

# FRONT END PARA SISTEMA DE INVENTÁRIO DE EQUIPAMENTO DO DEE

João Pedro da Silva Dias



Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

2021



Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Unidade Curricular de Projeto/Estágio, do 3º ano, da Licenciatura em Engenharia Eletrotécnica e de Computadores

Candidato: João Pedro da Silva Dias, N° 1181617, 1181617@isep.ipp.pt

Orientação científica: Jorge Manuel Estrela da Silva, jes@isep.ipp.pt



Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

6 de setembro de 2021



## *Agradecimentos*

A realização de um projeto desta natureza nunca seria possível sem a ajuda tanto de familiares, amigos que me acompanharam desde o primeiro dia de faculdade e professores que se cruzaram comigo durante o meu percurso académico no Instituto Superior de Engenharia do Porto (ISEP).

Queria começar por agradecer ao orientador Eng.º. Jorge Manuel Estrela da Silva, primeiramente pelo voto de confiança ao atribuir-me este projeto, por toda a disponibilidade, e flexibilidade nos momentos de maior incerteza e sobretudo pela sua capacidade de me desafiar sempre a levar o projeto mais além.

De seguida queria também agradecer, ao colega Gonçalo Morais (1180823), responsável pelo *back end* de suporte a esta tecnologia. Com ele foram partilhadas longas horas de pesquisa e trabalho sempre em busca da alternativa que melhor compusesse os nossos objetivos comuns.

Por último agradecer à minha namorada por estar sempre ao meu lado nos momentos mais difíceis ao longo de todo o projeto, bem como flexibilidade ao longo de todo o projeto.



## *Resumo*

Ao longo deste documento é descrito um *front end* concebido para um sistema de inventário de equipamentos do DEE (Departamento de Engenharia Eletrotécnica). A sua principal finalidade é a capacidade de realizar operações CRUD (*Create, Read, Update and Delete*).

As páginas desenvolvidas apresentam como destaque a tabela com os dados da página em questão. As principais tecnologias utilizadas no projeto foram o Bootstrap e AJAX. A primeira incorpora todo o contexto estético da página tais como tabelas, botões e formulários. A segunda é usada para aceder aos serviços fornecidos pelo *back end*, nomeadamente a implementação das operações CRUD respeitantes aos dados do sistema de inventário.

Durante o desenvolvimento do projeto testes foram realizados em servidor local e externo (servidor do DEE). Estes permitiram averiguar a fiabilidade e robustez da aplicação, bem como a sua maturidade em ambientes reais.

## *Palavras-Chave*

*Front end*, sistema de inventário, CRUD, Bootstrap, AJAX.





## *Abstract*

Throughout this document, a *front end* designed for an equipment inventory system of the DEE (Department of Electrical Engineering) is described. Its main purpose is the ability to perform CRUD operations (Create, Read, Update and Delete).

The developed pages feature as a highlight the table with the data of the page in question. The main technologies used in the project were Bootstrap and AJAX. The first incorporates all the aesthetic context of the page such as tables, buttons, and forms. The second is used to access the services provided by the *back end* services, mainly the implementation of CRUD operations concerning the data from inventory system.

During the development of the project tests were carried out on a local and external server (DEE server). These allowed us to verify the reliability and robustness of the application, as well as its maturity in real environments.

### ***Keywords***

*Front end*, inventory system, CRUD, Bootstrap, AJAX.



# Índice

<b>AGRADECIMENTOS .....</b>	<b>I</b>
<b>RESUMO .....</b>	<b>III</b>
<b>ABSTRACT .....</b>	<b>V</b>
<b>ÍNDICE .....</b>	<b>VII</b>
<b>ÍNDICE DE FIGURAS .....</b>	<b>IX</b>
<b>ÍNDICE DE TABELAS .....</b>	<b>XIII</b>
<b>ACRÓNIMOS.....</b>	<b>XV</b>
<b>1. INTRODUÇÃO .....</b>	<b>1</b>
1.1. CONTEXTUALIZAÇÃO .....	1
1.2. OBJETIVOS .....	2
1.3. CALENDARIZAÇÃO .....	2
1.4. ORGANIZAÇÃO DO RELATÓRIO .....	3
<b>2. TECNOLOGIAS .....</b>	<b>5</b>
2.1. HTML .....	5
2.2. CSS .....	6
2.3. PHP .....	7
2.4. JAVASCRIPT .....	7
2.5. BOOTSTRAP .....	12
2.6. BOOTSTRAP TABLES .....	14
2.7. FONT AWESOME .....	15
<b>3. ESTRUTURA.....</b>	<b>16</b>
3.1. MAPA DO WEBSITE .....	16
3.2. DESIGN .....	17
3.3. PÁGINA DE LOGIN.....	19
3.4. PÁGINAS DESENVOLVIDAS .....	20
3.5. LOGIN .....	23
3.6. LOGOUT.....	28
3.7. FORMULÁRIOS .....	29
3.8. TABELAS.....	36
3.9. CRIAÇÃO DA TABELA.....	49
3.10. FUNCIONALIDADES DISPONÍVEIS EM CADA PÁGINA .....	51
3.11. ACESSOS.....	51
3.12. SETUP .....	52

<b>4. RESULTADOS.....</b>	<b>53</b>
<b>5. CONCLUSÕES .....</b>	<b>55</b>
5.1. TRABALHO FUTURO .....	55
<b><i>REFERÊNCIAS DOCUMENTAIS.....</i></b>	<b>57</b>

## *Índice de Figuras*

Figura 1 Logótipo HTML 5 .....	5
Figura 2 Exemplo código HTML .....	6
Figura 3 Logótipo CSS.....	6
Figura 4 Link do ficheiro CSS externo .....	7
Figura 5 Logótipo PHP .....	7
Figura 6 Abrir/fechar instruções em PHP .....	7
Figura 7 Logótipo JavaScript .....	8
Figura 8 Link do ficheiro JavaScript externo .....	8
Figura 9 Logótipo jQuery.....	8
Figura 10 Exemplo DOM de uma página HTML [11] .....	9
Figura 11 Esquema de funcionamento do AJAX [12] .....	10
Figura 12 Função AJAX genérica para os pedidos realizados no projeto.....	12
Figura 13 Logótipo Bootstrap v.4 .....	12
Figura 14 Logótipo Bootstrap Tables.....	14
Figura 15 Logótipo Data Tables.....	14
Figura 16 Logótipo Font Awesome .....	15
Figura 17 Link de referência ao Kit de ícones .....	15
Figura 18 Mapa do Website parte 1 .....	16
Figura 19 Mapa do Website parte 2 .....	16
Figura 20 Mapa do Website parte 3 .....	16
Figura 21 Estrutura da página web.....	17
Figura 22 Esboço do output desejado .....	18
Figura 23 Banner DEE.....	18
Figura 24 Azul (#124f87), Cinza (#949ca4), Branco (#ffffff) .....	18
Figura 25 Código de cores adjacentes: #dc3545, #f7c5c9, #198754, #aae7b8 .....	19
Figura 26 Página de Login .....	20
Figura 27 Tabela e estruturas auxiliares.....	21
Figura 28 Ficheiro sem desfragmentação.....	22
Figura 29 <i>header.php</i> .....	22
Figura 30 <i>footer.php</i> .....	22
Figura 31 Ficheiro fragmentado .....	22
Figura 32 Estabelecimento da comunicação com API de Login.....	25
Figura 33 Função completa com API de Login .....	25
Figura 34 Decomposição dos dados retornados .....	25

Figura 35 Estrutura identificadora de erros .....	26
Figura 36 Tratamento dos erros retornados .....	26
Figura 37 Mensagem de erro ao utilizar credenciais erradas .....	26
Figura 38 Mensagem de erro ao utilizar uma palavra-passe errada .....	26
Figura 39. Estrutura de exibição de erros na página principal .....	27
Figura 40 Decomposição do <i>Token</i> nas suas três estruturas fundamentais .....	27
Figura 41 Aceder à estrutura "Data" do <i>Token</i> .....	27
Figura 42 Armazenamento em variáveis globais .....	28
Figura 43 Declaração das variáveis globais na página .....	28
Figura 44 Estrutura de verificação da autenticação .....	28
Figura 45 Ficheiro <i>logout.inc.php</i> .....	29
Figura 46 Mensagem de tempo de sessão excedido .....	29
Figura 47 Mensagem de <i>logout</i> com sucesso .....	29
Figura 48 Função de criação das opções para as caixas de seleção .....	31
Figura 49 Formulário colapsado .....	32
Figura 50 Formulário expandido .....	32
Figura 51 Inserção de novas alternativas às caixas de seleção .....	33
Figura 52 Função de repetição (100ms) .....	33
Figura 53 Esquema lógico implementado na função descrição .....	33
Figura 54 Código utilizado para realizar o objetivo .....	34
Figura 55 Esquema lógico de filtragem das descrições .....	35
Figura 56 Eliminação das opções das caixas de seleção .....	35
Figura 57 Caixa de pesquisa .....	37
Figura 58 Estruturas de paginação .....	37
Figura 59 Informações predefinidas no Bootstrap .....	37
Figura 60 Função para alterar a paginação predefinida .....	37
Figura 61 Filtragem por coluna .....	38
Figura 62 Tabela preenchida .....	39
Figura 63 Tabela sem entradas .....	39
Figura 64 Barra de ferramentas da tabela .....	39
Figura 65 Seleção de colunas visíveis .....	40
Figura 66 Formatos de exportação .....	41
Figura 67 Toolbar .....	41
Figura 68 Formulário de inserção .....	42
Figura 69 Obtenção dos dados preenchidos no formulário .....	42
Figura 70 Alerta de sucesso ao inserir .....	42
Figura 71 Código para pré preencher o formulário .....	43
Figura 72 Formulário de edição .....	43
Figura 73 Adição dos "id's" adicionar .....	44

Figura 74 Alerta de sucesso ao editar.....	44
Figura 75 Alerta de sucesso ao eliminar um equipamento.....	44
Figura 76 Estrutura do <i>modal</i> .....	45
Figura 77 <i>Modal</i> vista detalhada .....	45
Figura 78 Função de conversão dos caracteres especiais.....	46
Figura 79 <i>Modal</i> de Reciclagem .....	47
Figura 80 Função single select [38] .....	47
Figura 81 <i>Modal</i> de confirmação da restauração .....	48
Figura 82 Alerta de sucesso ao restaurar.....	48
Figura 83 Opções de exportação .....	49
Figura 84 Exemplo da utilização da função <i>data-field</i> .....	49
Figura 85 Exemplo de uma coluna invisível .....	49
Figura 86 Requisição de dados à API em PHP .....	50
Figura 87 Geração automática de colunas.....	50
Figura 88 Função para alterar os títulos das colunas.....	50
Figura 89 Aspeto gráfico global da página completa.....	54





## *Índice de Tabelas*

Tabela 1 Calendarização do Projeto.....	3
Tabela 2 GET vs. POST [23] .....	23
Tabela 3 Processos presentes por tabela.....	51
Tabela 4 Permissões de cada tipo de utilizador.....	51



## *Acrónimos*

AJAX	–	Asynchronous JavaScript and XML
API	–	Application Programming Interface
CRUD	–	Create, Read, Update and Delete
CSS	–	Cascading Style Sheets
DEE	–	Departamento de Engenharia Eletrotécnica
DOM	–	Document Object Model
HTML	–	HyperText Markup Language
ISEP	–	Instituto Superior de Engenharia do Porto
JSON	–	JavaScript Object Notation
LEEC	–	Licenciatura de Engenharia Eletrotécnica e Computadores
REST	–	Representational State Transfer
URL	–	Uniform Resource Locator



# 1. INTRODUÇÃO

Este projeto foi elaborado ao longo do 2º semestre, do 3º ano da Licenciatura de Engenharia Eletrotécnica e Computadores (LEEC), do Departamento de Engenharia Eletrotécnica (DEE), do Instituto Superior de Engenharia do Porto.

Neste capítulo é apresentado a contextualização, motivação, objetivos e organização do trabalho.

## 1.1. CONTEXTUALIZAÇÃO

Este projeto foi realizado no âmbito da unidade curricular PESTA (Projeto de estágio), nesse sentido foi do desejo do DEE (departamento de engenharia eletrotécnica) do ISEP (Instituto Superior de Engenharia do Porto) que este projeto fosse apresentado como proposta.

No arranque deste projeto, o DEE possuía já um protótipo de um sistema de inventário de equipamento, desenvolvido em PHP. Contudo a direção do DEE constatou que mesmo poderia beneficiar de algumas melhorias para se tornar ainda mais amigável.

O referido protótipo não apresenta uma separação estrita entre código da interface e código de acesso à base de dados, pelo que para facilitar futuros desenvolvimentos se decidiu dividir a aplicação em duas componentes:

- *Back end*, que fornece os serviços de acesso à base de dados através de uma API REST
- *Front end*, responsável pela interface com o utilizador.

O trabalho documentado neste relatório centra-se no desenvolvimento do *front end*.

É também importante referir que a realização deste projeto é apenas parte integrante da solução final pretendida, uma vez que o *software* que constitui o *back end* é da autoria do aluno Gonçalo Morais (1180823).

## **1.2. OBJETIVOS**

O processo de desenvolvimento do *front end* proposto revelou-se bastante desafiante não só pela falta de experiência na área, mas também pelo cumprimento de todos os objetivos propostos que este implicava.

Assim sendo os objetivos inicialmente impostos foram:

- Renovação do design gráfico da página, no sentido de tornar o website mais intuitivo e com uma interface mais em linha com os websites da atualidade
- Possibilidade de inserção, edição simples/múltipla e eliminação de registos
- Filtragem de dados
- Possibilidade de desfazer alterações, ou seja, quando um registo é alterado, os dados originais deveriam ser guardados para uma eventual reposição ou simplesmente para manter o histórico de alterações.
- Exportação de dados em diferentes formatos
- Visualização detalhada dos atributos de cada registo

## **1.3. CALENDARIZAÇÃO**

A organização e rentabilidade foi um processo chave para realizar este projeto com sucesso, de seguida é apresentada uma tabela onde é possível verificar a gestão temporal nas diferentes áreas.

Semana	21/03	28/03	04/04	11/04	18/04	25/04	02/05	09/05	16/05	23/05	30/05	06/06	13/06	20/06	27/06	04/07	11/07	18/07
Estudo das Tecnologias																		
Comparação Bootstrap vs Datatables																		
Estrutura e Design																		
Comunicação com API's																		
Formulários																		
Processos CRUD																		
Escrita do relatório																		
Testes																		
Apresentação, vídeo e poster																		

Tabela 1 Calendarização do Projeto

## 1.4. ORGANIZAÇÃO DO RELATÓRIO

O relatório é composto por cinco capítulos no total. No primeiro capítulo, introdução, são esclarecidas temáticas em relação ao contexto em que o projeto se insere, motivação e objetivos propostos.

Os três capítulos seguintes, 2, 3 e 4, são dedicados ao corpo deste projeto. No primeiro ponto deste apresentamos as diversas tecnologias utilizadas, bem como comparações entre alternativas que também foram consideradas. No capítulo seguinte, 3, é onde se concentra o principal elemento do corpo do projeto, nele figuram temáticas como mapa do website, design, páginas desenvolvidas entre outros. No capítulo 4 são apresentados os principais resultados obtidos ao longo dos diversos testes realizados.

O último capítulo, 5, é constituído pela conclusão, bem como propostas para que futuramente se possa melhorar a tecnologia.





## 2. TECNOLOGIAS

Neste capítulo são apresentadas todas as tecnologias que foram utilizadas ao longo do projeto e que permitiram a obtenção do resultado final tal como o conhecemos. Pontualmente são descritos pontos de debate travados aquando da escolha de uma tecnologia em detrimento de outra.

### 2.1. HTML

HTML (*HyperText Markup Language*) é uma linguagem utilizada em páginas web e aplicações que correm em browser. O hipertexto pode ser descrito como um bloco de texto que é usado como referência a outros textos, já a linguagem de anotação funciona como um estruturador geral da aplicação ou página criada.



Figura 1 Logótipo HTML 5

De um modo geral HTML não pode ser considerado uma linguagem de programação na sua génese [1], uma vez que não consegue criar funções dinâmicas como condições If/Else, ciclos de repetição, realizar eventos ou até mesmo manipular dados fornecidos [2]. Contudo é através da sua vertente estrutural que o HTML obtém o seu valor.

A funcionalidade de estrutura é obtida através das marcações em HTML, por outras palavras *tags*. Quando se deseja inserir novos conteúdos na página temos de abrir e fechar uma *tag* inserindo o conteúdo a exibir entre as mesmas [3].

```
1. <p>This is how you add a paragraph in HTML.</p>
```

Figura 2 Exemplo código HTML

## 2.2. CSS

A tecnologia CSS (Cascading Style Sheets) tem como principal função o controlo do aspeto gráfico da página [4].

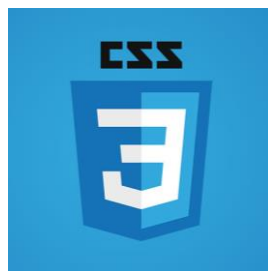


Figura 3 Logótipo CSS

Apesar de vulgarmente associada ao HTML, esta linguagem atua de forma independente em linguagens de anotação como neste caso acontece. A separação entre ambas facilita a manutenção de websites, a partilha de estilos entre diferentes páginas e grande versatilidade de adaptação a diferentes websites em contextos diversos [5].

Apesar de não atribuir novas funcionalidades à página o CSS é particularmente relevante no contexto de *front end*, área na qual este projeto se insere. A apresentação de informação num formato organizado ao utilizador é de máxima importância quando o objetivo reside em construir uma interface intuitiva e refinada.

Para intercalar um novo estilo numa página web é necessário abrir e fechar marcações da seguinte forma: `<style> </style>`. Contudo se quisermos referenciar um ficheiro de CSS

externo este terá de ser declarado no cabeçalho da página `<head>` através da anotação `<link>`.

```
1 <link rel="stylesheet" type="text/css" href="/css/style.css">
```

Figura 4 Link do ficheiro CSS externo

### 2.3. PHP

PHP (*Hipertext Preprocessor*) [6] é uma linguagem de programação *open-source* que é especialmente utilizada no desenvolvimento web. Através desta linguagem é possível criar trechos de código embebidos na estrutura da página de forma que este execute funções e tarefas que o HTML não consegue realizar. Diferente de outras linguagens que são utilizadas em desenvolvimento web, o PHP é executado no lado do servidor e desta forma o cliente apenas recebe os resultados do trecho de código executado.



Figura 5 Logótipo PHP

Para intercalar o PHP numa página web é necessário abrir e fechar uma instrução da seguinte forma.

```
1 <?php
2 //código
3 ?>
```

Figura 6 Abrir/fechar instruções em PHP

### 2.4. JAVASCRIPT

JavaScript é uma linguagem que proporciona dinamismo às páginas web. Ao contrário do HTML e do CSS o âmbito de aplicação desta linguagem não é ao nível estrutural, mas sim ao nível da dinâmica da página. Através da *framework* disponibilizada por esta linguagem é possível definir a reação a eventos que se realizam aquando do pressionar de um botão ou até mostrar conteúdo que se atualiza em intervalos de tempo definidos [7].



Figura 7 Logótipo JavaScript

Para intercalar um script numa página web é necessário abrir e fechar marcações da seguinte forma: `<script> </script>`. Contudo se quisermos referenciar um ficheiro de JavaScript externo teremos de incluir o caminho para este.

```
1 <script type="text/javascript" src="./js/instal.js"></script>
```

Figura 8 Link do ficheiro JavaScript externo

#### 2.4.1. JQUERY

O jQuery é uma biblioteca de JavaScript que facilita a escrita desta mesma linguagem. O principal objetivo desta *framework* é simplificar a sintaxe da linguagem, e assim obter um código mais simples e fácil de entender [8]. Esta alternativa foi utilizada sobretudo em pedidos efetuados às REST API's.



Figura 9 Logótipo jQuery

Para este caso é possível fazer download da ferramenta ou aceder à mesma através do URL do respetivo repositório, neste projeto optou-se pela segunda hipótese sendo posteriormente facilmente modificável.

#### 2.4.2. AJAX

Com vista a implementar uma solução dinâmica que ofereça satisfação na sua utilização é necessário recorrer algum mecanismo que execute pedidos assíncronos às diferentes API que servem aplicação para que esta permaneça em constante atualização.

O AJAX (*Asynchronous JavaScript and XML*) é um conjunto de técnicas de desenvolvimento web que visam a melhoria da experiência para o utilizador, essencialmente através da diminuição do número de vezes que a página tem que ser completamente recarregada, assim como da diminuição da quantidade de dados transferidos em cada interação com o servidor. O seu modo de atuação consiste em realizar um pedido assíncrono a um servidor web e atualizar o conteúdo apresentado na página sem que esta careça de atualização completa [9].

Para implementar uma técnica como esta são necessárias três ferramentas: HTML, DOM (*Document Object Model*) e JavaScript [10]. O DOM pode ser descrito como a estrutura da página HTML, este só é gerado após o carregamento total da página.

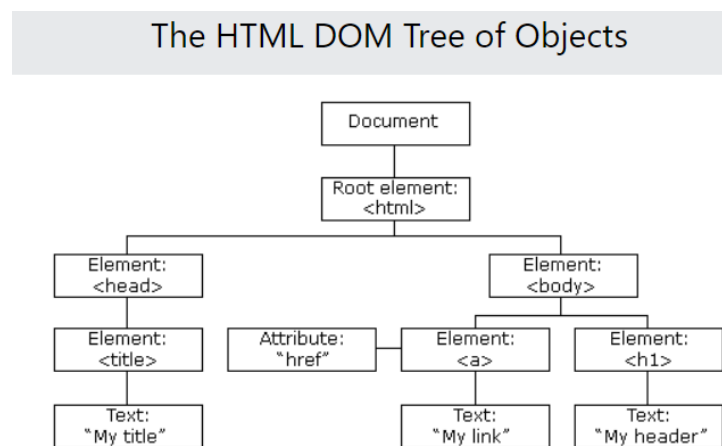


Figura 10 Exemplo DOM de uma página HTML [11]

Os pedidos apenas são efetuados após o carregamento completo da página em questão, seguindo os passos listados:

1. Um evento é gerado (carregar de um botão, carregamento da página)
2. Realiza-se um *XmlHttpRequest*, pedido assíncrono ao servidor.
3. O servidor processa a informação e envia a resposta
4. O cliente recebe a resposta
5. A DOM é atualizada

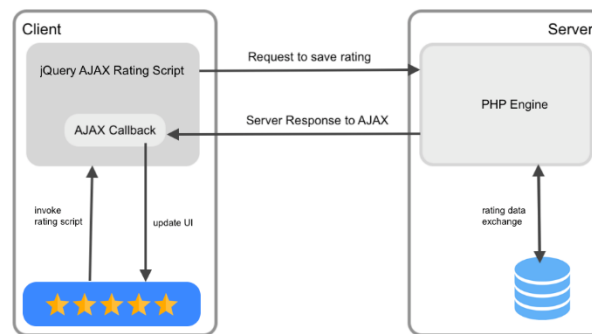


Figura 11 Esquema de funcionamento do AJAX [12]

É importante referir que a interface não fica bloqueada durante a realização das etapas acima referidas. Assim o AJAX confere algumas vantagens significativas nas aplicações em que esta figura, tais como:

- Menor ocupação de largura de banda
- Tempo de resposta da página inferior
- Experiência mais agradável ao utilizador

Embora esta técnica esteja disponível no JavaScript, decidiu-se recorrer à alternativa disponibilizada pelo jQuery devido à sua sintaxe mais facilitada.

Em detalhe uma função AJAX providenciada pelo jQuery, é composta por vários parâmetros que iremos passar a descrever [13].

- *Method*
  - Método pelo qual o pedido é realizado
    - GET para requisição de dados que aparecem nas tabelas e CRUD
    - POST para login
- URL
  - Contem o URL ao qual o pedido é efetuado
- *Data*

- Conteúdo enviado do cliente para o servidor, neste caso é sempre em formato JSON.
- *Datatype*
  - Formato do conteúdo em que a resposta é enviada, neste contexto é igual para todos os casos JSON.
  - No jQuery quando a resposta é processada em formato JSON esta é automaticamente convertida para um *object* facilmente manipulável em JavaScript.
- *beforeSend*
  - Permite customizar *headers*.
  - Em caso de retornar “*false*” o pedido será prontamente cancelado.
- *Success*
  - Esta função é executada em caso de sucesso do pedido, é no corpo desta função que procedemos à manipulação dos dados fornecidos.
- *Error*
  - Esta função é executada sempre que o pedido falhe, é no corpo desta função que iremos gerir os diferentes erros retornados.
  - Erro 401 (acesso não autorizado) [14], o utilizador é retornado para a página de login;
  - Erro na receção de dados para preencher a tabela

Na função *beforeSend* o *header Authorization* [15] é alterado no sentido de passar a conter o *token* de verificação do utilizador, para além disso é necessário fornecer o tipo de autenticação que usamos, *Bearer* [16], uma vez que estamos a utilizar um *token*.

```
$.ajax({
  method: 'GET',
  url: url + '/server/api_get_descricao Equipamento.php',
  data: {
    options: JSON.stringify({
      "coluna": "marca",
    })
  },
  dataType: "json",
  beforeSend: function(xhr) {
    xhr.setRequestHeader("Authorization", "Bearer " + "<?php echo $_SESSION['jwt'] ?>");
  },
  success: function(res) {
    //código executado em caso de sucesso
  },
  error: function(xhr, resp, text) {
    if (xhr.status == 401) {
      window.location.replace(url + "/includes/logout.inc.php?error=" + xhr.status + "&message=" + xhr.responseJSON.message);
    }
  }
});
```

Figura 12 Função AJAX genérica para os pedidos realizados no projeto

## 2.5. BOOTSTRAP

O Bootstrap é uma *framework* criada pela equipa do Twitter em meados de 2010 [17], o seu principal objetivo é facilitar a criação de páginas responsivas e facilmente adaptáveis a qualquer tipo de dispositivo. Para além de tudo isso esta *framework* possibilita uma personalização diversificada e fácil das mais diversas estruturas que oferece. Algumas das funcionalidades utilizadas foram os *modals* e botões.



Figura 13 Logótipo Bootstrap v.4

Atualmente o Bootstrap encontra-se na sua versão 5. Contudo, tendo em conta que esta versão ainda é recente, a base de apoio ao desenvolvimento não é ainda tão sólida como na versão 4, tendo-se então optado pelo uso da versão 4.

Adicionalmente, em testes preliminares, observaram-se alguns problemas de compatibilidade entre a versão 5 e a biblioteca Bootstrap tables (apresentada de seguida).

A utilização do Bootstrap pode ser feita de dois modos, local ou acesso ao repositório:

Local – O desenvolvedor necessita de realizar o download dos ficheiros para que posteriormente ao utilizar apenas seja necessário inserir o caminho em que este se encontra



no seu computador. No website da *framework* é possível verificar um separador onde são apresentados guias de como realizar o download e como utilizar a mesma.

Acesso ao repositório – Neste caso o desenvolvedor não poderá indicar o caminho do ficheiro na sua máquina, uma vez que não é efetuado o download, ao invés o desenvolvedor utiliza o URL fornecido pelo website que lhe indicará o repositório que deve aceder.

No desenvolvimento deste projeto optou-se por realizar o acesso ao repositório, contudo esta pode ser facilmente modificada.

### 2.5.1. *MODAL*

Os *modals* são estruturas disponibilizadas pelo Bootstrap [18], que têm bastante preponderância na aplicação desenvolvida. Estes funcionam como uma espécie de *pop-up*, e podem ser personalizados de acordo com a informação que queremos fazer aparecer. Ao longo do projeto iremos ver *modals* a serem utilizados como formulários, tabelas ou informação.

Assim como os *pop-ups* os *modals* não ocupam toda a página do website, e neste caso é importante notar que ao clicar nas zonas não abrangidas pelo elemento este irá prontamente fechar.

A localização e apresentação do *modal* merece também especial atenção, uma vez que sempre que este elemento aparece terá de ter máximo destaque e por isso optou-se que este fosse exibido no centro da página no maior formato disponibilizado pela *framework* que é o *large*.

Em termos de constituição a sua estrutura pode ser dividida em três grandes grupos: *header*, *body* e *footer*.

- *Header* é essencialmente composto pelo título e pelo botão fechar. Alternativamente ao método de fecho do *modal* acima apresentado existe também a possibilidade de dedicar um botão a esse efeito.
- *Body* é onde irá ser apresentada a informação que queremos mostrar, irá variar de acordo com os exemplos já anteriormente referidos.

- *Footer* é um conjunto de botões ou botão que efetua na maior parte dos casos o objetivo desejado, finaliza a ação.

## 2.6. BOOTSTRAP TABLES

O Bootstrap Tables [19] pode ser considerado uma *framework* para tabelas dentro do pacote principal que é o Bootstrap, contudo estes podem ser usados em separado.



Figura 14 Logótipo Bootstrap Tables

Neste projeto foram utilizados ambos em simultâneo para criar uma experiência mais agradável durante a navegação e utilização da solução desenvolvida. A utilização desta *framework* é efetuada nos mesmos moldes que a sua progenitora.

Alternativamente ao Bootstrap Tables foi também utilizado em fase experimental o plugin DataTables.



Figura 15 Logótipo Data Tables

Esta segunda alternativa foi desenvolvida para funcionar em consonância com o jQuery [20], contudo apresenta algumas limitações.

- Sintaxe complexa
- Design desconectado
- Informação disponível

Apesar desta ser uma alternativa bastante fiável e com bom desempenho, a sintaxe fica aquém do seu concorrente, apresentando uma curva de aprendizagem mais acentuada para o primeiro contacto com a tecnologia, como foi o caso.

O segundo recai sobre o design da página como um todo. Sendo esta solução um plugin para tabelas não oferece suporte para a customização dos botões que a acompanham. Desta forma iria criar a ideia de uma página desconectada em termos de design ao invés da vista harmoniosa que se pretende.

O terceiro ponto prendesse com a disponibilidade da informação. Após alguma pesquisa em fóruns e outras fontes durante a fase de comparação, a informação disponibilizada revelou-se bastante escassa face ao seu concorrente.

## 2.7. FONT AWESOME

O Font Awesome [21] é um conjunto de ícones que pode ser usado em qualquer área do website. Neste projeto obtive a sua utilização nos mais diversos botões criados.



Figura 16 Logótipo Font Awesome

Através deste conjunto de ícones personalizados foi possível criar botões apelativos e auto descritivos, que trazem uma elegância e destaque a esta área de grande relevância da página.

Para que este possa estar disponível no projeto apenas precisamos de inserir no *header* da página o *script* que contem o *link* da biblioteca.

```
1 <script src="https://kit.fontawesome.com/f9194420a8.js" crossorigin="anonymous"></script>
```

Figura 17 Link de referência ao Kit de ícones

# 3. ESTRUTURA

## 3.1. MAPA DO WEBSITE

De modo a facilitar a compreensão do documento, elaborou-se um mapa do website onde é possível verificar a organização dos diferentes ficheiros.

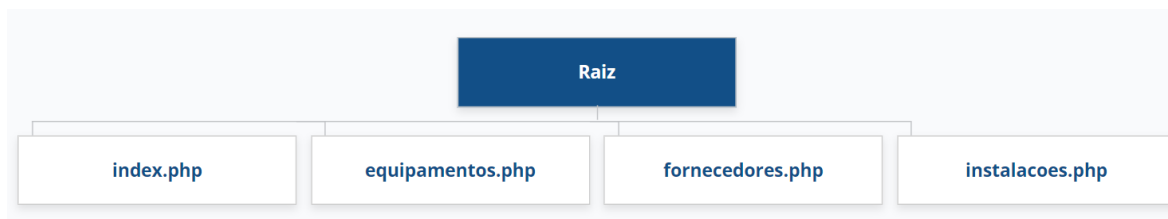


Figura 18 Mapa do Website parte 1

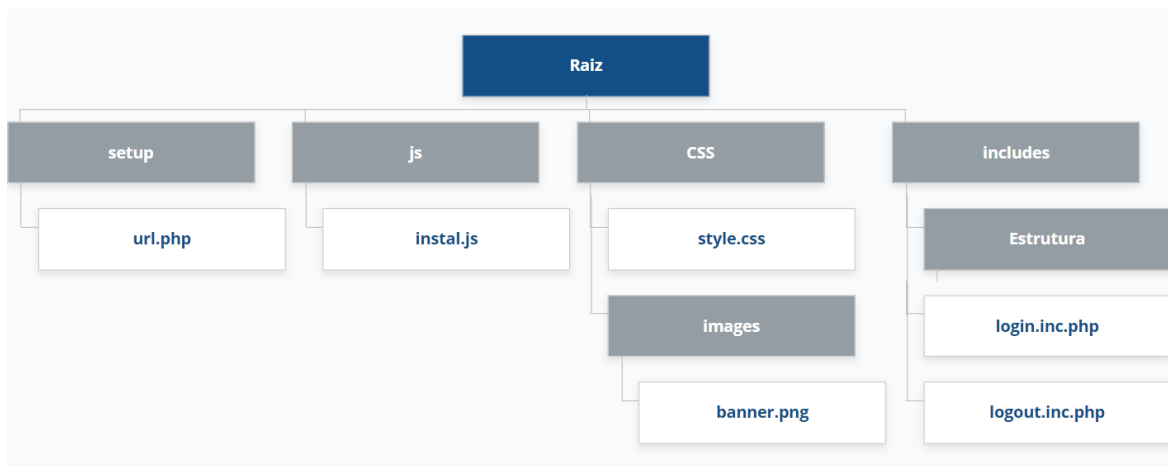


Figura 19 Mapa do Website parte 2

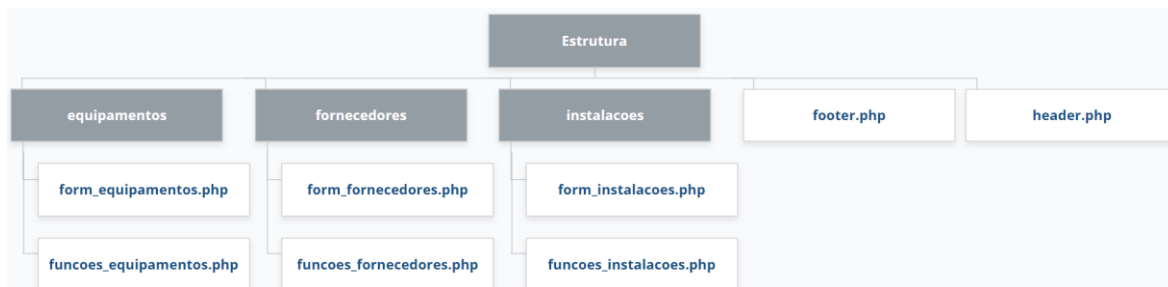


Figura 20 Mapa do Website parte 3

### 3.2. DESIGN

Ao definir a estrutura do *front end*, procurou-se seguir a estrutura do website do DEE. De um modo geral este incorpora três zonas distintas que albergam todo o *front end* desenvolvido, sendo elas a de entrada, o menu lateral e o corpo da página.

The screenshot displays the ISEP website interface. At the top, there is a blue banner with the ISEP logo and the text 'Instituto Superior de Engenharia do Porto' and 'DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA'. Below the banner is a navigation bar with 'INÍCIO' and 'BEM-VINDO PESTA'. On the left, there is a sidebar menu with options: 'Acesso rápido para:', 'LOGOUT', 'Equipamentos', 'Fornecedores', 'Instalações', and 'Ligações Úteis'. The 'Equipamentos' option is selected. The main content area shows a table of equipment with columns: 'Nº instalação', 'Tipo equipamento', 'Marca', 'Modelo', and 'Características'. The table contains several rows of equipment data.

Nº instalação	Tipo equipamento	Marca	Modelo	Características
F404	Multímetro digital portátil	Kiotto	KT-1000H	
F404	Osciloscópio digital	Sefram	5362 DC	2 x 60Mhz
F302	Computador		HP	
F302	Monitor		Haier	
F302	TV		LG	
F303	Computador		HP	
F303	Monitor		Haier	
F303	TV		LG	
F304	TV		LG	

Figura 21 Estrutura da página web

A entrada é constituída por um *banner* Figura 23 de entrada alusivo ao departamento, que dita a largura máxima da página, ou seja, todos os conteúdos posteriormente adicionados terão de estar horizontalmente alinhados com este. De seguida, mas ainda dentro da zona de entrada encontramos uma barra cujo conteúdo é inerente a hiperligações para outros websites do departamento, seguido de uma segunda barra que separa o conteúdo de entrada do corpo da página e o menu lateral. A segunda e terceira zonas são verticalmente iguais, mas horizontalmente diferentes. Através de um rácio 30/70 o menu lateral ocupa a zona de menor dimensão e neste temos os diferentes separadores que o utilizador pode visitar dentro do website. Já o corpo da página ocupa o restante espaço horizontal disponível, é então nesta área que se encontra o único conteúdo que sofre alteração durante a navegação dentro da página Figura 22.

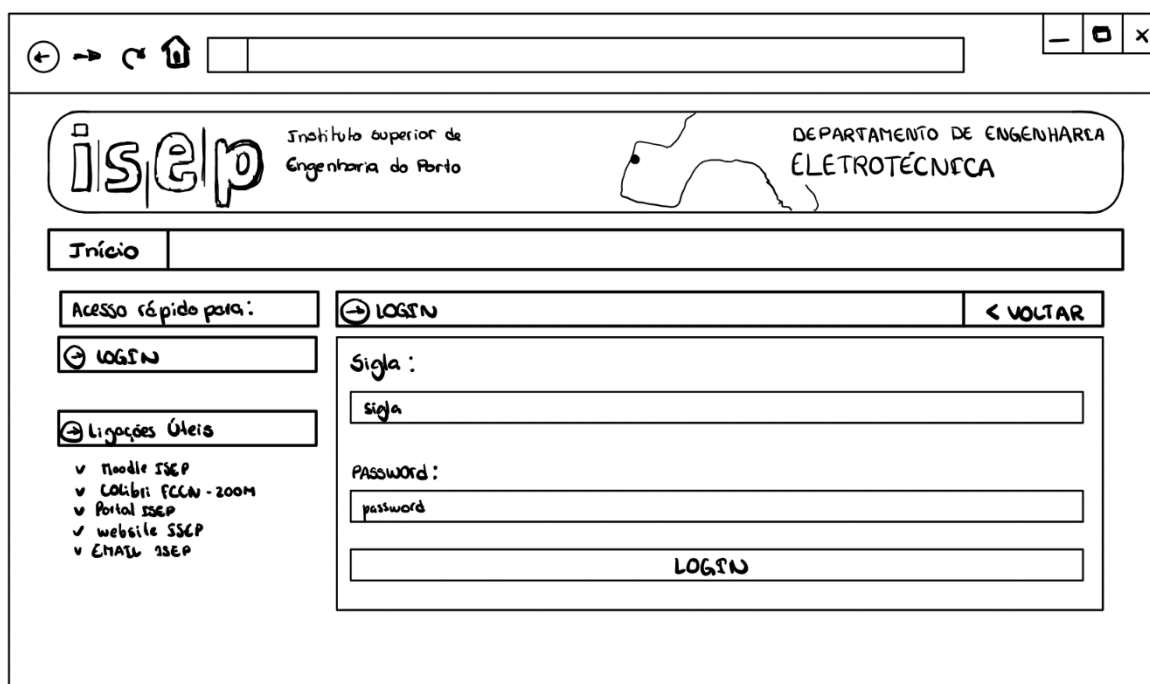


Figura 22 Esboço do output desejado

No contexto deste projeto o CSS revelou ser uma ferramenta muito poderosa uma vez que permitiu o desenvolvimento de uma página na qual se pudesse enquadrar nos requisitos definidos pelo orientador e pelas normas de estética do departamento. Em termos de paleta a sua panóplia de cores está reduzida apenas três, atuando como principal o azul (#124f87), secundária o cinzento (#949ca4) e por último, mas não menos importante o branco (#ffffff) que é sobretudo utilizado na cor de texto e ícones nas diferentes barras e menus desenvolvidos.



Figura 23 Banner DEE

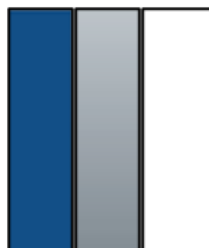


Figura 24 Azul (#124f87), Cinza (#949ca4), Branco (#ffffff)

Posteriormente cores adjacentes como o vermelho (#dc3545, #f7c5c9) e o verde (#198754, #a6e7b8) foram também utilizadas no âmbito de exibir mensagens ao utilizador.



Figura 25 Código de cores adjacentes: #dc3545, #f7c5c9, #198754, #a6e7b8

É importante ainda referir que todas as imagens, paleta de cores, formato e todo o conteúdo inerente ao design do website foram fornecidos no início do projeto pelo orientador.

Dado que a cor primária utilizada pelo Bootstrap não é consonante com a tonalidade de azul adotado pelo website, foi necessário alterar a mesma recorrendo ao CSS para esse efeito.

### **3.3. PÁGINA DE LOGIN**

A página de login é unicamente constituída por um formulário onde figuram dois campos de inserção sigla, palavra-passe e um botão para efetivar o login. Por vezes é necessário exibir mensagens em caso de erro ou sucesso, nesse caso estas irão aparecer situadas logo abaixo do formulário como é possível verificar na Figura 26.

Figura 26 Página de Login

### 3.4. PÁGINAS DESENVOLVIDAS

Para realizar este projeto foi necessário desenvolver três páginas: equipamentos, fornecedores e instalações. Apesar de nomes diferentes as funcionalidades implementadas entre elas são análogas.

Na sua génese cada página incorpora uma tabela com os dados referentes à página em questão. Agregada à mesma encontramos também dois conjuntos de botões localizados acima da tabela Figura 27.

O conjunto da esquerda dedica-se a operações CRUD (*Create, Read, Update and Delete*), já a da direita contém funcionalidades inerentes à tabela exclusivamente.



<div> <div> <div>+</div> <div></div> <div></div> <div></div> <div></div> </div> <div>Pesquisa</div> <div> <div></div> <div></div> <div></div> <div></div> </div> </div>				
ID	Características	Marca	Modelo	Local na sala
<input type="checkbox"/>				
<input type="checkbox"/> 673		lone		Bancada 3
<input type="checkbox"/> 674		NGS		Bancada 1
<input type="checkbox"/> 675		NGS		Bancada 2
<input type="checkbox"/> 676		NGS		Bancada 3
<input type="checkbox"/> 678	2 x 20Mhz	Kiotto	G-5020P	
<input type="checkbox"/> 679	2 x 30Mhz	Multimetrix	X03002	
<input type="checkbox"/> 680	2 Mhz	Topward	8110	
<input type="checkbox"/> 681	2 x 0-30V/2A + 5V/3A	Topward	6302A	
<input type="checkbox"/> 682	2 x 0-30V/3A + 5V/3A	Topward	6303A	
<input type="checkbox"/> 683	2 x 0-30V/3A + 5V/3A	Topward	6303D	

Mostrando 567 linhas

10 registros por página

< 1 2 3 4 5 ... 57 >

Figura 27 Tabela e estruturas auxiliares

Para facilitar o desenvolvimento do website foi necessário recorrer ao desmembramento da página em fragmentos de código que posteriormente podem ser acoplados a outras páginas que sejam criadas.

Como já foi aqui referido, existem zonas na página que não sofrem qualquer tipo de alteração aquando da navegação pelo website. Contudo não deixam de ser necessários de serem reescritos cada vez que uma nova página é criada.

Para este propósito foi utilizada a instrução `include_once` do PHP [22]. Esta função tem a capacidade de incluir o ficheiro especificado se este ainda não foi incluído na página onde esta instrução é invocada. Assim é possível criar um único ficheiro onde todas as zonas estáticas são declaradas e desta forma sempre que seja criada uma nova página é apenas necessário o desenvolvedor invocar o respetivo ficheiro através da função para essa finalidade.

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <div>Página de teste.</div>
</body>
</html>

```

Figura 28 Ficheiro sem desfragmentação

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>

```

Figura 29 *header.php*

```

</body>
</html>

```

Figura 30 *footer.php*

```

<?php
include_once "header.php";
?>

<div>Página de teste.</div>

<?php
include_once "footer.php";
?>

```

Figura 31 Ficheiro fragmentado

Com esta metodologia é possível albergar uma série de vantagens associadas:

1. Facilita o desenvolvimento e manutenção do sistema
2. Reutilização de código
3. Organização no ambiente de desenvolvimento
4. Ficheiros de menor dimensão e mais simples
5. Evita potenciais erros que poderiam advir da duplicação de código, garantindo maior robustez
6. Rentabilização de tempo

### 3.5. LOGIN

#### 3.5.1. COMUNICAÇÃO COM API DE LOGIN

Ao clicar no botão de login após inserir as credenciais de acesso no formulário destinado a este fim os dados são de seguida submetidos como parâmetros do script “*login.inc.php*”. Neste contexto utilizou-se o método POST em detrimento do GET, de forma a evitar a apresentação temporária dos dados de login na barra de endereços do browser. Na tabela seguinte é possível verificar as principais diferenças entre ambos os métodos.

	GET	POST
<b>Atualizar a página</b>	O pedido é reenviado, sem notificação ao cliente.	O pedido é reenviado, após a notificação e confirmação do utilizador.
<b>Histórico</b>	Dados permanecem guardados no histórico.	Os dados não são guardados.
<b>Restrições de extensão</b>	Dados enviados no URL (limitado a 2048 caracteres em alguns browsers)	Sem restrições
<b>Restrições no tipo de dados</b>	Apenas ASCII	Sem restrições
<b>Visibilidade</b>	Dados visíveis no URL	Dados não são apresentados

Tabela 2 GET vs. POST [23]

Após a receção das credenciais sigla e palavra-passe é então efetuado o pedido de um *token* de autorização usando a REST API do *back end*.

Em caso de sucesso é retornado um *token* que é usado para identificar o utilizador em cada pedido à REST API. Este *token* é decomposto [24] para obter nome e tipo de utilizador. Existem três tipos diferentes de utilizadores docente, técnico e administrador.

O nome é utilizado para mostrar uma mensagem de boas-vindas ao utilizador bem como mostrar menus personalizados ao nome do utilizador durante a navegação.

O tipo de utilizador é usado para seleccionar quais as funcionalidades que são disponibilizadas a cada utilizador.

Os detalhes da constituição e utilização do *token* são descritos na secção 3.5.3.

A biblioteca *libcurl* presente no PHP permite a conexão e comunicação com diferentes tipos de servidores utilizando os mais diversificados métodos [25]. Através das funções presentes nesta biblioteca foi estruturada uma função para obter a resposta da API.

Os conceitos para conceber esta função assentam na utilização de quatro funções presentes na biblioteca *libcurl*, como demonstra a Figura 32 [26].

- *curl\_init* [27]: Inicia uma sessão *curl*
- *curl\_setopt* [28]: Retorna os dados em *string*
- *curl\_exec* [29]: executa a sessão *curl* iniciada
- *curl\_close* [30]: Termina a sessão *curl*

```

$curl = curl_init();

curl_setopt($curl, CURLOPT_URL, $url);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);

$result = curl_exec($curl);

curl_close($curl);

```

Figura 32 Estabelecimento da comunicação com API de Login

As variáveis recolhidas no formulário do login são armazenadas na variável \$\_POST [31].

```

function CallAPI($url, $data = false)
{
    $curl = curl_init();

    curl_setopt($curl, CURLOPT_POST, 1);
    if ($data) {
        curl_setopt($curl, CURLOPT_POSTFIELDS, $data);
    }

    curl_setopt($curl, CURLOPT_URL, $url);
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);

    $result = curl_exec($curl);

    curl_close($curl);

    return $result;
}

$result = CallAPI($URL . '/server/api_login.php', '{"uname":"' . $_POST['uname'] . '","psw":"' . $_POST['psw'] . '"}');

```

Figura 33 Função completa com API de Login

A resposta do *back end* é codificada em formato JSON (JavaScript Object Notation), assim para que esta possa ser mais facilmente manipulada em PHP é realizada a conversão para um *array* PHP usando a função `json_decode` [24].

```

//apresentar os dados...
$json = json_decode($result, true);

```

Figura 34 Decomposição dos dados retornados

### 3.5.2. MENSAGENS

De seguida, a resposta retornada pelo *back end* é analisada com vista a avaliar se o pedido de login foi bem-sucedido. Esta filtragem é realizada do seguinte modo:

```

1 if ($json["message"] == "Login failed.") {
2 //código erro
3 }else{
4 //código sucesso
5 }

```

Figura 35 Estrutura identificadora de erros

Em caso de erro o utilizador é redirecionado para a página de login onde é visível uma mensagem com coloração vermelha que indica o motivo do erro.

```

if ($json["sigla"]) {
    echo '
    <script type="text/javascript" src="../../js/instal.js"></script>
    <script>
        window.location.replace(url + "?error=' . $mensagem . '" + "&type=credenciais");
    </script>';
}

if ($json["password"]) {
    echo '
    <script type="text/javascript" src="../../js/instal.js"></script>
    <script>
        window.location.replace(url + "?error=' . $mensagem . '" + "&type=password");
    </script>';
}

```

Figura 36 Tratamento dos erros retornados

São originados erros sempre que a sigla ou palavra-passe forem inválidas, sendo apresentadas as mensagens “Credenciais de acesso erradas” e “Palavra-passe errada”, respetivamente (Figura 37 e Figura 38).

**Credenciais de acesso erradas**

Figura 37 Mensagem de erro ao utilizar credenciais erradas

**Palavra-passe errada**

Figura 38 Mensagem de erro ao utilizar uma palavra-passe errada

Os erros são reconhecidos na página de login através do recurso à variável \$\_GET [32].

```

if (isset($_GET['error'])) {
    $a = $_GET['error'];
    if ($a == "Login failed.") {
        if ($_GET['type']=="credenciais"){
            echo "<div id='erro'>Credenciais de acesso erradas</div>";
        }
        if ($_GET['type']=="password"){
            echo "<div id='erro'>Palavra-passe errada</div>";
        }
    } else {
        echo "<div id='erro'>Tempo de sessão excedido</div>";
    }
}
}

```

Figura 39. Estrutura de exibição de erros na página principal

### 3.5.3. VARIÁVEIS GLOBAIS

Os *tokens* são compostos por três grupos delimitados pela *substring* “.”: *Header*, *Payload* e *Verify Signature* [33].

The image shows a web-based JWT token decoder. On the left, under 'Encoded', a long alphanumeric string is pasted. On the right, under 'Decoded', the token's structure is displayed in three sections: 'HEADER: ALGORITHM & TOKEN TYPE' showing 'typ': 'JWT' and 'alg': 'HS256'; 'PAYLOAD: DATA' showing issuer, audience, expiration, and user information; and 'VERIFY SIGNATURE' showing the HMACSHA256 algorithm and a checkbox for 'secret base64 encoded' which is checked.

Figura 40 Decomposição do *Token* nas suas três estruturas fundamentais

O nome e tipo de utilizador são armazenados no *Payload*, codificado em Base64 [34]. Para obter esses dados, essa secção do *token* é isolada e de seguida é feita a descodificação dos dados em base64 para variáveis de sessão PHP (Figura 41 e Figura 42).

```

$jwt = $json["jwt"];
$payload = explode(".", $jwt);
$decoded = urlsafeB64Decode($payload[1]);
$json_data = json_decode($decoded, true);

```

Figura 41 Aceder à estrutura "Data" do *Token*

```
session_start();
$_SESSION['jwt'] = $json["jwt"];
$_SESSION['nome'] = $json_data["data"]["nome_apelido_utilizador"];
$_SESSION['tipo'] = $json_data["data"]["tipo_utilizador];
```

Figura 42 Armazenamento em variáveis globais

De seguida o utilizador é redirecionado para a página principal onde poderá navegar pelo website livremente de acordo com o seu tipo de restrições.

A página para onde o utilizador é redirecionado irá reconhecer que o utilizador já se encontra autenticado, através do `session_start` Figura 43.

```
<?php
session_start();
?>

<!DOCTYPE html>
<html lang="pt">
```

Figura 43 Declaração das variáveis globais na página

Caso o utilizador não esteja autenticado e tente aceder alguma página apenas disponível após login este será imediatamente redirecionado para a página de login.

```
<?php
if (!isset($_SESSION['jwt'])) {
    header("location:../");
}
?>
```

Figura 44 Estrutura de verificação da autenticação

### 3.6. LOGOUT

A sessão mantém-se ativa até que as variáveis sejam libertadas (`session_unset()` [35]) e destruídas (`session_destroy()` [36]), o fragmento de código para esta finalidade encontra-se no “`logout.inc.php`”. Assim sendo este ficheiro será solicitado nos seguintes casos:

- Detecção de erro na verificação *token*



- O utilizador deliberadamente realizar *logout*

```
<?php
session_start();
session_unset();
session_destroy();

?>
<script type="text/javascript" src="../js/instal.js"></script>
<script>
    var url_atual = window.location.href;

    let params = (new URL(url_atual)).searchParams;

    if (params.has('error') === true) {
        window.location.replace(url + "?error=" + params.get('error') + "message=" + params.get('message'));
    } else {
        window.location.replace(url + "?success=logout");
    }
</script>
```

Figura 45 Ficheiro *logout.inc.php*

Os *tokens* têm um prazo de validade definido pelo *back end* na altura da sua criação. Caso esse prazo seja ultrapassado, o próximo pedido ao *back end* irá ser recusado. Nesse caso, o utilizador é reencaminhado para a página de login e é-lhe apresentada a mensagem "Tempo de sessão excedido" (Figura 46).

**Tempo de sessão excedido**

Figura 46 Mensagem de tempo de sessão excedido

A ação deliberada de realizar *logout* clicando no botão para essa finalidade irá redirecionar o utilizador para a página de login, porém uma mensagem de sucesso será exibida como a que se pode verificar Figura 47.

**Logout com sucesso.**

Figura 47 Mensagem de *logout* com sucesso

### 3.7. FORMULÁRIOS

A realização de formulários implica por vezes a utilização de diversas formas de inserção de dados, neste segmento explicamos o processo dinâmico de fazer aparecer ou esconder elementos dos formulários nas diferentes formas de inserção. Estes formulários serão utilizados no contexto das três páginas principais do *front end*: equipamentos, fornecedores e instalações.

### 3.7.1. CAIXAS DE SELEÇÃO

No contexto em que esta aplicação se insere dá-se o caso das opções presentes nas caixas de seleção serem provenientes de diferentes tabelas presentes no *back end* que visam auxiliar as tabelas que tomam o papel principal.

Para que as caixas de seleção sejam devidamente preenchidas sem que seja posta em causa a dinâmica e fluidez da página é necessário recorrer a um pedido AJAX, que em caso de sucesso irá fornecer as respetivas opções a serem agregadas à referida caixa de seleção.

- *createElement("option")* - criar uma nova opção
- *option.value* - valor da opção consonante com o id presente no *back end*
- *option.text* - texto apresentado ao utilizador proveniente do *back end*
- *appendChild(option)* - confirmar que a seleção é adicionada à seleção

```

success: function(res) {
    var valor = [];
    var texto = [];

    for (var i = 0; i < res.length; i++) {
        for (const key in res[i]) {
            if (key == "id_estado_equipamento" || key == "estado") {
                if (key == "id_estado_equipamento") {
                    valor.push(res[i][key]);
                }

                if (key == "estado") {
                    texto.push(res[i][key]);
                }
            }
        }
    }

    var select = document.getElementById("id_estado");

    var option = document.createElement("option");
    option.selected = "selected";
    select.appendChild(option);

    for (var a = 0; a < valor.length; a++) {
        var option = document.createElement("option");
        option.value = valor[a];
        option.text = texto[a];
        select.appendChild(option);
    }
},

```

Figura 48 Função de criação das opções para as caixas de seleção

### 3.7.2. DINÂMICA

Alguns dos registos de dados desta aplicação contêm atributos bastante extensos, que dificilmente poderiam ser apresentadas num único ecrã. De modo a tornar a visualização mais elegante optou-se por utilizar a ferramenta *collapse* [37] do Bootstrap que oferece uma forma fácil de esconder ou mostrar elementos através de um botão.

Formulário de inserção:

×

Tipo

▼

Adicione outro tipo...

Marca

▼

Adicione outra marca...

Modelo

▼

Adicione outro modelo...

Características

▼

Adicione outras características

Instalação

▼

Local

Armário, Bancada...

Fornecedor

▼

Estado

▼

▼

Guardar

Figura 49 Formulário colapsado

Formulário de inserção:

×

Tipo

▼

Adicione outro tipo...

Marca

▼

Adicione outra marca...

Modelo

▼

Adicione outro modelo...

Características

▼

Adicione outras características

Instalação

▼

Local

Armário, Bancada...

Fornecedor

▼

Estado

▼

▼

Custo

Data aquisição

dd/mm/aaaa

📅

Nº inventário DEE

Figura 50 Formulário expandido

A solução até aqui demonstrada seria suficiente se o utilizador se cingisse à escolha de opções predefinidas na base de dados. Contudo, verificou-se que era necessário permitir ao utilizador a introdução de novos tipos, marcas, modelos e características de equipamento (Figura 51).

Tipo	<input type="text"/>	<input type="text" value="Adicione outro tipo..."/>
Marca	<input type="text"/>	<input type="text" value="Adicione outra marca..."/>
Modelo	<input type="text"/>	<input type="text" value="Adicione outro modelo..."/>
Características	<input type="text"/>	<input type="text" value="Adicione outras características"/>

Figura 51 Inserção de novas alternativas às caixas de seleção

Para esconder ou fazer aparecer uma secção é necessário alterar a respetiva classe CSS. A deteção de alterações no elemento de entrada de dados é feita através de uma função que é executada a cada 100 ms.

```

1 function descricao() {
2   //código
3   repeater = setTimeout(descricao, 100);
4 }

```

Figura 52 Função de repetição (100ms)

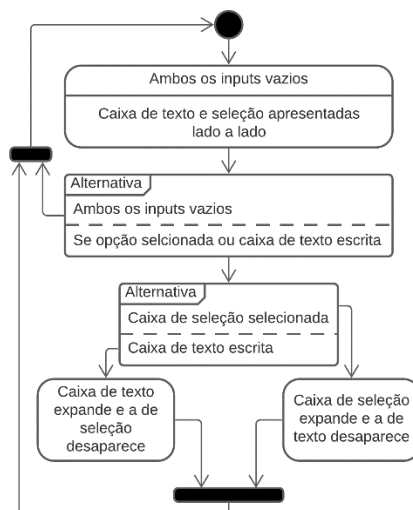


Figura 53 Esquema lógico implementado na função descrição

```

if ($('#marca option').filter(':selected').text() != "") {
    document.getElementById("marca_1").className = "col-sm-9";
    document.getElementById("marca_2").className = "d-none";
    document.getElementById('nova_marca').value = "";
    document.getElementById("nova_marca").required = false;
} else {
    document.getElementById("marca_1").className = "col-sm-4";
    document.getElementById("marca_2").className = "col-sm-5";
    document.getElementById("nova_marca").required = true;
}

```

Figura 54 Código utilizado para realizar o objetivo

### 3.7.3. FILTRAGEM

Um método de filtragem é uma característica muito importante e bastante requisitada em bases de dados que acomodam grandes conjuntos de dados. Filtrar o conteúdo duma caixa de seleção mediante uma escolha anterior facilita a experiência do utilizador na aplicação. Contrariamente ao que se sucede com as caixas de inserção, aqui o pedido só é efetuado caso o utilizador altere a opção que selecionou.

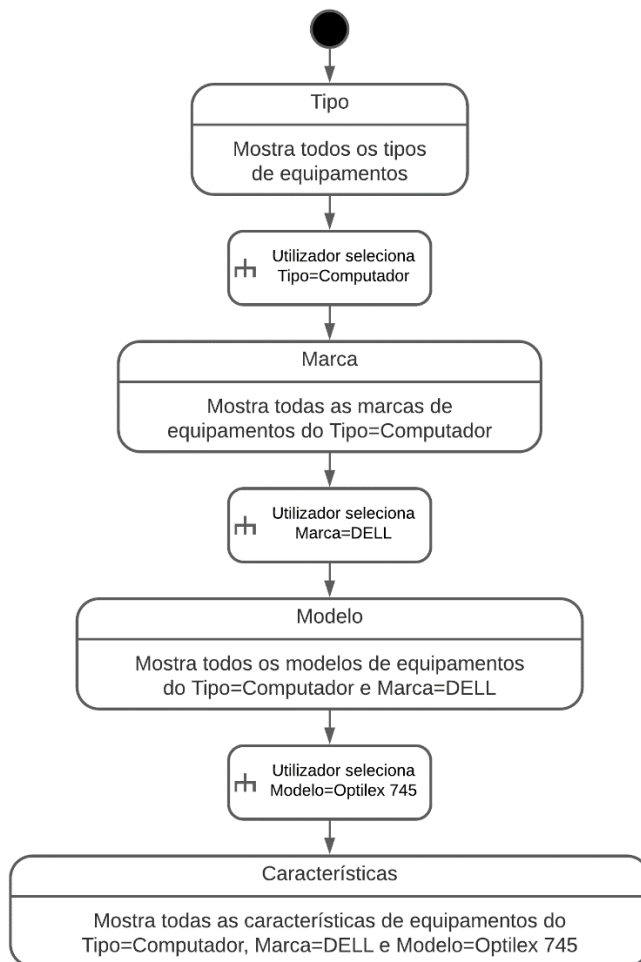


Figura 55 Esquema lógico de filtragem das descrições

O fragmento de código executado assenta num pedido AJAX. Para atualizar as opções nas caixas de seleção é necessário primeiramente remover as atuais para só depois introduzir as novas.

```

//ELIMINA AS OPÇÕES
function remove_options() {
    $("#id_tipo_equipamento").empty();
    $("#marca").empty();
    $("#modelo").empty();
    $("#caracteristicas").empty();
    $("#id_instalacao").empty();
    $("#id_fornecedor").empty();
    $("#id_estado").empty();
}
  
```

Figura 56 Eliminação das opções das caixas de seleção

As opções são atualizadas sempre que é aberto/fechado um formulário, se insere/edita com sucesso, carregue a página ou seja realizada uma ação de filtragem.

### 3.8. TABELAS

Todas as tabelas da página seguem as mesmas diretrizes de criação. Assim abriu-se as anotações HTML “*table*” (`<table></table>`) e definiu-se o id da mesma. Será através deste que a tabela se irá identificar. De seguida são descritos os restantes atributos do elemento “*table*” [38].

#### 3.8.1. ATRIBUTOS

- *class="table table-sm table-striped table-borderless"*

Os parâmetros definidos para este atributo são inerentes ao design geral da tabela, assim a seguinte lista é referente à nomenclatura utilizada no ficheiro CSS do Bootstrap Tables.

- *table* – Aplica a estética predefinida de qualquer tabela no Bootstrap Tables
  - *table-sm* – Reduz o tamanho das *padding*s das células
  - *table-striped* – Esquema de cores intercalado de branco e cinza-claro ao longo das linhas da tabela
  - *table-borderless* - Elimina as margens das células para criar uma visão mais limpa da tabela
- *data-toggle="table"*

Ativa a tabela sem a necessidade de escrita em JavaScript. Permite que a tabela seja formatada de acordo com os parâmetros definidos no ficheiro de JavaScript de tabelas do Bootstrap. Assim torna-se possível atribuir facilmente atribuir funcionalidades à tabela como paginação, mostrar/esconder colunas entre outras.

- *data-search="true"*

Habilita a existência de uma caixa de texto para realizar pesquisa na tabela.





Figura 57 Caixa de pesquisa

- `data-locale="pt-PT"`

Traduz para o idioma definido todas as mensagens apresentadas pela tabela bem como ferramentas que a auxiliam. Para o efeito deste projeto usamos a língua nativa do país da instituição.

- `data-pagination="true"`

Habilita a visualização do número de linhas por página, bem como a página em que o utilizador se encontra.

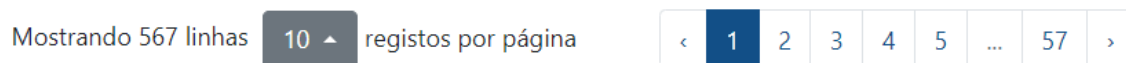


Figura 58 Estruturas de paginação

Adicionalmente, através de JavaScript são removidas informações como o número de linhas em cada página (Figura 59), mas que não se revelam interessantes para a solução final.

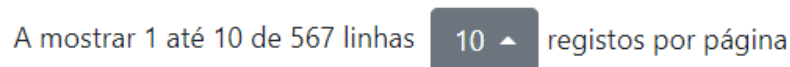


Figura 59 Informações predefinidas no Bootstrap

```
1 $('#table').bootstrapTable('refreshOptions', {  
2     paginationParts: ["pageInfoShort", "pageSize", "pageList"],  
3     exportTypes: ['json', 'xml', 'csv', 'txt', 'sql', 'excel', 'pdf']  
4 })
```

Figura 60 Função para alterar a paginação predefinida

- `data-filter-control="true"`

Esta característica permite criar caixas de pesquisa do tipo texto ou seleção para efetuar uma filtragem coluna a coluna.

	ID	Características	Marca	Modelo	Local na sala
<input type="checkbox"/>					

Figura 61 Filtragem por coluna

- *data-click-to-select="true"*

Permite habilitar a opção de seleção de linhas da tabela.

Com a seleção de uma linha é possível obter toda a informação nela contida. Este ato de selecionar uma linha ou várias precede vários eventos durante o uso da página, como por exemplo a edição, visualização detalhada, eliminação e exportação.

- *data-ajax="ajaxRequest"*

Neste segmento a função encarregue de fornecer dados à tabela é chamada.

Utilizando AJAX a função irá efetivar o pedido à API e em caso de sucesso retornar os dados inerentes à respectiva tabela Figura 62. Em caso de erro é apresentada uma mensagem de “Nenhum registo encontrado” Figura 63.

	ID	Características	Marca	Modelo	Local na sala
<input type="checkbox"/>					
<input type="checkbox"/>	673		Ione		Bancada 3
<input type="checkbox"/>	674		NGS		Bancada 1
<input type="checkbox"/>	675		NGS		Bancada 2
<input type="checkbox"/>	676		NGS		Bancada 3
<input type="checkbox"/>	678	2 x 20Mhz	Kiotto	G-5020P	
<input type="checkbox"/>	679	2 x 30Mhz	Multimetrix	X03002	
<input type="checkbox"/>	680	2 Mhz	Topward	8110	
<input type="checkbox"/>	681	2 x 0-30V/2A + 5V/3A	Topward	6302A	
<input type="checkbox"/>	682	2 x 0-30V/3A + 5V/3A	Topward	6303A	
<input type="checkbox"/>	683	2 x 0-30V/3A + 5V/3A	Topward	6303D	

Mostrando 567 linhas **10** registos por página

< 1 2 3 4 5 ... 57 >

Figura 62 Tabela preenchida

	ID	Características	Marca	Modelo	Local na sala
<input type="checkbox"/>					
Nenhum registo encontrado					

Figura 63 Tabela sem entradas

Barra de ferramentas da tabela:



Figura 64 Barra de ferramentas da tabela

- `data-show-refresh="true"`

Esta ação mostra um botão para atualizar a tabela, para evitar que o utilizador seja obrigado a recarregar a página inteira.

- *data-show-fullscreen="true"*

Este atributo cria um botão para poder ver a tabela e as suas barras de auxiliares em modo ecrã completo. Esta função acarreta grandes vantagens especialmente no contexto deste projeto, já que a área de trabalho (corpo da página) pode revelar-se pequena para tabelas de grande dimensão e através desta função este constrangimento deixa de existir.

- *data-show-columns="true"* e *data-show-columns-toggle-all="true"*

O primeiro destes dois atributos permite que o utilizador escolha quais as colunas que ficam visíveis ou escondidas na tabela, já o segundo atributo acrescenta uma camada extra a esta opção que permite fazer com que todas as colunas fiquem visíveis ou escondidas ao mesmo tempo.

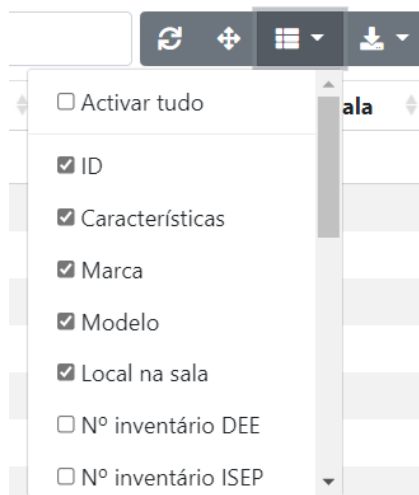


Figura 65 Seleção de colunas visíveis

- *data-show-export="true"*

Este atributo permite que o utilizador possa realizar a exportação da tabela em diferentes formatos: json, xml, csv, txt, sql, excel e pdf.

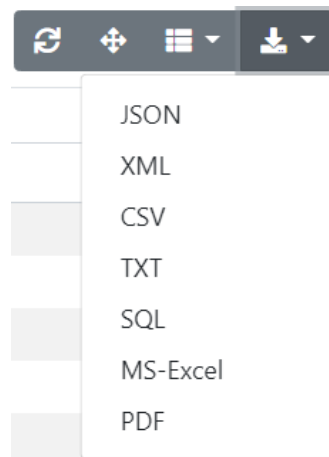


Figura 66 Formatos de exportação

- `data-toolbar="#toolbar"`

Este atributo indica à tabela que a anotação HTML com o `id="toolbar"`, contem um conjunto de botões ou outros eventos que devem ficar agregados à tabela do ponto de vista estético. Sendo que ficaram localizados no topo esquerdo da mesma, com a caixa de pesquisa e a barra de ferramentas da mesma a ocupar o canto superior oposto.



Figura 67 Toolbar

A descrição de cada botão é efetuada nas secções seguintes.

### 3.8.2. INSERÇÃO DE DADOS

O primeiro botão da barra de ferramentas permite inserir novos registos, através do preenchimento de um formulário contido num *modal*. Clicando no botão “guardar” este processo é finalizado.

Figura 68 Formulário de inserção

O envio da informação ao *back end* é efetuado através de um pedido AJAX em consonância com uma função que obtém os dados inseridos no formulário. Adicionalmente é também adicionado o tipo de dados que se pretende inserir.

```
event.preventDefault();
const data = new FormData(event.target);
const value = Object.fromEntries(data.entries());
var cart = [];
var id = [];

//ENVIO PARA A INSERÇÃO
value.table_name = "equipamentos";
cart.push(value);
```

Figura 69 Obtenção dos dados preenchidos no formulário

De seguida o utilizador recebe um alerta que lhe permite verificar se a inserção se deu com sucesso ou falhou.

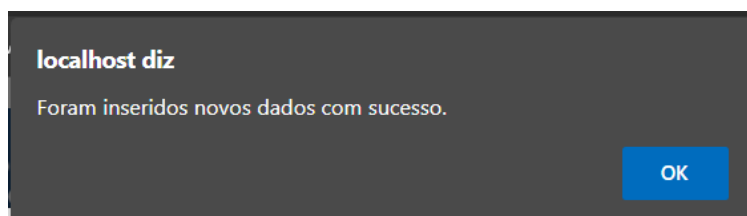


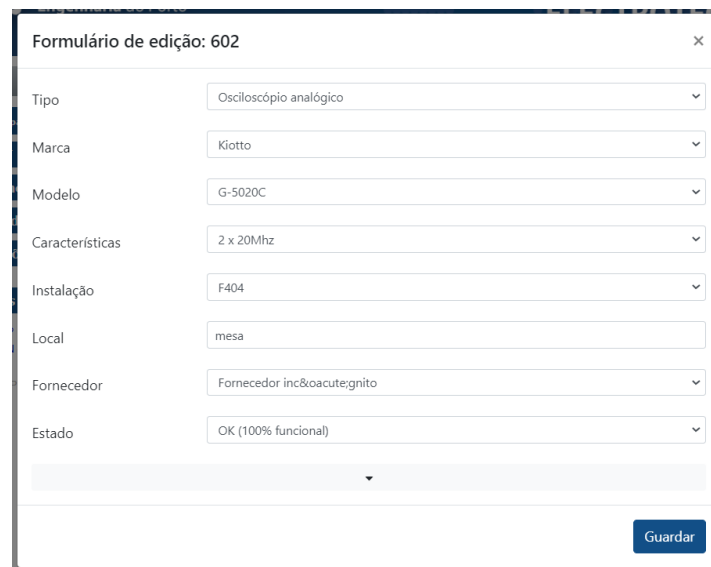
Figura 70 Alerta de sucesso ao inserir

### 3.8.3. EDIÇÃO DE REGISTOS

Tal como para o botão “Inserir”, o botão “editar” desencadeia a apresentação de um formulário numa janela *modal*. Contudo aquando do aparecimento do formulário este já se encontra preenchido com os dados que se encontram na linha seleccionada na tabela.

```
for (const key in resultado[0]) {  
  document.getElementById(key).value = resultado[0][key];  
}
```

Figura 71 Código para pré preencher o formulário



Formulário de edição: 602

Tipo	Osciloscópio analógico
Marca	Kiotto
Modelo	G-5020C
Características	2 x 20Mhz
Instalação	F404
Local	mesa
Fornecedor	Fornecedor inc&ocute;gnito
Estado	OK (100% funcional)

Guardar

Figura 72 Formulário de edição

No caso da página dos equipamentos, é possível editar vários equipamentos ao mesmo tempo. Nesse caso, os campos de inserção não irão aparecer pré preenchidos, sendo que apenas serão alterados os campos que forem preenchidos pelo utilizador no formulário.

A interação com o *back end* processa-se da mesma forma que na inserção. Contudo além de ser necessário indicar a tabela de destino, também é necessário fornecer as chaves (“id”) dos registos que são alterados.

```

//INDIVIDUAL
if (res.length == 1) {
    id.push(res[0].id_equipamento);
} else {
    //MÚTIPLA
    for (var i = 0; i < res.length; i++) {
        id.push(res[i].id_equipamento);
    }
}
value.id_equipamento = id;
value.table_name = "equipamentos";
cart.push(value);

```

Figura 73 Adição dos "id's" adicionar

De seguida o utilizador recebe um alerta que lhe permite verificar se a inserção se deu com sucesso ou falhou.



Figura 74 Alerta de sucesso ao editar

### 3.8.4. ELIMINAÇÃO DE REGISTOS

Para eliminar um ou mais registos é necessário primeiramente selecionar as linhas correspondentes da tabela e depois clicar no botão para esse efeito.

A função encarregue deste processo identifica as linhas que o utilizador pretende esconder, e assim como nos outros casos a informação é codificada no formato JSON sendo seguidamente enviada para a API correspondente através de um pedido AJAX.

De seguida o utilizador recebe um alerta que lhe permite verificar se a remoção se deu com sucesso ou falhou.

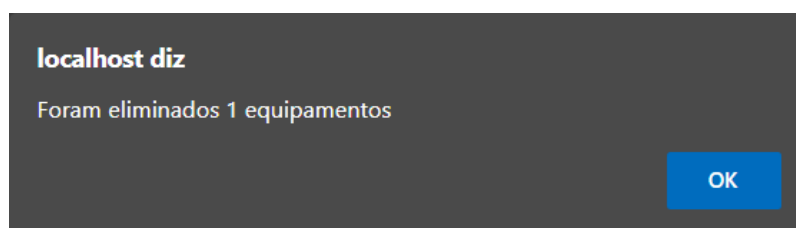


Figura 75 Alerta de sucesso ao eliminar um equipamento



### 3.8.5. VISTA DETALHADA

Esta propriedade permite que o utilizador possa visualizar todos os dados associados à linha selecionada na tabela. Esta informação é tratada e estruturada para encaixar num formato de duas colunas lado a lado num *modal*.

As colunas e linhas são criadas recorrendo à ferramenta *Grid* [39] do Bootstrap, que possibilita a criação de esquemas de página responsivos e organizados.

```
<div class="container">
  <div class="row" id="ficha">
    <div class="col-7" id="apresentar_dados"></div>
    <div class="col-5">
      <div class="row" id="foto_equipamento"></div>
      <div class="row no-gutters" id="sala_dados"></div>
    </div>
  </div>
</div>
```

Figura 76 Estrutura do *modal*

usando a *Grid*

<b>Ficha Detalhada do Equipamento: 681</b>	
<b>Características:</b> 2 x 0-30V/2A + 5V/3A	
<b>Marca:</b> Topward	
<b>Modelo:</b> 6302A	
<b>Nº inventário DEE:</b> DEE-FAL-023	
<b>Custo com IVA:</b> 0	
<b>Data aquisição:</b> 0000-00-00	
<b>Inserido em:</b> 2019-09-13 12:01:22	
<b>Inserido por:</b> JCPD	
<b>Última alteração em:</b> 2020-01-22 11:24:04	
<b>Última alteração por:</b> EACL	
<b>Estado:</b> OK (100% funcional)	
<b>Firma:</b> Fornecedor incógnito	
<b>Tipo :</b> Fonte de alimentação tripla	
	<b>Nº instalação:</b> F404
	<b>Nome da instalação:</b> Laboratório de Projectos
	<b>Director:</b> LBF
	<b>Técnico:</b> JCPD

Figura 77 *Modal* vista detalhada

Uma vez que a informação é gerada por JavaScript e segundo a pesquisa realizada não existe nenhum método *standard* para converter os caracteres especiais do HTML decidiu-se implementar uma função concebida para reconhecer estes caracteres e alterá-los.

```

function escapeHtml(str) {
    var str1;
    do {
        str1 = str;
        str = str.replace("&aacute;", "á");
        str = str.replace("&eacute;", "é");
        str = str.replace("&iacute;", "í");
        str = str.replace("&oacute;", "ó");
        str = str.replace("&uacute;", "ú");
        str = str.replace("&Aacute;", "Á");
        str = str.replace("&Eacute;", "É");
        str = str.replace("&Iacute;", "Í");
        str = str.replace("&Oacute;", "Ó");
        str = str.replace("&Uacute;", "Ú");
        str = str.replace("&agrave;", "à");
        str = str.replace("&egrave;", "è");
        str = str.replace("&igrave;", "ì");
        str = str.replace("&ograve;", "ò");
        str = str.replace("&ugrave;", "ù");
        str = str.replace("&Agrave;", "À");
        str = str.replace("&Egrave;", "È");
        str = str.replace("&Igrave;", "Ì");
        str = str.replace("&Ograve;", "Ò");
        str = str.replace("&Ugrave;", "Ù");
        str = str.replace("&acirc;", "â");
        str = str.replace("&ecirc;", "ê");
        str = str.replace("&icirc;", "î");
        str = str.replace("&ocirc;", "ô");
        str = str.replace("&ucirc;", "û");
        str = str.replace("&Acirc;", "Â");
        str = str.replace("&Ecirc;", "Ê");
        str = str.replace("&Icirc;", "Î");
        str = str.replace("&Ocirc;", "Ô");
        str = str.replace("&Ucirc;", "Û");
        str = str.replace("&atilde;", "ã");
        str = str.replace("&otilde;", "õ");
        str = str.replace("&Atilde;", "Ã");
        str = str.replace("&Otilde;", "Õ");
        str = str.replace("&ccedil;", "ç");
        str = str.replace("&Ccedil;", "Ç");
    }
    while (str1 != str);
    return str;
}

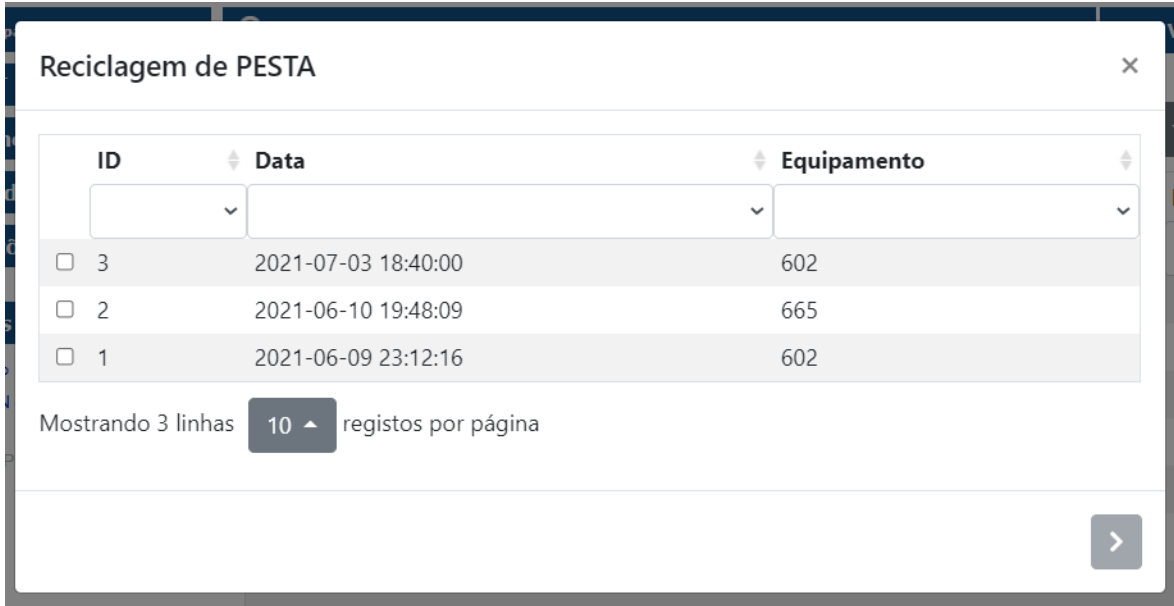
```

Figura 78 Função de conversão dos caracteres especiais

### 3.8.6. HISTÓRICO DE ALTERAÇÕES

Sempre que um elemento é editado, o *back end* guarda uma cópia do seu estado anterior. Desta forma, é possível ter um registo das alterações efetuadas e reverter alterações indesejadas.

Quando o utilizador carrega no símbolo de "desfazer", é-lhe apresentado uma lista de elementos anteriormente editados pelo mesmo, no estado anterior à alteração. Contudo um utilizador com privilégios máximo (tipo 0) não só pode ver as suas edições como também a dos outros utilizadores.



Reciclagem de PESTA

ID	Data	Equipamento
<input type="checkbox"/> 3	2021-07-03 18:40:00	602
<input type="checkbox"/> 2	2021-06-10 19:48:09	665
<input type="checkbox"/> 1	2021-06-09 23:12:16	602

Mostrando 3 linhas 10 registos por página

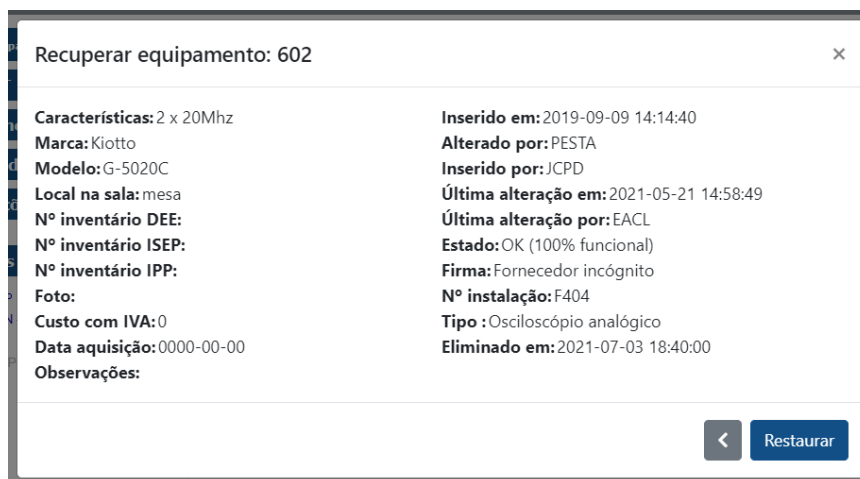
Figura 79 *Modal* de Reciclagem

Esta tabela apresenta características iguais às tabelas anteriormente descritas, porém neste caso é apenas possível selecionar uma única linha para que não aconteça que o utilizador tente restaurar duas entradas com o mesmo id ao mesmo tempo. Para esta finalidade é acrescentado o seguinte atributo:

```
data-single-select="true"
```

Figura 80 Função single select [38]

Após a seleção da entrada o utilizador pode avançar para a página seguinte, que apresenta uma vista detalhada estruturada em duas colunas lado a lado que contém os dados que o utilizador está a tentar restaurar. Esta página também recorre à função apresentada na Figura 76.



Recuperar equipamento: 602

<b>Características:</b> 2 x 20Mhz	<b>Inserido em:</b> 2019-09-09 14:14:40
<b>Marca:</b> Kiotto	<b>Alterado por:</b> PESTA
<b>Modelo:</b> G-5020C	<b>Inserido por:</b> JCPD
<b>Local na sala:</b> mesa	<b>Última alteração em:</b> 2021-05-21 14:58:49
<b>Nº inventário DEE:</b>	<b>Última alteração por:</b> EACL
<b>Nº inventário ISEP:</b>	<b>Estado:</b> OK (100% funcional)
<b>Nº inventário IPP:</b>	<b>Firma:</b> Fornecedor incógnito
<b>Foto:</b>	<b>Nº instalação:</b> F404
<b>Custo com IVA:</b> 0	<b>Tipo :</b> Osciloscópio analógico
<b>Data aquisição:</b> 0000-00-00	<b>Eliminado em:</b> 2021-07-03 18:40:00
<b>Observações:</b>	

<
Restaurar

Figura 81 *Modal* de confirmação da restauração

Para finalizar esta ação basta que o utilizador carregue no botão para restaurar para confirmar o pedido à API destinada e esta efetue a restauração.

De seguida o utilizador recebe um alerta onde poderá saber se a restauração se deu com sucesso ou falhou.

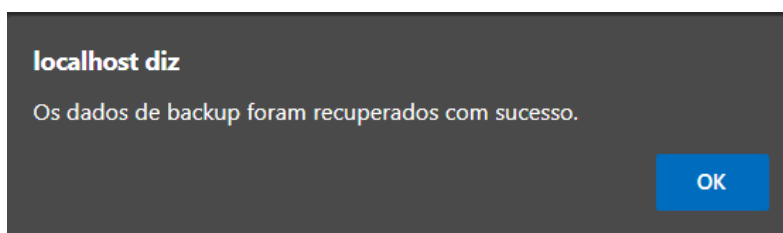


Figura 82 Alerta de sucesso ao restaurar

### 3.8.7. MODO DE EXPORTAÇÃO

Esta é a única opção da barra de ferramentas que não desencadeia nenhuma ação. Na verdade, este botão permite ao utilizador a seleção de três modos diferentes de exportação:

- Básico – Exporta as linhas que são visíveis na página
- Seleccionadas – Exporta as linhas seleccionadas

- Tudo – Exporta todas as linhas da tabela

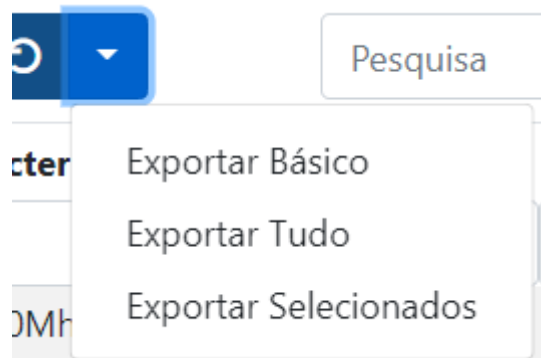


Figura 83 Opções de exportação

### 3.9. CRIAÇÃO DA TABELA

Através da função *data-field* [38] presente no Bootstrap Tables é possível definir para cada coluna qual o campo dos dados JSON que lhe é associado.

```

1 [
2 {
3   "id": "1",
4   "cor": "azul"
5 },
6 {
7   "id": "2",
8   "cor": "amarelo"
9 }
10 ]
11 <th data-field="id" data-filter-control="input" data-sortable="true">ID</th>
12 <th data-field="cor" data-filter-control="input" data-sortable="true">Tonalidade</th>

```

Figura 84 Exemplo da utilização da função *data-field*

Além disso é importante referir que os atributos *data-filter-control* e *data-sortable* [38] permitem que na coluna figure uma entrada dedicada à filtragem de dados e um organizador ascendente/descendente respetivamente.

Adicionalmente as colunas poderão incluir o atributo *data-visible="false"* [38], que por definição esconde a coluna aquando do carregamento da página. Contudo estas podem sempre passar a visíveis, uma vez que existe a possibilidade de mostrar/esconder colunas (Figura 65).

```

1 <th data-field="cor" data-filter-control="input" data-sortable="true" data-visible="false"></th>

```

Figura 85 Exemplo de uma coluna invisível

Visto que o pedido da lista de colunas ao *back end* é realizado em PHP, é utilizada uma função similar à da Figura 33 aquando do login.

Adicionalmente, foi criada um *header* dedicado ao envio do *token* Figura 86.

```
function CallAPI($url)
{
    $curl = curl_init();

    $authorization = "Authorization: Bearer " . $_SESSION["jwt"];
    curl_setopt($curl, CURLOPT_HTTPHEADER, array('Content-Type: application/json', $authorization));
    curl_setopt($curl, CURLOPT_URL, $url);
    curl_setopt($curl, CURLOPT_RETURNTRANSFER, 1);

    $result = curl_exec($curl);

    curl_close($curl);

    return $result;
}
```

Figura 86 Requisição de dados à API em PHP

Após a obtenção e tratamento dos dados é necessário fazer a construção da tabela da seguinte forma.

```
foreach ($json[0] as $key => $value) {
    if ($key == "numero_instalacao" || $key == "tipo_equipamento" || $key == "marca" || $key == "modelo" || $key == "caracteristicas") {
        echo '<th data-field="' . $key . '" data-filter-control="input" data-sortable="true">' . n_coluna($key) . '</th>';
    } else {
        if ($key == "id_equipamento" && $_SESSION['tipo'] == 0) {
            $name = str_replace("id_equipamento", "ID", $key);
            echo '<th data-field="' . $key . '" data-filter-control="input" data-sortable="true" data-visible="false">' . $name . '</th>';
        } else {
            echo '<th data-field="' . $key . '" data-filter-control="input" data-sortable="true" data-visible="false">' . n_coluna($key) . '</th>';
        }
    }
}
```

Figura 87 Geração automática de colunas

Tendo em conta a necessidade de obter uma apresentação cuidada é necessário alterar os nomes das colunas fornecidas para nomes mais intuitivos e fáceis de visualizar, esta tarefa é desempenhada pela função apresentada na Figura 88.

```
function n_coluna($key)
{
    $key1="";
    do {
        $key1 = $key;
        $key = str_replace("_", " ", $key);
        $key = str_replace("ris", "rís", $key);
        $key = str_replace("cao", "ção", $key);
        $key = str_replace("n inventario", "Nº inventário", $key);
        $key = str_replace("coes", "ções", $key);
        $key = str_replace("numero", "Nº", $key);
        $key = str_replace("ultima", "Última", $key);
        $key = ucfirst($key);
    } while ($key1 != $key);
    return $key;
}
```

Figura 88 Função para alterar os títulos das colunas

### 3.10. FUNCIONALIDADES DISPONÍVEIS EM CADA PÁGINA

Na tabela seguinte podemos verificar as funcionalidades disponíveis em cada página.

Funcionalidades/Página	Equipamentos	Fornecedores	Instalações
<b>Inserir</b>	✓	✓	✓
<b>Editar</b>	✓	✓*	✓*
<b>Eliminar</b>	✓	✗	✗
<b>Vista detalhada</b>	✓	✓	✓
<b>Modo de exportação</b>	✓	✓	✓

(\*) não podem realizar edições múltiplas

Tabela 3 Processos presentes por tabela

### 3.11. ACESSOS

Na tabela seguinte podemos verificar as funcionalidades autorizadas a cada tipo de utilizador.

Funcionalidades/Tipo	Administrador	Técnico	Docente
<b>Inserir</b>	✓	✓*	✗
<b>Editar</b>	✓	✓	✗
<b>Eliminar</b>	✓	✗	✗
<b>Vista detalhada</b>	✓	✓	✓
<b>Modo de exportação</b>	✓	✓	✗
<b>Coluna id's</b>	✓	✗	✗

(\*) não pode inserir instalações

Tabela 4 Permissões de cada tipo de utilizador

### 3.12. SETUP

Aplicação oferece a vantagem de rápida instalação em qualquer máquina para isso basta alterar a variável URL dentro dos ficheiros “*install.js*” e “*url.php*”. Esta variável contém o prefixo do URL dos serviços web fornecidos pelo *back end*.

1. Abra o ficheiro “*install.js*” localizado em `./js/install.js` Figura 19
2. Altere para o URL do servidor web
  - a. Ex: `var url = “http://localhost/tabela/server”;` (servidor local)
  - b. Ex: `var url = “https://direcao.dee.isep.ipp.pt/development/sigedee/API”;`  
(servidor final)
3. Abra o ficheiro “*url.php*” localizado em `./setup/url.php` Figura 19
4. Altere para o URL do servidor web
  - a. Ex: `$URL = “http://localhost/tabela/server”;` (servidor local)
  - b. Ex: `$URL = “https://direcao.dee.isep.ipp.pt/development/sigedee/API”;`  
(servidor final)



## 4. RESULTADOS

Após a realização de vários testes ao longo do projeto foi possível obter conclusões que consequentemente explicam algumas decisões tomadas.

A primeira destas prende-se com o método de filtragem nas colunas Figura 61. No software desenvolvido optou-se por usar caixas de entrada de texto, contudo a *framework* Bootstrap fornece uma segunda alternativa contendo listas de seleção. Através de algumas experiências foi possível averiguar que a opção de seleção se revelava bastante lenta no carregamento da página para tabelas cuja informação é bastante extensa, por conseguinte decidiu-se então recorrer ao método de filtragem por texto.

O tempo de resposta da aplicação foi sempre um tópico de grande ênfase desde o início deste projeto, graças às técnicas AJAX esta meta pode ser atingida com grande sucesso.

Durante o desenvolvimento do projeto foi também testado uma tecnologia de permuta da ordem das colunas nas três tabelas principais: equipamentos, fornecedores e instalações. Contudo a existência alguns problemas de compatibilidade com o resto da interface, levaram ao abandono desta alternativa.

É importante referir que todos os resultados/conclusões aqui apresentadas foram conseguidas utilizando como suporte um *back end* e hospedado no servidor final e o *front end* numa máquina distinta (servidor do ave) para que o ambiente de teste fosse o mais semelhante possível ao ambiente previsto para a utilização final.

**Formulário de edição: 665**

Tipo	Computador de mesa
Marca	DELL
Modelo	Optiplex 745
Características	Core 2 duo @ 1.86Ghz, 4GB RAM, 160GB HDD
Instalação	F404
Local	Bancada 1
Fornecedor	Fornecedor incógnito
Estado	OK (100% funcional)

**Guardar**

**Sidebar (Left):**

- INICIO
- Acesso rápido p...
- LOGOUT
- Equipam...
- Forneced...
- Instalaçã...
- Ligações
- ✓ Moodle ISEP
- ✓ Colibri FCCN
- ✓ Portal ISEP
- ✓ Website ISEP
- ✓ Email ISEP

**Table (Bottom):**

ID	Nome	Marca	Modelo	Características	Instalação	Local	Fornecedor	Estado
F404	Monitor LCD	DELL	E198FPb	17in, 1280x1024				

Figura 89 Aspeto gráfico global da página completa

# 5. CONCLUSÕES

Desde já queria destacar que este projeto foi o meu primeiro real contacto com desenvolvimento web aprofundado, uma vez que o assunto apenas tinha sido brevemente lecionado na unidade curricular de DEAPC da LEE.

Ao longo deste trabalho foram sendo discutidas e apresentadas as mais diversas soluções que visaram sempre a produção de uma tecnologia capaz de corresponder aos objetivos iniciais.

Sendo as operações CRUD o ponto principal de serviço, poderemos aqui destacar que a sua aplicação se revelou bastante desafiante, já que requereu aplicação de diversas tecnologias a funcionar em comunhão para poder criar a tão almejada simplicidade da interface.

Através dos testes efetuados no servidor final foi possível concluir que aplicação já se encontra suficientemente madura para o seu uso no quotidiano.

Dada a conjugação de todos estes acontecimentos estamos otimistas que esta solução possa desempenhar com sucesso todos os processos para os quais seja destacada e mais importante que contribua para o bom funcionamento do departamento para que este continue a evoluir como tem vindo a ser feito.

## 5.1. TRABALHO FUTURO

Como todas as tecnologias existe sempre espaço para a evolução e implementação de novas funcionalidades. Nesta perspetiva a solução aqui desenvolvida não será diferente.

Do ponto de vista do código desenvolvido e, dado que foi o meu primeiro contacto com a maior parte das tecnologias é possível que o código desenvolvido possa apresentar métodos mais complexos do que o que deveria. A simplificação ou mudança estrutural do código que sustenta este projeto pode trazer ainda mais aumentos na performance da mesma e até mesmo facilitar a sua manutenibilidade.

Outra perspectiva interessante de ser explorada seria a implementação de uma alternativa que não se necessita de passar de página em página e tivesse um comportamento mais próximo ao de uma aplicação móvel que apenas esconde ou mostra o conteúdo de acordo com a página. Esta solução pode ser alcançada facilmente através de outras *frameworks* como o React JS [40], Angular JS [41] ou Vue JS [42].

## REFERÊNCIAS DOCUMENTAIS

- [1] S. Curtis, “What Is HTML? Hypertext Markup Language Basics Explained,” Hostinger Tutorials, [Online]. Available: <https://www.hostinger.com/tutorials/what-is-html>. [Acedido em 1 7 2021].
- [2] B. Romy, “Why HTML is Not a Programming Language,” 5 4 2012. [Online]. Available: <https://ischool.syr.edu/why-html-is-not-a-programming-language/>. [Acedido em 1 7 2021].
- [3] MDN contributors, “HTML: Linguagem de Marcação de Hipertexto,” MDN Web Docs, [Online]. Available: <https://developer.mozilla.org/pt-BR/docs/Web/HTML>. [Acedido em 1 7 2021].
- [4] A. G., “O que é CSS? Guia Básico para Iniciantes,” Hostinger Tutorials, [Online]. Available: <https://www.hostinger.com.br/tutoriais/o-que-e-css-guia-basico-de-css>. [Acedido em 1 7 2021].
- [5] W3C, “HTML & CSS,” [Online]. Available: <https://www.w3.org/standards/webdesign/htmlcss.html>. [Acedido em 1 7 2021].
- [6] PHP, “What is PHP?,” [Online]. Available: <https://www.php.net/manual/en/intro-whatis.php>. [Acedido em 2 7 2021].
- [7] MDN Web Docs, “What is JavaScript?,” [Online]. Available: [https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First\\_steps/What\\_is\\_JavaScript](https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript). [Acedido em 5 7 2021].
- [8] w3schools, “jQuery Introduction,” [Online]. Available: [https://www.w3schools.com/jquery/jquery\\_intro.asp](https://www.w3schools.com/jquery/jquery_intro.asp). [Acedido em 5 7 2021].
- [9] V. H. L. Antunes, “Frontend Web 2.0 para Gestão de RADIUS,” Porto, 2009.
- [10] C. Draganova, *Asynchronous JavaScript Technology and XML (AJAX)*, Londres, p. 7.
- [11] w3school, “JavaScript HTML DOM,” [Online]. Available: [https://www.w3schools.com/js/js\\_htmlDOM.asp](https://www.w3schools.com/js/js_htmlDOM.asp). [Acedido em 10 7 2021].
- [12] Vincy, “Star Rating Script using PHP and MySQL with AJAX,” 4 2 2020. [Online]. Available: <https://phpspot.com/php/jquery-star-rating-script-using-php-and-mysql-with-ajax/>. [Acedido em 10 7 2021].
- [13] jQuery, “jQuery.ajax(),” [Online]. Available: <https://api.jquery.com/jquery.ajax/>. [Acedido em 27 7 2021].
- [14] MDN Web Docs, “401 Unauthorized,” MDN Web Docs, [Online]. Available: <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Status/401>. [Acedido em 8 7 2021].

- [15] MDN Web Docs, “Authorization,” [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Authorization>. [Acedido em 4 7 2021].
- [16] J. &. Hardt, “The OAuth 2.0 Authorization Framework: Bearer Token Usage,” 10 2012. [Online]. Available: <https://datatracker.ietf.org/doc/html/rfc6750>. [Acedido em 4 7 2021].
- [17] Bootstrap, “History,” [Online]. Available: <https://getbootstrap.com/docs/4.0/about/history/>. [Acedido em 5 7 2021].
- [18] Bootstrap, “Modal,” [Online]. Available: <https://getbootstrap.com/docs/4.0/components/modal/>. [Acedido em 10 7 2021].
- [19] Bootstrap Tables, “Introduction,” [Online]. Available: <https://bootstrap-table.com/docs/getting-started/introduction/>. [Acedido em 8 7 2021].
- [20] DataTables, “DataTables,” [Online]. Available: <https://datatables.net/>. [Acedido em 5 7 2021].
- [21] Font Awesome, “Icons,” [Online]. Available: <https://fontawesome.com/>. [Acedido em 9 7 2021].
- [22] PHP, “include\_once,” [Online]. Available: <https://www.php.net/manual/en/function.include-once.php>. [Acedido em 2 7 2021].
- [23] w3schools, “HTTP Request Methods,” [Online]. Available: [https://www.w3schools.com/tags/ref\\_httpmethods.asp](https://www.w3schools.com/tags/ref_httpmethods.asp). [Acedido em 5 7 2021].
- [24] PHP, “json\_decode,” [Online]. Available: <https://www.php.net/manual/en/function.json-decode>. [Acedido em 9 7 2021].
- [25] PHP, “Introduction,” [Online]. Available: <https://www.php.net/manual/en/intro.curl.php>. [Acedido em 2 7 2021].
- [26] Weichie, “cURL API calls with PHP, REST and JSON data (GET POST PUT DELETE),” 24 11 2020. [Online]. Available: <https://weichie.com/blog/curl-api-calls-with-php/>. [Acedido em 2 7 2021].
- [27] PHP, “curl\_init,” [Online]. Available: <https://www.php.net/manual/en/function.curl-init.php>. [Acedido em 2 7 2021].
- [28] PHP, “curl\_setopt,” [Online]. Available: <https://www.php.net/manual/en/function.curl-setopt>. [Acedido em 2 7 2021].
- [29] PHP, “curl\_exec,” [Online]. Available: <https://www.php.net/manual/en/function.curl-exec.php>. [Acedido em 9 7 2021].
- [30] PHP, “curl\_close,” [Online]. Available: <https://www.php.net/manual/en/function.curl-close>. [Acedido em 9 7 2021].
- [31] PHP, “\$\_POST,” [Online]. Available: <https://www.php.net/manual/en/reserved.variables.post>. [Acedido em 9 7 2021].

- [32] PHP, “\$\_GET,” [Online]. Available: <https://www.php.net/manual/en/reserved.variables.get.php>. [Acedido em 4 7 2021].
- [33] JWT, “JWT,” [Online]. Available: <https://jwt.io/>. [Acedido em 11 7 2021].
- [34] “What is base 64 encoding used for?,” stackoverflow, [Online]. Available: <https://stackoverflow.com/questions/201479/what-is-base-64-encoding-used-for>. [Acedido em 11 7 2021].
- [35] PHP, “session\_unset,” [Online]. Available: <https://www.php.net/manual/en/function.session-unset.php>. [Acedido em 4 7 2021].
- [36] PHP, “session\_destroy,” [Online]. Available: <https://www.php.net/manual/en/function.session-destroy.php>. [Acedido em 4 7 2021].
- [37] Bootstrap, “Collapse,” [Online]. Available: <https://getbootstrap.com/docs/4.0/components/collapse/>. [Acedido em 10 7 2021].
- [38] Bootstrap, “Table Options,” Bootstrap, [Online]. Available: <https://bootstrap-table.com/docs/api/table-options/#classes>. [Acedido em 27 7 2021].
- [39] Bootstrap, “Grid system,” [Online]. Available: <https://getbootstrap.com/docs/4.0/layout/grid/>. [Acedido em 10 7 2021].
- [40] React, “React,” [Online]. Available: <https://reactjs.org/>. [Acedido em 27 7 2021].
- [41] AngularJS, “AngularJS,” [Online]. Available: <https://angularjs.org/>. [Acedido em 27 7 2021].
- [42] Vue JS, “Vue JS,” [Online]. Available: <https://vuejs.org/>. [Acedido em 27 7 2021].
- [43] “Session Variables,” Middle Georgia State University, 13 11 2016. [Online]. Available: <https://itwebtutorials.mga.edu/php/chp8/session-variables.aspx>. [Acedido em 4 7 2021].
- [44] PHP, “curl\_error,” [Online]. Available: <https://www.php.net/manual/en/function.curl-error.php>. [Acedido em 9 7 2021].
- [45] PHP, “base64\_decode,” [Online]. Available: <https://www.php.net/manual/en/function.base64-decode.php>. [Acedido em 11 7 2021].