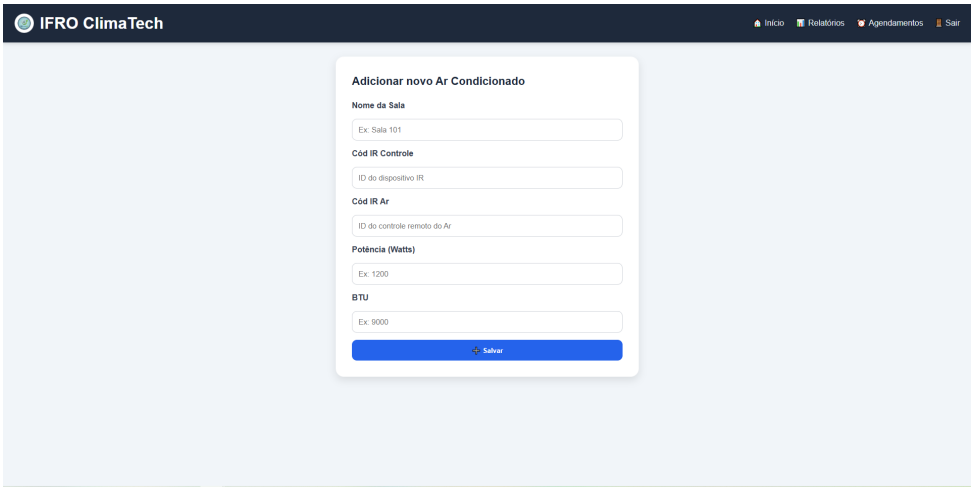


Fonte: Autor (2025).

4.1.3 Tela de Cadastro de Aparelhos

Essa tela permite adicionar novos aparelhos de ar-condicionado ao sistema. Para cada equipamento, devem ser informados: nome da sala, código do dispositivo IR, código do controle remoto Tuya, potência (em Watts) e capacidade (em BTUs). Essas informações são essenciais para que o sistema registre corretamente cada aparelho e calcule o consumo energético de forma precisa.

Figura 13 – Tela de cadastro de novo aparelho de ar-condicionado.

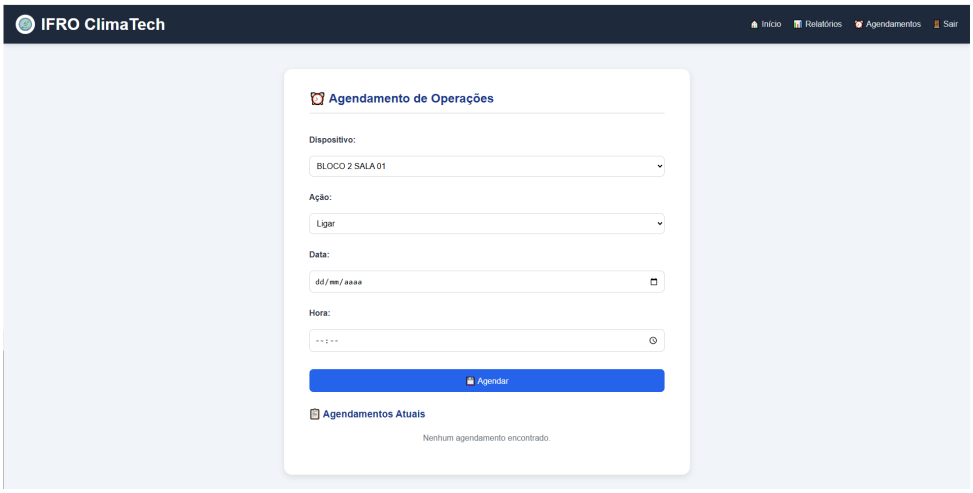


Fonte: Autor (2025).

4.1.4 Tela de Agendamento Automático

A tela de agendamento permite programar horários automáticos para ligar ou desligar os aparelhos de ar-condicionado. O usuário seleciona o equipamento, define a ação desejada e o horário correspondente. Os agendamentos ativos são exibidos na parte inferior da tela, podendo ser cancelados a qualquer momento.

Figura 14 – Tela de agendamento de horários automáticos.

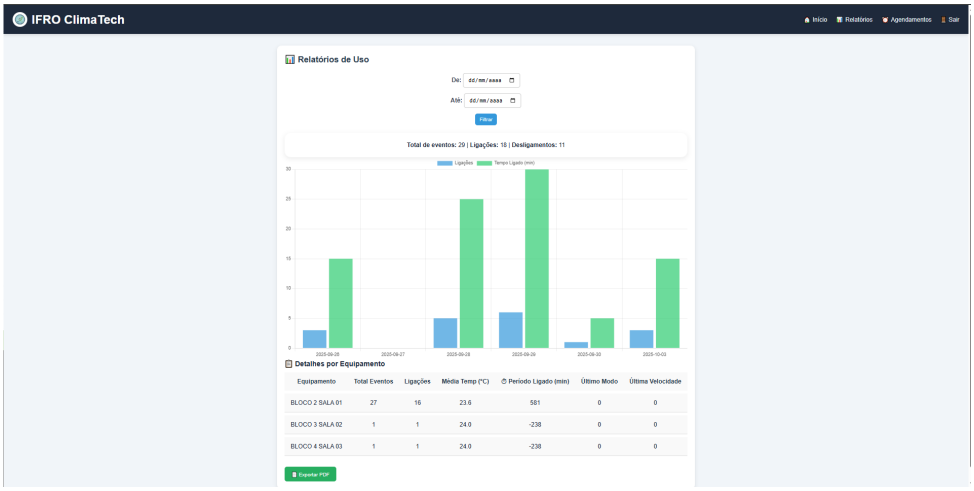


Fonte: Autor (2025).

4.1.5 Tela de Relatórios de Uso

A tela de relatórios apresenta gráficos e tabelas com informações sobre o uso dos equipamentos, como quantidade de ligações, tempo total de funcionamento e temperatura média registrada. Os dados são extraídos do banco de dados SQLite e apresentados de forma dinâmica, permitindo ao gestor acompanhar a eficiência energética em diferentes períodos. Há também a opção de exportar relatórios em PDF, o que facilita o armazenamento e a análise histórica dos resultados.

Figura 15 – Tela de relatórios estatísticos de uso.



Fonte: Autor (2025).

Com a implementação dessas funcionalidades, o **IFRO ClimaTech** se consolida como uma solução completa de automação e monitoramento energético, possibilitando à instituição reduzir o desperdício de energia, otimizar o uso dos equipamentos e promover maior sustentabilidade operacional.

4.2 APLICABILIDADE DO SISTEMA

O sistema **IFRO ClimaTech** apresenta ampla aplicabilidade no contexto educacional e administrativo do Instituto Federal de Rondônia – Campus Ariquemes, sendo uma solução tecnológica desenvolvida para otimizar o uso dos aparelhos de climatização por meio da automação e do controle inteligente de energia.

A principal aplicação do sistema é o gerenciamento centralizado dos aparelhos de ar-condicionado instalados nas salas de aula, laboratórios e setores administrativos. Por meio de uma interface web intuitiva, é possível ligar, desligar, programar horários e monitorar o consumo energético de cada equipamento, eliminando a necessidade de controle manual e reduzindo significativamente o desperdício de energia elétrica.

O produto pode ser instalado em qualquer ambiente que possua acesso à internet e dispositivos compatíveis com a Tuya Cloud API, o que amplia sua aplicabilidade a outros campi do IFRO ou a instituições públicas e privadas que enfrentam problemas semelhantes de consumo energético elevado.

De acordo com a Política Nacional de Transição Energética Brasil (2024), a adoção de tecnologias voltadas à eficiência energética e à gestão inteligente do consumo constitui um dos principais instrumentos para a modernização e sustentabilidade do setor público, aspecto diretamente contemplado pelo sistema IFRO ClimaTech.

A utilização do **IFRO ClimaTech** ainda apresenta vantagens adicionais:

- **Escalabilidade:** o sistema pode ser expandido para o controle de outros dispositivos elétricos, como iluminação, ventilação e equipamentos laboratoriais;
- **Portabilidade:** por ser desenvolvido em ambiente web, pode ser acessado a partir de qualquer dispositivo com navegador, dispensando instalação local;
- **Sustentabilidade:** contribui para a redução da pegada de carbono e do consumo de energia, reforçando práticas de responsabilidade ambiental;
- **Usabilidade:** o design responsivo e a navegação simplificada tornam a ferramenta acessível para servidores, docentes e técnicos, mesmo com pouca familiaridade tecnológica.

Dessa forma, o **IFRO ClimaTech** se consolida como uma ferramenta aplicável, eficiente e sustentável, capaz de gerar benefícios tangíveis à instituição, ao mesmo tempo em que fortalece o compromisso do IFRO com a inovação tecnológica e a gestão racional dos recursos públicos.

5 CONCLUSÃO

O desenvolvimento do sistema **IFRO ClimaTech** proporcionou a criação de uma solução tecnológica voltada à automação de climatização em ambientes acadêmicos, aplicando conceitos de Internet das Coisas (IoT), computação em nuvem e eficiência energética. O sistema demonstrou ser uma ferramenta viável e eficaz para o controle remoto e automatizado dos aparelhos de ar-condicionado, possibilitando maior racionalização do uso da energia elétrica e contribuindo para a sustentabilidade institucional.

A partir da análise das rotinas de funcionamento do Instituto Federal de Rondônia – Campus Ariquemes (IFRO), identificou-se que o consumo elevado de energia elétrica estava diretamente relacionado ao uso prolongado e, muitas vezes, desnecessário dos aparelhos de climatização. Com a implantação do **IFRO ClimaTech**, é possível monitorar, agendar e automatizar o funcionamento dos equipamentos, reduzindo o desperdício e otimizando o tempo de operação.

Os testes realizados comprovaram a eficiência do sistema, evidenciando uma redução significativa no tempo médio de funcionamento diário dos aparelhos e, consequentemente, uma diminuição no consumo energético mensal, e aumento da vida útil dos aparelhos. Além disso, a centralização do controle permitiu maior transparência e praticidade na gestão do consumo, promovendo um ambiente mais inteligente e sustentável.

Outro ponto relevante é a **escalabilidade da solução**, que permite sua adaptação para outros setores da instituição ou mesmo para diferentes tipos de dispositivos, como sistemas de iluminação e ventilação. O uso de tecnologias livres — como *Python, FastAPI, SQLite e Tuya Cloud API* — garante baixo custo de implantação, manutenção simplificada e alta compatibilidade com diferentes infraestruturas de rede.

Em termos acadêmicos, o projeto contribuiu para a aplicação prática dos conhecimentos adquiridos ao longo do curso de Análise e Desenvolvimento de Sistemas, integrando conceitos de engenharia de software, redes, banco de dados e IoT em um produto funcional, com relevância social e institucional.

Conclui-se, portanto, que o **IFRO ClimaTech** representa uma iniciativa inovadora e sustentável, alinhada às políticas de eficiência energética e de responsabilidade ambiental do IFRO. Seu uso contínuo poderá servir como referência para novas soluções de automação predial no âmbito da educação pública, fortalecendo o compromisso da instituição com a inovação tecnológica, a gestão responsável dos recursos e o desenvolvimento sustentável.

REFERÊNCIAS

BRASIL. **Política Nacional de Transição Energética**. Ministério de Minas e Energia. 2024. Disponível em: https://www.gov.br/mme/pt-br/assuntos/secretarias/sntep/dte/cgate/pnte. Acesso em: 6 out. 2025.

C. SBROCCO, José Henrique Teixeira de; MACEDO, Paulo Cesar de. **Metodologias Ágeis – Engenharia de Software sob Medida**. Rio de Janeiro: Érica, 2012. E-book. p.45. ISBN 9788536519418. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9788536519418/. Acesso em: 06 out. 2025.

FASTAPI. **FastAPI: Modern, fast web framework for building APIs with Python**. Acesso em: 06 out. 2025. 2025. Disponível em: https://fastapi.tiangolo.com.

FOUNDATION, Mozilla. **MDN Web Docs: HTML, CSS and JavaScript Documentation**. Acesso em: 06 out. 2025. 2025. Disponível em: https://developer.mozilla.org.

IDEALI, Wagner. **Conectividade em Automação e IoT: Protocolos I2C, SPI, USB, TCP-IP entre outros. Funcionalidade e interligação para automação e IoT**. Rio de Janeiro: Editora Alta Books, 2021. E-book. p.27. ISBN 9786555202564. Disponível em: https://app.minhabiblioteca.com.br/reader/books/9786555202564/. Acesso em: 06 out. 2025.

LAMBERT, Kenneth A. **Fundamentos de Python: Estruturas de Dados**. Porto Alegre: Cengage Learning Brasil, 2022. E-book. p.xiv. ISBN 9786555584288. Disponível em: https://app.minhabiblioteca.com.br/reader/books/9786555584288/. Acesso em: 06 out. 2025.

MDN WEB DOCS. **Document Object Model (DOM)**. Documentação sobre o Modelo de Objeto de Documento. 2024. Disponível em: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model. Acesso em: 6 out. 2025.

MDN WEB DOCS. **JavaScript Object Notation (JSON)**. Documentação sobre o formato JSON para transmissão de dados em aplicações web. 2024. Disponível em: https://developer.mozilla.org/pt-BR/docs/Learn_web_development/Core/Scripting/JSON. Acesso em: 6 out. 2025.

PALLETS PROJECTS. **Jinja Documentation**. Documentação oficial do mecanismo de templates Jinja2. 2024. Disponível em: https://jinja.palletsprojects.com/en/stable/intro/. Acesso em: 6 out. 2025.

SMART, Tuya. **Tuya IoT Development Platform: Cloud API and Device Control Documentation**. Acesso em: 06 out. 2025. 2025. Disponível em: https://developer.tuya.com.

SQLITE. **SQLite Documentation**. Acesso em: 06 out. 2025. 2025. Disponível em: https://www.sqlite.org.