Escola Superior de Tecnologia e Gestão

**Escola Inteligente**

Rafael dos Santos Cordeiro

Guilherme Seco Filipe Quaresma Pimentel

Trabalho de Projeto da unidade curricular de Tecnologias de Internet

Leiria, junho de 2025

# Lista de Figuras

Elemento a figurar, **quando aplicável**.

[Figura 1- Diagrama Representativo da Temperatura 4](#_Toc201396449)

[Figura 2 - Diagrama representativo do sensor de movimento 5](#_Toc201396450)

[Figura 3 - Ligações nos equipamentos 5](#_Toc201396451)

# Lista de tabelas

Elemento a figurar, **quando aplicável**.

**Não foi encontrada nenhuma entrada do índice de ilustrações.**

# Lista de siglas e acrónimos

Elemento a figurar, **quando aplicável**.

|  |  |
| --- | --- |
| ESTG | Escola Superior de Tecnologia e Gestão |
| IPLeiria | Instituto Politécnico de Leiria |
| API | Interface de programação de aplicações |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Índice

[Lista de Figuras iv](#_Toc201395990)

[Lista de tabelas v](#_Toc201395991)

[Lista de siglas e acrónimos vi](#_Toc201395992)

[1. Introdução 1](#_Toc201395993)

[2. Arquitetura 2](#_Toc201395994)

[3. Implementação 4](#_Toc201395995)

[4. Cenário de Teste 6](#_Toc201395996)

[5. Resultados obtidos 7](#_Toc201395997)

[6. Conclusão 8](#_Toc201395998)

[7. Bibliografia 9](#_Toc201395999)

# Introdução

O projeto desenvolvido pelo nosso grupo consiste num sistema de monitorização e atuação inteligente, com aplicação potencial em ambientes como edifícios inteligentes ou espaços escolares. A solução assenta numa arquitetura distribuída, composta por um microcontrolador (Arduino) e um microcomputador (Raspberry Pi), comunicando entre si através de uma API alojada num servidor web.

Este sistema tem como principais funcionalidades a monitorização da temperatura ambiente, deteção de presença de pessoas e o controlo automático de atuadores, como um servo motor que simula o funcionamento de um ar condicionado, reagindo de forma autónoma aos dados recolhidos pelos sensores.

A informação recolhida é disponibilizada numa plataforma online, onde é possível consultar os dados em tempo real e, no caso de utilizadores com permissões de administrador, aceder também a dados históricos apresentados em formato gráfico, bem como a imagens captadas por uma webcam.

Este projeto foi desenvolvido no âmbito da unidade curricular de Tecnologias da Internet, com o objetivo de explorar e aplicar conceitos de sistemas distribuídos, IoT e desenvolvimento web em contexto prático

**Justificação e Pertinência do Tema: A relevância deste trabalho reside na crescente necessidade de escolas mais propriamente politécnicos, de melhorarem a sua eficiência logística, reduzindo assim custos desnecessários, e melhorando a qualidade para os alunos, através de sistemas autónomos e dinâmicos.**

**Objetivo do Trabalho**: Desenvolver um Sistema distribuído composto por sensores e atuadores que comuniquem entre si através de uma API, permitindo assim o controlo e monotorização dos mesmos.

**Métodos e Técnicas Utilizados**: Foram utilizados scripts baseados em métodos HTTP (“GET” e “POST”), programação em Python e C++, e ferramentas de desenvolvimento como o Visual Studio Code, XAMPP, e bibliotecas específicas para controlo de hardware.

# Arquitetura

O sistema desenvolvido apresenta uma arquitetura distribuída composta por diferentes componentes de hardware e software, integrados para formar um ecossistema IoT funcional. Abaixo descreve-se a composição e o funcionamento de cada parte:

## Arduino 1010 WIFI:

* Equipado com um sensor digital de temperatura (DHT11).
* Conectado via Wi-Fi para envio periódico de dados em formato JSON para a API.
* Possui dois LEDs (verde e vermelho) que ligam conforme o movimento detetado pelo sensor PIR conectado ao Raspberry Pi:
* LED verde: indica entrada de pessoas.
* LED vermelho: indica saída de pessoas.

## Raspberry Pi 4:

* Controla um servo motor SG90, que simula a operação de um sistema de ar condicionado.
* Ligado a um sensor de movimento PIR HC-SR501, utilizado para detetar a presença de pessoas.
* Executa scripts em Python que enviam dados via HTTP (método POST) para a API e leem dados do ficheiro valor.txt.
* Utiliza bibliotecas como gpiozero, RPi.GPIO e requests.

## Servidor Web (XAMPP)

* A API é desenvolvida em PHP e responde a pedidos HTTP para registo (POST) e consulta (GET) de dados.
* Estrutura organizada em pastas separadas por sensores e atuadores.
* Inclui uma dashboard local para visualização dos dados em tempo real.

## Comunicação

* O Arduino envia os dados para o endpoint api.php utilizando o método HTTP POST com payload em JSON.
* O Raspberry Pi pode aceder aos dados através de leitura direta de ficheiros (valor.txt) ou via API (GET).
* Todos os dispositivos estão interligados em rede local (Wi-Fi).

## Webcam

* Utilização da aplicação DroidCam num smartphone para simular uma webcam.
* Acionada automaticamente sempre que o sensor PIR deteta movimento de entrada.

## Dashboard dinâmica

* Apresenta dados atualizados em tempo real.
* Diferentes permissões de acesso:
* Administrador (admin): acesso completo, incluindo gráficos históricos e imagens da webcam.
* Utilizador comum (user): acesso limitado, sem permissões administrativas.

# Implementação

## Arduino

* Mede temperatura com um sensor digital (DHT11).
* Envia via WiFi para a API:
  + {"nome": "Temperatura", "valor": 22.5, "hora": "2025-06-02 21:00:00"}
* Envia a cada 5 segundos.
* Recebe o valor do sensor PIR, e pisca um dos leds(verde, vermelho) se a pessoa entrou ou saiu

## Raspberry Pi

* Lê o **valor.txt** da temperatura.
* Aciona o servo motor se a temperatura for < 15 ou > 25ºC.
* Lê o sensor PIR (GPIO4) e envia 1 se detetar movimento.
* Utiliza biblioteca gpiozero, requests e RPi.GPIO.

Uma imagem com texto, diagrama, captura de ecrã, file

Os conteúdos gerados por IA podem estar incorretos.

Figura 1- Diagrama Representativo da Temperatura

Uma imagem com diagrama, texto, file, captura de ecrã

Os conteúdos gerados por IA podem estar incorretos.

Figura 2 - Diagrama representativo do sensor de movimento

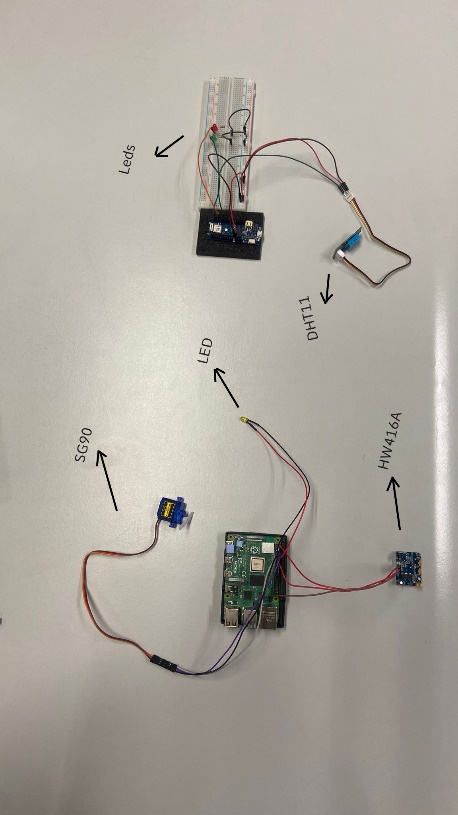


Figura 3 - Ligações nos equipamentos

# Mutli Thread

Durante o desenvolvimento do projeto, identificámos um problema funcional relacionado com a execução sequencial das tarefas no Raspberry Pi: quando o servo motor era acionado para oscilar (simulando o funcionamento de um ar condicionado), o processo de oscilação ocupava o tempo total do programa, impedindo que outros sensores fossem lidos ou que outras ações, como a deteção de movimento, fossem executadas corretamente.

## Importância no Projeto

No contexto do nosso projeto, várias tarefas independentes precisam de acontecer simultaneamente ou de forma assincrona , nomeadamente :

* **Leitura do sensor de movimento PIR**, que deve reagir imediatamente à presença de pessoas.

Sem o uso de multithreading, estas ações teriam de ser realizadas sequencialmente, o que causaria **delays significativos**, especialmente na leitura e resposta aos sensores.

## Uso de Multi Thread no projeto



A black background with white text

AI-generated content may be incorrect.A screen shot of a computer program

AI-generated content may be incorrect.

# Cenário de Teste

**Arduino:**

* **Sensor de temperatura** ligado ao Arduino em A0.
* **Wi-Fi SSID**: "labs", **Password**: "1nv3nt@r2023\_IPLEIRIA".
* **Dois leds (verde= entrada de pessoas, vermelho=saída de pessoas)**

**Raspberry:**

* **Sensor PIR(HC-SR501): VCC(pin 1), OUT(GPIO4), GND(pin 9)**
* **Servo motor (SG 90): sinal no GPIOO17, VCC 5V, GND**
* **Led (Amarelo = contador >= 20 pessoas)**

**Webcam:**

* **Usamos o DROIDCAM no smartphone para tirar uma foto quando o raspberry deteta movimento**

**Dashboard:**

* **Dashboard dinâmica**
* **3 utilizadores com permissões diferentes(admin tem acesso a gráficos do historico e á webcam, o user tem acesso a tudo menos aos do admin)**

**Código Utilizado:**

* **RaspCodeProject.py, para o raspberry que deteta o movimento, liga o servo motor e liga um led**
* **Projeto.ino, para o Arduíno que mede a temperatura, liga dois leds**

**Observação**: Tanto no Arduino como no Raspberry Pi, optou-se por concentrar todas as funcionalidades num único ficheiro em cada plataforma, para facilitar a organização e compreensão do projeto.

**Software Utilizado:**

* **Servidor XAMPP local** com pasta /Uni-Project/api/
* **VS Code**
* **PuTTY**

# Resultados obtidos

**5.1 Temperatura:**

Relativamente à monitorização da temperatura, o sistema envia corretamente os dados do Arduino para a API, utilizando o código desenvolvido no ficheiro Projeto.ino, com resposta de sucesso (HTTP 200). Quando a temperatura ultrapassa os limites definidos, o servo motor entra em funcionamento, oscilando de forma contínua para simular a ativação de um sistema de ar condicionado.

**5.2 Sensor Movimento:**

O sensor de movimento está ligado ao raspberry que deteta se entraram pessoas ou saíram, de seguida o arduino recebe os valores do sensor de movimento, e dependendo do resultado ele pisca um de dois leds(Verde: entram pessoas, Vermelho: Saíram pessoas).

**5.3 A webcam:**

A aplicação **DroidCam**, instalada num smartphone, atua como webcam de vigilância. Sempre que o sensor PIR deteta movimento, o Raspberry Pi aciona a captura de uma imagem. Esta funcionalidade foi testada com sucesso, garantindo a integração entre hardware e software.

**5.4 Dashboard**

A dashboard apresenta os dados dos sensores em tempo real de forma clara e organizada. Os diferentes perfis de utilizador funcionam corretamente, conforme o esperado:

* **Administrador**: tem acesso aos dados históricos, apresentados em forma de gráfico, e às imagens captadas pela webcam.
* O **utilizador comum** tem acesso apenas aos dados em tempo real e ao historicos mas sem a apresentação grafica.

# Conclusão

O projeto permitiu integrar sensores e atuadores reais com uma arquitetura IoT baseada em APIs. A divisão de responsabilidades entre o Arduino e o Raspberry Pi mostra-se eficaz, permitindo um sistema modular e escalável. O uso de tecnologias conhecidas como HTTP, PHP e Python facilita a manutenção. Como melhorias futuras, poderá ser implementada autenticação na API, armazenamento em base de dados e integração com sistemas de notificacão em tempo real.

# Bibliografia

<https://app.diagrams.net/>

<https://www.w3schools.com/>

<https://getbootstrap.com/>