Escola Superior de Tecnologia e Gestão

**Fábrica/Indústria Inteligente**

Pedro Afonso Monteiro Pedro

Ricardo dos Santos Franco

Trabalho de Projeto da unidade curricular de Tecnologias de Internet

Leiria, junho de 2021

# Lista de Figuras

Elemento a figurar, quando aplicável.

[Figura 1 - Ilustração da Arquitetura do Projeto. 3](https://myipleiria-my.sharepoint.com/personal/luis_frazao_ipleiria_pt/Documents/DEI-ESTG-IPLEIRIA/201920/EI/TI/TI_2020/2_PRATICA/PROJETO/1_ENUNCIADO/Template_Relatorio_Projeto_TI.docx#_Toc38661057)

[Figura 2 - Fluxograma do funcionamento da ventoinha 4](#_Toc38661058)

[Figura 3 - Fluxograma do funcionamento do extintor 5](#_Toc38661058)

[Figura 4 - Fluxograma do funcionamento da porta 5](#_Toc38661058)

[Figura 5 - Fluxograma do funcionamento da alarme 6](#_Toc38661058)

[Figura 6 - Fluxograma do funcionamento da luz 6](#_Toc38661058)

[Figura 7 - Dispositivos ligados ao switch 7](#_Toc38661058)

[Figura 8 - Dispositivos ligados ao Access-Point\_PT do interior 8](#_Toc38661058)

[Figura 9 - Dispositivos ligados ao MCU 8](#_Toc38661058)

[Figura 10 - Dispositivos ligados ao SBC 9](#_Toc38661058)

[Figura 11 - Dispositivos ligados ao Access-Point\_PT do exterior 9](#_Toc38661058)

[Figura 12 - Dispositivos ligados ao solar panel 10](#_Toc38661058)

# Lista de siglas e acrónimos

|  |  |
| --- | --- |
| API  CSS  ESTG | Interface de Programação de Aplicações  Cascading Style Sheets  Escola Superior de Tecnologia e Gestão |
| IPLeiria  HTML | Instituto Politécnico de Leiria  Linguagem de Marcação de Hipertexto |
| IoT | Internet of Things |
| PHP | Hypertext Preprocessor |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Índice

[Lista de Figuras iv](#_Toc67320774)

[Lista de siglas e acrónimos v](#_Toc67320776)

[1. Introdução 1](#_Toc67320777)

[2. Arquitetura 2](#_Toc67320778)

[3. Implementação 4](#_Toc67320779)

[4. Cenário de Teste 7](#_Toc67320780)

[5. Resultados obtidos 11](#_Toc67320781)

[6. Conclusão 12](#_Toc67320782)

[7. Bibliografia 13](#_Toc67320783)

[8. Anexos 14](#_Toc67320784)

[8.1. Código do Microcontrolador 1 14](#_Toc67320785)

[8.2. Código do SBC 1 … 23](#_Toc67320786)

[8.3. Código Python do ficheiro projeto.py… 34](#_Toc67320787)

[8.4. Código HTML e PHP do ficheiro index … 36](#_Toc67320788)

[8.5.](#_Toc67320789) Código HTML e PHP do ficheiro dashboard… 38

# Introdução

Neste projeto tivemos como objetivo desenvolver uma fábrica inteligente.

Optamos por este tema pois é algo que pode ser aplicado facilmente numa situação real e também por representar a evolução da realidade.

Os objetivos deste trabalho são poder monitorizar as condicionantes de uma sala de uma fábrica.

Sendo estas condicionantes divididas em duas partes, os sensores e os atuadores.

Os sensores, sendo estes a temperatura, humidade, movimento, detetor de fogo e luminosidade, têm como objetivo coletar dados para haver controlo sobre a sala.

Já os atuadores, porta, ventoinha, camara, armazém, alarme, extintor e luz serão os responsáveis por controlar a sala, através dos valores dos sensores, numa dashboard.

Usamos também várias ferramentas tais como o packet tracer e SQL para criar e fazer alterações à bases de dados e phpmyadmin para administrar a mesma.

Utilizamos também várias linguagens de programação, como java script, python, php e HTML.

# Arquitetura

Neste trabalho pretendemos usar dois microcontroladores.

Um SBC, de modo a controlar os atuadores, realizar GETs e assim dependendo resultado destes controlar determinado atuador.

O outro microcontrolador é um MCU. Este tem como objetivo controlar os sensores, fazer a leitura de dados e realizar POSTs para a api.

A nível de sensores teremos um sensor de temperatura para registar a temperatura ambiente, um sensor de humidade de modo a registar a humidade da sala, sensor de movimento que abrirá a porta ao registar movimento perto desta, um sensor de fogo de modo a verificar se houve incêndios na sala e sensor de luminosidade de modo a verificar a luminosidade da sala.

Sobre atuadores temos uma ventoinha de modo a tentar regular a temperatura da sala, um extintor que será ativado quando o sensor de fogo registar algum fogo, uma luz que será ligada dependendo da luminosidade da sala e um alarme que esta relacionado com uma porta e um portão de armazém.

Usamos também um RFID para controlar quem entra na fábrica verificando as credenciais da pessoa.

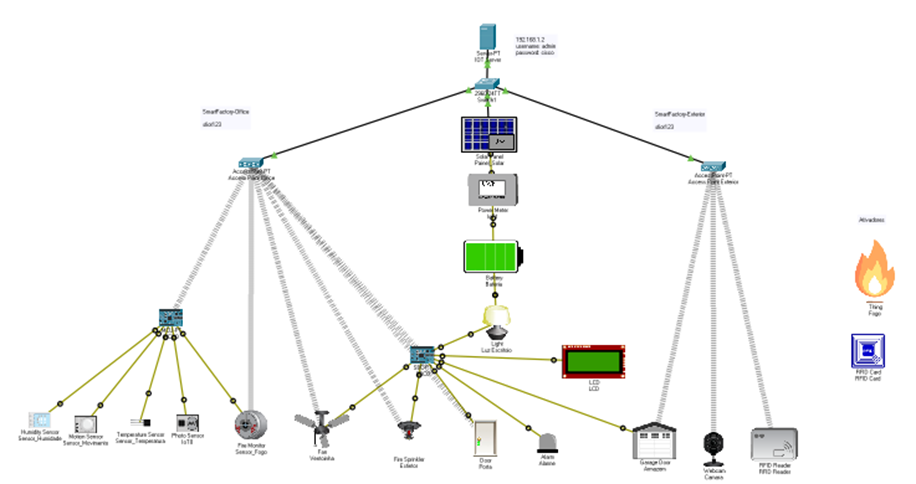
Para podermos visualizar os resultados dos sensores, estes valores são enviados automaticamente para a dashboard onde poderão ser monitorizados.

Figura 1 – Ilustração da Arquitetura do Projeto

# Implementação

Para cada evento que possa ocorrer no projeto, decidimos representá-lo através de um fluxograma.

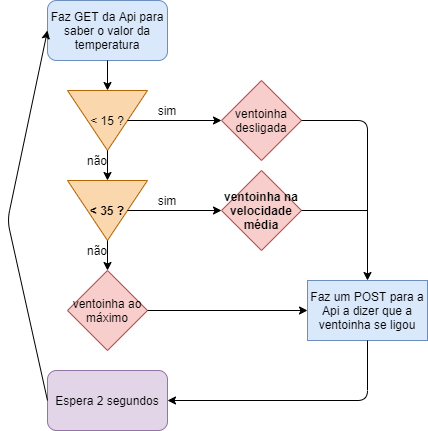


Figura 2 – Fluxograma do funcionamento da ventoinha

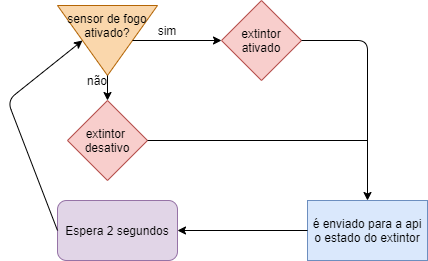


Figura 3 – Fluxograma do funcionamento do extintor

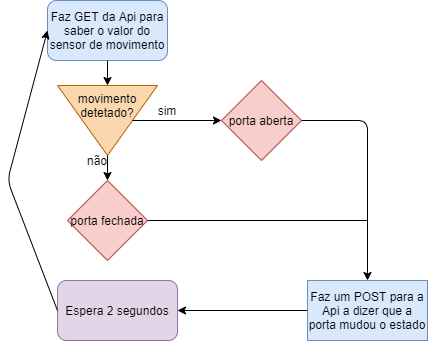
****

Figura 4 – Fluxograma do funcionamento da porta

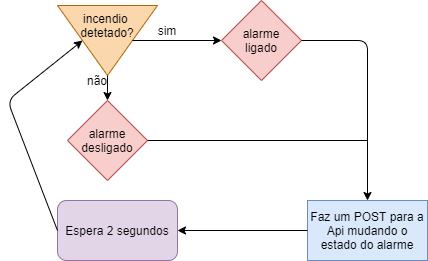


Figura 5 – Fluxograma do funcionamento do alarme

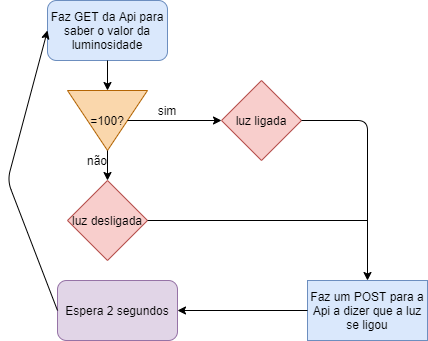


Figura 6 – Fluxograma do funcionamento da luz

# Cenário de Teste

Para que todos os elementos do nosso projeto tivessem interligados usamos o xampp e o uniserver. Este cria uma rede interna. Serve também como servidor php-apache, phpmyadmin e hospeda uma base de dados mysql.

A base do nosso projeto é o cisco packet tracer, o HTML, PHP e Python.

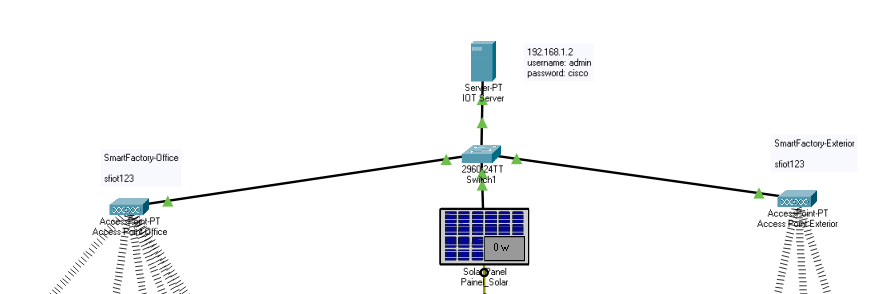
Dentro do cisco packet tracer criamos uma rede interna usando um switch 2960-24TT, onde se encontra ligado a um Server-PT (ligado na porta FastEthernet0 no Servidor e na porta GigabiteEthernet 0 do Router) , 1 solar panel e 2 Access-Point-PT, um que será para o interior da fábrica e o outro para o exterior.

Figura 7 – Dispositivos ligados ao switch

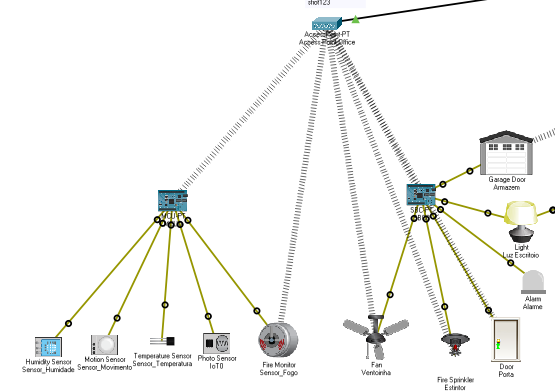
No interior, o Access-Point-PT está ligado a um MCU-PT e a um SBC-PT, que estão ligados aos sensores e aos atuadores respetivamente.

Figura 8 – Dispositivos ligados ao Access-Point\_PT do interior

Ligado ao MCU temos os sensores, nas portas digitais os sensores de movimento, fogo e luz, e nas portas analógicas os sensores de humidade e temperatura.

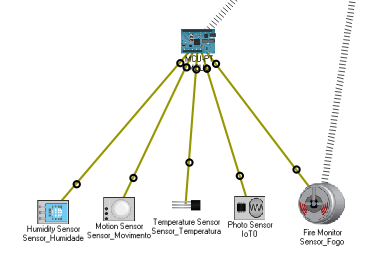


Figura 9 – Dispositivos ligados ao MCU

No SBC temos os atuadores, todos eles ligados nas portas digitais.

Temos a ventoinha que a sua intensidade irá depender da temperatura.

Ou seja, se a temperatura for menor que 15º a ventoinha estará desligada, se for menor que 35º estará a velocidade média e se superior a 35º a ventoinha estará ao máximo.

O atuador extintor irá receber através do sensor de fogo a indicação de quando terá de ser ativado.

A porta irá ser aberta quando o sensor de movimento detetar movimentação perto desta.

A luz terá 2 estados, ligada ou desligada, o estado dependerá do sensor de luminosidade que está responsável por obter a luminosidade do environment do cisco packet tracer.

O LCD irá mostra a informação que o utilizador pretender e pode ser mudado a partir do python.

E por fim o alarme que irá funcionar respetivamente ao sensor de fogo, o alarme é ativado quando há fogo e só pode ser desativado pelo admin na dashboard .

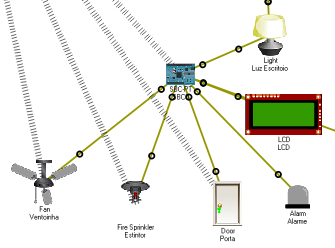


Figura 10 – Dispositivos ligados ao SBC

O Access-Point-PT do interior está também conectado via wifi aos atuadores e ao sensor de fogo.

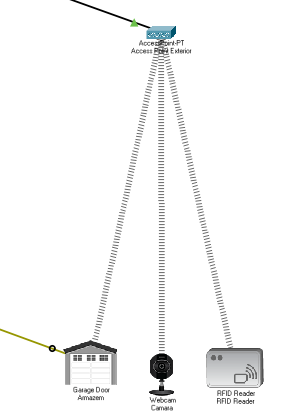
No exterior, o Access-Point-PT está ligado via wifi, à garagem, à webcam e ao RFID reader.

Figura 11 – Dispositivos ligados ao Access-Point\_PT do exterior

No exterior encontra-se também o solar panel, que se encontra ligado a um power meter conectado a uma bateria, isto para poder dar energia ao atuador Luz.

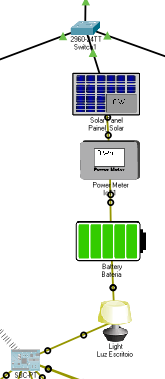


Figura 12 – Dispositivos ligados ao solar panel

Saindo do cisco packet tracer, temos o nosso website feito com HTML e php.

Como utilizamos uma estrutura MVC o index é onde é iniciada a navegação e posteriormente o utilizar será redirecionado para outras rotas ( url`s ) ao longo da sua navegação pelo website.

É constituído por uma página de login, dashboard e o histórico.

Na dashboard é onde estão todas as informações disponíveis dos componentes do projeto que estão a ocorrer no cisco packet tracer.

Cada componente tem também direito a um histórico, exceto a câmara, onde vão ser armazenados através de uma tabela e um gráfico os dados coletados ao longo do tempo.

# Resultados obtidos

Todos os objetivos referidos foram implementados com sucesso e funcionam como esperado.

No cisco packet tracer, configuramos as variáveis ambientais para tentar simular o melhor possível a nossa realidade.

Com isso foi possível obter dados realistas e assim uma boa dinâmica entre os componentes do projeto.

O MCU e o SBC estão a comunicar corretamente com a api.

Por fim, o python também está a funcionar corretamente com a dashboard e api.

# Conclusão

Neste projeto desenvolvemos uma fábrica inteligente onde se pretendia reunir as condições para o funcionamento desta.

Esta fábrica tem a particularidade de se poder controlar todos os mecanismos á distancia, o que se poderá ser algo usado em próximas fabricas.

Cumprimos todos os objetivos propostos no projeto.

Através do projeto obtivemos conhecimento de IoT e desenvolvimento web front-end e back-end.

# Bibliografia

Matéria: pdf’s das aulas teóricas

Algum código: Exercícios feito em aula

# Anexos

Anexos com código desenvolvido.

## Código do Microcontrolador 1 (MCU 0)

var url="http://127.0.0.1/projetoV2/api/updateDisp";

var loginurl ="http://127.0.0.1/projetoV2/api/auth";

var login={

"username":"Admin",

"password":"cisco"

};

var isAuth=false;

var http;

var valor;

var valores;

var state;

var SENSOR\_MOVIMENTO = 0;

var SENSOR\_TEMPERATURA = A0;

var SENSOR\_FOGO = 1;

var SENSOR\_LUZ = 2;

var SENSOR\_HUMIDADE = A1;

function setup(){

pinMode(SENSOR\_MOVIMENTO,INPUT);

pinMode(SENSOR\_TEMPERATURA,INPUT);

pinMode(SENSOR\_FOGO,INPUT);

pinMode(SENSOR\_LUZ,INPUT);

pinMode(SENSOR\_HUMIDADE,INPUT);

http = new RealHTTPClient();

}

function loop(){

// Caso não esteja autentificado irá ser adicionado ao obejto "http"

// os headers de autentificação à api para ser feita a autentificação

if(!isAuth){

http.postWithHeader(loginurl,login,{});

}

// Função que irá ser executada depois do request post for efetuado

http.onDone = function(status, data, replyHeader) {

// Status code 403 = acesso negado

if(status == 403){

isAuth=falso;

Serial.println("Autentificação falhada");

}else{

isAuth=true;

}

};

humidade();

luz();

sensorMovimento();

temperatura();

fogo();

delay(2000);

}

function humidade(){

valor = digitalRead(SENSOR\_HUMIDADE)\*(200/1023)-99.75;

valor = Math.round(valor);

Serial.println(valor);

if(digitalRead(SENSOR\_HUMIDADE) == HIGH){

valor=100;

}else{

valor=0;

}

valores={

"name":"Humidade",

"type":"sensor",

"value":valor,

"state":0,

};

RealHTTPClient.post(url,valores);

}

function luz(){

if(digitalRead(SENSOR\_LUZ) == HIGH){

valor=100;

}else{

valor=0;

}

valores={

"name":"Luminosidade",

"type":"sensor",

"value":valor,

"state":0,

};

RealHTTPClient.post(url,valores);

}

function fogo(){

if(digitalRead(SENSOR\_FOGO) == LOW){

valor="Sem Fogo";

state=0;

}else{

valor="Fogo Detetado";

state=2;

// Como o alarme só via ser possivel desligar no dashboard

// só quando o fogo é detetado é que o ativamos

valores={

"name":"Alarme",

"type":"atuador",

"state":state,

};

RealHTTPClient.post(url,valores);

}

Serial.println(valor);

valores={

"name":"Detetor de Fogo",

"type":"sensor",

"value":valor,

"state":state,

};

RealHTTPClient.post(url,valores);

}

function sensorMovimento(){

if(digitalRead(SENSOR\_MOVIMENTO) == LOW){

valor="Sem Movimento";

state=0;

}else{

valor="Movimento Detetado";

state=2;

}

Serial.println(valor);

valores={

"name":"Sensor Movimento",

"type":"sensor",

"value":valor,

"state":state,

};

RealHTTPClient.post(url,valores);

}

function temperatura(){

valor = analogRead(SENSOR\_TEMPERATURA)\*(200/1023)-99.75;

valor = Math.round(valor \* 100) / 100;

Serial.println(valor);

valores={

"name":"Temperatura",

"type":"sensor",

"value":valor,

"state":0,

};

RealHTTPClient.post(url,valores);

}

## Código do Microcontrolador 2 (SBC 0)

var urlGet="http://127.0.0.1/projetoV2/api/findDisp";

var urlUpdate="http://127.0.0.1/projetoV2/api/updateDisp";

var loginurl ="http://127.0.0.1/projetoV2/api/auth";

var login={

"username":"Admin",

"password":"cisco"

};

var isAuth=false;

var http;

var PIN\_PORTA = 0;

var PIN\_VENTOINHA = 1;

var PIN\_ARMAZEM = 2;

var PIN\_ALARME = 3;

var PIN\_EXTINTOR = 4;

var PIN\_LUZ = 5;

var valores;

var state;

function setup(){

pinMode(PIN\_PORTA,OUTPUT);

pinMode(PIN\_VENTOINHA,OUTPUT);

pinMode(PIN\_ARMAZEM,INPUT);

pinMode(PIN\_ALARME,OUTPUT);

pinMode(PIN\_EXTINTOR,INPUT);

pinMode(PIN\_LUZ,OUTPUT);

http = new RealHTTPClient();

}

function loop(){

// Caso não esteja autentificado irá ser adicionado irá

// ser feito um pedido de autentificação à api

if(!isAuth){

http.postWithHeader(loginurl,login,{});

}

// Função que irá ser executada depois do request post for efetuado

http.onDone = function(status, data, replyHeader) {

// Status code 403 = acesso negado

if(status == 403){

isAuth=falso;

Serial.println("Autentificação falhada");

}else{

isAuth=true;

}

};

luz();

porta();

ventoinha();

armazem();

alarme();

extintor();

delay(2000);

}

function luz(){

RealHTTPClient.get(urlGet.concat("?name=Luminosidade&type=sensor"), function(status,data){

data = JSON.parse(data);

data = data.split("%")[0];

if(data == "100"){

customWrite(PIN\_LUZ, "2");

Serial.println("Luz ligada");

state=0;

} else{

customWrite(PIN\_LUZ, "0");

Serial.println("Luz desligada");

state=2;

}

valores={

"name":"Luz",

"type":"atuador",

"state": state,

};

RealHTTPClient.post(urlUpdate, valores, function(status,data){});

});

}

function extintor(){

state = customRead(PIN\_EXTINTOR);

if(state == LOW){

digitalWrite(PIN\_ALARME,LOW);

Serial.println("Extintor desligado");

state=0;

} else{

digitalWrite(PIN\_ALARME,HIGH);

Serial.println("Extintor ligado");

state=2;

}

valores={

"name":"Extintor",

"type":"atuador",

"state": state,

};

RealHTTPClient.post(urlUpdate, valores, function(status,data){});

}

function alarme(){

RealHTTPClient.get(urlGet.concat("?name=Alarme&type=atuador"), function(status,data){

data = JSON.parse(data);

if(data == "green"){

digitalWrite(PIN\_ALARME,LOW);

Serial.println("Alarme desligado");

} else{

digitalWrite(PIN\_ALARME,HIGH);

Serial.println("Alarme ligado");

}

});

}

function porta(){

RealHTTPClient.get(urlGet.concat("?name=Sensor\_Movimento&type=sensor"), function(status,data){

data = JSON.parse(data);

if(data == "Movimento Detetado"){

customWrite(PIN\_PORTA,"1");

Serial.println("Porta Aberta");

state=0;

}else{

customWrite(PIN\_PORTA,"0");

Serial.println("Porta fechada");

state=2;

}

valores={

"name":"Porta",

"type":"atuador",

"state": state,

};

RealHTTPClient.post(urlUpdate, valores, function(status,data){});

});

}

function ventoinha(){

RealHTTPClient.get(urlGet.concat("?name=Temperatura&type=sensor"), function(status,data){

data = JSON.parse(data);

data = data.split("º")[0];

if(data < 15){

customWrite(PIN\_VENTOINHA,"0");

Serial.println("Ventoinha Desligada");

state=0;

}else if(data < 35){

customWrite(PIN\_VENTOINHA,"1");

Serial.println("Ventoinha Velocidade Média");

state=1;

}else{

customWrite(PIN\_VENTOINHA,"2");

Serial.println("Ventoinha no Máximo");

state=2;

}

valores={

"name":"Ventoinha",

"type":"atuador",

"state":state,

};

RealHTTPClient.post(urlUpdate, valores, function(status,data){});

});

}

function armazem(){

if(customRead(PIN\_ARMAZEM) == 1){

Serial.println("Armazem aberto");

state=0;

}else{

Serial.println("Armazem fechado");

state=2;

}

valores={

"name":"Armazem",

"type":"atuador",

"state":state,

};

RealHTTPClient.post(urlUpdate, valores, function(status,data){});

}

## Código Python do ficheiro projeto.py

import upload\_image as up

import lcd

import requests, threading

LOGIN={"username":"Admin", "password":"cisco"}

urlLogin="http://127.0.0.1/projetoV2/api/auth"

# Função de login que irá proceder ao login à api e devolver a

# sessão com a autentificação

def login():

session=requests.Session()

# Request de login à api

response=session.post(urlLogin, data=LOGIN)

if response.status\_code == 200:

print("Autentificação bem sucedida")

return session

print("Erro na autentificação")

return null

print("\n\*\*\*\*\*\*\* Projeto de python iniciado \*\*\*\*\*\*\*\n\n")

session=login()

lcd.init(session)

up.init(session)

## Código HTML e PHP do ficheiro index

<?php

session\_start();

date\_default\_timezone\_set("Europe/Lisbon");

/\* Como é necessário incluir ou requerir cada ficheiro de todas classes e modelos o php tem uma

função já defenida em que facilita esse processo \*/

spl\_autoload\_register(function ($model\_name) {

if (file\_exists("../app/core/" . $model\_name . ".php")) {

require\_once "../app/core/" . $model\_name . ".php";

} elseif (file\_exists("../app/models/" . $model\_name . ".php")) {

require\_once "../app/models/" . $model\_name . ".php";

} elseif (file\_exists("../app/controllers/" . $model\_name . ".php")) {

require\_once "../app/controllers/" . $model\_name . ".php";

} else {

echo "Ficheiro não existe";

}

});

// Criação de uma variável do tipo "App"

$app = new App(dirname(\_\_DIR\_\_) . "/app");

// Incluir todas as routes presentes no ficheiro "Routes.php"

require\_once "../app/Routes.php";

// Chamar a função "run" da App

$app->run();

## Código HTML e PHP do ficheiro dashboard

<div class="container-fluid" id="dashboard-container">

<?php

$this->title = "Dashboard";

if (isset($this->deviceTypes)) {

foreach ($this->deviceTypes as $deviceType) {

echo '<div class=" text-center" style="padding-top:1%">

<h2>' . $deviceType["title"] . '</h2>

</div>

<hr>';

echo '<div class="row d-flex justify-content-center text-center">';

if (!empty($deviceType["devices"])) {

foreach ($deviceType["devices"] as $device) {

echo '

<div class="col-md-3 m-3 d-flex justify-content-center text-center">

<div class="card card-' .

$device->state .

'">

<h6 style="padding-top:10%">' .

$device->name .

'</h6>

<span class="circle circle-' .

$device->state .

'"></span>

<div class="card-body d-flex justify-content-center align-items-center">

<img src="' .

$device->image .

'" width="100" alt="' .

$device->name .

'">

</div>

<h6>'; ?>

<?php

echo $device->value ?? "";

echo '</h6> <h6>' .

$device->date .

'</h6>

<div class="card-footer">

<a class="mb-5" href="./history?name=' .

$device->linkName .

'&type='. $device->type .'">Histórico</a>

</div>

</div>

</div>';

}

} else {

echo "sem " . $deviceType["title"];

}

echo '</div>';

}

}

?>

<?php

if ($\_SESSION["priv"] == 2) {

?>

<div class="col-md-12 mt-5 pb-5 d-flex justify-content-center text-center">

<div class="card">

<div class="card-body">

<img src="public/assets/images/foto\_armazem.jpg" style="border-radius: 15px" width="100%"

alt="Foto Armazem">

</div>

<h6>Camara do Armazem</h6>

</div>

</div>

<div class=" text-center" style="padding-top:1%">

<h2>Ações</h2>

</div>

<hr>

<div class="pb-5">

<button type="submit" onclick="updateDisp('Alarme', 'atuador', null , '2' )"

class="btn btn-primary btn-lg btn-block">Ativar Alarme

</button>

<button type="submit" onclick="updateDisp('Alarme', 'atuador', null , '0' )"

class="btn btn-secondary btn-lg btn-block">Desativar Alarme

</button>

</div>

<?php } ?>

</div>