Relatório de Estágio

Mestrado em Engenharia Informática – Computação Móvel

Desenvolvimento de aplicações que marcam presença na *web*

Nuno Filipe Lopes Duarte

Leiria, *Março de 2016*

Relatório de Estágio

Mestrado em Engenharia Informática – Computação Móvel

Desenvolvimento de aplicações que marcam presença na *web*

Nuno Filipe Lopes Duarte

Estágio de Mestrado realizado sob a orientação do Doutor Sílvio Priem Mendes, Professor da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria

Leiria, *Março de 2016*

À Minha Família

Esta página foi intencionalmente deixada em branco

Agradecimentos

O trabalho que aqui se apresenta resultou de um percurso realizado com muito esforço e dedicação, na qual não era possível sem o apoio de vários amigos, colegas e companheiros.

Em primeiro lugar, quero agradecer à minha família que me deu oportunidade de estudar e de aumentar as minhas competências quer a nível pessoal quer a nível académico. Para além disto, também me proporcionaram todas as condições materiais e emocionais necessárias para que este percurso tivesse sucesso.

Agradeço também a todos os meus colegas pelos momentos de descontracção e pelo apoio, pois tornaram este percurso mais suave e divertido.

Quero agradecer ao Professor Sílvio Mendes, meu orientador, por me ter ajudado neste ano em todas as dificuldades que fui encontrando, e por me ter dado a conhecer a empresa onde trabalhei, pois foi nela que isto tudo se tornou possível.

Quero também agradecer ao Wagner Fernandes e Andreia Faísca da Empresa The Silver Factory por me terem guiado neste ano, e por me terem dado a conhecer o que é o mundo do trabalho, por me terem ajudado a ultrapassar todos os obstáculos que fui encontrando, pela paciência que tiveram e pelos bons conselhos que me deram.

Por fim quero agradecer à Escola Superior de Tecnologia e Gestão pelo facto de ter sido a minha instituição de escolha do meu percurso universitário académico.

Esta página foi intencionalmente deixada em branco

Resumo

O principal objectivo deste relatório é documentar a experiência como estagiário na empresa The Silver Factory – Marketing, Design e Gestão Comercial, Lda, no âmbito da unidade curricular de Estágio do Mestrado em Engenharia-Informática – Computação Móvel (MEI-CM).

Sendo este estágio uma parte integrante para a obtenção do grau de Mestre, foram postos em prática conhecimentos adquiridos ao longo do meu percurso universitário (Licenciatura e 1º ano de Mestrado) de modo a concluir com sucesso todos os projetos que me foram propostos.

Durante este estágio trabalhei na área da comunicação *online*, através do desenvolvimento de *websites* e auxiliando na gestão do alojamento dos clientes. O principal objetivo foi desenvolver um *backoffice* e os seus respetivos componentes. Dos vários componentes desenvolvidos os mais importantes foram o *Content Management System* que permite fazer a gestão do conteúdo apresentado em *frontend*, e o *Projetct Management Tool*, que permite a uma empresa gerir os seus clientes, e os projetos que desenvolve para os mesmos, com a possibilidade de visualização do progresso de desenvolvimento desses projetos utilizando um gráfico de Gantt interativo.

Palavras-chave: estágio, mestrado, comunicação online, backoffice, componentes

Esta página foi intencionalmente deixada em branco

Abstract

The main goal of this report is to document my experience as an intern in the company "The Silver Factory – Marketing, Design and Commercial Management, Lda", under the curricular Internship unit as part of the Master of Computer Science – Mobile Computing (MEI-CM). This internship allowed me to put in practice the knowledge acquired during the university course so that I could successfully complete the projects that where proposed to me. The main focus of the work developed was in the field of online communication. This includes the development of websites and the assistance of the client regarding hosts management. The main objective was the development of a backoffice and its underlying components. Of the many components developed, the most important were the Content Management System, that allows you to manage all the frontend content, and the Project Management Tool that allows a company to manage their clients as well as the projects developed for them, with the possibility to visualize the development progress of these projects using an interactive Gantt chart.

Key-Words: internship, masters, online communcation, backoffice, components

Esta página foi intencionalmente deixada em branco

Índice de Figuras

Índice de Listagens

Índice de Quadros

Esta página foi intencionalmente deixada em branco

Lista de Acrónimos

ACRÓNIMO	DESCRIÇÃO
AJAX	Asynchronous Javascript and XML
API	Application Programming Interface
CLI	Command Line Interface
CMS	Content Management System
CPU	Central Processing Unit
CRUD	Create, Read, Update e Destroy
CSRF	Cross-Site Request Forgery
CSS	Cascading Style Sheets
DOM	Document Object Model
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JPEG	Joint Photographic Experts Group
JWT	JSON Web Token
JSON	JavaScript Object Notation
MVC	Model View Controller
ORM	Object-Relational Mapping
PDF	Portable Document Format
PDO	PHP Data Object
PHP	PHP: Hypertext Preprocessor
PNG	Portable Network Graphics
REST	Representational State Transfer
REP	Robots Exclusion Protocol
SEO	Search Engine Optimization
SQL	Structured Query Language
URL	Uniform Resource Locator
XML	Extensible Markup Language
XSS	Cross-Site Scripting

Esta página foi intencionalmente deixada em branco

Índice

Esta página foi intencionalmente deixada em branco

1 - Introdução

O trabalho apresentado neste documento foi elaborado no âmbito do estágio de Mestrado em Engenharia Informática – Computação Móvel, na Escola Superior de Tecnologia e Gestão (ESTG) pertencente ao Instituto Politécnico de Leiria. O estágio decorreu no período compreendido entre 01 de Agosto de 2013 e 31 de Agosto de 2014 na empresa The Silver Factory, sediada em Leiria e foi realizado sob orientação do Professor Sílvio Mendes.

Este relatório surge como memória descritiva do estágio efetuado, tendo como objetivo dar a conhecer ao leitor, o trabalho realizado ao longo deste.

1.1 Enquadramento

A empresa The Silver Factory – Marketing, Design e Web e Gestão Comercial é uma agência de publicidade, sediada em Leiria, cujo posicionamento no mercado é o de estruturar, aconselhar e desenvolver soluções na área da publicidade e *marketing*. O papel que uma empresa como a The Silver Factory desempenha no mercado começa pela análise (compreensão) da posição atual que a marca cliente possui no mercado, traçando então uma estratégia de comunicação e *marketing*. Esta é geralmente iniciada pelo desenvolvimento da entidade corporativa da marca cliente, recorrendo para isso a diferentes suportes comunicacionais, tais como: brochuras, catálogos, *flyers*, anúncios de imprensa, páginas *web*, entre outros.

A empresa The Silver Factory encontrava-se deficitária no que se refere às soluções *web*, tendo até ao início deste estágio recorrido a *outsourcing*. Com a minha integração na empresa, enquanto estagiário, foi criado um departamento inteiramente dedicado à *web*, sendo que todos os projetos referidos no presente relatório foram inteiramente desenvolvidos por mim.

1.2 Motivação e objetivos

O que motiva o meu estágio na The Silver Factory, é dotar a empresa da capacidade de dar resposta às solicitações dos clientes na área da comunicação *online*, conseguindo desenvolver páginas *web* e ajudar na gestão de alojamentos dos clientes da empresa. A alta diversidade de áreas, objetivos e tipos de páginas *web* que as empresas necessitam obriga a que a The Silver Factory consiga implementar soluções estáveis e de continuidade, o que por *outsourcing* era impossível de garantir devido à grande dependência existente neste tipo de projetos.

Hoje em dia, a comunicação *online* é um fator importante para o setor empresarial. Esta permite uma grande expansão tanto a nível de credibilidade como de público-alvo a um custo bastante reduzido. Do mesmo modo, o uso de *smartphones*, *tablets* e computadores é cada vez mais comum, e a Internet é cada vez mais acessível em qualquer lugar e a qualquer momento, pelo que, uma posição confiável na Internet requer uma análise profissional para escolher o meio mais indicado, o tipo de página *web* mais indicado, a organização de conteúdos mais apropriada, o esquema de cores mais indicados, enfim o estudo completo da melhor solução para o problema que o cliente necessita de resolver.

O objetivo do estágio ficou definido com os seguintes tópicos:

- Desenvolver um *backoffice* e os seus respetivos componentes, principalmente o *Content Management System* (*CMS*), que tem como objetivo a simplicidade de gestão de conteúdos em *backend* e o desempenho na apresentação em *frontend;*
- Análise de requisitos com um *briefing* recolhido em reunião com o cliente;
- Desenvolver páginas *web* tendo por base o *backoffice* desenvolvido, respondendo às solicitações de cada projeto;
- Analisar o produto desenvolvido, executando testes de usabilidade e desempenho;
- Otimizar o produto para que seja o mais rápido possível e assim potenciar o regresso do utilizador.

1.3 Organização do documento

O documento encontra-se dividido em seis capítulos.

Capítulo dois (Tecnologias): Este capítulo faz referência às tecnologias abordadas e utilizadas no desenvolvimento dos vários projetos, tal como o Hypertext Markup Language (HTML), os Cascading Style Sheets (CSS), bem como a *framework* Laravel. Referencia também a principal linguagem de programação associada à *framework,* ou seja o PHP, PHP: Hypertext Preprocessor. É também feito uma referência à *stack web* utilizada para o ambiente de desenvolvimento.

Capítulo três (Introdução ao Desenvolvimento e Implementação): Este terceiro capítulo faz uma primeira introdução à metodologia abordada pela equipa de desenvolvimento, para a realização dos vários projetos, fazendo referência de seguida às duas principais linhas orientadoras de desenvolvimento. É feita ainda uma explicação detalhada do *backoffice* e dos seus componentes.

Capítulo quatro (Memória Descritiva dos Projetos Realizados): No capítulo quatro são apresentados de uma maneira simples e objectiva todos os projetos desenvolvidos, com referência para alguns *plugins* utilizados, e para o uso do *backoffice* e dos seus componentes.

Capítulo cinco (Validação dos Trabalhos): Neste capítulo, são explicados os critérios pela qual cada projeto tem de ser submetido, antes de ser dado como finalizado, e por conseguinte, ser colocado *online*. Faz também referência a testes de usabilidade e de desempenho.

Capítulo seis (Conclusão): Neste capítulo é descrita a conclusão do presente relatório.

2 - Tecnologias

Este capítulo apresenta as várias tecnologias utilizadas para o desenvolvimento dos vários projetos, abordando o HTML, os estilos CSS, e a linguagem de cliente JavaScript. Faz também referência à *stack web* utilizada por excelência, e aos componentes que a compõem. Por último, faz referência à principal linguagem de programação abordada juntamente com a *framework* utilizada.

2.1 HTML e CSS

O HTML [1], como o próprio nome indica, é uma linguagem de marcação que assenta no protocolo Hypertext Transfer Protocol (HTTP) [2] e que serve para criar e representar visualmente as páginas *web*. Estas páginas *web* são constituídas por *tags* HTML e pelo conteúdo específico da página, isto é, texto, imagens, etc. Estas *tags* HTML são elementos de marcação que definem como o *browser* deve formatar o conteúdo que será apresentado.

Todos os elementos estruturais de uma página HTML podem ser formatados com a utilização de CSS [3], que é uma linguagem de estilo.

A versão mais recente do HTML é o HTML5 [4] que tem algumas diferenças quando comparado com o HTML4, nomeadamente alterações a nível do *DOCTYPE* e ainda a introdução de novos elementos, quer a nível semântico (estrutural), a nível de formulários, a nível de grafismo e a nível de multimédia. Contudo, também houve elementos que foram removidos. Para além disso, ainda foram introduzidas novas API's (Application Programming Interface) como a "Geolocation", "Drag and Drop", "Local Storage", entre outras.

A nível de CSS, a versão mais recente é a CSS3 [5], que veio também trazer novas características a nível de *backgrounds* e *borders*, a nível de efeitos para os textos, novas transformações 3D, novas animações, entre outras.

2.2 JavaScript e jQuery

O JavaScript [6] é uma linguagem de programação interpretada, que é utilizada do lado do cliente e que permite ao utilizador interagir com a página *web*. Juntamente com o HTML e o CSS é uma das três tecnologias essenciais para a produção de conteúdo *web*. É suportado pelos *browsers* modernos sem recorrer à utilização de *plugins (add-on)*, ou seja, sem o uso de extensões do *browser*.

O jQuery [7] é uma biblioteca JavaScript que torna mais simples a manipulação dos documentos HTML, tratamento de eventos, animações, entre outros. Esta biblioteca JavaScript foi utilizada em todos os projetos realizados durante o estágio. Vai ser devidamente explicada no seu respetivo capítulo.

Foi também utilizado *Asynchronous JavaScript and XML (AJAX)* [8], uma técnica de comunicação assíncrona que é utilizada do lado do cliente que permite interagir com o lado servidor. De acordo com a Ilustração 1, temos que do lado do *browser* há um pedido enviado para o servidor, que depois este processa esse pedido e devolve uma resposta de novo para o *browser*. Assim que o *browser* recebe a resposta, a mesma é processada por forma a permitir a atualização dos dados que se encontram na página *web*, sendo para tal manipulado o Document Object Model (DOM).

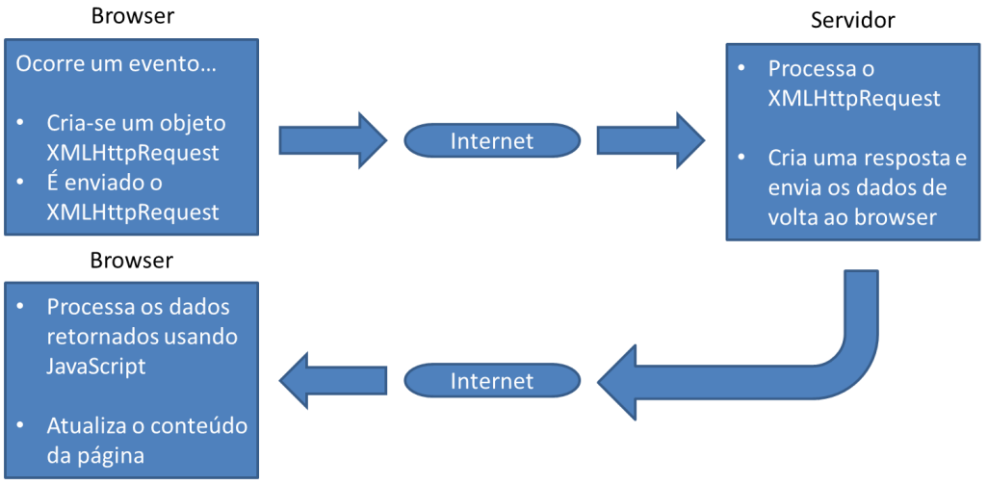


Ilustração 1 - Exemplo ilustrativo de um pedido AJAX (adaptado de [8])

Esta técnica foi utilizada em alguns projetos, nomeadamente para funcionalidades de envio de email, preenchimento dinâmico de *dropdowns* quando outra(s) era(m) clicada(s). Foi também utilizada em acessos à base de dados para as efetuar as várias ações Create Read Update Delete (CRUD).

2.3 PHP

O PHP [9] é uma linguagem de programação que é utilizada do lado do servidor, independente da plataforma e utilizada para gerar conteúdos dinâmicos. É ainda uma linguagem orientada a objectos, com uma elevada base de classes, rápida e com um conjunto elevado de bibliotecas. É utilizada sobretudo para desenvolvimento *web*.

Esta linguagem foi utilizada em todos os projetos desenvolvidos durante o estágio, mesmo aqueles que não utilizaram o *backoffice*, tendo sido mais explorada com o uso da *framework* Laravel, que será devidamente explicada no seu respetivo capítulo.

2.4 MySQL Workbench

O *MySQL Workbench* [10] é uma ferramenta para desenvolver Bases de Dados em MySQL [11], que permite a modelação de dados, desenvolvimento Structured Query Language (SQL) [12], e disponibiliza ferramentas administrativas para a configuração do servidor. Está disponível para *Windows*, *Linux* e Mac OS X, e é uma ferramenta bastante simples e intuitiva de utilizar.

2.5 *WampServer e phpMyAdmin*

O Wamp é uma *stack web,* utilizado no sistema operativo Windows, que é constituído por três componentes essenciais: o servidor Apache; o módulo de interpretação PHP e ainda o módulo de base de dados MySQL [13].

O módulo de interpretação PHP é o phpMyAdmin que permite facilmente gerir Bases de Dados através de uma interface gráfica [14].

2.6 Framework Laravel

A *framework* Laravel [15] é uma *framework open-source*, desenvolvida em PHP que está em constante desenvolvimento e disponibiliza uma boa documentação bem como uma vasta

comunidade de utilizadores. Foi utilizada para o desenvolvimento integral do *backoffice* que foi utilizado por alguns dos projetos desenvolvidos ao longo do estágio. Disponibiliza uma linha de comandos chamada Artisan CLI (Command Line Interface) com um elevado conjunto de comandos úteis que permitem automatizar tarefas associadas ao desenvolvimento da aplicação [16].

Esta *framework* utiliza o padrão de desenho Model-View-Controller (MVC) [17] que divide uma aplicação em controladores (*controllers*), o modelo de domínio (*model)* e as vistas (*view*).

O modelo é a camada responsável pela lógica do negócio, com as entidades e classes para o acesso à base de dados.

A vista é a representação visual do modelo, de acordo com o contexto. É normalmente o resultado que a *framework* apresenta no *browser,* sendo essencialmente constituído por código HTML.

O controlador é o elo de ligação entre o modelo e a vista. É responsável por processar a informação, obtida através do modelo, se for esse o caso, e decidir que ação deve ser efectuada, como por exemplo apresentar essa informação numa vista. Associado ao termo de vista, temos o Blade, que é um *template engine* fornecido pelo Laravel [18].

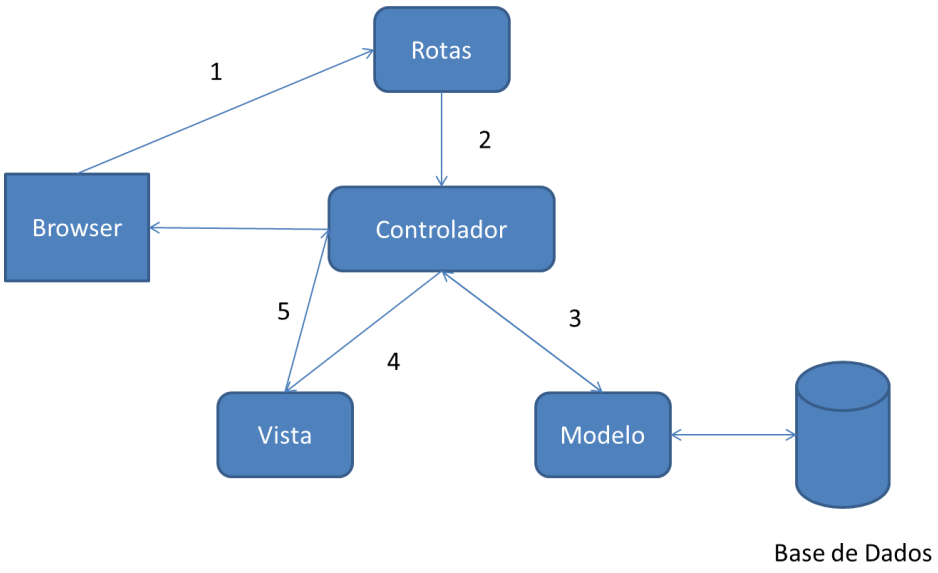


Ilustração 2 – Fluxo do padrão MVC com a *framework* Laravel

A Ilustração 2 faz referência ao fluxo do padrão de desenho MVC utilizado em associação com a *framework* Laravel. Deste modo, quando se utiliza uma aplicação desenvolvida com estra *framework*, o processo é iniciado quando o *browser* envia um pedido HTTP (1), que é recebido pelo servidor *web* e passado para o *routing engine* do Laravel. O *router* do Laravel recebe o pedido, e redireciona-o para o método do controlador apropriado, baseado no padrão de *routing* do Uniform Resource Locator (URL) (2).

De seguida, pode dar-se o caso de o controlador fazer logo o *render* da vista (HTML, CSS e imagens) que depois é enviado de volta para o *browser*. Contudo, no caso de *websites* que são constituídos por conteúdo dinâmico, o controlador interage com o modelo (3), que é um objeto PHP que representa uma entidade, e que é responsável por comunicar com a base de dados. Depois de comunicar com o modelo, o controlador faz então o *render* da vista *(4)* (HTML, CSS e imagens) e devolve a página *web* completa para o *browser (5)*.

Com este padrão tem-se então uma arquitetura bem definida que permite ao programador ter o código separado e bem estruturado, com a vantagem de permitir com facilidade, efetuar futuras alterações e adicionar novas funcionalidades.

Esta *framework* será tida em conta e em mais pormenor no seu respetivo capítulo, apresentado mais à frente no documento.

3 - Introdução ao Desenvolvimento e Implementação

Este capítulo faz uma introdução ao conceito de metodologia, abordando aquela que mais se enquadrou no desenvolvimento dos projetos. Apresenta também um cronograma associado ao estágio, com uma lista dos vários projetos desenvolvidos e uma aproximação ao seu tempo de duração. Por último, faz referência ao desenvolvimento de *websites* com e sem *backoffice* e aos vários componentes desenvolvidos que fazem parte desse mesmo *backoffice*.

3.1 Metodologia

Todas as empresas no actual mercado do mundo de trabalho são cada vez mais competitivas e têm o objetivo de gastar o menos tempo e dinheiro possível no desenvolvimento dos seus produtos de *software*. Mas para assegurar a qualidade dos produtos e ter ciclos de entrega rápidos é necessário usar uma metodologia de desenvolvimento [19]. Existem vários tipos de metodologias de desenvolvimento, entre elas as metodologias ágeis e outras mais clássicas, como o desenvolvimento em cascata, ou o tradicional desenvolvimento sequencial.

A entidade de acolhimento, The Silver Factory, não tinha qualquer abordagem ao uso de metodologias para a realização de projetos, pelo que de acordo com a constituição da equipa de trabalho (gestor de projetos, *web designer/designer* gráfico e programador), e ainda de acordo com as características de cada projeto, os métodos de trabalho eram definidos *ad-hoc*, ou seja, exclusivos para cada projeto, seguindo geralmente sempre o mesmo fluxo. Ainda assim, estes métodos estavam mais enquadrados numa abordagem do tipo Agile.

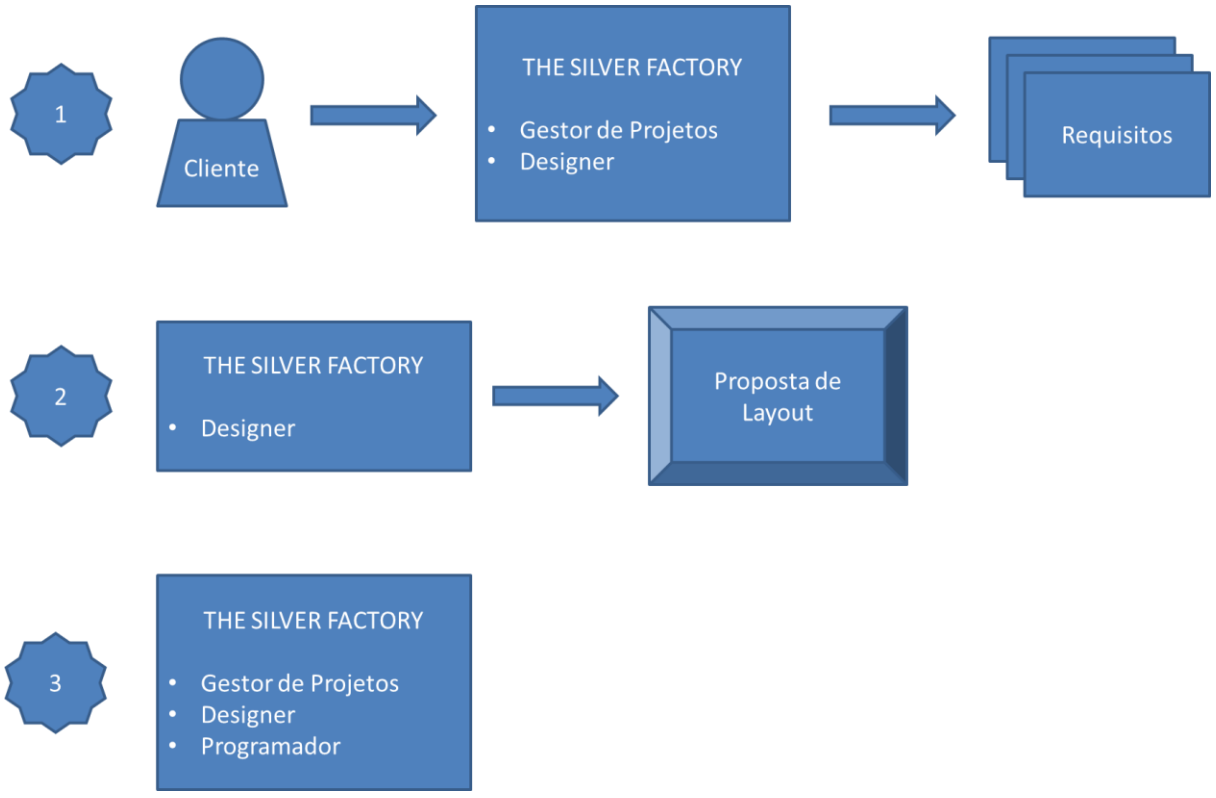


Ilustração 3 - Processo de aceitação e desenvolvimento de um novo projeto

De acordo com a Ilustração 3, o processo de desenvolvimento de um novo projeto inicia com uma primeira abordagem por parte do cliente à empresa, havendo uma primeira reunião entre o mesmo, o gestor de projetos e, neste caso, a *web designer*. Nesta primeira reunião é discutido o trabalho a ser realizado, onde é efetuada uma análise de requisitos, percebendo todas as características e funcionalidades do projeto (1). A próxima fase do trabalho é assumida pela *web designer,* que desenvolve uma proposta de *layout*, que depois é apresentada numa nova reunião. Nesta reunião são discutidos pormenores do *layout* (2). Caso a proposta seja aceite, há uma nova e última reunião, neste caso interna e sem a presença do cliente. Nesta reunião o programador também está presente, onde lhe é então explicado todo o trabalho a ser desenvolvido, abordando todas as funcionalidades, tanto a nível de *frontend* como de *backend*, e é ainda definido um tempo de realização do mesmo (3).

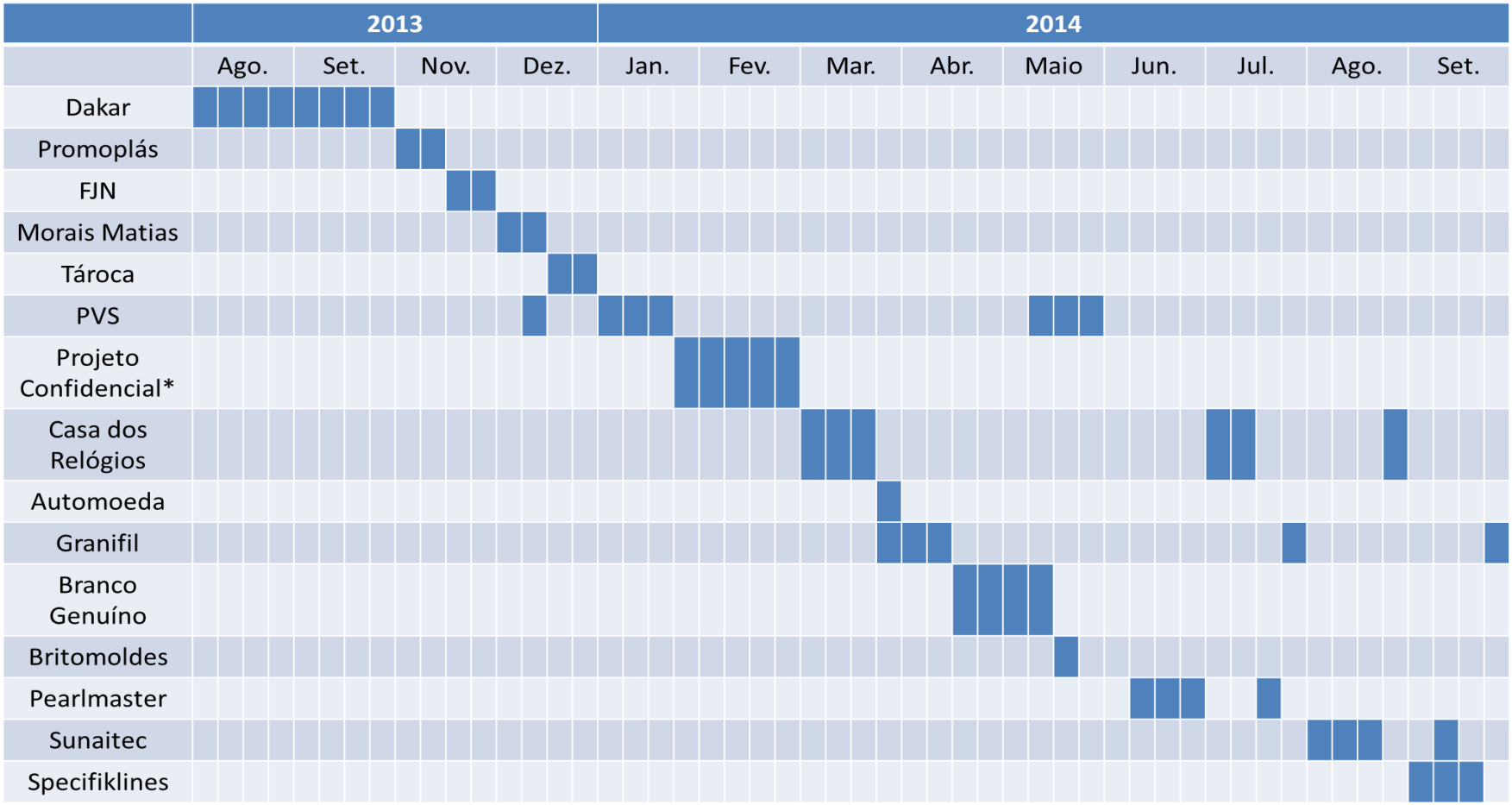
A metodologia Agile é constituída por um conjunto de vários princípios [20] que foram seguidos durante a realização dos vários trabalhos. Dos vários princípios, há a necessidade de satisfazer o cliente através de um processo de entregas rápidas do *software*. No caso do estágio, todos os dias, ao final do dia, o trabalho realizado era atualizado no servidor do cliente ou no da empresa. Deste modo, o cliente poderia esta a par do progresso de desenvolvimento do seu projeto.

Este tipo de metodologia prevê também alterações de funcionalidades durante o desenvolvimento, que por sua vez, aconteceram em alguns projetos, onde o cliente pedia uma nova funcionalidade ou até mesmo a alteração de uma já implementada. Houve também um importante trabalho de equipa entre os vários elementos responsáveis pelo projeto. Este trabalho de equipa é também um princípio defendido pela metodologia. Por último, defende também o facto de que a melhor maneira de fazer chegar a informação a uma equipa de desenvolvimento é pessoalmente, e de facto, era desta maneira que toda a informação era passada pelos vários elementos da equipa de desenvolvimento em questão.

Existem várias metodologias ágeis apresentadas em [21], das quais a SCRUM [22], é que mais se enquadra na metodologia utilizada no desenvolvimento dos vários projetos.

3.2 Cronograma

Tabela 1 - Cronograma



* Por motivos confidenciais não me é permitido revelar o verdadeiro nome deste projeto, contudo é também apresentado no cronograma por ter sido desenvolvido por mim durante a realização do estágio.

Para uma melhor perspetiva dos vários projetos desenvolvidos, bem como a sua duração, é apresentado um cronograma, de acordo com a Tabela 1. Neste cronograma, cada rectângulo preenchido, corresponde sensivelmente a uma semana de trabalho na qual houve uma envolvência da minha parte no respetivo projeto.

3.3 Linhas Orientadoras de Desenvolvimento

No desenvolvimento integral de *websites* e dentro do ambiente de trabalho onde estava integrado, foram abordadas duas principais linhas orientadoras. O desenvolvimento de *websites* sem a utilização de *backoffice*, e o desenvolvimento de *websites* com a utilização de *backoffice*.

3.3.1 Websites sem backoffice

Os *websites* sem *backoffice* são essencialmente desenvolvidos em HTML. As várias páginas são constituídas por conteúdo estático, que muito esporadicamente é atualizado. Neste exemplo de *websites* todo o tempo utilizado para o seu desenvolvimento é dedicado exclusivamente à construção do *layout* das páginas e aos vários tipos de animações, caso estas existam. Todos os *websites* desenvolvidos durante o estágio tinham um *layout* único, apelativo e devidamente pensado, de maneira a oferecer ao utilizador uma melhor *user experience*.

3.3.1.1 HTML, CSS e Introdução ao Responsive

Como referido acima, estes *websites* são desenvolvidos utilizando essencialmente a linguagem de marcação HTML e pelos respetivos estilos de CSS. Relacionado com o desenvolvimento de *websites*, surge o conceito de *responsive*, ou seja a capacidade de um *website* se ajustar à dimensão dos vários dispositivos de onde é acedido. Dos vários dispositivos que existem, como *smartphones*, *tablets*, computadores portáteis e de mesa, há uma característica importante a ter em consideração para o conceito de *responsive,* e que é diferente em todos eles: a dimensão do ecrã. Como tal, é necessário que o *layout* das várias páginas *web* seja alterável de acordo com o tamanho dos dispositivos e das suas capacidades [23].

Em primeiro lugar é necessário definir um *viewport*. O *viewport* controla como uma página *web* é apresentada nos dispositivos móveis. Sem o *viewport*, os dispositivos móveis fazem *render* da página para uma largura de computador portátil ou de mesa, de maneira a preencher todo o ecrã [24].

Esta configuração é feita na cabeça (*head*) do documento, através da *tag* HTML apresentada na Listagem 1:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

Listagem 1 - Meta Tag referente ao viewport

O uso do valor “width=device-width” instrui a página a fazer com que a largura do ecrã coincida em *pixeis* com a largura do dispositivo. Isto permite à página refluir o conteúdo para coincidir com os diferentes tamanhos dos ecrãs. O atributo “inicial-scale=1” instrui o *browser* para estabelecer uma relação de 1 para 1 entre os *pixeis* do CSS e os *pixeis* do dispositivo, independentemente da orientação, e permite à página aproveitar toda a largura horizontal.

Ainda associado ao conceito de *responsive*, temos as CSS Media Queries [25], que foram adicionadas no CSS3. Estas Media Queries permitem que o conteúdo apresentado seja adaptado para uma gama específica de dispositivos, sem ser necessário alterar o conteúdo em si.

```
@media (max-width: 767px) {

}
```

Listagem 2 - Exemplo de uma media query

As Media Queries são definidas nos ficheiros de CSS e um exemplo de uma delas é apresentado pela Listagem 2. Neste caso, quando a largura do ecrã for inferior a 767 *pixeis*, todas as regras que se encontrarem no interior da *query* serão executadas, seguindo as regras normais de cascada.

Com o intuito de desenvolver projetos para marcarem presença na *web,* oferecendo uma melhor *user experience*, temos o conceito de animações. Uma animação no mundo dos CSS permite animar transições de um estilo de configuração de CSS para outro. Estas animações são essencialmente constituídas por dois componentes, o estilo que descreve a animação CSS (Listagem 3) e um conjunto de *keyframes* [26] que indicam o início e o fim do estado da animação, bem como *waypoints* intermediários (Listagem 4) [27].

```
.da-slide-fromright .da-img-botas{
    animation: fromRightAnim4 0.6s ease-in 0.8s both;
}
```

Listagem 3 - Exemplo de um classe CSS com a utilização de uma *keyframe*

```
@keyframes fromRightAnim4{
    0%{ left: 110%; opacity: 0; }
    100%{ left: 60%; opacity: 1; }
}
```

Listagem 4 - Exemplo de uma keyframe

Cada *keyframe* descreve como é que o elemento que vai ser animado se deve comportar a uma altura específica da sequência da animação. De acordo com a Listagem 4, a percentagem de 0% indica o primeiro momento da sequência da animação, com a propriedade de opacidade definida a 0. A percentagem de 100% indica o estado final da animação, com a alteração do valor da propriedade de opacidade de 0 para 1.

3.3.1.2 Bibliotecas JavaScript

Como linguagem de cliente, para o desenvolvimento dos vários trabalhos foi utilizado o JavaScript e algumas das suas bibliotecas, tal como o jQuery e o Modernizr [28].

O jQuery foi utilizado tanto para as mais simples manipulações de elementos, como para os mais complexos *plugins,* ou extensões jQuery. Juntamente com as propriedades CSS, o jQuery permite criar os mais diversos tipos de animações e despoletar ações interativas que dão ao utilizador uma maior *user experience*. Destaca-se essencialmente pela utilização de selectores [29] e da utilização de uma menor quantidade de código para o mesmo resultado quando comparado com a utilização apenas de JavaScript.

```javascript
// JavaScript
var d = document.getElementsByClassName("example");
var i;

for (i = 0; i < d.length; i++) {
    d[i].className = d[i].className + " new_class_name";
}



// jQuery
$(".example").addClass("new_class_name");
```

Listagem 5 - JavaScript vs jQuery

De acordo com a Listagem 5, é possível verificar a existência de diferenças a nível da seleção de elementos com a utilização de JavaScript e jQuery, e ainda na ação de adicionar uma nova classe CSS aos respetivos elementos selecionados.

Relativamente ao Modernizr, este tem a função de detetar se o *browser* que está a ser utilizado tem suporte para certas funcionalidades de HTML5 e CSS3. Caso não tenha suporte para algumas destas funcionalidades, utiliza propriedades já predefinidas por omissão. Alguns dos componentes utilizados já tinham na sua integração o Modernizr implementado.

Animações Simples de Elementos

Tal como referido, o jQuery também foi utilizado para o desenvolvimento de animações. A função utilizada com maior frequência foi a "animate()" [30]. Esta função pode ser utilizada em qualquer elemento HTML, e com ela podem ser animadas todas as propriedades CSS.

```javascript
$('.nome_class').animate({"margin-left": "335"  },1500);
```

Listagem 6 - Função jquery "animate"

A Listagem 6 ilustra um exemplo de um determinado elemento estrutural HTML onde vai ser aplicado a função "animate()". Neste caso, será animada durante 1500 milissegundos, ou seja, 1,5 segundos a propriedade "margin-left" para o valor final de "335" *pixeis*.

Galeria de Imagens (Slide Show)

A maior parte dos trabalhos realizados tinha na sua constituição galerias de imagens. Um dos principais componentes utilizados para o desenvolvimento destas galerias foi o Flexslider [31]. Este componente é uma extensão jQuery, que permite a navegação entre os vários elementos utilizando os controlos de navegação (seta anterior e de seguinte). Permite também, a navegação através de *thumbnails,* ou seja, quando é clicado um certo *thumbnail*, o *slider* mostra a imagem associada ao *thumbnail* que foi clicado. Permite ainda a navegação pelos botões de controlos. Disponibiliza um conjunto de funcionalidades configuráveis, como o *autoplay,* o tempo de duração das transições, o tipo de animação da transição (*fade* ou *slide*), entre muitas outras.

Ilustração 4 - Exemplo da utilização do componente Flexslider

A Ilustração 4 mostra um exemplo de uma galeria de imagens, utilizada num projeto desenvolvido, recorrendo ao componente Flexslider, utilizando as setas de anterior e seguinte para a navegação.

Ilustração 5 - Exemplo da utilização do componente Flexslider com *thumbnails*

Por outro lado, a Ilustração 5 mostra um outro exemplo de uma galeria de imagens, utilizada também num outro projeto desenvolvido, recorrendo novamente ao mesmo componente Flexslider, mas com os *thumbnails* como controlos de navegação dos vários elementos, para além das setas anterior e seguinte.

Outro *plugin* jQuery utilizado para um efeito semelhante em galerias de imagens, foi o jCarousel [32]. Este *plugin* permite uma navegação tanto horizontal como vertical e um exemplo da sua utilização é apresentado pela Ilustração 6.

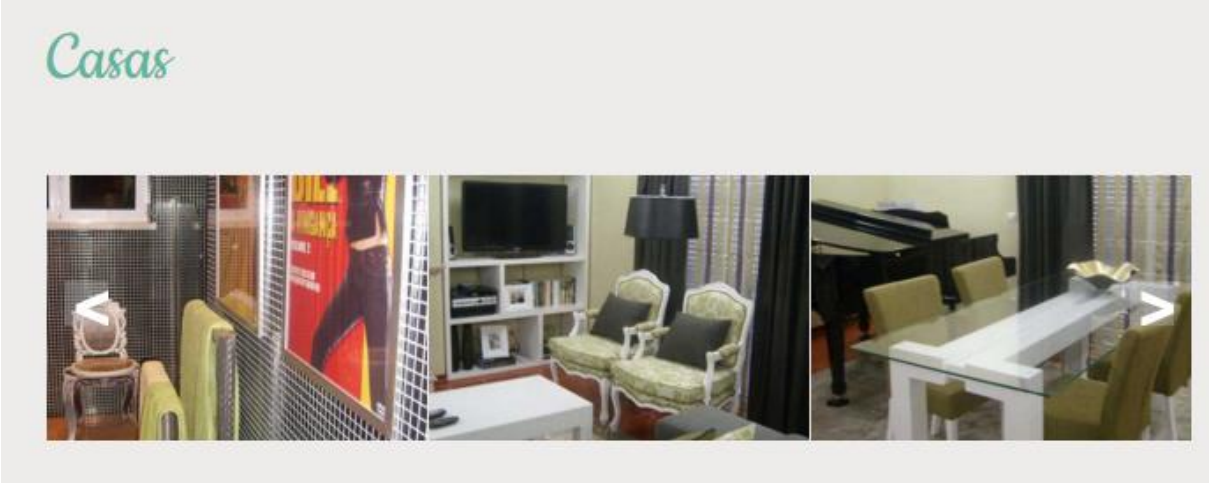


Ilustração 6 - Exemplo da utilização do componente jCarousel

O "Google Grid Gallery" [33] é o nome de uma galeria com um *slideshow* associado, disponível num exemplo *online*. Este exemplo é constituído por uma biblioteca JavaScript chamada de Masonry [34], que coloca os elementos de uma *grid* numa determinada posição, com base no espaço disponível verticalmente. Para um determinado projeto desenvolvido, houve a necessidade de criar uma galeria e um *slideshow* com o comportamento idêntico ao apresentado no exemplo *online*, pelo que foram utilizados como base de desenvolvimento os ficheiros disponíveis por este mesmo exemplo.

De acordo com o "Google Grid Gallery", quando por exemplo o elemento dois é clicado, aparece um *slideshow,* onde a primeira imagem desse *slideshow* corresponde ao elemento clicado. De cada lado da imagem é visível o elemento anterior e próximo, bem como as setas de navegação entre ambos, como apresentado na Ilustração 7.

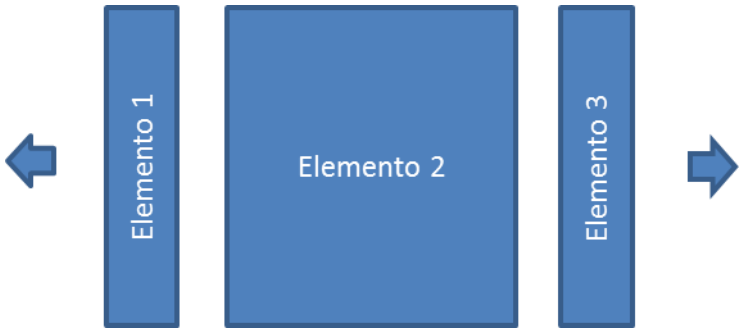


Ilustração 7 - Exemplo do *slideshow* utilizado no *"Google Grid Gallery"*

Contudo, para o projeto em questão, o que era pretendido era que caso o elemento dois fosse clicado, aparecesse um *slideshow*, mas em vez de ser de todos os elementos da *grid*, seria um *slideshow* de imagens associadas apenas ao elemento que fora clicado, como uma subgaleria, ou seja, elemento 2.1, 2.2, etc.

Para tal, no momento da criação da *grid*, para cada elemento foi atribuído um *data* atributo [35], com o nome de "project", como apresentado na Listagem 7.

```
<div id="grid-gallery" class="grid-gallery" style="display:none;">
    <section class="grid-wrap">
        <ul class="grid">
            <li class="grid-sizer"></li><!-- for Masonry column width -->
            <li data-project="1">
                <figure>

                </figure>
            </li>
            <li data-project="2">
                <figure>

                </figure>
            </li>
        </ul>
    </section>
</div>
```

Listagem 7 - Estrutura HTML da *grid* correspondente ao "*Google Grid Gallery*" utilizada em um projeto

Depois, referente ao *slideshow*, este é constituído por vários grupos *<ul>* também com o mesmo atributo *data,* o "project", que na sua constituição, tem então as várias imagens associadas, como apresenta a Listagem 8.

```
<section class="slideshow">
    <ul data-project="1">
        <li >
            <figure>
                <img src="img1" alt="img1"/>
            </figure>
        </li>
        <li >
            <figure>
                <img src="img2" alt="img2"/>
            </figure>
        </li>
    </ul>
    <ul data-project="2">
        <li >
        <figure>
            <img src="img1" alt="img1"/>
            </figure>
        </li>
        <li >
            <figure>
                <img src="img2" alt="img2"/>
            </figure>
        </li>
    </ul>
</section><!-- // slideshow -->
```

Listagem 8 - Estrutura HTML refente ao *slideshow*

De seguida, foi necessário alterar o ficheiro JavaScript com o nome "cbpGridGallery.js" disponível no exemplo *online* [33], na função responsável por apresentar o *slideshow*, isto é, a função "_openSlideshow".

```
//Função original
CBPGridGallery.prototype._openSlideshow = function( pos ) {
    this.isSlideshowVisible = true;
    this.current = pos;

    classie.addClass( this.el, 'slideshow-open' );

    /* position slideshow items */

    // set viewport items (current, next and previous)
    this._setViewportItems();

    //restante código igual
};


//Função alterada
var proj = 0;

CBPGridGallery.prototype._openSlideshow = function(item,  pos ) {
    proj = $(item).data('project');

    // slideshow grid items
    $('section.slideshow > ul[data-project!="'+proj+'"]').hide();
    $('section.slideshow > ul[data-project="'+proj+'"]').show();

    this.isSlideshowVisible = true;
    this.current = pos;

    classie.addClass( this.el, 'slideshow-open' );

        /* position slideshow items */

        // set viewport items (current, next and previous)
        this._setViewportItems();

    //restante código igual
};
```

Listagem 9 - Função original e função alterada responsável por apresentar o slideshow

De acordo com a Listagem 9, comparando as duas funções, é possível reparar que existe mais um argumento passado para a função, ou seja, o "*item*", que é depois utilizado para obter o valor do projeto a que está associado, através do seu atributo "data-project". Com base no valor obtido, apenas é apresentado o *slideshow* correspondente.

Transições de Páginas

O JavaScript e as suas bibliotecas permitem criar elegantes transições entres as várias secções de uma página HTML. Em um dos projetos desenvolvidos, foi utilizado como base o exemplo *online* com o nome de "A Collection of Page Transitions" [36], que foi desenvolvido com jQuery e com a biblioteca Modernizr.

```
<div id="pt-main">
    <div class="pt-page pt-page-1">
        <h1>
            Page 1
        </h1>
    </div>
    <div class="pt-page pt-page-2">
        <h1>
            Page 2
        </h1>
    </div>
    <div class="pt-page pt-page-3">
        <h1>
            Page 3
        </h1>
    </div>
</div>
```

Listagem 10 - Exemplo da estrutura HTML de uma página de um determinado projeto

De acordo com a Listagem 10, é possível verificar que existem três secções, que para efeitos finais de visualização do utilizador, correspondem a páginas HTML individuais. Estas secções estão colocadas umas depois das outras, contudo, apenas uma está visível de cada vez no ecrã. As transições entre secções ocorrem quando os botões de navegação são pressionados.

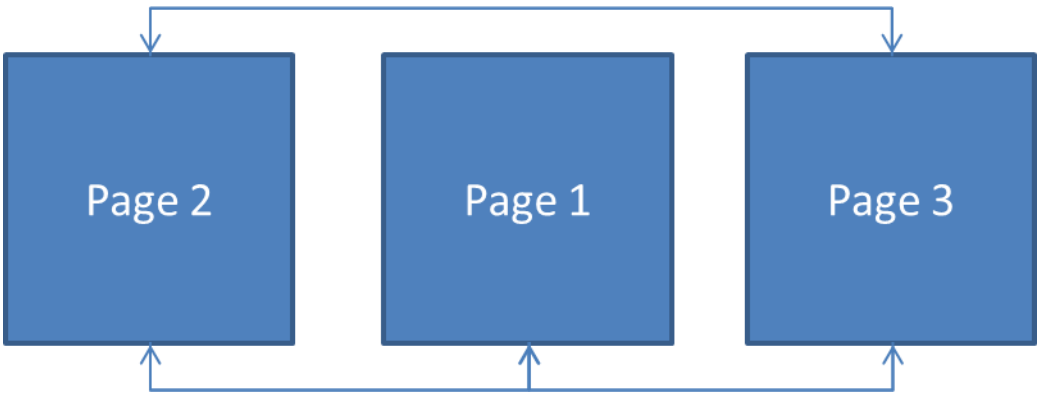


Ilustração 8 - Exemplo da estrutura das páginas utilizando o componente "A Collection of Page Transitions"

De maneira a melhor entender, na Ilustração 8, tem-se então a "Page 1" como sendo a única secção (página HTML) visível. Depois, ao utilizar os botões de navegação, e consoante o efeito da transição desejado, será mostrado a "Page 2", "Page 3", e por aí adiante.

Ainda relativamente às transições entre secções de uma página HTML, também foi utilizado como base o exemplo *online* com o nome de "Create a Parallax Scrolling Website Using Stellar.js" [37]. Este exemplo foi desenvolvido com a utilização de uma extensão jQuery com o nome de Stellar.js [38].

Nos projetos onde este *plugin* foi utilizado, o pretendido era existir uma animação na transição vertical entre as várias secções quando os elementos do menu forem carregados. Esta animação é possível observar no exemplo *online* do respetivo componente [39].

```
var htmlbody = $('html,body');
var dataslide = 2; //Exemplo, este valor é dinâmico

htmlbody.animate({
    scrollTop: $('.slide[data-slide="' + dataslide + '"]').offset().top
}, 2000);
```

Listagem 11 - Código responsável por efetuar a animação entre as secções (slides)

Consoante a Listagem 11, a transição entre secções (páginas) ocorre através da função "animate()", que já foi mencionada anteriormente, com a propriedade "scrollTop" do valor correspondente à distância em *pixeis* que a *div* "slide" com o atributo *data "*slide", neste caso igual a 2, se encontra do topo da página. Em complemento a esta animação, para além na navegação pelo menu, também era pretendido ocorrer a mesma animação, mas aquando a utilização do *scroll* do rato. Para isso, foi adicionado um novo *plugin* chamado "jquery-mousewheel" [40].

```
$('#elemento').mousewheel(function(event, delta, deltaX, deltaY) {
    //TODO
});
```

Listagem 12 - Evento "Mousewheel" disponível com o *plugin*

Este *plugin* tem uma função que é despoletada quando deteta o uso do *scroll* do rato, como apresentado na Listagem 12. Neste evento, é então executado um código semelhante ao da Listagem 11, para um "data slide" específico.

Novamente associado às transições das secções de uma página HTML, mas desta vez, para uma orientação horizontal, foi utilizado como base o exemplo *online* com o nome "Create a Funky Parallax Background Effect Using Jquery" [41]. Este exemplo foi desenvolvido com a utilização de uma extensão jQuery com o nome de "jquery.scrollTo-1.4.2-min.js" [42].

O *website* onde este componente foi utilizado, é constituído por quatro *slides* (secções). Cada *slide* ocupa sempre toda a largura e altura da janela, consoante os estilos de CSS, estando apenas um *slide* visível de cada vez. A Ilustração 9 mostra como estão estruturados os vários *slides*.

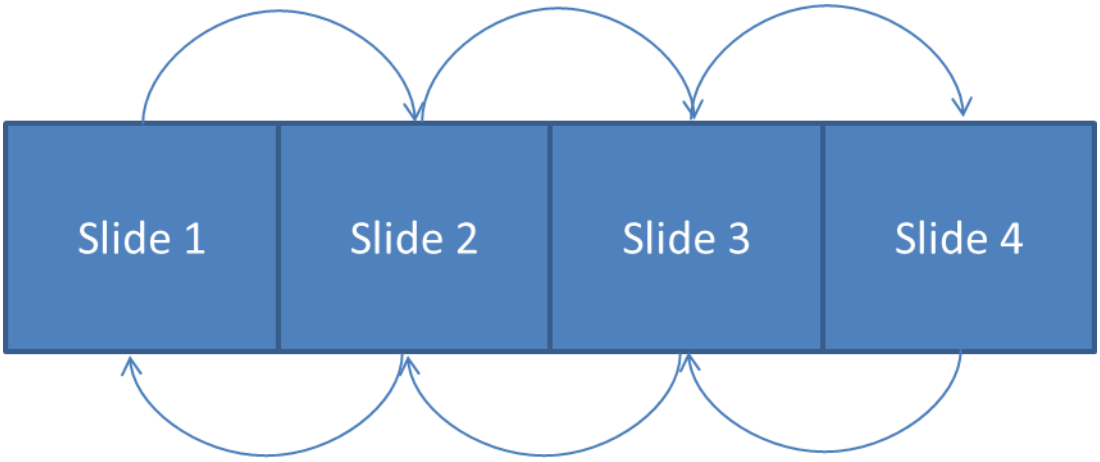


Ilustração 9 - Exemplo da estrutura dos vários *slides* do componente "Create a Funky Parallax Background Effect Using Jquery"

Quando os vários *links* do menu de navegação deste mesmo *website* são clicados, há então uma transição de *slide* para *slide*. O código responsável por realizar este comportamento é apresentado na próxima Listagem 13.

```
var index; //Valor do index da página - 1,2,3 e 4

$('#wrapper').scrollTo("#box"+index, 1000);
```

Listagem 13 - Código referente ao plugin jQuery "scrollTo"

Storage

O HTML Local Storage [43] é uma nova ferramenta apenas disponível no HTML5 que permite guardar informação local dentro do *browser.* Antes do HTML5, os dados de uma aplicação eram guardados nos Cookies [44]. A Local Storage é mais segura pois a informação não passa para o servidor através do HTTP, sendo acedido apenas do lado do cliente. Ou seja, o servidor não consegue ler nem escrever diretamente na "web storage". Para além disto, tem maior capacidade de armazenamento que os Cookies.

Esta ferramenta foi utilizada em dois projetos, onde numa determinada página HTML, um determinado *link* era clicado, e através da API, o valor correspondente a esse *link* era guardando numa variável da Local Storage. De seguida, quando o *website* faz o redirecionamento do utilizador para a nova página, correspondente ao *link* selecionado, com novamente a utilização da API, obtinha-se o valor correspondente à variável, e consoante o valor obtido, uma determinada ação era efetuada.

Outros Comportamentos

O MixItUp [45] é também uma extensão jQuery, que foi utilizado em dois projetos. Este *plugin* permite filtrar e ordenar elementos através de animações.

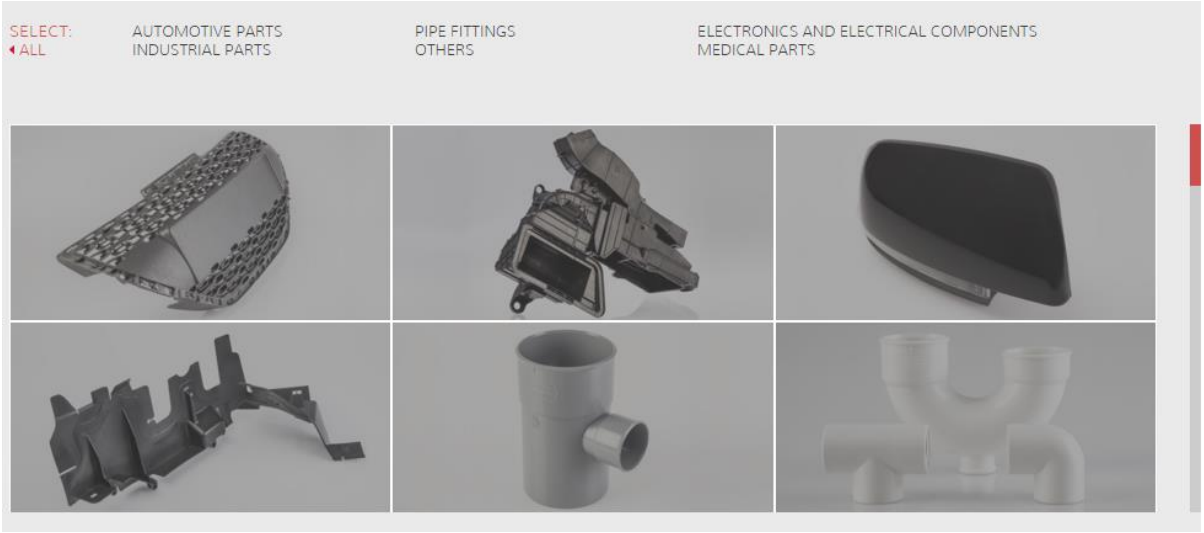


Ilustração 10 - Exemplo de uma *grid* de imagens, que pode ser filtrada por categorias

De acordo com a Ilustração 10, é possível verificar a existência de uma *grid* com várias imagens. No topo dessa *grid,* existe um menu com as várias categorias existentes. Quando cada um dos elementos do menu é clicado, a *grid*, recorrendo ao uso de animações, é filtrada, mostrando apenas a imagens correspondentes à categoria anteriormente selecionada.

Por último, em um dos projetos desenvolvidos, foi necessário implementar uma *sprite animation*, que corresponde a um conjunto de imagens individuais que fazem parte de uma sequência. Para esta animação, foi utilizado uma extensão jQuery com o nome "jQuery.animateSpirte" [46]. Este *plugin* necessita de uma imagem que na sua constituição contém outras imagens, como mostra a Ilustração 11.

Ilustração 11 - Imagem usada para uma sprite animation

O *plugin* percorre a imagem total, pegando nos elementos individualmente que têm um tamanho fixo predefinido tanto em altura como largura, e anima-os numa sequência.

3.3.1.3 Search Engine Optimization

Na área das aplicações *web*, há um fator importante a ter em conta aquando o desenvolvimento de uma aplicação. Um bom lugar no *ranking* de resultados dos motores de pesquisa, como o Google, Bing, Yahoo e outros, é fundamental para que um *website* seja facilmente encontrado, e desse modo, acedido. Para isso, existem conjuntos de estratégias, técnicas e táticas, à qual se dá o nome de *Search Engine Optimization* (SEO) [47]. Estas estratégias, técnicas e táticas são pequenas modificações que se fazem numa aplicação *web*, que quando vistas individualmente parecem coisas

insignificantes, mas quando combinadas podem ter um impacto enorme na experiência do utilizador a navegar na aplicação e na performance em *organic search results* [48].

Das técnicas que vão ser explicadas de seguida, todas elas foram utilizadas, embora não tenham sido utilizadas ao mesmo tempo na mesma aplicação *web*. É ainda importante referir que estas não são todas as técnicas de SEO existentes.

Title Tags

O *title tag* é um elemento que deve ser único e preciso. Deve ser colocado em todas as páginas de uma aplicação *web*. Este elemento tem como objetivo ser um título alusivo ao conteúdo de cada página. É um elemento crítico tanto para a experiência do utilizador como para os motores de pesquisa [49]. A Ilustração 12 apresenta um resultado obtido após uma pesquisa por "pvs moldes". O primeiro URL que aparece, "Moulds for the plastic injection industry | PVS Moldes" corresponde a uma *title tag*.

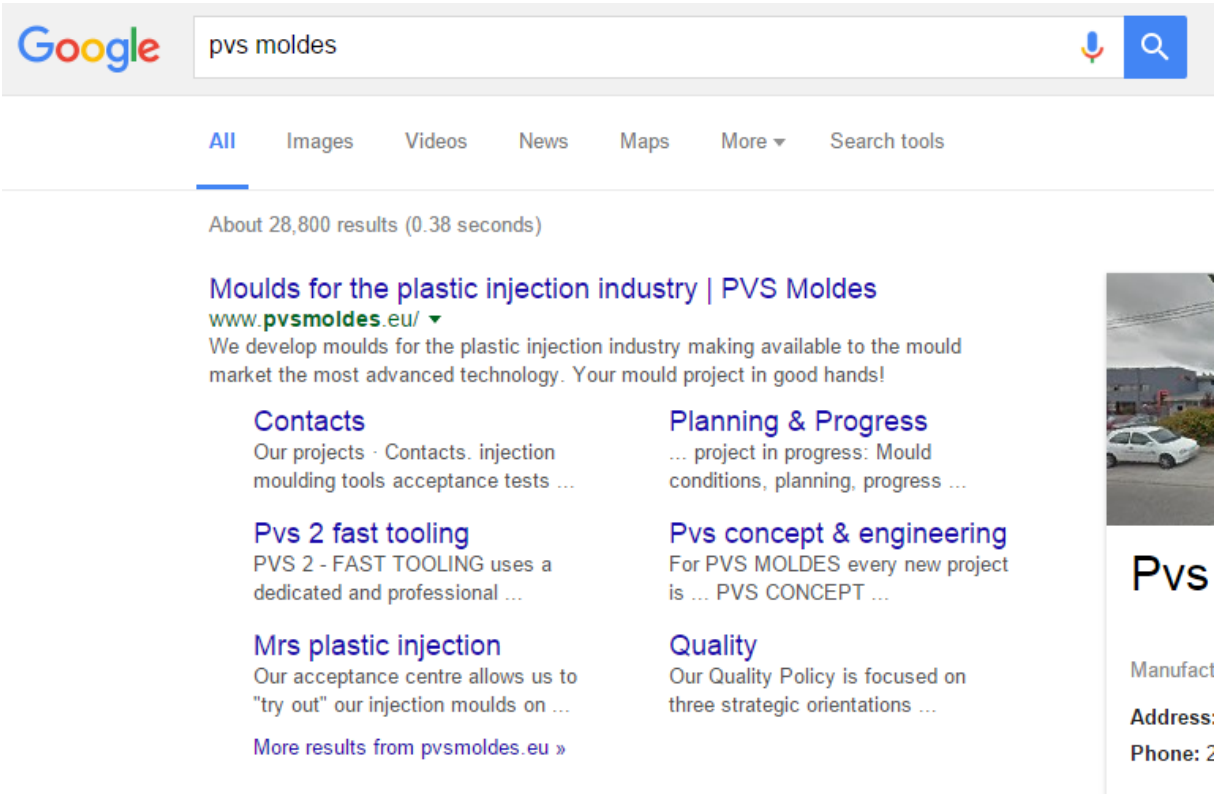


Ilustração 12 - Exemplo do resultado obtido quando efetuada uma pesquisa no Google

Meta Description

A *meta* description é um elemento que tal como o t*itle tags* deve ser único e deve constar em cada uma das páginas da aplicação *web*. Este elemento tem como objetivo apresentar uma explicação do conteúdo que cada página contém. A m*eta description,* tal como o t*itle tags,* é um elemento crítico tanto para a experiência do utilizador como para os motores de pesquisa.

É importante referir que a m*eta description* não interfere no *ranking* do Google. Apenas interfere na experiência do utilizador e nos motores de pesquisa [50]. A Ilustração 13 apresenta novamente o resultado obtido numa pesquisa de outro *website*, onde é possível verificar os *title tags* e as *meta description*.

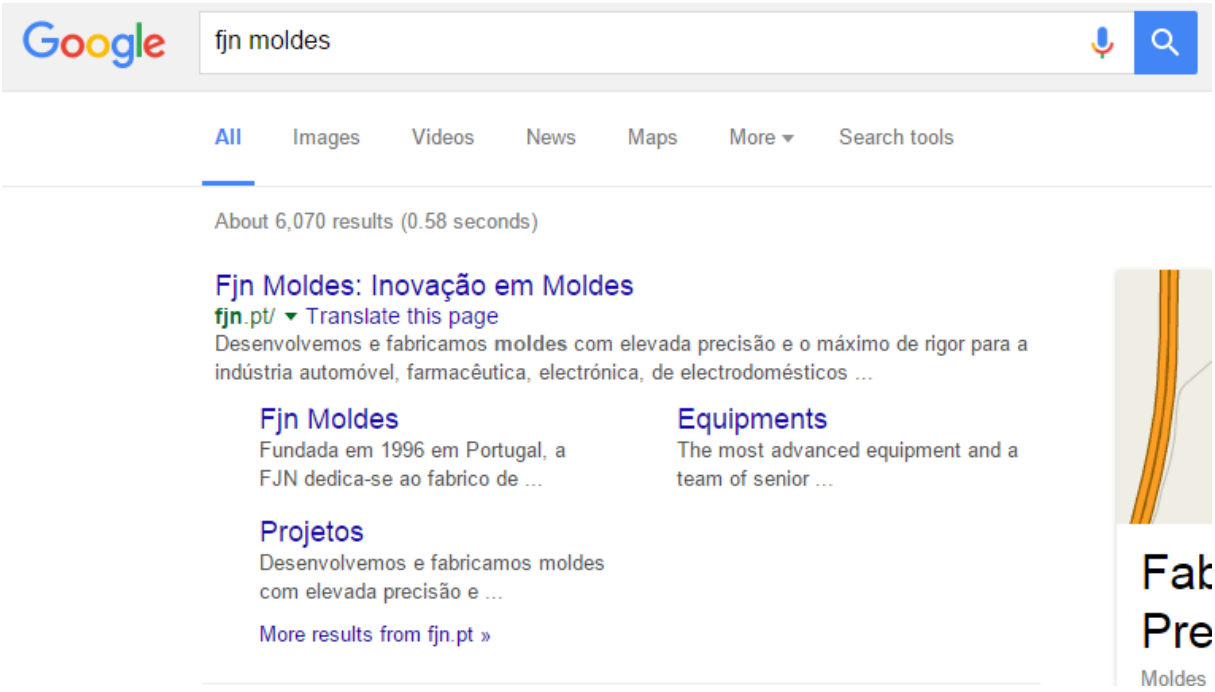


Ilustração 13 - Exemplo da *meta description* quando utilizado um motor de busca

Friendly Url

Os friendly url ou *url* amigáveis são uma das técnicas de SEO utilizadas que permitem ao utilizador entender melhor o URL em si. Também têm um grau de importância para os motores de pesquisa pois ajuda-os a descrever a página.

Exemplo de URL não amigáveis:

- http://www.meusite.pt/contato.php
- http://www.meusite.pt/produtos.php?categoria=5
- http://www.meusite.pt/noticia.php?id=2

Como é possível verificar acima, os URL contêm informação que para um utilizador habitual é difícil de entender, tal como por exemplo a extensão do ficheiro ".php", ou até o "?id=". De maneira a contornar esta situação, para ajudar então os utilizadores e os motores de pesquisa, utiliza-se por exemplo:

- http://www.meusite.pt/contato
- http://www.meusite.pt/produtos/camisas
- http://www.meusite.pt/noticias/2

Deste modo, o utilizador consegue ter uma melhor percepção do que vai encontrar ao aceder à respetiva página [51].

Sitemap

Existem dois tipos de *sitemap*. O *sitemap* que contém a estrutura do *website* de uma maneira hierárquica, tal como apresentado na Ilustração 14, e o *sitemap* correspondente a um ficheiro Extensible Markup Language (XML) constituído por todos os URL do *website*. Este tipo de *sitemap* ajuda os motores de pesquisa a encontrar e classificar o conteúdo do *website*, que por alguma razão, pode não ser encontrado pelos motores de pesquisa [52].

Ilustração 14 - Exemplo de um sitemap do *website* da Granifil

Após a criação do ficheiro XML, que deve ser colocado na raiz do *website*, de maneira a informar o Google que queremos que ele faça o rastreamento desse *website*, é necessário aceder à ferramenta Webmaster Tool [53] e adicionar uma nova propriedade. Após adicionar a propriedade, é necessário validá-la, colocando um ficheiro ".html" de maneira a que o endereço para aceder a esse ficheiro seja o seguinte, http://www.website.com/google1234556789.html.

Existem métodos alternativos para a validação da propriedade, tal como o uso do Google Analytics, ou adicionar uma *meta tag* à página inicial do *website*. Após a validação, apenas é necessário dizer qual o caminho do ficheiro XML do *sitemap*. Normalmente o caminho é o seguinte, http://www.website.com/sitemap.xml.

Canonicalization

A *"canonicalization"* [54] faz referência a páginas *web* individuais que podem ser acedidas através de múltiplos URL's. Isto é um problema pois quando várias páginas têm o mesmo conteúdo, mas URL's diferentes, os *links* que estão destinados a ir para a mesma página, são divididos entre os múltiplos URL. O que significa que a "popularidade" (importância na *web*) da página também é dividida. A próxima lista, apresenta os erros mais comuns da *"canonicalization"*:

- http://www.example.com/
- http://www.example.com/index.html
- http:/example.com/
- http://example.com/index.html

Para prevenir então este problema, tem-se então a utilização de uma *meta tag,* que está visível na Listagem 14, que deve ser colocada no *head* da página HTML.

```
<link rel="canonical" href="http://www.example.com/" />
```

Listagem 14 - Exemplo da *meta tag* utilizada para prevenir problemas de "*canonicalization*"

Para além da *tag*, é possível também no ficheiro ".htaccess", adicionar uma regra de redirecionamento de código 301 para obter o mesmo resultado.

Robots.txt

O REP (Robots Exclusion Protocol) ou "robots.txt" é um ficheiro utilizado para instruir os *robots* dos motores de pesquisa de como, e que paginas devem ser indexadas. [55].

Neste ficheiro "robots.txt" colocam-se quais as diretorias ou páginas que o criador do *website* quer ou não que sejam indexadas quando é feita uma pesquisa, bem como a lista dos vários motores de busca que possam vir a ser utilizados, que se pretende bloquear.

Dado o ficheiro "robots.txt" ser público, para efeitos de segurança, é recomendado usar palavras-chave caso se queira prevenir que certas páginas sejam indexadas. Por exemplo através do ficheiro ".htaccess" [48].

Texto de Âncora

O texto de âncora (*anchor text*) é um texto clicável numa hiperligação, que redireciona o utilizador para uma outra página *web*.

```
Product of  <a href="http://www.thesilverfactory.pt/" target ="_blank" >THE SILVER FACTORY</a>
```

Listagem 15 - Exemplo de um "Anchor Text"

A Listagem 15 é um exemplo de um "texto de âncora" utilizado em todos os projetos desenvolvidos.

Deste modo, este *link* informa os motores de busca que quando o termo "The Silver Factory" for pesquisado, os *websites* que contiverem este texto de âncora assumem que http://thesilverfactory.pt é um *website* relevante. Posto isto, se muitos *websites* tiverem como importante uma página específica, essa página pode ser bem classificada nos *rankings* do Google.

Optimização de Imagens

Uma outra técnica de SEO é o uso de imagens nos ficheiros HTML, mas de maneira correta, ou seja, utilizar imagens que contenham um nome único e ilustrativo, e que contenham sempre o atributo *alt*, onde este *alt* permite especificar do que se trata a imagem caso ela, por alguma razão, não possa ser apresentada pelo *browser*. Para além destas características, é importante também utilizar imagens com um tamanho adequado, sem por em causa a sua resolução.

Uso de Headings

Por último, mas não menos importante, é o uso de *heading tags* [56]. Estas *tags* são utilizadas para ilustrar um tipo de estrutura hierárquica nas páginas, e têm impacto do ponto de vista do SEO [57], tal como o aspeto da relevância, onde os motores de pesquisa comparam as palavras nas *header tags* com o conteúdo da secção associada; o aspeto da consistência de palavras-chaves, onde os motores de pesquisa verificam se estas palavras estão espalhadas pelas várias páginas do *website*. A *tag* H1 é a *tag* mais importante e deve ser sempre utilizada em cada uma das páginas. Os motores de pesquisa dão especial atenção a esta *tag*, verificando se contém a descrição base do conteúdo da página, tal como acontece com a *title tag*. Por último, também dão ao utilizador uma ideia clara do que se vai encontrar no documento.

3.3.2 *Websites com backoffice*

Os constituintes base para o desenvolvimento de *websites* sem *backoffice* foram devidamente mencionados na secção anterior (3.3.1), isto é, a linguagem de marcação HTML, a linguagem de estilos CSS e ainda o JavaScript para a linguagem de cliente, com as respectivas bibliotecas. Contudo, também foram desenvolvidos *websites,* com os mesmos constituintes, mas com um *backoffice* integrado. Este *backoffice* disponibiliza um suporte essencial para a administração dos *websites* onde foi utilizado.

3.3.2.1 *Framework Laravel*

Todo o *backoffice* foi desenvolvido com uma *framework* em PHP, já mencionada na secção 2.6, no capítulo das Tecnologias, chamada Laravel. Esta *framework* utiliza o padrão de desenho MVC que também já foi referido nesse mesmo capítulo. A versão da *framework* que foi utilizada para o desenvolvimento do *backoffice* e dos respetivos projetos começou por ser a 4.0, cuja data de lançamento foi em Maio de 2013 e que requeria uma versão de PHP 5.3.7 ou superior. Contudo, foi depois atualizada para versão 4.2, cuja data de lançamento foi em Maio de 2014. Para esta versão do Laravel, era necessário uma versão de PHP 5.4 ou superior.

Rotas e Controladores

A *framework* Laravel é constituída por um sistema de regras que são definidas no ficheiro "routes.php". Este sistema de *routing* realiza o reencaminhamento da aplicação para os métodos funcionais de um controlador [58].

```
Route::get('/administration/destaques', array('as' => 'destaques' ,'uses' => 'DestaquesController@index'));
```

Listagem 16 - Exemplo de uma rota no ficheiro "routes.php"

Com base na Listagem 16, quando é colocado na barra de endereços o seguinte URL www.dominio.pt/administration/destaques, a aplicação verifica se existe alguma regra definida no ficheiro "routes.php", e caso se verifique, é efetuado o mapeamento para o método do controlador correspondente. Neste caso o controlador é o "DestaquesController" e o método é o "index".

Deste modo, tem-se então uma estrutura de caminhos logicamente bem divididos. Esta estrutura pode ser demonstrada na Tabela 2.

Tabela 2 - Métodos disponibilizados por um Resource Controller (adaptado de [59])

Método HTTP	Caminho (URL)	Ação (Método)	Nome da Rota
GET	/destaques	index	destaques.index
GET	/destaques/create	create	destaques.create
POST	/destaques	store	destaques.store
GET	/destaques/{id}	show	destaques.show
GET	/destaques/{id}/edit	edit	destaques.edit
PUT/PATCH	/destaques/{id}	update	destaques.update
DELETE	/destaques/{id}	destroy	destaques.destroy

Na *framework* Laravel, como foi referido anteriormente, é possível definir regras, para associar caminhos a controladores. É ainda possível, associar caminhos a controladores de modo a que a *framework* mapeie automaticamente os métodos REST (Representational State Transfer) presentes nesses controladores. São os controladores com o nome de RESTFull Resource Controllers [60] que disponibilizam os métodos que se encontram na Tabela 2. É possível também adicionar novos métodos a estes controladores, contudo o mapeamento tem de ser realizado manualmente no ficheiro "routes.php".

Também é possível associar funcionalidades específicas (métodos) a controladores. Para isso, são utilizados os Implicit Controllers [61]. Estes controladores mapeiam todos os métodos que estejam declarados nesse mesmo controlador, desde que o nome siga uma determinada nomenclatura. Esta nomenclatura é composta por um nome com o prefixo do método HTTP.

```
class RemindersController extends Controller {

    public function getRemind()
    {
        //TODO
    }

}
```

Listagem 17 - Exemplo de um método utilizado por um Implicit Controller

Eloquent

Para questões de persistência de dados, a *framework* Laravel utiliza um Object-Relational Mapping (ORM) [62] que é uma técnica usada para o acesso à base de dados que permite definir e gerir o mapeamento entre as tabelas da base de dados e as classes do modelo de domínio. Com a utilização de um ORM, há uma abstracção total do SQL, e trabalha-se apenas com classes. O módulo ORM fica responsável por fazer a interface entre os objetos e as tabelas da base de dados (Modelo de Objetos e Modelo Relacional).

Este ORM que o Laravel utiliza, tem o nome de Eloquent [63] que implementa o padrão Active Record [64]. Este padrão é essencialmente utilizado em aplicações baseadas em métodos CRUD. Com ele, cada tabela da base de dados corresponde a um "Modelo", "Modelo" esse que é então utilizado para interagir com essa mesma tabela. Com o Eloquent é mais fácil de trabalhar com relações entre as várias entidades, bem como com as várias operações CRUD.

O Eloquent prevê que para cada tabela, seja criada uma classe que vai herdar da classe *Eloquent*.

```
class Projeto extends Eloquent {

    protected $primaryKey = 'id_projeto';

    public $timestamps = false;

    public function fotosProjeto(){
        return $this->hasMany("FotosProjeto","id_projeto");
    }

}
```

Listagem 18 - Classe Projeto que herda da classe Eloquent

De acordo com a Listagem 18, a classe apresentada corresponde à entidade "Projeto", que é representada pela tabela "projetos". Por convenção, o nome da tabela correspondente à classe, é o plural do nome dessa mesma classe, e cada tabela tem uma coluna da chave primária com o nome de "id". Contudo, neste caso, essa regra foi redefinida através da variável *$primaryKey*.

É ainda possível verificar que existe uma variável do tipo *boolean*, a *$timestamps*, que tem o valor de "false". O que significa que a tabela não irá conter duas colunas, a "created_at" e "updated_at", que servem para guardar as datas de criação e alteração de uma linha.

É importante também referir que, embora não tenha estejam todas presentes na classe anterior, o Eloquent permite definir as várias relações entre entidades, tais como 1 para 1, 1 para N e N para M, o que facilita a obtenção de todos os objetos relacionados.

Neste caso específico da Listagem 18, é possível verificar que existe um método, chamado "fotosProjeto", que permite que a entidade "Projeto" se relacione com a entidade "FotosProjeto". Este tipo de relação corresponde a uma relação de 1 para N (*hasMany*).

```php
//Create
$projeto = new Projeto;
$projeto->nome = 'Nome';
$projeto->save();

//Read
$projeto = Projeto::find(1); //Baseado no ID
$projeto = Projeto::first(); //Primeiro objeto

//Update
$projeto = Projeto::first();
$projeto->nome = 'Outro nome';
$projeto->save();

//Delete
$projeto = Projeto::first();
$projeto->delete();
```

Listagem 19 - Exemplo da utilização do Eloquent

De acordo com a Listagem 19, é possível verificar um exemplo dos métodos CRUD que podem ser utilizados com o Eloquent.

Embora as funcionalidades relativas a relacionamentos que o Eloquent disponibiliza sejam muito úteis, existem alguns cuidados a ter em conta quando são utilizadas. Um dos principais cuidados a ter é quando se utilizam os métodos de relacionamento dentro de ciclos. Nesse caso, pode existir um elevado número de *queries* feitas à base de dados, que vai ter como consequência a diminuição do desempenho da aplicação. O Eloquent disponibiliza funções que permitem contornar este problema.

```php
//Total de livros = 25;

//Mau exemplo
$books = Book::all(); //SELECT * FROM books - 1 query executada
foreach($books as $book){
    echo $book->author->nome; //Query executada de dentro do loop, uma por cada livro, para obter o nome do autor - 25 queries executadas
}

/*
    SELECT * from author WHERE book_id = 1;
    SELECT * from author WHERE book_id = 2;
    SELECT * from author WHERE book_id = 3;
    SELECT * from author WHERE book_id = 4;
    SELECT * from author WHERE book_id = 5;
    etc....

    Total = 26 queries executadas
*/

//Bom exemplo
$books = $books->with("author"); //SELECT * FROM books - 1 query executada
                                 //SELECT * FROM author WHERE book_id in (1, 2, 3, 4, 5, ...) - 1 querie executada

foreach($books as $book)
{
  echo $book->author->nome;
}

//Total = 2 queries executadas
```

Listagem 20 - Bom e mau exemplo da utilização do Eloquent

Temos por exemplo a Listagem 20, que apresenta dois ciclos *foreach* que imprimem para o ecrã o nome do autor a que pertence cada livro. Imaginando que cada livro pertence a apenas 1 autor, isto é, uma relação de 1 para 1, e imaginando ainda que existem 25 livros na tabela "books", após serem obtidos todos os livros, em cada iteração do ciclo *foreach*, é executada uma *querie* para obter o nome do autor do livro. Ora se existirem 25 livros, no total serão executadas 25+1 *queries* (N+1) [65].

Por outro lado, ainda na Listagem 20 no segundo exemplo, e imaginando o mesmo cenário, logo na primeira *query* serão obtidos todos os livros, e ainda os respetivos autores, consoante o "id" do livro. Com isto, serão apenas executadas 2 *queries*, o que, comparando com o exemplo anterior, irá aumentar o desempenho da aplicação.

No entanto este tipo de ORM não tem como objetivo substituir as tradicionais queries SQL, embora tenha sido utilizado na maioria dos casos.

Query Builder

Relacionado também com o conceito de persistência de dados, foi utilizado o Query Builder [66], que disponibiliza uma interface eficiente para criar e executar *queries*. Utiliza ainda PHP Data Object (PDO) [67] para proteger a aplicação contra ataques de SQL injection [68]. O Query Builder é utilizado quando se pretende optimizar manualmente uma consulta à base de dados, ou então quando se pretende realizar *queries* mais complexas.

```php
$projetos = Projetos::find($id);  // Eloquent version
$projetos = DB::table('projetos')->where('id_projeto',$id)->get(); // Fluent version
```

Listagem 21 - Exemplo da utlização do Eloquent vs Query Builder

Na Listagem 21 temos um exemplo da utilização do Eloquent e do Query Builder. A nível de leitura, é mais acessível o Eloquent. A nível de funcionalidade, todas as funções que podem ser executadas no Query Builder, podem também ser executadas no Eloquent. Novamente, quando se trata de *queries* mais complexas, é aconselhável o uso do Query Builder.

Autenticação e autorização

A segurança é um dos aspectos fundamentais a ter em consideração no desenvolvimento de aplicações *web*. Como tal, a *framework* Laravel aborda medidas de segurança que são importantes de mencionar.

Começando pela autenticação, por omissão a *framework* utiliza o Eloquent como *driver* de autenticação, agregado a um "Modelo" chamado *User*.

Algo que é preciso ter em atenção quando se trata de *logins,* é a maneira como são armazenadas as palavras-chave na base de dados. É impossível impedir que a base de dados sofra ataques e que a pessoa ou entidade que organizou o ataque tenha acesso a todos os dados dos vários utilizadores. Em muitos casos, os utilizadores apenas têm uma palavra-chave para o email, conta do banco, etc, pelo que um simples *hack* se poderia transformar em um roubo de identidade. Para prevenir isto, usam-se métodos de cifragem. A *framework* Laravel disponibiliza a classe *Hash* que usa uma Bcrypt Hash [69] para guardar as palavras-chaves na base de dados.

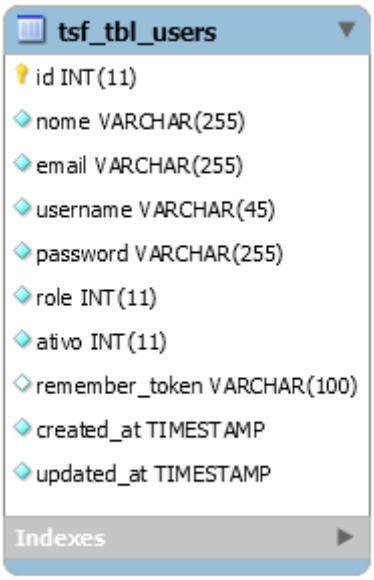


Ilustração 15 – Tabela "tsf_tbl_users"

A tabela "tsf_tbl_users", apresentada na Ilustração 15, é a tabela principal utilizada para armazenar os dados de acesso dos vários utilizadores, tal como o nome, o email, o *username* e a *password*. Cada utilizador tem apenas um nível de acesso às aplicações, pelo que esse valor é armazenado diretamente na tabela na coluna "role".

Após os dados dos utilizadores terem sido previamente armazenados, é possível então efetuar o processo de autenticação. Para tal, a *framework* utiliza o método chamado "*attempt*" [70] da classe *Auth,* apresentado na Listagem 22, que valida se os dados introduzidos coincidem com os dados que estão armazenados na tabela "tsf_tbl_users".

```php
$userdata = array(
    'username' => Input::get('username'),
    'password' => Input::get('password')
);

if(Auth::attempt($userdata)){
    //Login com sucesso
}else{
    //Erro no login
}
```

Listagem 22 - Exemplo da implementação do método "attempt"

O campo "*remember_token*" da tabela "users" é utilizado para guardar um *token* para a opção de "*remembre me*" do método "*attempt*". A *flag* igual a "*true*" deve ser colocada como segundo parâmetro do método "*attempt*". Desta forma, o utilizador fica indefinitivamente logado sistema, ou até fazer manualmente o *logout (*Listagem 23*)*.

```php
$userdata = array(
    'username' => Input::get('username'),
    'password' => Input::get('password')
);

if(Auth::attempt($userdata, true)){
    //Login com sucesso
}else{
    //Erro no login
}
```

Listagem 23 - Exemplo da implementação do método "attemp" com a particularidade do "remember me"

Caso um utilizador se esqueça dos seus dados de acesso, a *framework* disponibiliza também um processo para fazer *reset* às palavras-chave através de *password-reminders.* Para tal, é também necessário criar uma tabela chamada "*password-reminders" (*Ilustração 16*)* que irá guardar os *tokens* associados aos respetivos emails. Há uma relação de 1 para 1 entre esta nova tabela e a tabela "users". Por último, é necessário criar o respetivo "Controlador".

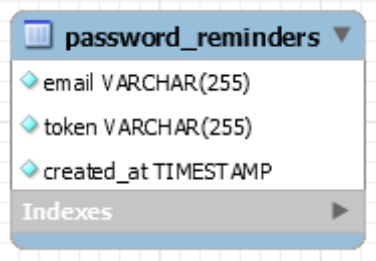


Ilustração 16 - Tabela "*password_reminders*"

Este “Controlador”, apresentado pela Listagem 24, tal como outros elementos estruturais da aplicação, pode ser criado automaticamente utilizando a linha de comandos Artisan CLI.

```php
class RemindersController extends Controller {

    /**
     * Display the password reminder view.
     *
     * @return Response
     */
    public function getRemind()
    {
        //código
    }

    /**
     * Handle a POST request to remind a user of their password.
     *
     * @return Response
     */
    public function postRemind()
    {
        //código
    }

    /**
     * Display the password reset view for the given token.
     *
     * @param  string  $token
     * @return Response
     */
    public function getReset($token = null)
    {
        //código
    }

    /**
     * Handle a POST request to reset a user's password.
     *
     * @return Response
     */
    public function postReset()
    {
        //código
    }

}
```

Listagem 24 - Controlador "Reminders" gerado automaticamente através da Artisan CLI

Após ser criado o controlador, é necessário criar as respectivas “*views*” e implementar o respetivo código nas mesmas.

Ainda relativamente ao tópico da segurança, é importante fazer um controlo de acesso às várias rotas que os utilizadores podem aceder para não haver configurações indesejáveis no sistema. Para isso foram utilizados filtros [59]. Os filtros permitem executar funções antes ou depois da execução do método correspondente ao controlador, consoante a *key* “*before”* ou “*after”* respetivamente.

O filtro mais utilizado foi o “*auth*”, com a *key “before”,* como é possível ver na Listagem 25. Este filtro foi atribuído a um “*group*” que é constituído por um conjunto de rotas. A sua principal função é verificar se existe algum utilizador logado no sistema. Se esse caso não se verificar, a aplicação redireciona o utilizador para a página de *login* do *backoffice*. Por outro lado, se o utilizador existir, o sistema verifica ainda o seu *role*. Se o *role* for o pretendido, o controlador realiza toda a lógica

correspondente ao pedido efetuado. Se o *role* não for o pretendido a aplicação redireciona novamente o utilizador para a página de *login* do *backoffice,* fazendo automaticamente o seu *logout.*

```php
// Filtro "auth"
Route::filter('auth', function()
{
    $user = Auth::user();

    if($user)
    {
        if($user->role != 1)
        {
            Auth::logout();
            return View::make('login')->with('title','Login');
        }
    }else{
        return View::make('login')->with('title','Login');
    }
});

// Grupo constituido por várias rotas onde é aplicado o filtro "auth"
Route::group(array('before' => 'auth'), function() {
    // Rotas correspondentes a este grupo
});
```

Listagem 25 - Código correspondente ao filtro "auth" aplicado a um conjunto de rotas

Protecção contra ataques

O Cross-Site Request Forgery (CSRF) é um ataque comum a nível de aplicações *web*, e consiste na realização de certos comandos ou pedidos (*requests*) por um outro utilizador sem ser aquele que está autenticado no sistema [71] [72]. Para proteger de tal ataque, é utilizado um outro filtro, com o nome "*csrf*". Este filtro foi utilizado em todas as rotas correspondentes aos métodos HTTP de POST/PUT.

Nas *views* correspondentes a estas rotas, no formulário é introduzido um campo oculto que contem um *csrf-token*, e o filtro é responsável por verificar se esse *token* corresponde ao mesmo *token* da sessão, como é possível verificar na Listagem 26.

```php
<input type="hidden" name="_token" value="<?php echo csrf_token(); ?>">

Route::filter('csrf', function()
{
    if (Session::token() != Input::get('_token'))
    {
        throw new Illuminate\Session\TokenMismatchException;
    }
});
```

Listagem 26 - Input hidden do crsf-token e o código do filtro "csrf"

Se não existir qualquer problema com a utilização do filtro na respetiva rota, o controlador realiza toda a lógica correspondente ao pedido HTTP.

Para além de ataques CSRF, existem outros tipos de ataques que devem também ser protegidos. São por exemplo o Cross-Site Scripting (XSS) e o SQL Injection. O XSS refere-se à injeção de código do lado do cliente para executar *scripts* maliciosos numa aplicação *web* legítima [73], normalmente

código JavaScript. Com a *framework* Laravel, este tipo de ataque pode ser protegido, utilizando *triple curly braces* [74], ou seja, "{{{ }}}", que são disponibilizadas pelo *template engine*, Blade. Usando estes parênteses, não são tidas em conta HTML *entities*, isto é, no *output* de conteúdo na página *web*, por exemplo, na *string* "bold", os caracteres "" e "" são convertidos em entidades HTML, "" e "", respetivamente, pelo que não será executada a sua função de colocar o texto a *bold*.

Ataques de SQL Injection são também um tipo de ataques frequentes em aplicações *web*, onde é possível injectar comandos SQL numa expressão SQL (*SQL statement*), que podem expor ou alterar dados importantes que estão armazenados na base de dados. Para evitar este tipo de ataques, a *framework* utiliza PDO através do Query Builder, como referido anteriormente.

Email

O envio de emails foi uma funcionalidade que foi implementada em todos os projetos desenvolvidos ao longo do estágio. Todos os *websites* com ou sem *backoffice*, tinham na sua constituição pelo menos uma página equivalente à página dos "Contactos". Nessa página existia um formulário de contactos, que após o seu preenchimento poderia ser submetido. Para esta funcionalidade, para os *websites* com *backoffice*, o Laravel disponibiliza uma API que assenta sobre o SwiftMailer [75] (ficheiro "root/app/config/mail.php"). É possível utilizar vários *drivers* de *mail*, como o "smtp", o "mail" e ainda o "sendmail". À versão 4.2 da *framework*, foram adicionadas duas novas *drivers* de *mail* chamadas Mailgun [76] e Mandrill [77]. Contudo, a *driver* que foi mais frequentemente utilizada foi a "smtp".

Para os *websites* sem *backoffice*, foi utilizada a função mail [78] do PHP.

Localização

Em todos os projetos desenvolvidos, existiam algumas palavras ou frases que necessitavam de tradução, tanto a nível de *frontend* para títulos de páginas ou *placeholders* em formulários, bem como para o *backoffice* para as validações dos campos dos formulários. Para este tipo de funcionalidade, foi utilizada a classe "Lang" [79], disponível pela *framework* Laravel.

```
// definir a linguagem da aplicação
App::setLocale('pt');

// imprimir a traducao correspondente à expressão
Lang::get('nome_ficheiro.expressao');
```

Listagem 27 - Exemplo da utilização da classe "Lang" para tradução de expressões

Com base na Listagem 27, é necessário primeiro definir a linguagem da aplicação, para depois se usar a classe "Lang", que neste caso, recebe como parâmetro o nome do ficheiro que está no interior da pasta "pt", onde está guardada a expressão e a respetiva tradução. Se não for definido a linguagem da aplicação, é utilizada a que está configurada por omissão no ficheiro "app/config/app.php".

3.4 Principais Componentes Desenvolvidos

O *backoffice* foi desenvolvido com o objetivo de oferecer ao cliente um controlo do seu *website,* permitindo-lhe utilizar ferramentas ou componentes com determinadas características para determinados fins.

Deste modo, tem-se então os seguintes componentes desenvolvidos:

- *Content Management System (CMS)*
- *Project Management Tool*
- *Dashboard – Analytics*
- Gestão de Ficheiros (Área Reservada)
- SEO

Consoante a necessidade do cliente, apenas estavam visíveis no *backoffice* os componentes necessários ao projeto em questão.

3.4.1 Infra-Estrutura do Backoffice

Como qualquer outro *backoffice* existente, este não podia deixar de ter na sua constituição um sistema de *login*, com a possibilidade de efetuar um *reset* à *password* quando necessário. Estas duas funcionalidades são comuns em todos os projetos e já foram devidamente explicadas na secção 3.3.2.1 no tópico "Autenticação e autorização".

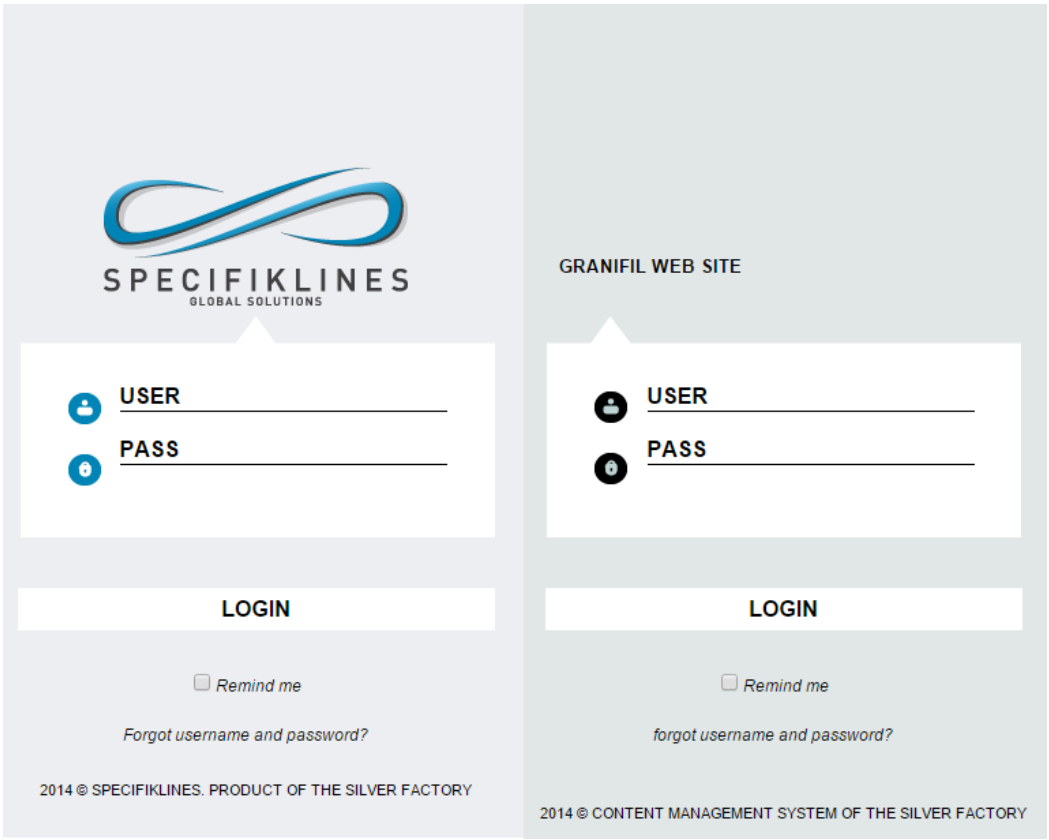


Ilustração 17 - Exemplo de duas páginas de *login* para o *backoffice*

A Ilustração 17 apresenta um exemplo de duas páginas de *login* que são apresentadas ao utilizador quando este acede ao endereço do *backoffice*. Nestas páginas pode haver diferenças a nível do *layout,* isto é, diferentes cores e diferentes logotipos, consoante o projeto.

Após o *login* ser efetuado com sucesso, o utilizador é reencaminhado para uma nova página, com a possibilidade de escolher entre os vários componentes constituintes do *backoffice*.

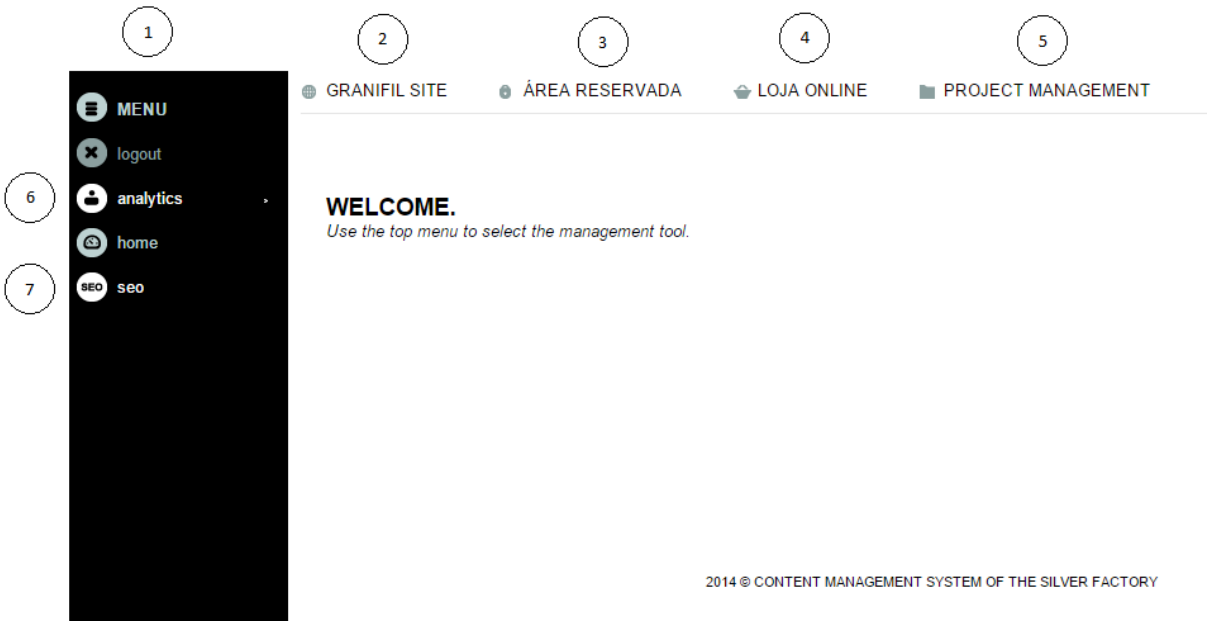


Ilustração 18 - Exemplo da página apresentada após o sucesso do *login*

A Ilustração 18 apresenta um exemplo da página visível após o sucesso no *login*, e nela é possível verificar alguns pontos-chave, tal como:

1. Menu principal de navegação do *backoffice*;
2. Componente do *Content Management System*;
3. Componente da Gestão de Documentos e Ficheiros, ou Área Reservada;
4. Componente de E-Commerce/Loja Online, que embora esteja visível na imagem, nunca foi desenvolvido;
5. Componente do *Project Management Tool*;
6. Componente do *Dashboard – Analytics*.
7. Componente de integração de algumas características de SEO no *backoffice*.

Tal como referido anteriormente, para cada projeto, apenas aparecem visíveis os componentes que fazem parte do mesmo.

3.4.2 Content Management System

O *CMS* foi o primeiro componente a ser desenvolvido, com o objetivo de simplificar a gestão de conteúdos em *backend*.

Para aceder a este componente do *backoffice*, é necessário utilizar o elemento correspondente ao número 2 da Ilustração 18. Com esta ação, o utilizador é novamente redirecionado para uma nova

página, específica apenas deste componente. Nessa página, é possível verificar a existência de dois novos elementos do menu lateral, o "Pages Management" e o "Gallery".

Pages Management

Este elemento do menu apresenta uma lista de todas as páginas existentes no *frontend* às quais se pode alterar o seu conteúdo. Se estas páginas tiverem subpáginas associadas, aparecem também neste menu.

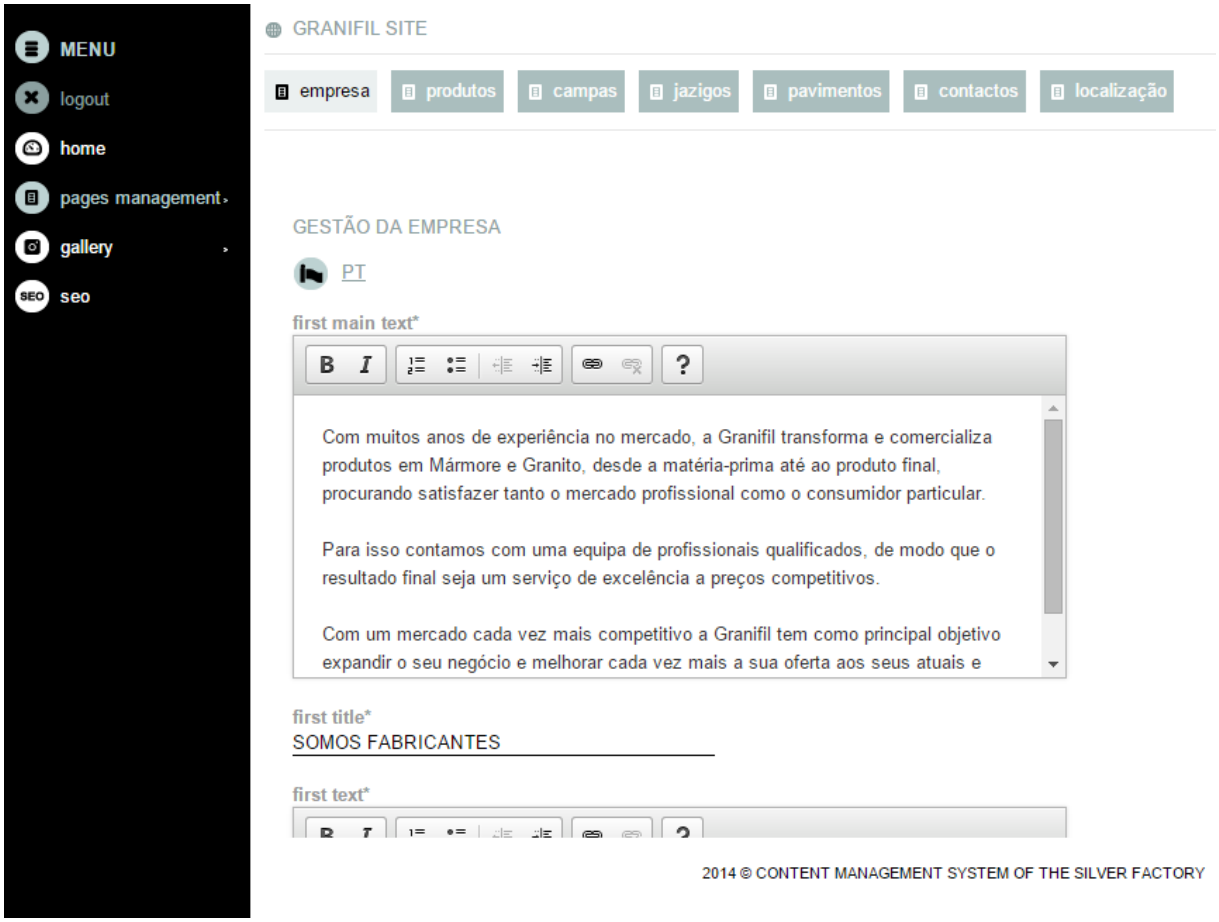


Ilustração 19 - Exemplo do formulário de edição de dados do conteúdo de uma página do *frontend*

Quando se seleciona uma destas páginas ou subpáginas do menu, o utilizador é redirecionado para uma página com um formulário de edição de conteúdo, tal como apresenta a Ilustração 19.

No topo dessa página, é sempre apresentado o título da respetiva página que se está a editar, seguido dos vários *links* para escolha dos vários idiomas, caso existam. De seguida, aparecem então os campos necessários que compõem a página.

Para os campos que necessitassem de uma formatação mais personalizada, tal como o "*bold*", o *"italic",* ou a hiperligação, entre outros, foi utilizado um editor de texto HTML com o nome de CKEditor [80].

O *backoffice* foi desenvolvido de maneira a que para cada página visível em *frontend,* correspondesse uma única tabela na base de dados MySQL para o armazenamento da informação. Um exemplo desta tabela pode ser observado na Ilustração 20. O conteúdo armazenado pode variar

consoante a necessidade, contudo, os campos sempre fixos de cada uma destas tabelas são “meta_description”, “keywords” e “idioma”.

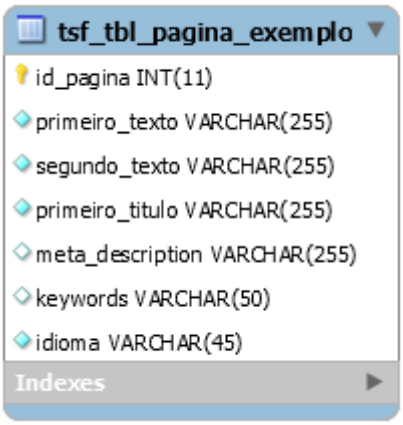


Ilustração 20 - Exemplo de uma tabela para ao armazenamento do conteúdo de uma página

Gallery

Este elemento do menu apresenta uma lista de todas as galerias existentes nas várias páginas do *frontend.*

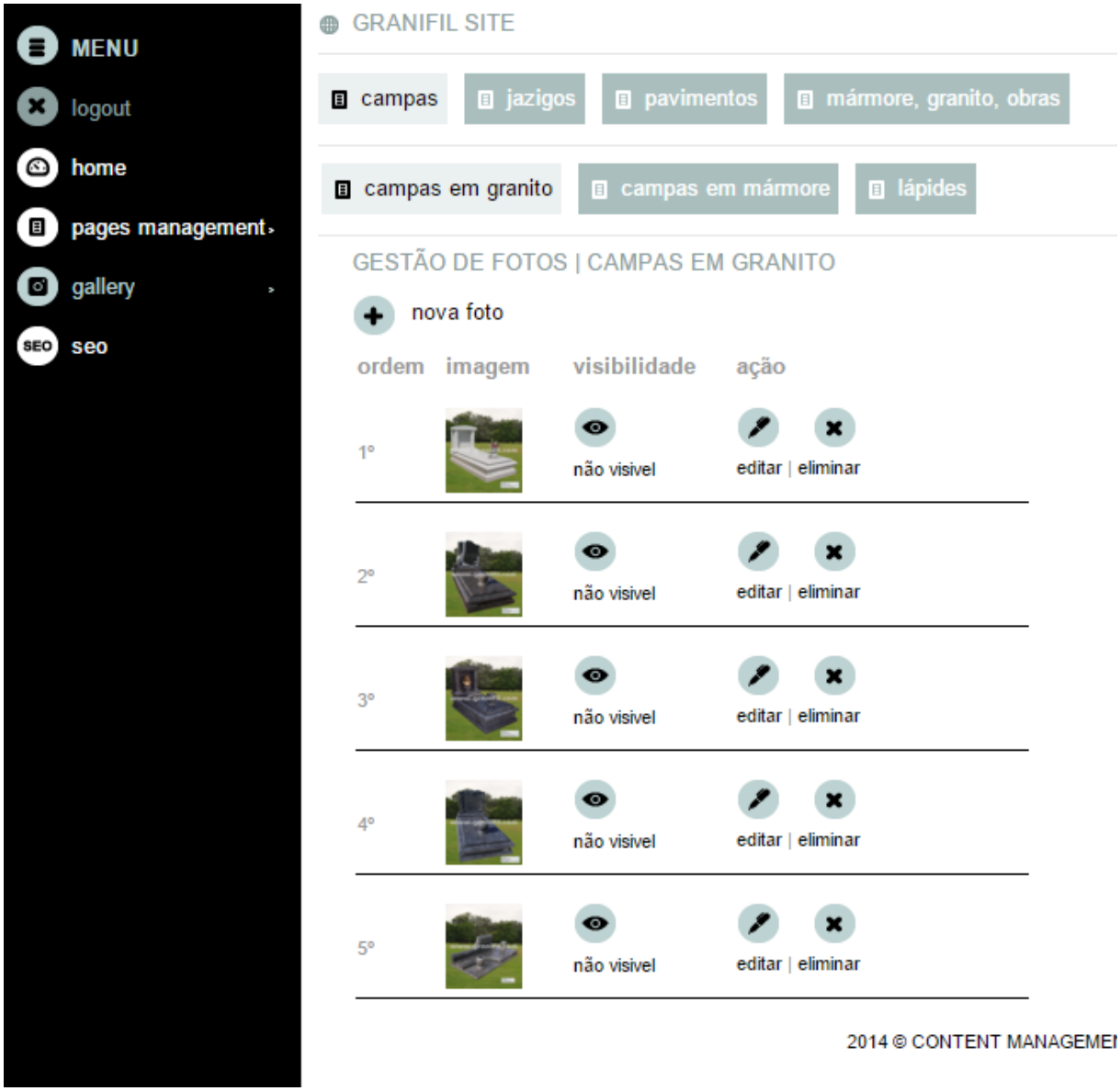


Ilustração 21 - Exemplo da página de uma galeria de imagens

Quando uma das galerias existentes é selecionada, o utilizador é redirecionado para uma página com a listagem de todas as imagens associadas à galeria em questão, tal como apresenta a Ilustração 21.

No topo da página, é sempre apresentado o título da página que contém a galeria bem como uma listagem das imagens constituintes dessa galeria. Nesta listagem é possível eliminar uma imagem e alterar o seu estado de visibilidade no *frontend*. Caso se pretenda alterar uma das imagens, basta selecionar o botão de "editar" que redireciona o utilizador para uma página com o formulário de edição. Caso se pretenda adicionar uma nova imagem, basta selecionar o botão "nova foto".

Quando se adiciona uma nova imagem, é possível escolher a ordem pela qual se pretende que esta apareça no *frontend*, um texto alternativo (*alt*) e ainda um título (*title*). Estes últimos dois campos são importantes para questões de SEO.

Todos os ficheiros inseridos através do *backoffice* são armazenados fisicamente no servidor e apenas os nomes dos mesmos são guardados na base de dados. Um exemplo da tabela responsável por armazenar a informação referente às imagens que constituem uma galeria é apresentado na próxima Ilustração 22.

Como se trata de uma tabela que apenas contém imagens, aquando da inserção de um ficheiro, a aplicação valida a extensão do mesmo, permitindo apenas a inserção dos ficheiros do tipo Portable Network Graphics (PNG) e Joint Photographic Experts Group (JPEG) ou JPG.

Ilustração 22 - Exemplo de uma tabela para o armazenamento de imagens de uma galeria

3.4.3 *Project Management Tool*

Muitos dos trabalhos desenvolvidos durante o estágio foram produzidos para empresas produtoras de moldes. Estes moldes são projetos requisitados por outras empresas e durante a sua produção passam por um conjunto de fases até chegar ao produto final, sendo depois entregue ao cliente. Deste modo, e de maneira a facilitar estas empresas, foi desenvolvido o componente do *Project Management Tool*.

O *Project Management Tool* é uma ferramenta que permite fazer uma gestão de clientes e seus projetos. Esta ferramenta permite a visualização do progresso de fabrico dos vários projetos, utilizando um componente externo com um gráfico de Gantt [81]. Para além dos projetos, é também possível fazer uma gestão de ficheiros, fotos e documentos desses mesmos projetos.

Para o desenvolvimento deste componente foram tidos em conta os requisitos que são a presentados na Tabela 3.

Tabela 3 - Listagem de requisitos funcionais do *Project Management Tool*

Requisitos Funcionais	Prioridade
O sistema deve permitir efetuar autenticação	Alta
O sistema deve proteger as várias rotas com credenciais de acesso	Alta
O sistema deve permitir aos administradores realizar as ações de "CRUD" sobre os outros administradores	Alta
O sistema deve permitir aos administradores realizar as ações de "CRUD" sobre as empresas e respetivos utilizadores	Alta
O sistema deve permitir aos administradores realizar as ações de "CRUD" sobre os projetos	Alta
O sistema deve permitir aos administradores realizar as ações de" CRUD" sobre os desenhos	Alta
O sistema deve permitir aos administradores realizar as ações de "CRUD" sobre os documentos	Alta
O sistema deve permitir aos administradores realizar as ações de "CRUD" sobre as fotos	Alta
O sistema deve permitir aos administradores realizar as ações de "CRUD "sobre as fases	Alta
O sistema deve permitir aos clientes realizar as ações "C" sobre os clientes das empresas	
O sistema deve permitir aos clientes realizar as ações de "CR "sobre as fotos	Alta
O sistema deve permitir aos clientes realizar as ações de "CR" sobre os documentos	Alta
O sistema deve permitir aos clientes realizar as ações de "CR" sobre os desenhos	Alta
O sistema deve permitir aos clientes realizar as ações de "R" sobre os projetos a que estão associados	Alta
O sistema deve permitir aos administradores visualizar ações realizadas pelos clientes	Média
O sistema deve permitir aos clientes visualizar ações realizadas pelos administradores	Média
O sistema deve permitir o envio de emails	Média

Para aceder a esta zona do Project Management, é necessário utilizar o elemento correspondente ao número 5 da Ilustração 18.

Neste componente com base no *login* efetuado, há uma distinção entre o *role* do administrador e do cliente. De acordo com a Tabela 4, tem-se então as várias ações permitidas pelos vários *roles* para cada funcionalidade disponível no *Project Management Tool*.

Tabela 4 - Tabela de ações que cada *role* pode efetuar

Ações	Administrador	Cliente
Gestão de Empresas	CRUD	-
Gestão de Administradores	CRUD	-
Gestão de Fases	CRUD	-
Gestão de Projetos	CRUD	R
Gestão de Fotos	CRUD	CR
Gestão de Documentos	CRUD	CR
Gestão de Desenhos	CRUD	CR

É de salientar que tanto para um administrador como para um cliente, o processo de *login* segue sempre a mesma estrutura, tal como explicado na secção 3.3.2.1 no tópico "Autenticação e autorização". Contudo, com base no *role* do utilizador logado, a apresentação do *backoffice* e dos seus componentes é diferente.

3.4.3.1 Gestão de Empresas

As Empresas são os clientes da empresa responsável pela ferramenta do *Project Management Tool*. Quando se adiciona uma nova empresa no sistema, para além da informação que é necessária preencher no formulário como o nome da Empresa e a sua morada, é também necessário preencher a informação relativa aos dados de acesso para um utilizador dessa Empresa.

Como é possível verificar na Tabela 4, todas as ações referentes à Gestão de Empresas apenas podem ser efetuadas pelo(s) administrador(es) do sistema.

Após a inserção de uma Empresa e do seu primeiro utilizador, este pode efetuar o *login* no *backoffice* e aceder à zona do *Project Management Tool* para efetuar alteações dos seus dados pessoais, bem como visualizar os Projetos da qual faz parte, caso existam. Pode posteriormente também, adicionar Fotos, Desenhos e Documentos a esses mesmos Projetos.

Caso ainda pretenda, na sua área pessoal, o utilizador logado no sistema pode também inserir um novo utilizador para a sua Empresa. Contudo, antes de este puder efetuar o *login*, é necessário primeiro ser ativado por pelo menos um administrador do sistema.

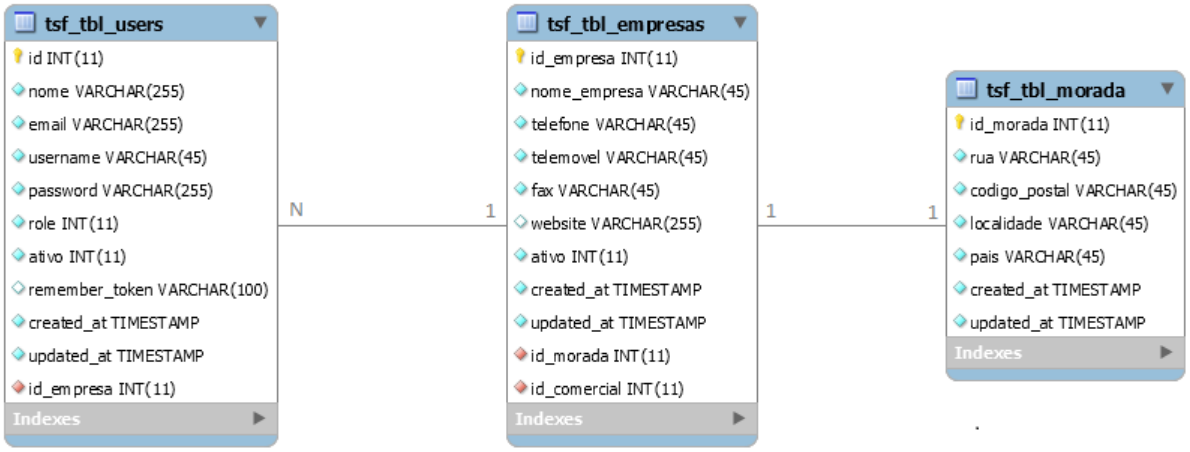


Ilustração 23 - Modelo de Dados da Gestão de Empresas

Para este componente, é necessário acrescentar duas novas tabelas à tabela "tsf_tbl_users", tal como apresentado na Ilustração 23. Deste modo existe uma relação de 1 para N entre a tabela "tsf_tbl_empresas" e a tabela "tsf_tbl_users" e uma relação de 1 para 1 entre a tabela "tsf_tbl_empresas" e a tabela "tsf_tbl_morada".

A entidade "Users" corresponde a um utilizador do sistema, com os campos respetivos ao seu nome, email, *username* e *password*. A entidade "Empresas" corresponde a uma empresa que vai ser registada no sistema, e é constituída por vários utilizadores. Nesta entidade existem os atributos como o nome da empresa, o telefone, telemóvel, entre outros. Por último, temos a entidade "Morada" que corresponde à morada de um "Empresa", onde cada "Empresa" tem uma e apenas uma "Morada" associada. Nesta entidade são então guardados os atributos correspondentes à rua, código postal, localidade e país.

3.4.3.2 Gestão de Administradores

Os administradores são responsáveis por toda a gestão do *Project Management Tool*. De acordo novamente com a Tabela 4, apenas os administradores podem efetuar ações sobre os próprios administradores.

Para esta gestão, apenas se tem em conta a entidade "Users" que já foi previamente explicada na secção 3.3.2.1 no tópico "Autenticação e autorização".

3.4.3.3 Gestão de Fases (templates)

A Gestão de Fases é o requisito mínimo necessário para a criação de um novo Projeto no sistema. É necessário existir um conjunto de fases e subfases já previamente criadas, isto é, um *template*. Nesta gestão é possível então criar as várias fases e subfases e ainda alterar a ordem de apresentação das mesmas.

Na altura em que esta etapa foi desenvolvida, falou-se na existência de um *template* único com todas as fases e subfases, sendo esse o cenário que está implementado. Contudo, numa outra fase de desenvolvimento, foi discutida uma nova abordagem, com a existência de vários *templates*, tendo cada um as suas respectivas fases e subfases. Esta abordagem acabou por não ser sido aprofundada nem desenvolvida porque os clientes assim o desejaram.

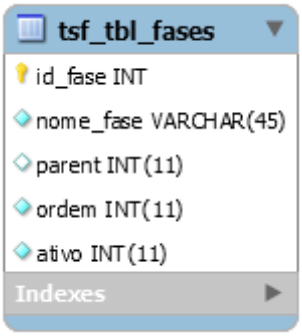


Ilustração 24 - Modelo de Dados da Gestão de Fases

Para esta gestão, como é visível na Ilustração 24, existe apenas uma tabela, que corresponde à entidade "Fases", onde é guardada apenas informação referente ao nome das fases e subfases de um Projeto.

3.4.3.4 Gestão de Projetos

A Gestão de Projetos foi uma funcionalidade implementada no *Project Management Tool* que permite ao(s) administrador(es) adicionar Projetos e associá-los às Empresas e aos respetivos utilizadores. Deste modo, no momento da criação de um novo Projeto, para além da Empresa, é necessário também escolher quais os utilizadores dessa Empresa que poderão estar também associados ao Projeto em questão. Assim sendo, estes utilizadores, quando acederem ao *backoffice* à área correspondente ao *Project Management Tool*, poderão visualizar toda a informação dos seus Projetos.

De acordo com a secção 3.4.3.3, é obrigatória a existência de um *template* das fases e subfases para se poder criar um novo Projeto. Caso esta situação não se verifique, é apresentada uma mensagem de aviso ao utilizador.

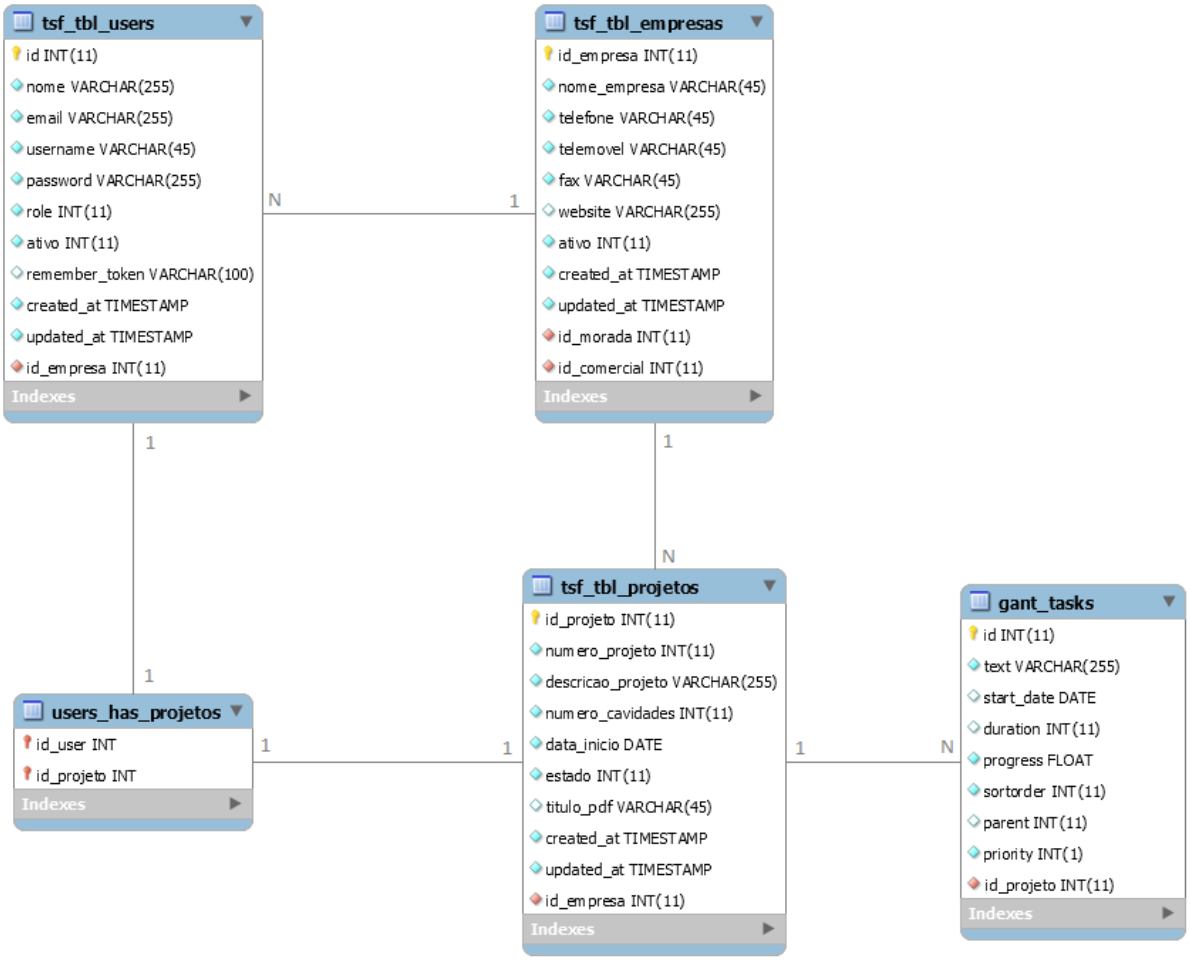


Ilustração 25 - Modelo de Dados da Gestão de Projetos

A gestão de Projetos tem um modelo de dados um pouco mais completo, abrangendo novas entidades, e utilizando outras já existentes. De acordo com a Ilustração 25, a entidade principal tem o nome de "Projetos", que tal como o próprio nome indica, é referente a um projeto faz parte do sistema. Este projeto tem um número de identificação, uma descrição, um número de cavidades e ainda uma data de início.

Uma empresa estabelece uma relação de 1 para N com a entidade projetos. Deste modo, no momento da criação de um novo projeto, é possível selecionar uma empresa à qual se quer associar o projeto, e por conseguinte apenas os utilizadores apenas dessa empresa. Posto isto, um utilizador tem uma relação de N para M com a entidade projetos, onde um utilizador pode estar associado a mais que um projeto, do mesmo modo que um projeto pode pertencer a mais que um utilizador.

Por último, esta entidade projeto ainda tem uma relação de 1 para N com a entidade das tarefas "*gant_tasks*", que representa todas as tarefas que fazem parte de um projeto. São essencialmente as fases e subfases referidas na secção 3.4.3.3, ou seja, no momento da inserção de um novo Projeto no sistema, a aplicação internamente obtém todas as fases e subfases presentes na tabela "tsf_tbl_fases" e adiciona-as à tabela "gant_tasks" para o "id_projeto" correspondente.

Um exemplo de uma listagem de projetos é apresentado na próxima Ilustração 26. Nesta tabela há a possibilidade de selecionar a opção de editar um projeto, onde o utilizador é redirecionado para uma página com um formulário, e nele pode editar os campos como o nome e descrição do projeto, e

ainda os vários utilizadores que fazem parte do mesmo. Há também a possibilidade de remover um projeto ou de o fechar, dando por completa a sua produção. Para se proceder à visualização do gráfico de Gantt de um projeto, basta selecionar o *link* com o nome "*see progress report*".

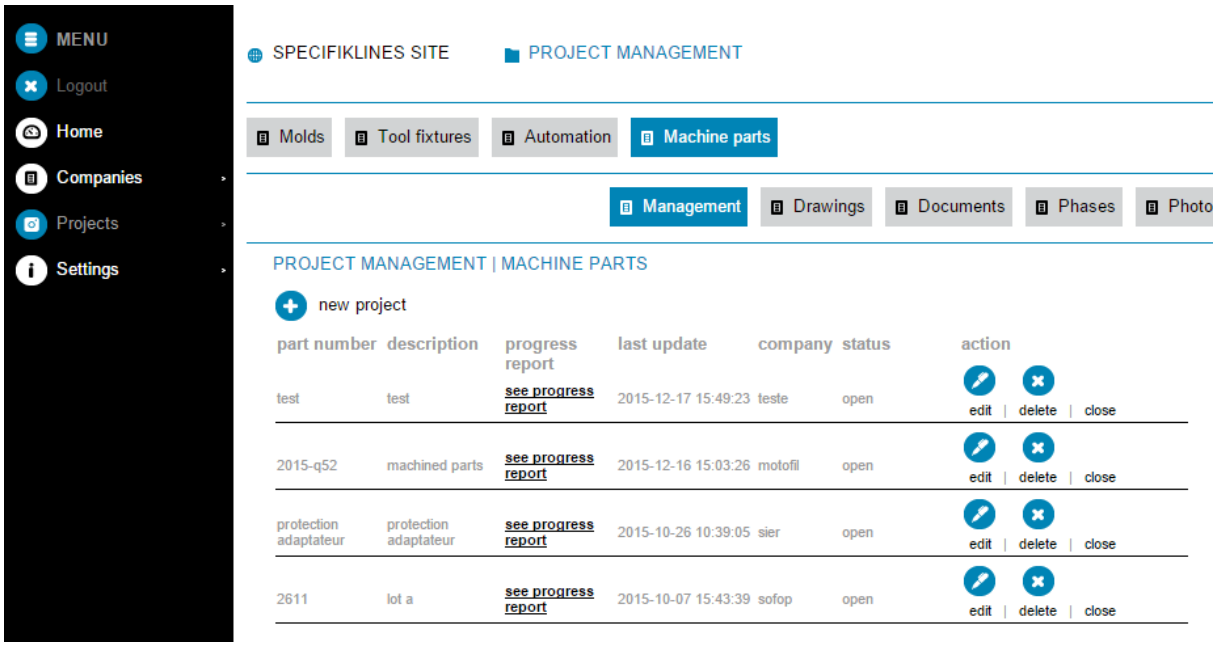


Ilustração 26 - Exemplo de uma listagem dos vários projetos existentes no sistema

Gráfico de Gantt

O gráfico de Gantt foi um elemento necessário a implementar no *Project Management Tool*. Para tal, foi utilizado um componente existente com o nome de dhtmlxGantt [82]. O dhtmlxGantt é um gráfico de Gantt desenvolvido em JavaScript que permite mostrar as dependências entre as várias tarefas e atribuir diferentes relações entre as mesmas. Disponibiliza uma API flexível e um elevado número de eventos que dão uma grande liberdade para personalizar o gráfico às várias necessidades.

Para integrar este gráfico no componente do *Project Management Tool*, foi necessário efetuar o seu *download*, extrair os ficheiros e coloca-los no interior da pasta do projeto. Na vista responsável por apresentar o gráfico, deve fazer-se o *include* de dois ficheiros essenciais:

- DhtmlxGantt.js – Ficheiro que contém toda a lógica para a configuração do gráfico com a respetiva API;
- DhtmlxGantt.css – Ficheiro que contém todos os estilos para a apresentação do *layout*;

Existem duas maneiras possíveis de carregar a informação no gráfico de Gantt. Com conteúdo estático utilizando um objecto JavaScript Object Notation (JSON) declarado em JavaScript, ou com conteúdo dinâmico obtido com uma ligação à base de dados local.

Por uma questão de persistência de dados, foi apenas abordado o processo dinâmico para o carregamento de informação para o gráfico. Como tal, é necessário criar uma tabela para armazenar a informação das tarefas e subtarefas. Esta tabela já foi mencionada na secção 3.4.3.4, e tem o nome de "gant_tasks".

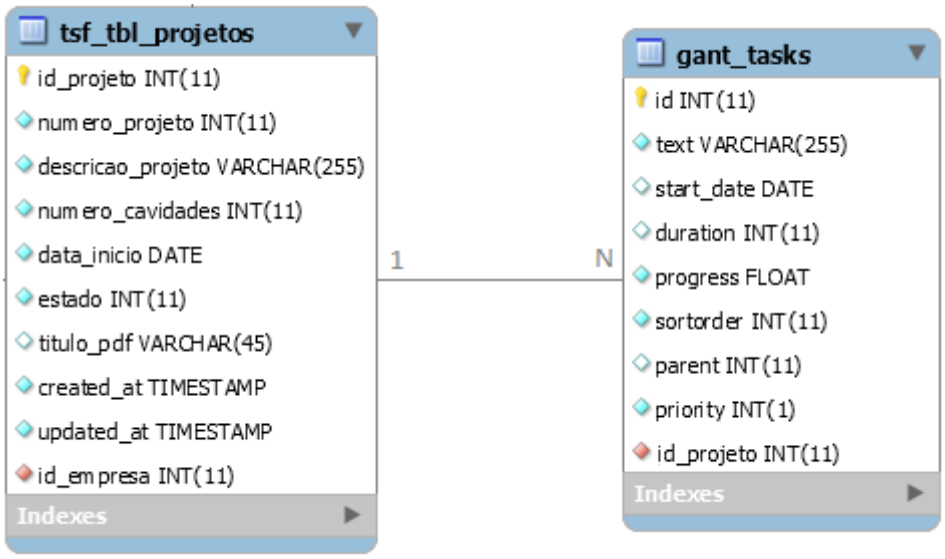


Ilustração 27 – Tabela "tsf_tbl_projetos" e "gant_tasks"

Como foi dito anteriormente no capítulo 3.4.3.4 (Gestão de Projetos), e como é também possível verificar pela Ilustração 27, um projeto é constituído por várias tarefas. Esta relação é estabelecida através da coluna "id_projeto" que corresponde à chave estrangeira de um projeto.

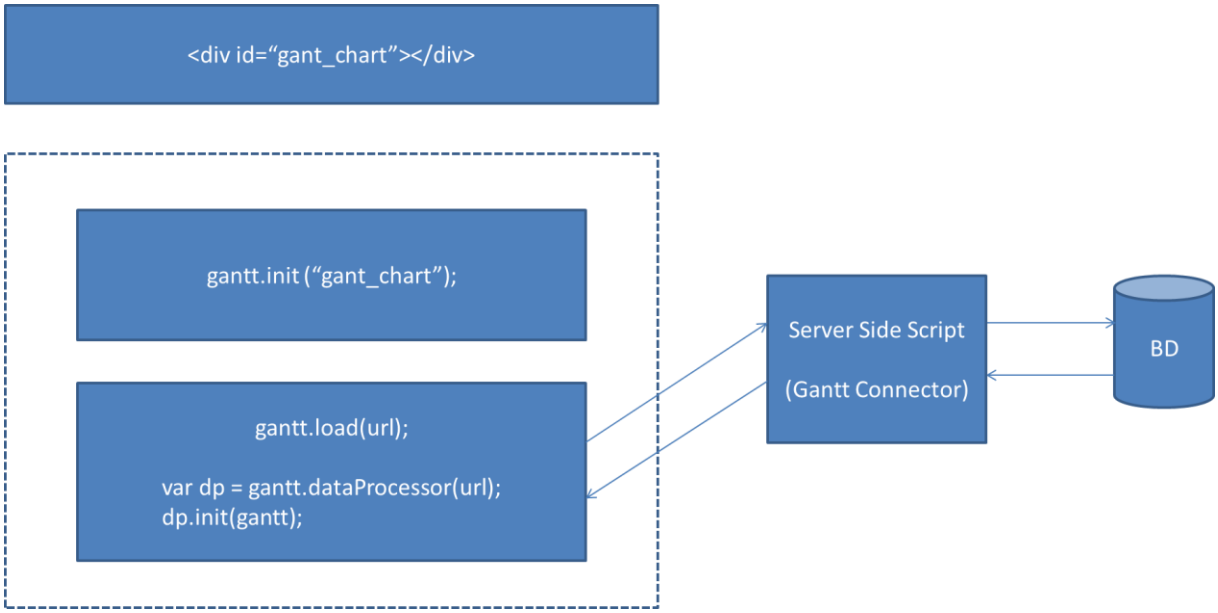


Ilustração 28 - Diagrama do processo de inicialização do componente dhtmlxGantt

Para melhor entender o processo de inicialização do componente dhtmlxGantt, temos então a Ilustração 28, onde primeiramente é necessário existir um elemento estrutural HTML que irá conter o gráfico, neste caso uma *div*. De seguida, é necessário configurar o formato dos dados provenientes da base de dados, de maneira a que o *output* dos mesmos seja compatível com o formato do dhtmlxGantt. Só depois, é que se pode inicializar o objecto dhtmlx.

Por último, utiliza-se o método *load* [83] disponível pela Gantt API [84], para efetuar a ligação ao *script* do lado do servidor. Este método realiza um pedido AJAX do tipo GET ao *url* do ficheiro que contém então o *script* do lado do servidor, tal como é possível verificar na Listagem 28.

```
<!-- Div container do Gráfico de Gantt -->
<div id="gantChart" style='width:1500px; height:600px;'></div>
<script type="text/javascript">
    // configurar o formato da data
    gantt.config.xml_date="%Y-%m-%d";

    // inicialização do objecto dhtmlx
    gantt.init("gantChart");

    // carregar o gráfico com dados provenientes da base de dados
    gantt.load("path_to_file/data.php?id="+{{ $id }});
</script>
```

Listagem 28 - Código responsável inicializar o dhtmlxObject

O *script* do lado do servidor é responsável por receber um parâmetro GET com o nome "id", que corresponde ao ID do projeto à qual se pretende fazer *render*. De seguida, utiliza-se o *connector* necessário, que neste caso é o "Gantt Connector", para efetuar a ligação ao lado do servidor. Depois de a ligação estar estabelecida, utiliza-se o ID para efetuar um SELECT à base de dados, por todas as tarefas e subtarefas correspondentes ao ID do projeto em causa. E por último, utiliza-se o método "render_sql" [85], disponível pela PHP Connector API [86] para configurar o *connector* e obter a informação de acordo com a *$query* realizada, tal como é possível visualizar na Listagem 29.

```
<?php
    include ('../../connector/gantt_connector.php');

    /* ID do Projeto */
    $id = $_GET['id'];

    $res=mysql_connect("localhost","root","");
    mysql_select_db("progressive_db");

    $gantt = new JSONGanttConnector($res);

    /* Select à base de dados pelas tarefas correspondentes ao projeto em causa */
    $query = "SELECT * FROM gantt_tasks WHERE projetos_id_projeto =".$id;

    /* Opção que faz com que todas as tarefas e subtarefas estejam expandidas aquando o render do gráfico de gantt */
    $gantt->mix("open", 1);

    /* Configura o connector e devolve a informação baseada numa expressão SQL */
    $gantt->render_sql($query,"id","start_date,duration,text,progress,sortorder,parent,projetos_id_projeto");
?>
```

Listagem 29 - Código correspondente ao ficheiro responsável por efetuar a ligação ao lado do servidor

Com este processo, já é possível visualizar o gráfico de Gantt, contudo, para este componente, ainda era necessário efetuar mais uma configuração, de maneira a que qualquer alteração efetuada no gráfico de Gantt fosse diretamente armazenada na base de dados. Para tal, é necessário utilizar o DataProcessor [87], uma biblioteca que permite comunicar com o lado do servidor quando é efetuada uma qualquer ação no gráfico de Gantt. A sua configuração é visível na Listagem 30.

```
// inicialização do dhtmlxDataProcessor para updates à tabela da base de dados
var dp=new gantt.dataProcessor("path_to_file/data.php?id="+{{ $id }});
dp.init(gantt);
```

Listagem 30 – Inicialização do dhtmlxDataProcessor para realizar updates na tabela da base de dados

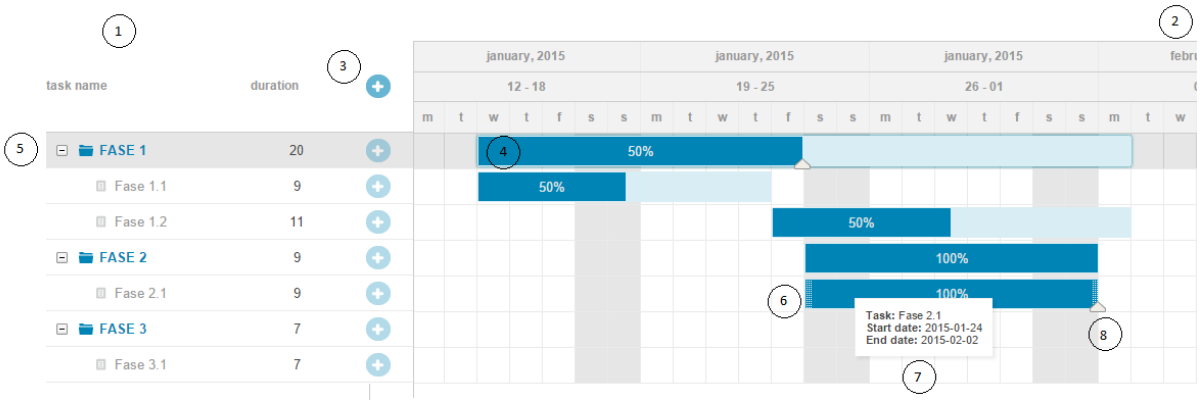


Ilustração 29 - Exemplo de um gráfico de Gantt com o componente dhtmlxGantt

O resultado final do gráfico de Gantt é apresentado pela Ilustração 29. Nele existem vários pontos que são importantes para melhor se compreender o bom funcionamento do mesmo. Esses pontos são os seguintes:

1. Zona correspondente ao eixo dos Y com os nomes de todas as tarefas e respetiva duração;
2. Zona correspondente ao eixo dos X com a escala do tempo;
3. Botão "+" que permite em tempo real a interação com o gráfico para adicionar uma nova tarefa ou subtarefa;
4. Barra que representa uma tarefa. Apresenta também o valor do seu progresso em %;
5. *Highlight* de uma tarefa selecionada;
6. Parte da barra que é possível arrastar, aumentando ou diminuindo a duração da tarefa correspondente;
7. Tooltip com informação adicional sobre a tarefa correspondente;
8. Ícone que pode ser arrastado para alterar o valor do progresso da tarefa.

Neste gráfico, é possível realizar uma ação responsável por adicionar uma nova tarefa ou subtarefa. Esta ação acontece quando o elemento correspondente ao número 3 é selecionado, apresentando um formulário, tal como o visível na Ilustração 30.

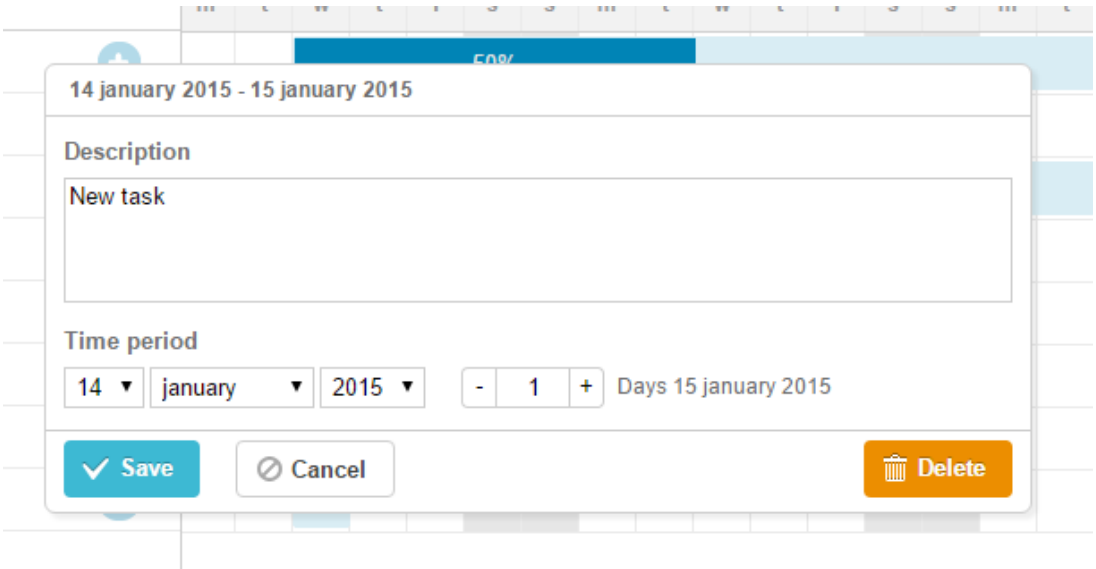


Ilustração 30 - Formulário de inserção de uma nova tarefa/subtarefa

Contudo, ao adicionar uma nova tarefa ou subtarefa, por si só, não há nenhuma indicação a que Projeto estas tarefas vão ser adicionadas, pois por omissão, estas tarefas apenas são adicionadas à tabela "gant_tasks". Para se especificar então a que número de Projeto a que se pretende associar estas tarefas, utiliza-se o evento "onBeforeTaskAdd" [88] , disponível pela Gantt API, tal como é possível verificar pela Listagem 31.

```
/* Evento que será executado quando se pressionar o botão "Save", antes da tarefa ser realmente adicionada */
gantt.attachEvent("onBeforeTaskAdd", function(id,item){
    item.projetos_id_projeto = {{ $id }};
});
```

Listagem 31 - Código referente ao evento "onBeforeTaskAdd"

Este componente dhtmlxGantt permite também efetuar a exportação dos seus dados em formato Portable Document Format (PDF) e PNG.

Caso o utilizador logado no sistema tenha o *role* de administrador, na página referente ao gráfico de Gantt, terá à sua disposição um botão com a funcionalidade de notificar todos os utilizadores (cliente) que estão associados ao Projeto. Esta notificação é efetuada com o envio de email.

Caso o *role* do utilizador logado no sistema seja o de cliente, é necessário adicionar um conjunto de novas configurações ao gráfico de Gantt de maneira a proibir o utilizador de fazer quaisquer ações no mesmo, excepto o da exportação da informação para um ficheiro PNG ou PDF, tal como é possível verificar na Listagem 32.

```
<script type="text/javascript">
    gantt.config.drag_move = false; //Impossibilita mover as tarefas
    gantt.config.drag_progress = false; //Impossibilita mexer no progresso de cada tarefa
    gantt.config.drag_resize = false; //Impossibilita fazer resize nas tarefas, isto é, aumentar ou diminuir a sua duração
    gantt.config.details_on_dblclick = false; //Impossibilita o evento de double click nas tarefas
    gantt.config.details_on_create = false; //Impossibilita a possibilidade de adicionar novas tarefas e subtarefas
    gantt.config.select_task  = false; //Impossibilita a seleção de tarefas e subtarefas
</script>
```

Listagem 32 - Código referente às configurações para o *role* de cliente

3.4.3.5 Gestão de Desenhos, Documentos e Fotos

A gestão de Desenhos, Documentos e Fotos são funcionalidades que apenas podem ser efetuadas quando existem Projetos adicionados no sistema. Novamente, como é possível verificar na Tabela 4, o(s) administrador(es) pode(m) fazer toda a gestão destes ficheiros, contudo os clientes apenas os podem adicionar, realçando o facto de que apenas podem adicionar e visualizar os ficheiros associados aos Projetos da qual fazem parte.

Ilustração 31 - Modelo de Dados da Gestão de Desenhos, Fotos e Documentos

Para o modelo de dados apresentado pela Ilustração 31, existem três novas entidades, todas elas relacionadas com a entidade Projeto. Estas entidades são então “Desenhos”, “Fotos” e “Documentos”, que representam os desenhos, fotos e documentos que um Projeto pode ter, respetivamente. Existe uma relação de 1 para N entre a entidade Projetos e quaisquer umas das três novas entidades. Ambas têm na sua constituição campos semelhantes, tal como a descrição, o *url* do ficheiro, uma data de expiração, uma data de aviso para a remoção e ainda um comentário.

Tal na secção 3.4.3 no tópico “Gallery”, os vários desenhos, documentos e fotos aquando do momento da sua inserção, sofrem uma verificação da sua extensão, sendo apenas inseridos aqueles que contêm a extensão permitida. Os que não tiverem uma extensão válida, são apresentados ao utilizador, para que possam ser alterados.

No momento da inserção de imagens ou de documentos (excluindo as fotos), é possível inserir ficheiros individuais ou um conjunto de ficheiros, isto é, um ficheiro ZIP. Neste caso, a aplicação

descompacta automaticamente o ficheiro ZIP, e todos os ficheiros que forem encontrados dentro desse ficheiro ZIP, se tiverem também a extensão válida, são automaticamente inseridos no sistema.

Ao adicionar os vários ficheiros é necessário preencher o campo da data de remoção. Este campo corresponde à data em que os mesmos vão ser removidos do sistema, e trata-se de um campo obrigatório. Associado a este campo, existe um outro com o nome de "numero_dias_aviso" onde se coloca o número de dias que se pretende que o(s) cliente(s) seja(m) notificado(s) antes do(s) ficheiro(s) ser(em) automaticamente removido(s) do sistema, caso ainda não tenha sido efetuado o seu *download.* Se este campo não for preenchido, por defeito, o sistema atribui quinze dias.

No caso de a inserção destes ficheiros ser efetuada por parte de um cliente, estes dois campos não aparecerem no formulário, sendo atribuído por omissão à data de remoção, trinta dias, e quinze dias ao número de dias de aviso.

As funcionalidades de remoção automática dos vários ficheiros do sistema, bem como o envio de emails para os clientes a informar que estes ficheiros irão ser removidos do sistema, foram desenvolvidas com o uso de uma ferramenta online chamada de SetCronJob [89]. Nesta ferramenta, foram adicionados dois URL's do *website* em questão para serem executados automaticamente uma vez por dia, todos os dias da semana.

Houve também a necessidade de guardar as ações efetuadas pelos administradores para os clientes, e vice-versa. Para tal foi criada uma tabela, tal como é visível na Ilustração 32, que guarda informações referentes às ações realizadas tanto pelos administradores como pelos clientes do sistema. Estas ações são por exemplo a adição, edição ou remoção de projetos, ou então a adição, edição ou remoção de ficheiros (desenhos, documentos e fotos).

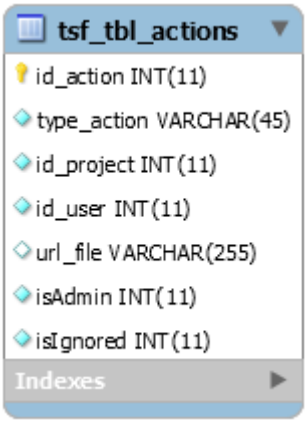


Ilustração 32 - Tabela correspondentes às ações efetuadas na aplicação

Estas ações são apresentadas, tanto para os administradores como para os clientes sob a forma de uma tabela (*grid*), e nela é possível ignorar as notificações; no caso de existirem ficheiros, fazer o *download* dos mesmo, e no caso de ser adicionar um novo projeto, ou até a sua edição, aceder ao *url* disponível para verificar o gráfico de Gantt do mesmo, tal como aparece na Ilustração 33.

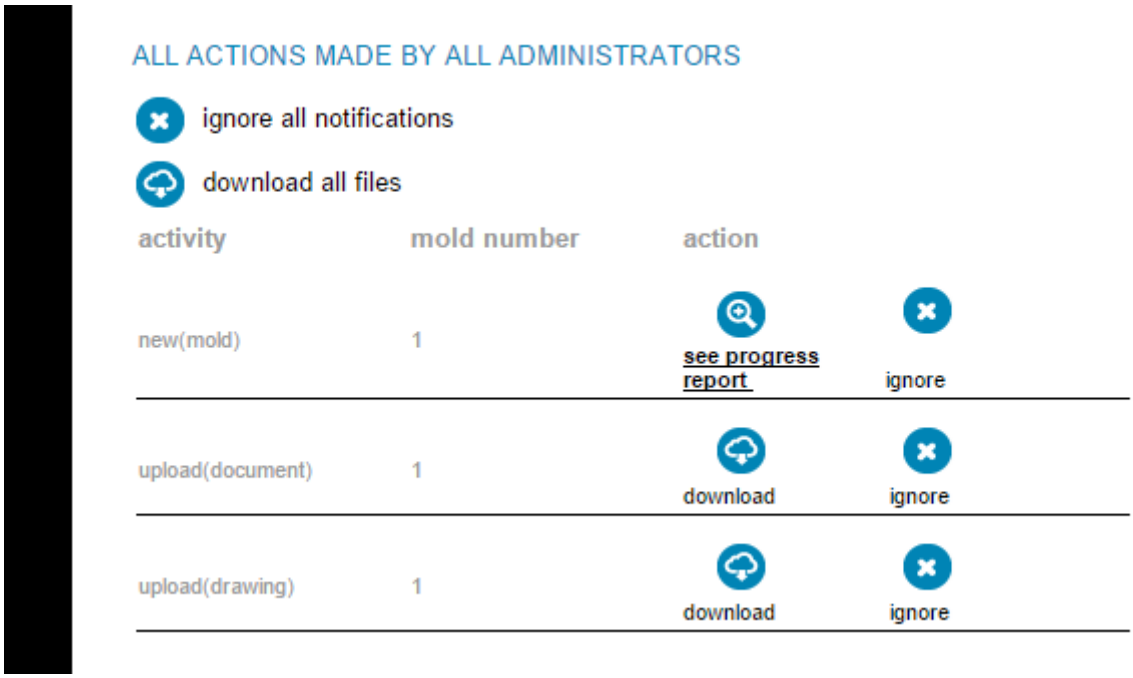


Ilustração 33 - Exemplo de ações efetuadas pelos administrador

3.4.3.6 Gestão de Comerciais

A funcionalidade associada à gestão de Comerciais não tinha sido planeada no início da realização do componente, tendo sido por isso mais tarde adicionada. Os Comerciais são entidades que podem estar associadas a uma Empresa existindo a relação de 1 para 1 entre as duas entidades, sem a participação obrigatória da entidade correspondente ao Comercial.

Foi então acrescentada uma nova tabela à gestão de Empresas, tal como apresentado na Ilustração 34. Nesta tabela é armazenado o nome e o email do Comercial.

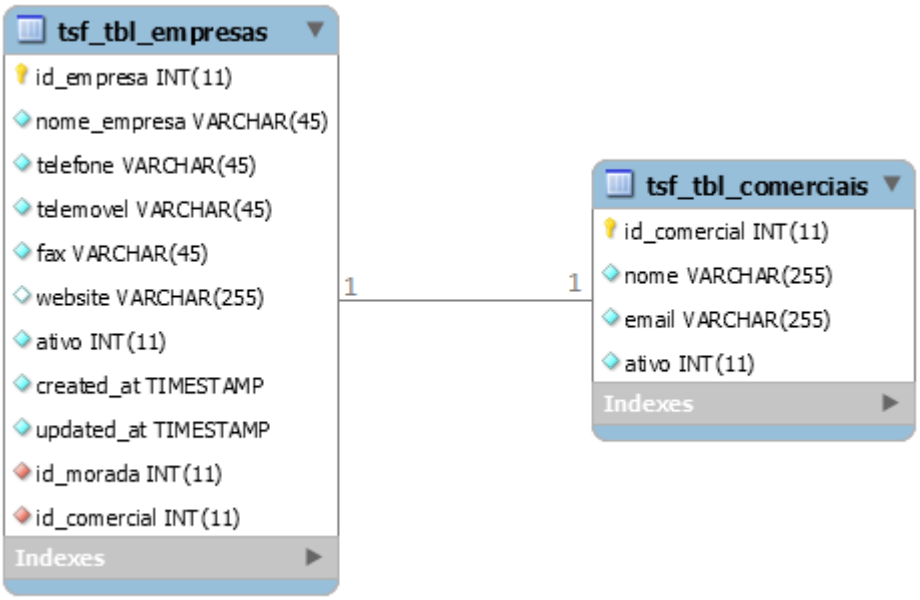


Ilustração 34 - Modelo de Dados da Gestão de Comerciais

Quando alguma ação referente a uma Empresa é efetuada, como por exemplo a inserção/edição ou remoção de Projetos, fotos, desenhos, etc, se existir um Comercial associado, este é notificado da respetiva ação via email.

3.4.4 Gestão de Documentos e Ficheiros (Área Reservada)

A Gestão de Documentos e Ficheiros é um componente do *backoffice* constituído por uma área reservada onde é possível ao administrador do sistema inserir vários ficheiros ou documentos e associa-los aos vários utilizadores. Os utilizadores apenas podem fazer o *download* dos ficheiros ou documentos que lhes estão associados.

Para o desenvolvimento deste componente foram impostos alguns requisitos que foram tidos em conta. Segue-se de seguida a Tabela 5 que ilustra os requisitos funcionais do componente desenvolvido.

Tabela 5 - Tabela de Requisitos Funcionais da componente da Área Reservada

Requisitos Funcionais	Prioridade
O sistema deve permitir efetuar autenticação	Alta
O sistema deve proteger as várias rotas com credenciais de acesso	Alta
O sistema deve permitir ao administrador realizar ações "CRUD "sobre os ficheiros ou documentos	Alta
O sistema deve permitir obter a lista de todos os utilizadores	Alta
O sistema deve permitir ao utilizador visualizar os documentos associados ao seu *role*	Alta
O sistema deve permitir ao utilizador fazer o *download* de ficheiros	Alta
O sistema deve permitir ao utilizador fazer a alteração dos dados pessoais	Baixa

Após a identificação dos vários requisitos funcionais, foi necessário criar um modelo de dados que melhor se enquadrasse às necessidades deste componente.

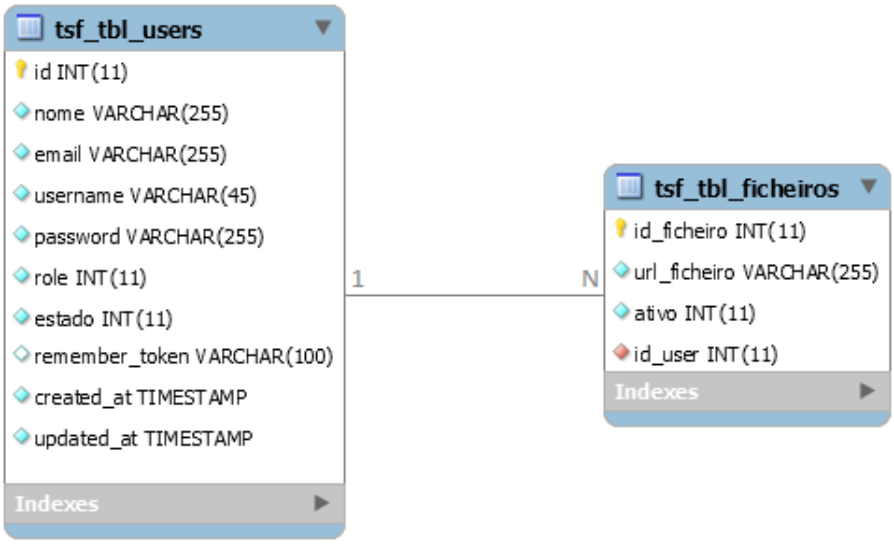


Ilustração 35 - Modelo de Dados do componente da Gestão de Documentos e Ficheiros

Para o modelo de dados apresentado pela Ilustração 35, existem duas entidades, a dos utilizadores que já foi previamente explicada, e a entidade correspondente aos vários ficheiros inseridos no sistema. Existe uma relação de 1 para N entre as duas entidades, ou seja, um utilizador pode ter vários ficheiros associados, contudo, um ficheiro apenas pode pertencer a um utilizador.

Este componente apenas foi utilizado em um projeto, e a pedido do cliente, existiam três utilizadores, cada um com o seu único *role, username* e *password*. Deste modo, existiam então três *roles* para o acesso à ferramenta, o *role* igual a 2 para "distribuidores", igual a 3 para "instaladores" e igual a 4 para "arquitectos, engenheiros e designers". Assim sendo, o cliente não tinha possibilidade de adicionar novos utilizadores. Esta funcionalidade apenas era permitida via código.

Como se trata de um componente com distinção de *roles*, é necessário haver um controlo de acessos para as várias rotas. Este controlo foi efetuado com a utilização de filtros, disponíveis pela *framework*, que já foram explicados na secção 3.3.2.1 no tópico "Autenticação e autorização".

Posto isto, se um utilizador efetuar com sucesso o *login* no sistema, de acordo com o *role* do mesmo, apenas lhe é apresentado a lista de ficheiros que lhe estão associados. Nesta lista de ficheiros é possível efetuar o *download* individual de cada ficheiro, ou de todos os ficheiros, de uma vez, tal como é visível na Ilustração 36.

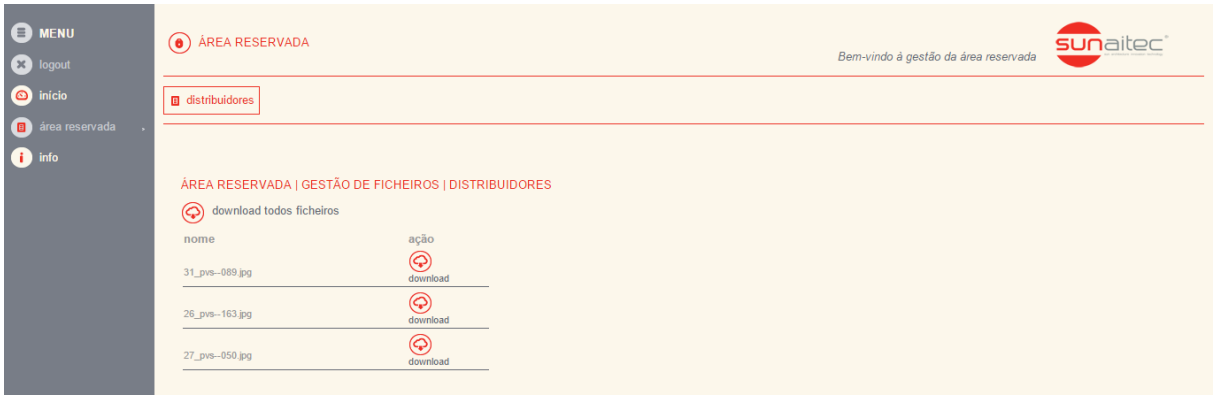


Ilustração 36 - Lista de ficheiros associados ao utilizador com o *role* de "distribuidor"

3.4.5 Dashboard – Analytics

A melhor maneira de compreender e optimizar o uso da *web* é através do Web Analytics. O Web Analytics fornece um conjunto de informações acerca de quem visita o *website*, o que os utilizadores procuram, e ainda como foram ter ao *website*. Toda a informação obtida através de uma ferramenta de análise é usada para melhorar o *website* e oferecer aos utilizadores o que eles procuram.

O *Dashboard* foi outro componente que foi desenvolvido com o intuito de ser possível, no interior do *backoffice*, visualizar informação obtida pelo Google Analytics, referente ao *website* onde o componente foi adicionado.

3.4.5.1 Google Analytics

O Google Analytics [90] é a aplicação de análise mais utilizada que ajuda a entender o padrão de comportamento dos visitantes de um *website* [91].

Para usar o Google Analytics é necessário registar uma conta no *website* do Google Analytics, e adicionar um *website* a essa conta. Depois dessa associação, é gerado um código com o seguinte formato, "UA-XXXXXXXX-X". Este código deve ser colocado em todas as páginas do *website* de onde se quer obter a informação.

3.4.5.2 API Google Analytics

A API do Google Analytics (*Analytics API*), tal como outras, requer autenticação e autorização quando utilizada numa aplicação. Isto pressupõe uma interação por parte do utilizador aquando do aparecimento de um *consent screen* OAuth [92], ou seja um ecrã onde é pedida autorização ao utilizador para prosseguir/recuar com o pedido. Contudo, este não era o cenário pretendido, dado não ser a maneira mais cómoda e transparente para o utilizador. Deste modo, utilizou-se o OAuth2 [93] para aplicações *server-to-server*. Para este cenário, é necessário utilizar uma *service account* [94]. Com isto, há uma conta associada à aplicação, em vez de um indivíduo ou utilizador.

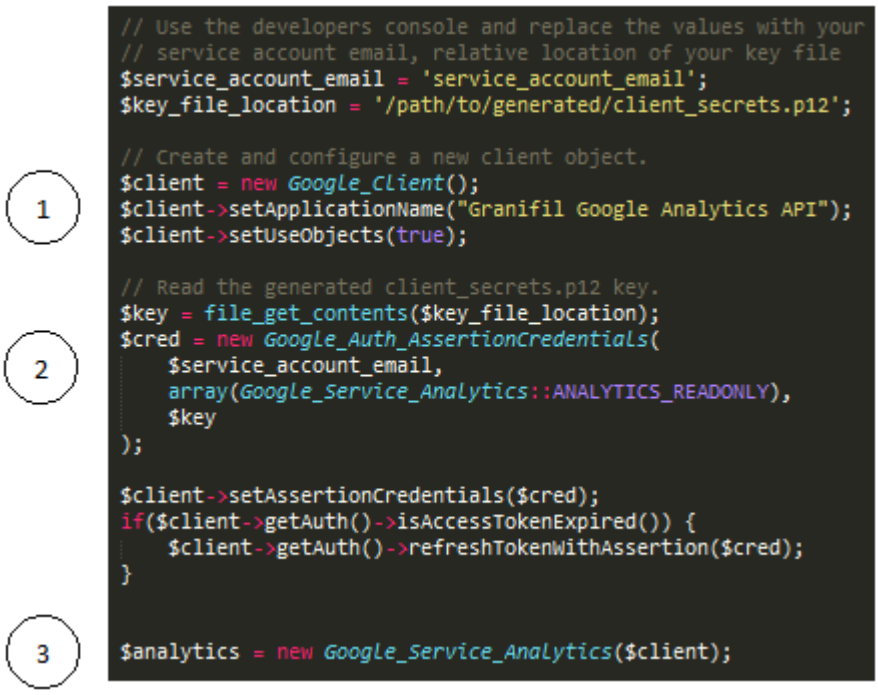
Uma *service account* é constituída por um email gerado automaticamente, que é único, por um ID de cliente e pelo menos um par de chaves publica/privada. Para criar as credenciais de uma *service account* é necessário seguir os seguintes passos:

- Aceder à zona de *Google Developers Console* [95];
- Selecionar um projeto, ou criar um de novo;
- Depois de selecionado o projeto, aceder a "Ative e gerencie APIs" e escolher a API que se quer utilizar. Neste caso foi a *Analytics API;*
- Na página onde foi activada uma API, carregar em "Ir para Credenciais"*;*
- Escolher a opção "Novas Credenciais", e escolher "Chave da conta de serviço" (*Service Account)*
- Escolher o tipo de chave, JSON ou P12. No caso de P12, que foi o tipo utilizado, deve-se guardar num local seguro a chave que irá ser gerada, para posterior utilização;
- Em *Permissions* atribuir todas as permissões ao email gerado da *service account*

Como neste caso se trata da *Analytics API*, é necessário ainda na conta do Google Analytics adicionar o novo email correspondente à *service account*, e atribuir-lhe todas as permissões [96]. Deste modo,

quando a aplicação fizer a chamada à API, não haverá qualquer problema de autenticação/autorização.

Como o componente do Dashboard – Analytics foi introduzido em um *backoffice* desenvolvido em PHP, mais propriamente com a *framework* Laravel, é necessário adicionar às dependências a *Google APIs Client Library For PHP* [97] , através da utilização do Composer [98].



Listagem 33 - Código necessário para efetuar uma ligação aos Servidores da Google

De acordo com a Listagem 33, inicialmente é necessário criar um novo objeto PHP (*$client*) que será depois utilizado para atribuir o nome da aplicação e definir a utilização de objetos, em vez de *arrays* associativos, como é visível pelo ponto 1.

De seguida temos o ponto 2, onde, com a utilização da chave privada e do email associado à *service account*, presentes na "*Developers Console",* a aplicação internamente realiza os seguintes passos:

- Criar um JSON Web Token (JWT)
- Usar o JWT para fazer um pedido ao serviço do Google (pedir um *access token*)
- Tratar da resposta dada pelo *Google Server*

Se a resposta incluir um *access token* a aplicação usa esse mesmo *token* para chamar a respetiva API dos serviços da Google. Este exemplo é melhor ilustrado através da Ilustração 37.

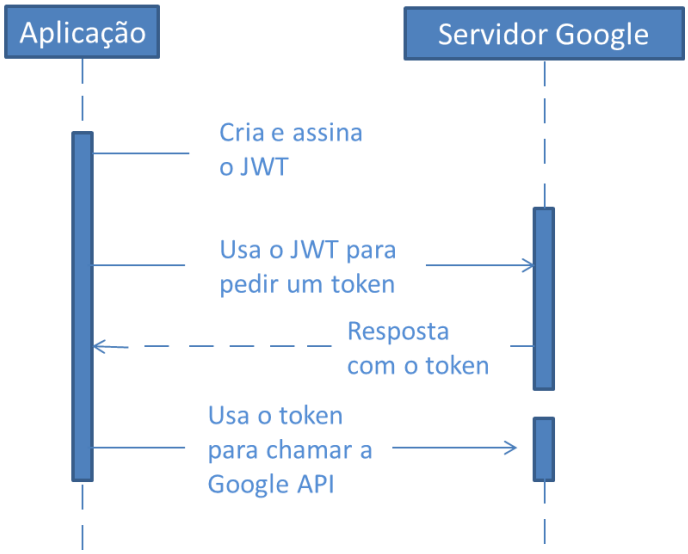


Ilustração 37 – Exemplo de uma chamada à API por parte da aplicação (adaptado de [94])

De uma maneira transparente ao programador, para evitar erros que podem ter um impacto grave na segurança da aplicação, a aplicação cria e criptograficamente assina um JWT, com o uso do Google *APIs Client Library* [94].

Com este processo, a aplicação está autorizada a efetuar chamadas à API.

De seguida, com a criação do novo *analytics service object*, visível na Listagem 34 no ponto 3, utiliza-se a Management API [99], que permite aceder aos dados de configuração do Google Analytics.

3

```php
$analytics = new Google_Service_Analytics($client);


function getFirstprofileId(&$analytics) {
    // Get the user's first view (profile) ID.

    // Get the list of accounts for the authorized user.
    $accounts = $analytics->management_accounts->listManagementAccounts();

    if (count($accounts->getItems()) > 0) {
        $items = $accounts->getItems();
        $firstAccountId = $items[0]->getId();

        // Get the list of properties for the authorized user.
        $properties = $analytics->management_webproperties
            ->listManagementWebproperties($firstAccountId);

        if (count($properties->getItems()) > 0) {
            $items = $properties->getItems();
            $firstPropertyId = $items[0]->getId();

            // Get the list of views (profiles) for the authorized user.
            $profiles = $analytics->management_profiles
                ->listManagementProfiles($firstAccountId, $firstPropertyId);

            if (count($profiles->getItems()) > 0) {
                $items = $profiles->getItems();

                // Return the first view (profile) ID.
                return $items[0]->getId();

            } else {
                throw new Exception('No views (profiles) found for this user.');
            }
        } else {
            throw new Exception('No properties found for this user.');
        }
    } else {
        throw new Exception('No accounts found for this user.');
    }
}
```

4

Listagem 34 - Utilização da Management API para a obtenção do primeiro *profile ID*

De acordo com o ponto 4, primeiramente obteve-se a lista de contas para o utilizador autorizado. Desta lista de contas (*$accounts*), restringiu-se apenas à primeira conta, para obter o seu ID (*$firstAccountId*). De seguida, com este *$firstAccountId* obteve-se uma lista de propriedades (*$properties*), novamente para o utilizador autorizado. Desta lista, restringiu-se apenas à primeira propriedade, para também obter o seu ID (*$firstPropertyId*). Por último, com o ID da primeira conta mais o ID da primeira propriedade, obteve-se uma lista de *view (profiles)* também do utilizador autorizado. Desta lista, apenas se pretende a primeira *view (profile) ID.*

Este *Profile ID* é essencial para depois se utilizar a Core Reporting API [100].

3.4.5.3 Core Reporting API

Após a obtenção do Profile ID, já é possível utilizar a Core Reporting API que devolve informação proveniente dos dados obtidos através do código de rastreamento do Google Analytics. Para obter esta informação é necessário utilizar *queries* que são constituídas por um conjunto de parâmetros que especificam que e como os dados vão ser devolvidos.

Um dos mais importantes parâmetros da *query* é o "*ID*" ou o "*table ID".* Este parâmetro específica de que Google Analytics *view profile* se vai buscar a informação. Este valor está no formato "ga:xxxx" que é a concatenação do *namespace* "ga:" com o *view (profile)* ID. Para além do ID, é ainda necessário obrigatoriamente uma data de início, uma data de fim e pelo menos uma métrica. É também possível colocar parâmetros adicionais, como dimensões, filtros, segmentos, entre outros [101] [102].

A data de início e a data de fim são obrigatórias, pois é necessário especificar um *range* para qual se quer obter informação.

O parâmetro da dimensão é opcional e separa as métricas por critérios comuns, isto é, por *browser*, sistema operativo, cidade, entre muitos outros. Ou seja, é possível saber o número total de pessoas que acederam a um *website*, e com o parâmetro de dimensões, é possível saber o número total de pessoas que acederam ao *website* e por que *browser*s o fizeram. Este resultado é obtido através da dimensão: "ga:browser".

A métrica é também um campo obrigatório, e é uma estatística agregada referente à actividade dos utilizadores, isto é, número de cliques, o número de vezes que uma página é apresentada, entre outros. Se não for especificada uma dimensão, o resultado apresenta valores agregados para as datas introduzidas, como o total de visitas às páginas ou o total de rejeições. Contudo, se uma dimensão for especificada, os valores são segmentados pelo valor da dimensão.

Imaginando que se está a utilizar a métrica: "ga:sessions". Com esta métrica é possível obter um valor geral do número sessões que acederam a um *website* num determinado range. Contudo, ao utilizarmos esta métrica com a dimensão "ga:country", obtém-se o número total de sessões por país que acederam ao *website.*

Os filtros são utilizados para restringir a informação que é devolvida. Um exemplo de um filtro é "ga:browser==Firefox", que restringe a informação ao *browser* Firefox.

Para além destes parâmetros, existem muitos outros, contudo, os que foram mencionados são aqueles que foram unicamente utilizados.

```php
$profileId = getFirstProfileId($service);

//Visitas por Browser
function getVisitByBrowser(&$service, $profileId, $start_date, $end_date) {
    $optParams = array(
     'dimensions' => 'ga:browser',
      'sort' => '-ga:sessions');

    return $service->data_ga->get(
    'ga:' . $profileId,
    $start_date,
    $end_date,
    'ga:sessions,ga:sessionDuration,ga:pageviews,ga:percentNewSessions,ga:bounceRate',
    $optParams);
}

$browser_visits = getVisitByBrowser($service, $profileId,$start_date, $end_date);
```

Listagem 35 - Exemplo da query para obter informação sobre os browsers utilizados

Para melhor entender, temos então a Listagem 35 que apresenta o exemplo de uma *query* criada para obter informação relativamente aos *browsers* utilizados quando o *website* da Granifil (http://www.granifil.com/) foi acedido. Para esta *query* foi utilizada a dimensão "ga:browser".

Tabela 6 - Tabela de Métricas usadas nas várias queries

Métrica	Definição
Ga:session	Número total de sessões
Ga:sessionDuration	Duração total da sessão do utilizador em segundos
Ga:pageViews	Número de páginas vistas (novas e repetidas) no intervalo de tempo estabelecido
Ga:percentNewSessions	Percentagem de sessões dos utilizadores que nunca visitaram o website
Ga:bounceRate	Percentagem da sessão de uma única página, isto é, sessão na qual o utilizador abandona o *website* a partir da primeira página

Existem muitas métricas que são possíveis de aplicar de maneira a obter os mais variados resultados, contudo as apresentadas na Tabela 6, foram todas as que foram utilizadas para este componente.

Para além da dimensão do *browser*, foram utilizadas outras dimensões, como é possível verificar na Tabela 7, associadas contudo sempre as mesmas cincos métricas referidas anteriormente.

Tabela 7 - Tabela de Dimensões usadas nas várias queries

Dimensão	Definição
Ga:browser	Nome dos *browsers* utilizados pelos utilizadores no *website*. Ex: Internet Explorar ou Firefox
Ga:medium	Tipo de referência. Isto é, se o utilizador acede ao *website* através de um motor de pesquisa, o valor é "organic". Se não for por um motor de buscar, o valor é "referral". Se os utilizadores acederem diretamente pelo URL, o valor é "non".
Ga:deviceCategory	Categoria do dispositivo. Ex: Desktop, tablet ou mobile.
Ga:language	Linguagem fornecida pelo pedido http para o *browser*. Ex: pt para Portugal.
Ga:country	O país do utilizador, derivado do endereço IP do "Geographical IDs".
Ga:city	A cidade do utilizador, derivado do endereço IP do "Geographical IDs".
Ga:pagePath	Página no *website* especificada pelo caminho. Usado em conjunto com o *hostname* para obter o endereço URL completo da página.
Ga:operatingSystem	Sistema Operativo usado pelos utilizadores. Ex: Windows, Linux, Macintosh, iPhone, iPod.

O *output* gerado pelas diferentes métricas e respetivas dimensões assumiu sempre a apresentação presente na Ilustração 38.

browser	sessões	% novas sessões	novos utilizadores	taxa de rejeições	página/sessão	duração média da sessão
chrome	558	89.61	500	52.33	5.32	00:01:52
internet explorer	144	86.81	125	25	6.58	00:02:06
firefox	98	88.78	87	60.2	4.84	00:01:01
safari	82	85.37	70	36.59	6.57	00:02:29
android browser	30	96.67	29	30	9.87	00:04:42
opera	21	100	21	76.19	8.05	00:01:51
(not set)	15	100	15	40	0.4	00:00:00
yabrowser	12	100	12	100	1	00:00:00
mozilla compatible agent	10	100	10	100	1	00:00:00
edge	9	100	9	33.33	3.56	00:00:46
opera mini	2	50	1	50	1.5	00:00:41

Ilustração 38 – Informação relativa aos *browsers* utilizados quando navegaram no *website*

Apesar de todas estas métricas e dimensões terem sido utilizadas no componente do *Dashboard*, este não ficou com todas as funcionalidades que tinham sido idealizadas, implementadas, devido ao surgimento de novos projetos com *deadlines* mais apertadas e prioridades mais elevadas.

3.4.6 *Integração do SEO no Backoffice*

O que é o SEO, e como utilizar algumas das suas técnicas para obter uma boa classificação nos *rankings* dos motores de busca como o Google, Bing ou Yahoo, já foi devidamente explicado no capítulo 3.3.1.3. Como tal, é importante agora referir como é que algumas destas técnicas foram implementadas no *backoffice.*

Relembrando que o *backoffice* foi desenvolvido de maneira a que para cada página presente em *frontend* existisse uma tabela correspondente na base de dados, responsável por armazenar a sua informação, era possível no *backoffice* gerir as *title tags* e as *meta description* de cada página para os vários idiomas, caso existissem.

Ilustração 39 - Página correspondente à gestão de algumas características de SEO

A Ilustração 39 corresponde a uma página apresentada quando é selecionada a opção "SEO" no menu lateral esquerdo. Nesta página aparece um menu no topo com as várias páginas existentes em *frontend*, onde para cada um dos elementos do menu é apresentado um formulário com o campo *title tag* e *meta description* para cada um dos idiomas. Em *frontend*, consoante o idioma da aplicação, são então apresentados os valores correspondentes.

Quando se trata de imagens e o conceito de SEO, para além do nome do ficheiro e do seu tamanho e resolução, é também importante referir a *tag alt*, correspondente ao texto alternativo quando por alguma razão a imagem não é apresentada. Este campo está presente para ser definido no formulário de inserção/edição de uma nova imagem, tal como apresenta a Ilustração 40.

Ilustração 40 - Exemplo de um formulário de inserção de uma nova imagem

A nível de *friendly url*, estes foram todos definidos no ficheiro "*routes.php",* isto é, no ficheiro onde são definidas todas as rotas da aplicação. Deste modo, foram todos devidamente pensados para serem intuitivos e bem estruturados.

4 – Memória Descritiva dos Projetos Realizados

Neste capítulo são ilustrados todos os trabalhos que foram realizados ao longo do estágio por ordem cronológica, sendo explicado o motivo pela qual foi desenvolvido cada projeto, e ainda algumas das suas características mais importantes.

Na Tabela 8 é possível observar então quais os projetos que foram desenvolvidos e quais deles utilizaram o *backoffice* e os seus componentes. É possível também verificar quais dos projetos é que são *responsive*, e se tiveram no seu desenvolvimento a utilização de tecnologias do lado de cliente, tal como JavaScript e AJAX. Por último, é ainda possível observar se foram utilizadas técnicas de SEO.

Tabela 8 - Tabela ilustrativa dos projetos desenvolvidos ao longo do estágio

Projetos	CMS	Project Management Tool	Dashboard	Gestão Docs., Fich. e Imag.	JavaScript e/ou AJAX	Técnicas SEO	Responsive
Dakar	-	-	-	-	Sim	Sim	-
FJN	-	-	-	-	Sim	Sim	-
Morais Matias	-	-	-	-	Sim	Sim	-
Tároca	Sim	-	-	-	Sim	Sim	-
PVS	Sim	Sim	-	-	Sim	Sim	-
Casa dos Relógios	Sim	-	-	-	Sim	Sim	-
Automoeda	-	-	-	-	Sim	Sim	-
Granifil	Sim	-	Sim	-	Sim	Sim	-
Branco Genuíno	Sim	-	-	-	Sim	Sim	Sim
Britomoldes	-	-	-	-	Sim	Sim	-
Pearlmaster	Sim	-	-	-	Sim	Sim	Sim
Sunaitec	Sim	-	-	Sim	Sim	Sim	Sim
Specifiklines	-	Sim	-	-	Sim	Sim	-

Para cada projeto individualmente, serão mencionados alguns comportamentos e funcionalidades que já foram previamente explicadas nos capítulos 3.3.1.2 (Bibliotecas JavaScript) e 3.4 (Principais Componentes Desenvolvidos).

4.1 Projeto Dakar | Boots and Shoes

A "Dakar | Boots and Shoes" trata-se de um projeto referente a uma empesa representante da DAKAR – marca de calçado de prestígio internacional. Esta marca caracteriza-se na sua essência por ser um calçado para Homem, Senhora e Criança, primando pela sua superior qualidade, *design* exclusivo e estilo próprio e moderno.

No momento em que este projeto surgiu, a Dakar | Boots and Shoes já tinha uma presença na *web*, caracterizada por uma loja *online,* pelo que este projeto surgiu como complemento a essa loja. Foi desenvolvido um *website* institucional com informação referente à empresa, e com a apresentação dos vários produtos vendidos pela DAKAR, divididos pelas suas três categorias.

Foi desenhado para ser navegável com a utilização do *scroll* do rato ou através do menu de navegação, mas com uma orientação horizontal [41], com um conjunto variado de animações entre os elementos.

Nas várias galerias de imagens, para cada produto existente, há um botão de "Like", semelhante ao do *facebook*, com a possibilidade de ser clicado. Quando esse evento é realizado, através de AJAX é efetuada uma actualização na base de dados, para incrementar o número de likes do respetivo produto, fazendo com que este valor de "*like"* seja apenas um valor meramente informativo.

A página *home* do projeto é apresentada pela Ilustração 41 e o seu endereço de acesso é o http://www.dakarbootsandshoes.pt/ .

Ilustração 41 - Página *home* do *website* da "Dakar | Boots and Shoes"

4.2 *Projeto FJN Moldes*

O Projeto da "FJN Moldes" é um projeto referente a uma empresa de fabrico de moldes de precisão injeção plástica para os mais diversos setores.

O objetivo deste projeto foi desenvolver um *website* institucional para promover esta empresa, passando desta forma a empresa FJN Moldes a ter uma presença na *web* que até ao momento era inexistente. Trata-se então de um *website multi-page* que disponibiliza informação referente à empresa, aos equipamentos que utiliza, e ainda apresenta uma gama de projetos desenvolvidos para um conjunto variado de setores.

Para a apresentação dos vários equipamentos utilizados pela empresa, foi utilizado o *slideshow* que melhor se enquadrou para o resultado pretendido, neste caso o Flexslider [31]. Para a página dos vários Projetos, foi utilizado o *plugin* MixItUp [45], que permite filtrar os vários projetos por uma determinada categoria.

A HTML Local Storage [43] foi pela primeira vez utilizada neste projeto. O seu principal objetivo foi guardar o valor de um elemento HTML selecionado, para que quando o utilizador fosse redirecionado para a página dos "Projetos", após ter selecionado o elemento HTML, o *plugin* MixItUp efetuasse automaticamente o filtro respetivo ao valor guardado.

O AJAX foi utilizado para o envio dos dados do formulário de contactos.

A página *home* do *website* é apresentada pela Ilustração 42 e o seu endereço de acesso é o http://www.fjn.pt/ .

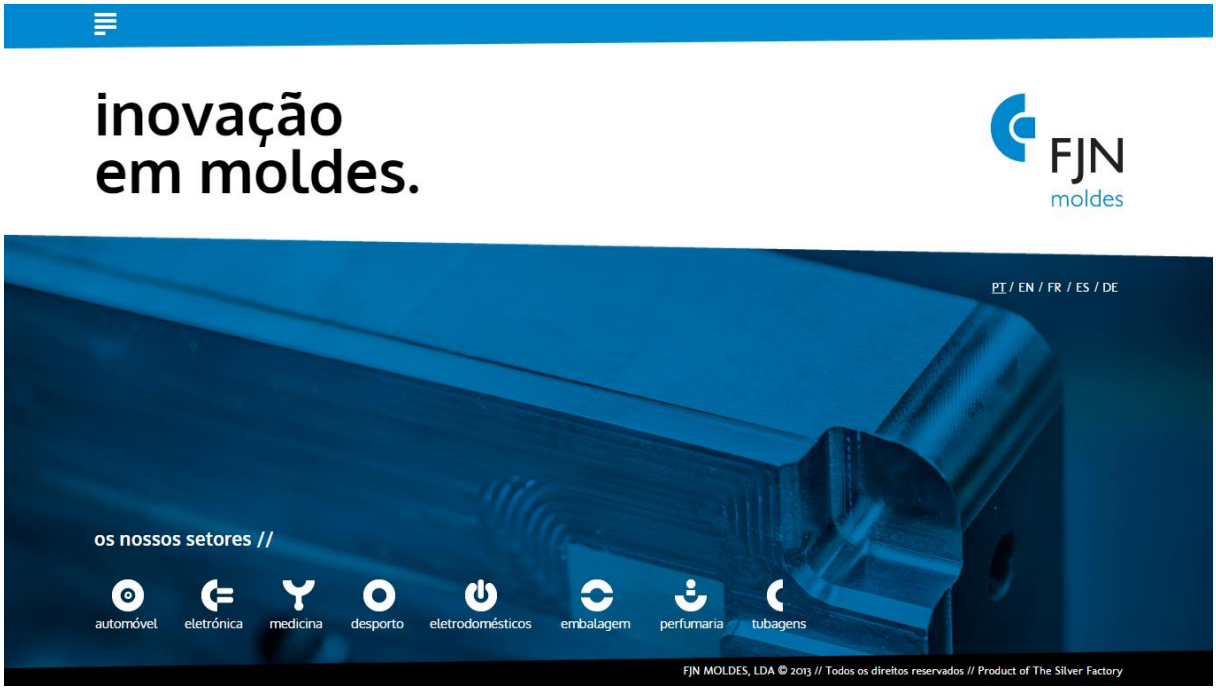


Ilustração 42 - Página *home* do website da "FJN Moldes"

4.3 *Projeto Morais Matias*

O Projeto da "Morais Matias, S.A" trata-se de um projeto referente a uma empresa de fabrico de ampolas em vidro neutro, tendo Portugal como principal mercado, mais propriamente para a indústria farmacêutica, cosmética, entre outros.

Para este projeto o objetivo foi desenvolver novamente um *website* institucional que promovesse a empresa Morais Matias. Trata-se então de um *website multi-page* que disponibiliza informação sobre a empresa em si e o seu principal mercado de trabalho. Apresenta também uma linha cronológica, desenvolvida com o *plugin* jQuery com o nome jCarousel [32], com informação desde o seu nascimento até à atualidade e ainda um catálogo dos produtos que fabrica.

A página *home* do *website* é apresentada pela Ilustração 43, e o endereço *web* para o acesso do mesmo é http://www.moraismatias.pt/ . É na página *home* que é utilizado o componente chamado "A Collection of Page Transitions" [36], responsável por realizar as animações aquando da transição das várias secções presentes.

Ilustração 43 - Página *home* do *website* da "Morais Matias, S.A."

4.4 *Projeto Taróca Decoração & Mobiliário*

O Projeto da "Taróca Decoração & Mobiliário" é um projeto referente a uma empresa de referência no setor da decoração e *design* de interiores.

Tal como os projetos anteriores, para este projeto foi desenvolvido um *website* institucional para promover esta empresa. Trata-se de um *website multi-page* e a informação disponibilizada é referente à própria empresa, às suas duas lojas e a alguns dos projetos realizados. Tem ainda uma área reservada, onde é possível efetuar o *download* de ficheiros.

Foi o primeiro projeto onde foi utilizado o *backoffice*, estando este ainda numa fase inicial, onde o *layout* do mesmo ainda não era o que foi apresentado no capítulo 3.4.1. O único componente constituinte do *backoffice* foi o *CMS*.

Relativamente às duas lojas apresentadas no *website,* cada uma delas tinha uma galeria associada, galeria essa que foi desenvolvida com o *plugin* jQuery já referido, o Flexslider [31].

Para a galeria referente aos projetos, foi utilizado um *plugin* jQuery chamado jCarousel [32], também previamente mencionado.

A página *home* do *website* é apresentada na Ilustração 44 e o seu endereço de acesso é o http://www.tarocadecoracao.com/ .

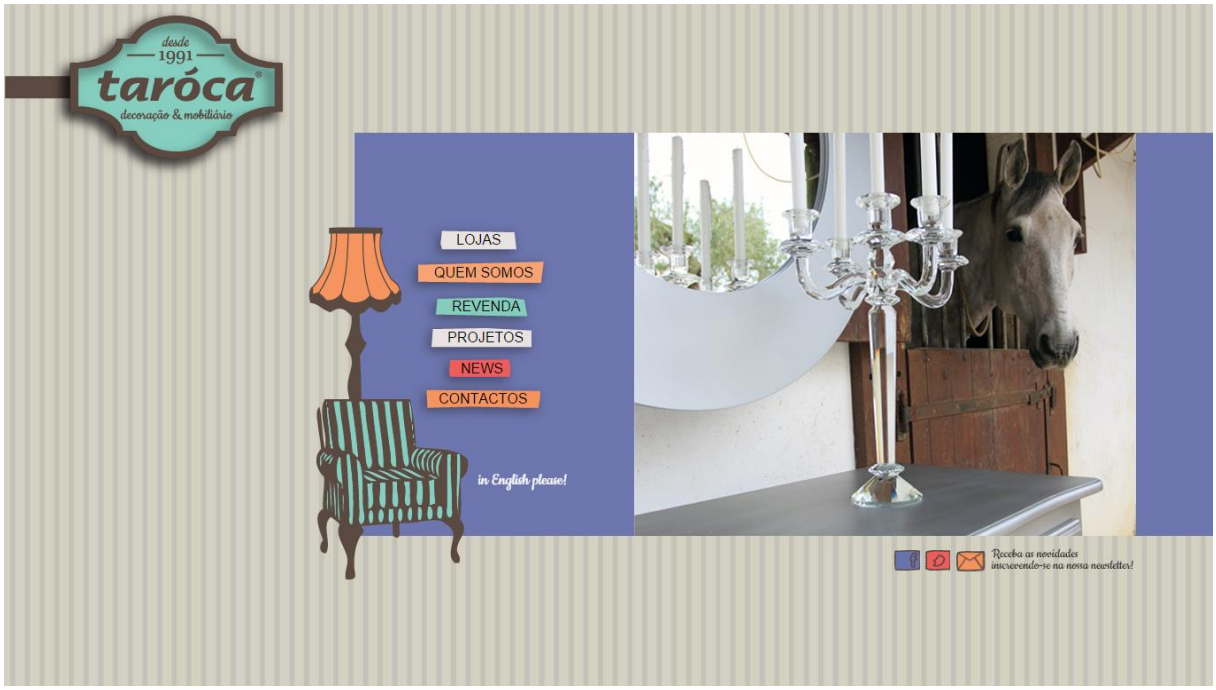


Ilustração 44 - Página *home* do *website* da "Tároca Decoração & Mobiliário"

4.5 Projeto PVS

O Projeto da "PVS. SA Portugal" trata-se de um projeto referente ao grupo PVS, constituído pela PVS Concept & Engineering, PVS 2 Fast Tooling e MRS Plastic Injection. Este grupo tem como principal função o fabrico de moldes para a indústria de injeção plástica.

Na altura que este projeto foi concebido, a PVS já tinha uma presença na *web* caracterizada pelo seu *website,* contudo este tinha sido desenvolvido em Flash. Deste modo, houve a necessidade de atualizar o *website* para uma tecnologia mais recente, e com a possibilidade de se editar o conteúdo a nível de *frontend*. Como tal, o projeto da PVS foi desenvolvido com o *backoffice*, estando este na altura numa versão mais atualizada. O componente *CMS* foi utilizado para a gestão do conteúdo.

O antigo *website* tinha na sua constituição uma ferramenta que permitia gerir os seus clientes e os seus projetos, onde para cada projeto era possível visualizar uma imagem com o gráfico de Gantt correspondente ao projeto. Do mesmo modo que se desenvolveu um novo *website*, também foi desenvolvido uma nova plataforma para a gestão de clientes e projetos da PVS. Esta plataforma tem o nome de *Project Management Tool*, um componente integrado no *backoffice,* com a particularidade de ter um gráfico de Gantt interativo.

O componente Flexslider [31] foi o *plugin* utilizado por excelência para as várias galerias existentes no *website*. Tal como o projeto da FJN Moldes apresentado em 4.2, também a PVS tinha uma galeria de imagens, com a possibilidade de ser filtrada pelas várias categorias existentes. Para tal, foi utilizado o componente MixItUp [45].

A página *home* do *website* é apresentada Ilustração 45 e o seu endereço de acesso é o http://www.pvsmoldes.eu/ .

Ilustração 45 - Página *home* do *website* "PVS. SA Portugal"

4.6 *Projeto Casa dos Relógios*

A "Casa dos Relógios" é um projeto referente a uma relojoaria e joalharia, que disponibiliza um catálogo de produtos para as mais variadas marcas.

Este projeto foi desenvolvido com o objetivo de criar também uma presença na *web*, caracterizada mais uma vez por um *website*. Como se pretendia ter um controlo da gestão dos conteúdos apresentados em *frontend*, foi utilizado o *backoffice* com o componente *CMS*.

A página *home* do *website* é apresentada na Ilustração 46 e o seu endereço de acesso é o http://www.casadosrelogios.pt/ .

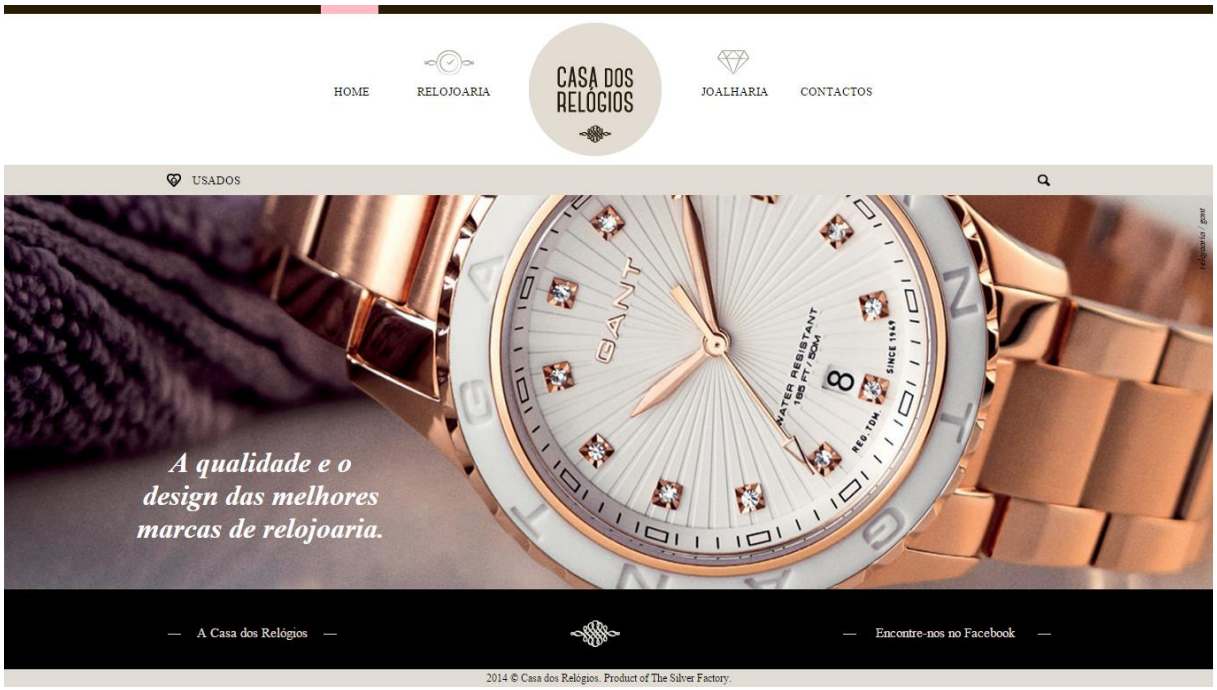


Ilustração 46 - Página *home* do *website* "Casa dos Relógios"

4.7 *Projeto Automoeda*

O Projeto da "Automoeda" trata-se de um pequeno projeto referente a uma oficina de manutenção e reparação automóvel. Foi apenas desenvolvida uma *landing page* com a informação referente à localização da empresa no Google Maps e com um pequeno formulário de contactos.

A única particularidade desta *landing page* é o conjunto de animações utilizadas para as transições entre as várias secções, provenientes do componente "A Collection of Page Transitions" [36].

Os dados do formulário são enviados através de AJAX para que não haja o carregamento integral da página após o submit dos dados.

A página *home* da *landing page* é apresentada na Ilustração 47 e o seu endereço de acesso é o http://www.automoeda.pt/.

Ilustração 47 - Página home do *website* da "Automoeda"

4.8 Projeto Granifil

O Projeto da "Granifil" é um projeto referente a uma empresa que transforma e comercializa produtos em granito, mármore e outras pedras calcárias, desde a seleção da matéria-prima até à venda do produto final.

O *website* da Granifil já tinha sido desenvolvido, contudo tratava-se de um *website* estático, e desatualizado. Como tal, o objetivo deste projeto foi de integrar o *backoffice* no *website*¸ juntamente com o componente *CMS* para permitir a gestão dos conteúdos.

O componente utilizado para as galerias foi atualizado para um já mencionado com o nome, Flexslider [31]. Para além da atualização das galerias, também algumas noções de SEO foram atualizadas, tal como os *friendly url*, as *alt tags* das imagens, as *title tags* e ainda as *meta descriptions*.

O componente *Dashboard – Analytics* foi especialmente desenvolvido para este projeto, embora sem a intenção de ser exclusivo apenas para o mesmo.

A página *home* do *website* é apresentada na Ilustração 48 e o seu endereço de acesso é o http://www.granifil.com/ .

Ilustração 48 - Página *home* do *website* da "Granifil"

4.9 Projeto Branco Genuíno

O Projeto da "Branco Genuíno" trata-se de um projeto referente a uma empresa de limpezas, manutenção de residências e reparações domésticas.

Trata-se de um *website* institucional com informação sobre a empresa e dos serviços que presta. Foi desenvolvido com a utilização do *backoffice* e do seu componente *CMS* para a gestão de conteúdos. Foi o primeiro projeto desenvolvido *responsive*, isto é, adaptável às várias dimensões dos vários dispositivos.

Não tem qualquer galeria de imagens, mas é composto por três *sliders*, que foram desenvolvidos com o *plugin* jQuery chamado Flexslider [31]. Também foi usada a HTML Local Storage para guardar informação referente a um qualquer serviço selecionado pelo utilizador, situado no *header* da página. Deste modo, após o utilizador ser redirecionado para página dos "Serviços", o serviço que aparece como destaque corresponde ao valor guardado na Local Storage.

A página *home* do *website* é apresentada na Ilustração 49, e o seu endereço de acesso é o http://www.brancogenuino.pt/ .

Ilustração 49 - Página home do *website* da "Branco Genuíno"

4.10 Projeto Britomoldes

O Projeto da "Britomoldes" é um projeto referente a uma empresa que oferece um serviço completo no fabrico de moldes, isto é, projeto, produção e assistência pós-venda de moldes de pequenas, médias e grandes dimensões.

Trata-se de um *website single page*, desenvolvido com base no exemplo *online* "Create a Parallax Scrolling Website Using Stellar.js" [37]. Tem numa das secções informação apresentada sobre a forma de um acordeão, cujo seu desenvolvimento foi baseado no *widget* jQuery "Accordion" [103]. No mesmo *widget*, para alguns elementos, há a possibilidade de visualizar uma galeria de imagens, correspondente ao elemento selecionado. Estas galerias são apresentadas num *modal*, e foram desenvolvidas com o *plugin* Flexslider [31].

A página *home* do *website* é apresentada na Ilustração 50, e o seu endereço de acesso é o http://www.britomoldes.pt/ .

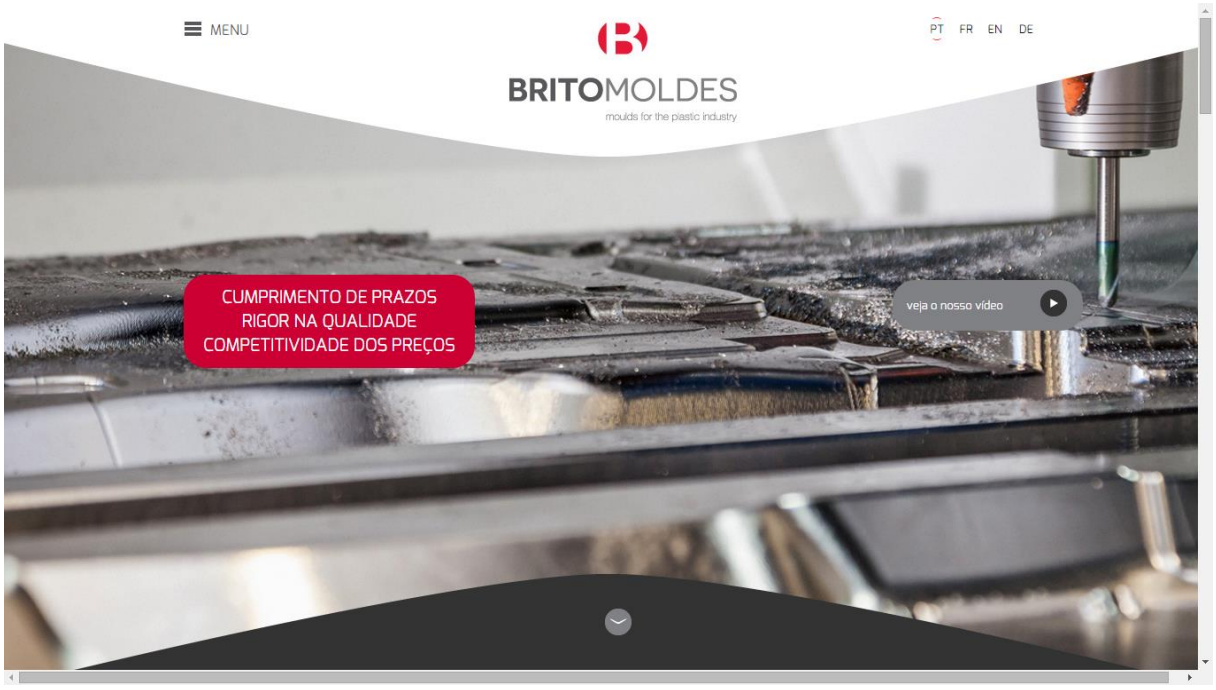


Ilustração 50 - Página home do *website* da "Britomoldes"

4.11 Projeto Pearlmaster

O Projeto da "Pearlmaster" trata-se de um projeto referente a uma empresa que oferece uma gestão completa de projeto de moldes e oferece também o serviço de consultadoria para outras empresas que precisem de apoio no desenvolvimento de projetos.

Na altura, a empresa já tinha uma presença *online*, contudo estava desatualizada e não oferecia condições de gestão de conteúdos. Como tal, foi desenvolvido um novo *website* com tecnologias mais recentes, com o *backoffice* e com o componente *CMS*, para permitir a gestão integral do conteúdo do *frontend*. É também um *website responsive*, isto é, adaptável aos vários dispositivos.

Tem na sua constituição uma única galeria de imagens, desenvolvida com o uso do *plugin* jQuery Flexslider [31].

Foi também usada a HTML Local Storage, com o mesmo comportamento dos outros projetos onde foi utilizada, ou seja, armazenar informação referente a um elemento quando selecionado. Deste modo, a respetiva página para a qual o utilizador foi redirecionado, iria ter um determinado comportamento, com base no valor selecionado.

A página *h*ome do *website* é apresentada na Ilustração 51, e o seu endereço de acesso é o http://www.pearlmaster.pt/ .

Ilustração 51 - Página home do *website* da "Pearlmaster"

4.12 Projeto Sunaitec

O Projeto da "Sunaitec" é um projeto referente a uma empresa que desenvolve, produz e comercializa soluções avançadas na área da energia solar.

À semelhança de outras empresas, também esta já tinha uma presença *online*. Tratava-se porém de um *website* desenvolvido em Flash. Como tal, este projeto surgiu como necessidade de atualizar a presença na *web*, tendo sido desenvolvido um novo *website* institucional com tecnologias mais recentes e com a característica de ser *responsive*. Para o seu desenvolvimento, foi utilizado o *backoffice* e o componente do *CMS*.

O *website* é *single page* e a sua estrutura foi desenvolvida com base num exemplo *online* chamado "Create a Parallax Scrolling Website Using Stellar.js" [37]. A página *home* apresentada pela Ilustração 52 tem na sua constituição dois *sliders*, desenvolvidos com o *plugin* jQuery Flexslider [31]. O "jQuery.animateSpirte" [46] foi utilizado especificamente neste projeto, para desenvolver uma *sprite animation*.

A única galeria de imagens presente no *website,* foi desenvolvida com base no exemplo online chamado "Google Grid Gallery" [33], cuja implementação já foi previamente explicada no capítulo 3.3.1.2, se secção "Galeria de Imagens (Slide show)

Para além do componente CMS, existente no *backoffice,* o componente Gestão de Documentos e Ficheiros (Área Reservada) foi desenvolvido e utilizado unicamente para este projeto.

O endereço de acesso ao *website* é http://www.sunaitec.pt/ .

Ilustração 52 - Página home do *website* da "Sunaitec"

4.13 Projeto Specifiklines

O Projeto da "Specifiklines" trata-se de um projeto referente a uma empresa que oferece um grande e completo conjunto de serviços, desde engenharia ao desenvolvimento de projetos chave na mão, e foi o último projeto desenvolvido por mim durante o estágio na empresa The Silver Factory.

Trata-se de uma *landing page* simples, com conteúdo referente à empresa e aos serviços que disponibiliza. Esta *landing page* tem no seu desenvolvimento o componente chamado "A Collection of Page Transitions" [36], para realizar as animações aquando da transição entre as várias secções presentes. A informação referente aos serviços é apresentada sobe a forma de um *slider*, e este foi desenvolvido com o *plugin* jQuery Flexslider [31].

Apesar de se tratar de uma *landing page*, o projeto tem também a utilização do *backoffice* e do componente *Project Management Tool*, para a gestão de clientes e dos seus projetos, com a particularidade do gráfico de Gantt interativo.

A página *home* do *website* é apresentada na Ilustração 53, e o seu endereço de acesso é o http://www.specifiklines.com/ .

Ilustração 53 - Página home do *website* da "Specifiklines"

5 - Validação dos Trabalhos

Neste capítulo são explicados os processos pela qual passaram os vários projetos desenvolvidos para que pudessem ser publicados *online*. É também referido uma análise ao resultado dos testes de usabilidade realizado a dois desses projetos, e ainda o resultado dos testes de desempenho realizados a apenas um desses projetos.

5.1 Critérios de Qualidade

Um projeto é dado como concluído quando cumpre todos os requisitos. É fundamental que para além deste cumprimento de requisitos, sejam testadas todas as funcionalidades e a maneira como a aplicação se comporta. No caso de a aplicação ser *responsive*, ou seja, adaptável a todas as resoluções dos vários dispositivos, é essencial que a aplicação apresente a informação de uma maneira correta em todas as plataformas (*mobile, tablet* e *desktop/laptop*). Os testes às várias resoluções foram realizados com a utilização de uma extensão do *browser* Google Chrome chamado "Window Resizer" [104] e ainda de uma ferramenta *online* chamada de "Screenfly" [105]. Para além destes dois métodos referidos, os testes às resoluções foram também efetuados pelos vários elementos da empresa nos respetivos dispositivos pessoais.

Atualmente existem ao dispor de cada utilizador, um conjunto variado de *browsers*, como o Google Chrome, Mozilla Firefox, Internet Explorer, Opera, Safari, entre muitos outros, pelo que é de esperar que os *websites* funcionem e se comportem todos sempre da mesma maneira. Contudo isso não se verifica, pois os padrões de tecnologia *client-side* estão em constante evolução e isso causa uma grande inconsistência em como os diferentes *browsers* se comportam para a mesma página *web* [106]. Deste modo, e de acordo com o conceito de compatibilidade *cross-browser*, surge então a necessidade de se verificar o comportamento dos *websites* nos vários *browsers*. Esta verificação foi realizada também pelos vários elementos da empresa nos seus dispositivos pessoais.

Para o *browser* Internet Explorer, apenas foram tidas em conta as versões mais recentes até à data da realização do estágio, ou seja, a versão 10 e 11. Para testar estas versões, no próprio *browser*, na secção das ferramentas do programador, é possível realizar uma emulação, com a possibilidade de escolher a versão e o perfil, isto é, *desktop* ou Windows Phone.

No Google Chrome também existe a ferramenta de programador, onde também é possível fazer uma emulação por dispositivo, como por exemplo para a gama Google Nexus, Samsung Galaxy e ainda iPhone, entre muitos outros.

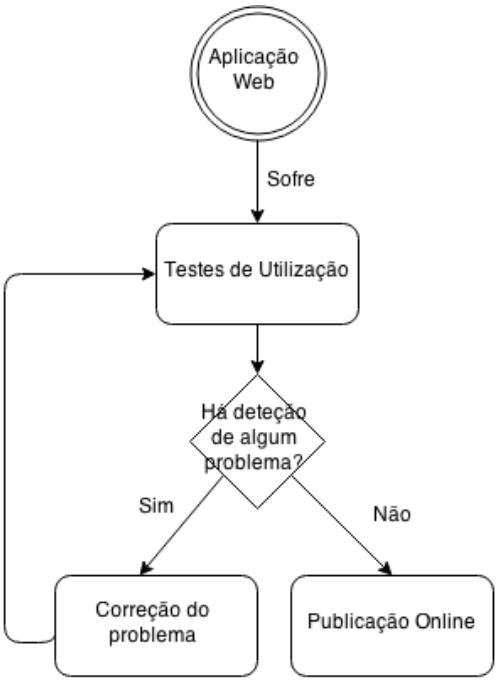


Ilustração 54 - Diagrama ilustrativo referente à validação dos trabalhos

A Ilustração 54 serve para de uma maneira simples, mas objectiva, demostrar o processo de validação dos vários projetos, tendo já este sido devidamente explicado anteriormente, com a particularidade de que no rectângulo de "Testes de Utilização", são então abordados os testes às funcionalidades e ao comportamento da aplicação às várias resoluções.

5.2 Testes de Usabilidade

A realização de testes de usabilidade são um fator importante no desenvolvimento de aplicações *web*, pois permitem avaliar a complexidade do *layout*, isto é, como as várias páginas estão estruturadas, e como é apresentada a informação. Permite avaliar se os *websites* são intuitivos, através da velocidade com que os utilizadores realizam as tarefas e permite ainda observar as reações do utilizador, bem como recolher opiniões e recomendações.

Os testes de usabilidade serviram apenas para um estudo pessoal, fora do âmbito do estágio, tendo sido escolhidos para a sua realização, dois projetos já referidos anteriormente. O da Pearlmaster, referido na secção 4.11, por ser *multi-page* e o da Sunaitec, referido na secção 4.12, por ser *single page*. Ambos os projetos são *responsive.*

Uma vez que os *websites* escolhidos continham um vasto conjunto de informação, de forma a guiar o utilizador durante a navegação dos mesmos, foi criado um questionário, que para além de ter questões a nível pessoal, também continha um conjunto de questões relativas às tarefas que os utilizadores tinham de realizar.

Para se obterem resultados mais próximos da realidade, foi utilizado um computador de mesa e um dispositivo móvel com interação táctil, ou seja, que funcione por gestos e toques. Tanto o computador e o dispositivo móvel tinham ligação à internet durante a realização das tarefas.

Um aspeto importante a ter em consideração nos testes de usabilidade é a fase de recrutamento e seleção de potenciais utilizadores ou participantes para realizarem os testes. Para tal é importante

definir o perfil desejados dos participantes. Como foi referido anteriormente neste documento, o *website* da Pearlmaster apresenta informação sobre a produção de moldes, e o *website* da Sunaitec apresenta informação sobre painéis solares, pelo que este fator dificultou-me a seleção dos utilizadores, dado que, a nível pessoal não ter conhecimento de alguém que pudesse estar interessado nestas áreas. Sendo assim, foram escolhidos cinco utilizadores que em nada estavam relacionados com as áreas descritas acima. Este número foi eleito com base num estudo realizado por Jakob Nielsen [107].

Para além da seleção dos participantes, também é necessário definir o local e as condições na qual os participantes devem realizar os testes. Os testes foram realizados em locais bem iluminados, e sem barulho de fundo que pudesse causar distracções ou desconcentrações aos participantes. Para além disso, o teste foi realizado a cada participante de forma individual, tendo a minha postura sido imparcial. Os participantes estavam ainda em posições confortáveis e a dedicarem toda a sua atenção às tarefas.

Cada participante, durante a sessão, teve de preencher um questionário dividido em três partes. A primeira parte do questionário era constituída por questões pessoais e questões sobre a experiência do utilizador com o manuseamento de equipamentos tecnológicos, tal como computador ou dispositivos móveis. A segunda parte era constituída por um conjunto de tarefas que cada participante teve de realizar ao mesmo tempo que navegava no *website*. A terceira e última parte é constituída por questões relacionadas com a segunda parte do questionário, de modo a obter *feedback* do participante relativamente à realização dessas mesmas tarefas.

Por último, é de salientar que toda a informação gerada nas sessões será mantida anónima, salvaguardando a identidade de todos os participantes.

O Anexo A refere-se a um exemplo do questionário apresentado aos participantes.

5.2.1 Resultados e Conclusões dos Testes de Usabilidade

Como referido anteriormente, os testes de usabilidade foram realizados por cinco utilizadores. Quatro do sexo feminino, e um do sexo masculino. Destes cinco utilizadores, dois faziam parte do setor de trabalho da área de comunicação e aplicações *web* enquanto que os restantes três faziam parte do setor geral.

Dado que os dois *websites* escolhidos são *responsive,* as tarefas foram realizadas duas vezes para cada *website,* perfazendo um total de quatro, ou seja, uma vez no computador para cada um dos *websites* e outra vez no dispositivo móvel de novo para cada um dos *websites*. Definindo como "tarefa bem-sucedida" o facto de se conseguir realizar a tarefa apresentada pelo questionário, independentemente do tempo levado, então todas as tarefas foram realizadas com sucesso, e por conseguinte, o mesmo para o teste geral em si. Naturalmente, houve tarefas que demoraram um pouco mais de tempo a ser realizadas, mas nada que valha a pena de referir.

Tabela 9 - % de respostas dadas para cada questionário

	Muito Fácil	Fácil	Difícil	Muito Difícil	Impossível
Pearlmaster (Computador)	61,82%	34,55%	3,63%	-	-
Pearlmaster (Dispositivo Móvel)	43,33%	45%	11,67%	-	-
Sunaitec (Computador)	53,75%	41,25%	5%	-	-
Sunaitec (Dispositivo Móvel)	36,47%	60%	3,53%	-	-

A Tabela 9 apresenta o resultado em percentagem, para o grau de dificuldade das respostas dadas aos vários questionários, onde é possível verificar que em nenhum dos questionários, as tarefas foram classificadas como "Muito Difícil" ou "Impossível". Também se verifica que a resposta correspondente a "Difícil" foi a que teve um valor mais reduzido, sobrando a classificação de "Muito Fácil" e "Fácil" com os melhores valores.

Após uma análise das várias respostas dadas, foi possível concluir que ambos os *websites* têm uma navegação intuitiva, e que o computador é a ferramenta principal para a navegação dos *websites*, dando como justificação o facto de que usar um rato é mais fácil e prático, associado ainda ao facto de que os elementos clicáveis têm dimensões maiores, ou seja, são mais visíveis. Algo que também foi tido em conta, foi que o menu de navegação é mais intuitivo no computador.

Quanto ao facto do *website* da Pearlmaster ser *multi-page* quando comparado com o Sunaitec que é *single page,* houve algumas opiniões que diferiram por parte dos vários utilizadores. Mais de metades dos utilizadores preferiram o *website* da Sunaitec, sendo este *single page*, por acharem que estando toda a informação presente numa única página, e não existindo a necessidade de se navegar entre várias páginas, se consegue encontrar tudo o que se precisa, apenas com a utilização do *scroll* do rato.

Concluiu-se também que como os *websites* apresentam caminhos alternativos para a execução das funcionalidades, os vários participantes escolheram o caminho que lhes era mais intuitivo, não perdendo muito tempo na leitura e entendimento da pergunta da realização da tarefa.

Não foram encontrados também quaisquer erros a nível de programação, isto é, todas as funcionalidades funcionaram como esperado. Das várias sugestões dadas, como aumentar os tamanhos dos elementos do *website,* bem como o espaçamento entre alguns desses elementos, e ainda a necessidade de reorganização dos elementos do menu, estas sugestões não foram efetuadas, pois como mencionado anteriormente, estes testes serviram apenas para estudo pessoal, não tendo sido possível efetuar qualquer alteração.

Após a análise do resultado dos testes práticos, conclui que os 5 participantes seriam capazes de utilizar os vários *websites* para consultar a informação desejada.

5.3 Testes de Desempenho

Todas as empresas que têm um *website* como principal elemento presencial *online*, querem que este seja acedido pelo maior número de pessoas, e que tenha elevadas taxas de retorno. Contudo, quantos mais utilizadores estiverem a navegar no *website*, maior será o tempo de resposta do servidor, o que se traduz numa maior lentidão do *website*, e por conseguinte, um maior tempo de espera até que todo o conteúdo da página seja apresentado.

É essencialmente para avaliar a capacidade, robustez e disponibilidade de um *website* e ainda a quantidade de acessos simultâneos, que os testes de desempenho servem. A ferramenta que foi utilizada para realizar alguns dos testes, foi a ferramenta do Google Developers, chamada de "Page Speed Insights", que mede a performance de uma página em dispositivos móveis e em computadores e portáteis, analisando o *url* duas vezes, uma com um *mobile user-agent*, e outra com um *desktop user-agent* [108]. Como a performance da rede (*network*) está em constante variação, esta ferramenta apenas considera os aspetos independentes da performance de uma página, ou seja, a configuração do servidor, a estrutura HTML da página e o uso de recursos externos como imagens, ficheiros JavaScript e CSS.

Como tal, o Page Speed Insight analisa duas vertentes importantes, a Velocidade e a Experiência do utilizador, que depois se dividem nas respectivas características.

5.3.1 Velocidade

Deste modo, relativamente à Velocidade são analisados os seguintes aspectos:

1. Reduzir HTML;
2. Reduzir CSS;
3. Reduzir JavaScript;
4. Optimizar Imagens;
5. Ativar Compressão;
6. Eliminar JavaScript e CSS de bloqueio;
7. Tirar partido da cache do *browser*;
8. Reduzir o tempo de resposta do servidor;
9. Dar prioridade ao conteúdo visível;
10. Evitar redirecionamento de páginas destino.

Reduzir HTML

Tal como o próprio nome indica, deve-se compactar o conteúdo HTML, sem afetar o processamento da página pelo *browser*, como por exemplo, remover código que não é usado, reduzir nome de variáveis ou funções, etc.

A estrutura de uma página HTML básica pode ser representada pela Listagem 36.

```
<html>
<head>
<style>
    /* awesome-container is only used on the landing page */
    .awesome-container { font-size: 120% }
    .awesome-container { width: 50% }
</style>
</head>
<body>
    <!-- awesome container content: START -->
    <div>_</div>
    <!-- awesome container content: END -->
    <script>
        awesomeAnalytics(); // function
    </script>
</body>
</html>
```

Listagem 36 - Exemplo de uma estrutura HTML

Contudo, se no exemplo da Listagem 36, forem removidos os comentários, os espaços em branco, e por exemplo, no caso da classe CSS com o nome de "*awsome-container*", à qual está a ser atribuída dois tipos diferentes de estilos, se apenas se usar uma vez a classe estando os depois estilos nessa única classe, será possível reduzir o número de caracteres, tal como apresentado na Listagem 37.

```
<html><head><style>.awesome-container{font-size:120%;width: 50%}
</style></head><body><div>_</div><script>awesomeAnalytics();
</script></body></html>
```

Listagem 37 - Exemplo de uma estrutura HTML reduzida

É verdade que este tipo de redução não tem uma boa leitura. Contudo, não tem de ter. Usa-se o primeiro exemplo enquanto "versão de desenvolvimento" e depois aplicam-se todas as reduções quando o *website* for para ser colocado *online* [109]. Para esse efeito, existem várias ferramentas *online*.

Reduzir CSS e Reduzir JavaScript

Tal como "Reduzir HTML", também é importante reduzir ficheiros CSS e JavaScript. Esta redução pode ser realizada através de ferramentas de compactação, isto é, para os ficheiros JavaScript temos o YUI Compressor [110] e o JSMin [111], e para os ficheiros CSS tem-se também o YUI Compressor e o cssmin.js [112].

Optimização de Imagens

A utilização de imagens numa aplicação *web* afeta bastante o tempo de resposta do servidor, pelo que é importante utilizar imagens com um tamanho reduzido, sem afetar a qualidade da mesma [113].

Ativar compressão

Todos os *browsers* modernos suportam compressão *gzip* para todos os pedidos HTTP [114]. Deste modo, ativar esta compressão reduz o tamanho da resposta do servidor, que diminui o tempo que o servidor demora a fazer o *download* dos vários recursos. Este tipo de compressão pode ser activado no ficheiro ".htacces". Um exemplo desta configuração pode ser visível no Anexo 2.

Eliminar JavaScript e CSS de bloqueio de conversão no conteúdo da parte superior

Antes do *browser* efetuar o *rendering* da página, tem de construir a árvore DOM, fazendo *parse* da estrutura HTML. Durante este processo, sempre que o *parser* encontrar um *script* tem de parar e executá-lo antes de continuar o *parser* do HTML. Mas caso encontre um *script* externo, o *parser* é obrigado a esperar pelo seu *download*, antes de o executar. Neste caso, vai atrasar o carregamento da página. É este tipo de ocorrência que se deve evitar, utilizando *scripts* ou CSS's *in-line* no HTML. Mas mesmo *in-line* é necessário que sejam pequenos, e rápidos a executar.

Os *scripts* que não forem críticos para o *render* inicial, devem ser carregados de maneira assíncrona [115].

Tirar partido da cache do *browser*

É possível usar a cache do *browser* para guardar informação, para quando a página carregar, não carregar os recursos pela rede, mas sim a partir do disco local, para onde já foram previamente transferidos.

Uma das maneiras para se proceder a esta configuração, é no ficheiro ".htaccess", onde se define uma data de validade ou uma idade máxima dos recursos. Um exemplo desta configuração pode ser também visível no Anexo 2.

Reduzir tempo de resposta do servidor

Existem inúmeros fatores que podem aumentar o tempo de resposta do servidor. Temos por exemplo *queries* lentas à base de dados, as *frameworks* utilizadas, excesso de consumo de memória e de CPU (Central Processing Unit) e ainda como dito anteriormente, não guardar os recursos na cache do servidor [116], pelo que é importante ter em conta todos estes fatores durante o desenvolvimento das várias aplicações *web*.

Dar prioridade ao conteúdo visível

A parte inicial de uma página *web* é chamada de "above-the-fold". Se a quantidade de dados que a página precisa para apresentar o primeiro conteúdo excede a congestão inicial da janela (cerca de 14.6kB comprimidos), é necessário mais tempo para o servidor carregar completamente a página, e o seu conteúdo ser apresentado pelo *browser*. Deve-se então estruturar o HTML juntamente com os ficheiros de CSS e JavaScript, de maneira, que o primeiro conteúdo visível da página, não demore muito tempo a ser carregado [117].

Evitar redirecionamento de páginas destino

Um redirecionamento faz com que exista um ciclo pedido-resposta HTTP adicional, atrasando o carregamento de uma página. No melhor cenário, cada redirecionamento vai realizar um simples *roundtrip* (pedido-resposta HTTP) e no pior caso, pode resultar em múltiplos *roundtrips* adicionais. Como tal, é necessário minimizar o uso de redirecionamentos para aumentar a performance de um *website*.

Um caso prático que deve ser evitado é por exemplo quando se acede a www.exemplo.com. Neste caso, como este endereço não está preparado para ser acedido por um dispositivo móvel, o utilizador é redirecionado para www.m.exemplo.com. Logo aqui, já existe um redirecionamento

adicional. Para evitar este tipo de situação, deve-se desde logo desenvolver o *website* para que esteja preparado para dispositivos móveis, ou seja, um *website responsive*.

5.3.2 Experiência do Utilizador

Exclusivamente para a análise a nível dos dispositivos móveis, temos a vertente da Experiência do Utilizador, que tem os seguintes aspetos em consideração:

- Configurar *viewport*;
- Dimensionar conteúdo em função da janela atual;
- Dimensionar elementos táteis adequadamente;
- Utilizar tamanhos de tipos de letras legíveis.

Configurar viewport

O *viewport* controla como é que uma página é apresentada nos vários dispositivos. Sem este tipo de configuração, os dispositivos móveis vão fazer *render* à página com uma largura típica da janela de um computador/portátil. Desde modo atribuir um *viewport* dá controlo sob a largura da página e a escala dos vários dispositivos [118]. Para tal, como referido anteriormente no capítulo 3.3.1.1, na Listagem 1, é necessário utilizar a *tag* HTML no *head* do documento.

Dimensionar conteúdo em função da janela atual e utilizar tamanhos de tipos de letras legíveis

Normalmente, quando se navega num *website,* os utilizadores estão habituados a usar apenas o *scroll* vertical, e não horizontal, e forçar o uso do *scroll* horizontal ou o *zoom* resulta numa má experiência para o utilizador. Contudo, ao desenvolver *websites* responsive, utilizando a *meta viewport tag*, pode dar-se o caso do conteúdo da página não caber no *viewport*, como por exemplo uma imagem que tem uma largura superior à do viewport. Neste caso, obriga o utilizador a utilizar o *scroll* horizontal. O uso de *media-queries*, que já foi referido no capítulo 3.3.1.1, é um importante recurso para contornar este problema [119], bem como o uso de valores relativos de largura de imagens, tal como 100% do *container* em que está inserido.

Dimensionar elementos táteis adequadamente

Todos os *websites* têm na sua constituição elementos que são clicáveis. Ao utilizar um computador/portátil, onde se tem um cursor (rato), torna mais fácil carregar nesses mesmos elementos. Deve-se por isso, ter em atenção o tamanho e espaçamento dos elementos que se tem mais certeza que serão clicados dando especial atenção a *websites* que podem ser acedidos por um dispositivo móvel. Aqui, caso o tamanho dos elementos seja mais pequeno, pode sujeitar o utilizador e cometer erros na altura de pressionar os elementos, podendo por exemplo, ser pressionados elementos que não se queira, que depois poderá executar eventos indesejados [120].

5.3.3 Resultados e Conclusões dos testes de Desempenho

A realização deste teste serviu apenas para identificar eventuais falhas que possam ter ocorrido, e explicar de que maneiras podiam ter sido evitadas, resultando numa maior performance do *website.* Não houve a possibilidade de efetuar qualquer alteração após o resultado deste teste, pois tal como os testes de usabilidade, este teste foi realizado fora do âmbito do estágio.

Ao utilizar a ferramenta do Page Speed Insights num determinado *website*, é possível ter logo em consideração os três tipos de situações que podem acontecer, tal como se pode ver na Tabela 10.

Tabela 10 - Legenda da análise do Page Speed Insight (adaptado de [108])

Ícone	Nome	Descrição
	Ponto de Exclamação Vermelho	Ao corrigir, haverá um impacto significativo na *performance* da página.
	Ponto de Exclamação Amarelo	Considerar corrigir.
	Marca Verde (*Check*)	Não existem problemas significativos.

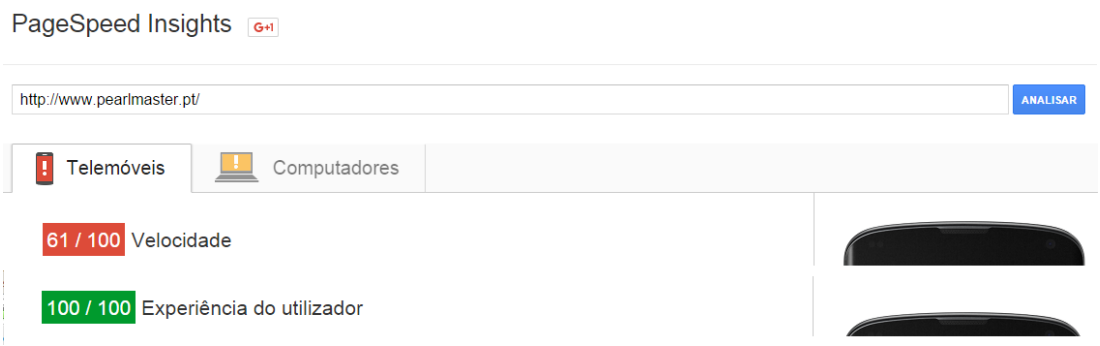


Ilustração 55 - Resultado obtido usando o Page Speed Insights no *website* da Pearlmaster versão mobile

Após utilizar a ferramenta no *website* da Pearlmaster, o primeiro resultado obtido para a componente *mobile* é apresentado pela Ilustração 55. Neste caso, o resultado obtido foi de 61 em 100. Analisando a vertente da velocidade, a informação aparece detalhada para cada aspecto que foi tido em conta no capítulo anterior. Deste modo, de acordo com os critérios do Page Speed Insights, tem-se então que:

É necessário corrigir os seguintes aspectos:

- Eliminar JavaScript e CSS de bloqueio de conversão no conteúdo na parte superior;
- Tirar partido da cache do navegador.

Devem-se considerar corrigir os seguintes aspectos:

- Reduzir JavaScript;
- Reduzir CSS;
- Reduzir HTML;
- Optimizar Imagens;

Foram implementados com sucesso os seguintes aspectos:

- Ativar compressão;
- Dar prioridade a conteúdo visível;
- Evitar redirecionamento de páginas de destino.
- Reduzir tempo de resposta do servidor.

Ainda para a análise em *mobile*, em Experiência de Utilizador, foi obtido o resultado de 100 em 100, pelo que não foi encontrado nenhum problema nesta componente.

Ilustração 56 - Resultado obtido usando o Page Speed Insights no website da Pearlmaster versão desktop

Verificando o separador "Computadores", tal como apresenta a Ilustração 56, é possível verificar que nesta análise foi obtido o resultado de 71 em 100. Analisando o "Resumo de sugestões", a informação aparece detalhada, novamente para cada aspecto que foi tido em conta no capítulo anterior. Novamente, de acordo com os critérios do Page Speed Insights, tem-se então que:

É necessário corrigir os seguintes aspectos:

- Tirar partido da cache do navegador.

Devem-se considerar corrigir os seguintes aspectos:

- Reduzir JavaScript;
- Reduzir CSS;
- Reduzir HTML;
- Optimizar Imagens;
- Eliminar JavaScript e CSS de bloqueio de conversão no conteúdo na parte superior.

Foram implementados com sucesso os seguintes aspectos:

- Ativar compressão;
- Dar prioridade a conteúdo visível;
- Evitar redirecionamento de páginas de destino.
- Reduzir tempo de resposta do servidor;

De acordo com a análise efetuada com a ferramenta do Page Speed Insights, o fator mais importante que seria necessário corrigir, tanto para o *mobile user-agent* como para o *desktop user-agent*, seria o campo correspondente ao "tirar partido da cache do navegador". Outros fatores que também afetariam o resultado para um melhor valor, seriam o de "eliminar JavaScript e CSS de bloqueio de conversão no conteúdo na parte superior", a "optimização de imagens" e "reduzir JavaScript, CSS e HTML".

Relativamente aos aspectos "ativar compressão", "dar prioridade ao conteúdo visível", "evitar redirecionamento de páginas de destino" e ainda de "reduzir a resposta do servidor", estes foram implementados com sucesso, de acordo com as regras definidas pela ferramenta.

Por último, a nível a experiência do utilizador foi obtido o resultado máximo, o que significa que de acordo com a ferramenta, o *website* está preparado para ser acedido por dispositivos móveis e permitir uma boa acessibilidade.

6 - Conclusão

Existem grandes diferenças do mundo académico para o mundo do trabalho, e a uma determinada altura é necessário transpor a barreira que existe entre ambos. Essa altura surgiu com a realização de um estágio de final de mestrado. Os vários conhecimentos adquiridos ao longo da licenciatura, e do primeiro ano do mestrado, juntamente com a interação dos elementos da entidade acolhedora e do Professor Orientador, permitiram-me realizar as várias tarefas e com elas, obter uma visão mais ampla do que é desenvolvimento de aplicações *web* e da área de *marketing* digital.

Deste modo, o trabalho apresentado no presente relatório focou diversos aspectos relativos ao desenvolvimento de aplicações com o objetivo de marcar presença na *web*. De acordo com o ambiente de trabalho onde estava integrado, foram tidas em conta duas principais linhas orientadoras, o desenvolvimento de *websites* com e sem a utilização de *backoffice*.

Para o desenvolvimento do *backoffice* e dos seus componentes, foi escolhida a *framework* Laravel que utiliza o padrão de desenho MVC. Contudo, apenas alguns projetos utilizaram o *backoffice*, e do mesmo, apenas alguns componentes foram utilizados. Como tal, o principal componente desenvolvido foi o CMS que permite gerir todo o conteúdo apresentado no *frontend*. Para além do CMS, foi também desenvolvido um outro componente com o nome de Project Management Tool que permite a uma determinada empresa, gerir os seus clientes e os projetos que produzem para os respetivos clientes, permitindo a visualização do progresso de cada projeto utilizando um Gráfico de Gantt. Foi utilizado unicamente em empresas produtoras de moldes, mas o seu desenvolvimento não foi exclusivo esta indústria. O Dashboard – Analytics foi um outro componente também desenvolvido. Este, utiliza a API do Google Analytics para apresentar informação relativa ao tráfego realizado num determinado *website*, informação essa que foi obtida através do código de rastreamento do Google Analytics aplicado no respetivo *website*. Por último, foi desenvolvido o componente com o nome de Gestão de Documentos e Ficheiros, que funciona essencialmente como uma área reservada, com o intuito de permitir a um administrador do *website* associar ficheiros e documentos a determinados utilizadores.

Com o objetivo de impulsionar os vários projetos para um elevado *ranking* dos motores de busca, tal como o Google, Yahoo, Bing e outros, foram utilizadas várias estratégias de SEO.

No desenrolar deste estágio fui confrontado várias vezes com prazos e metas a atingir, existindo um compromisso da empresa para com o cliente, para o qual nós (empresa) não podíamos falhar, pois estava em causa o nome da entidade patronal. Assim sendo, todos os prazos foram respeitados e todos os objetivos idealizados no início do estágio foram cumpridos com sucesso.

Para além disso, todos os projetos desenvolvidos até à data da finalização do estágio, foram publicados e estão disponíveis para ser acedidos.

Bibliografia

[1] “W3C HTML,” [Online]. Available: https://www.w3.org/html/.

[2] “HTTP - Hypertext Transfer Protocol Overview,” [Online]. Available: https://www.w3.org/Protocols/.

[3] “Cascading Style Sheets,” [Online]. Available: https://www.w3.org/Style/CSS/.

[4] W3Schools, “HTML5,” [Online]. Available: https://www.w3.org/TR/html5/.

[5] “Introduction to CSS3,” [Online]. Available: https://www.w3.org/TR/2001/WD-css3-roadmap-20010523/.

[6] W3Schools, “JavaScript Tutorial,” [Online]. Available: http://www.w3schools.com/js/.

[7] j. F. -. jquery.org, “jQuery,” [Online]. Available: https://jquery.com/.

[8] W3Schools, “AJAX Introduction,” [Online]. Available: http://www.w3schools.com/ajax/ajax_intro.asp.

[9] 2.-2. T. P. Group, “PHP: Hypertext Preprocessor,” [Online]. Available: http://php.net/.

[10] O. Corporation, “MySQL :: MySQL Workbench,” [Online]. Available: http://www.mysql.com/products/workbench/.

[11] O. Corporation, “MySQL,” [Online]. Available: https://www.mysql.com/.

[12] W3Schools, “SQL Tutorial,” [Online]. Available: http://www.w3schools.com/sql/.

[13] R. Bourdon, “WampServer, the web development platform on Windows - Apache, MySQL, PHP,” [Online]. Available: http://www.wampserver.com/en/.

[14] p. contributors, “phpMyAdmin,” [Online]. Available: http://www.phpmyadmin.net/home_page/index.php.

[15] T. Otwell, “Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/.

[16] T. Otwell, “Artisan CLI - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/artisan.

[17] M. Ehsan, “Architecture of Laravel Applications - Laravel Book,” [Online]. Available: http://laravelbook.com/laravel-architecture/.

[18] T. Otwell, “Templates - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/templates.

[19] M. P. Eva del Nuevo, *Scrum-based Methodology for Distributed Software Development,* 2011.

[20] rener, “Metodologia AGILE,” [Online]. Available: http://pt.scribd.com/doc/209170/Metodologia-AGILE.

[21] S. A. D. N. E. E. M. S. Z. S. A. Ahmed, *Agile Software Development: Impact on Productivity and Quality,* 2010.

[22] M. James, “Scrum Methodology,” [Online]. Available: http://scrummethodology.com/.

[23] P. LePage, “Responsive web design basics | Web Fundamentals - Google Developers,” [Online]. Available: https://developers.google.com/web/fundamentals/layouts/rwd-fundamentals/.

[24] “Configure the Viewport - PageSpeed Insights - Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/ConfigureViewport.

[25] 2.-2. M. D. N. a. i. contributors, “CSS media queries - Web developer guide | MDN,” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Media_queries.

[26] W3Schools, “CSS3 @keyframes Rule,” [Online]. Available: http://www.w3schools.com/cssref/css3_pr_animation-keyframes.asp.

[27] 2.-2. M. D. N. a. i. contributors, “Using CSS animations - CSS | MDN,” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations.

[28] P. A. R. P. S. a. R. Faruk, “Modernizr: the feature detection library for HTML5/CSS3,” [Online]. Available: https://modernizr.com/.

[29] j. F. -. jquery.org, “Selectors | jQuery API Documentation,” [Online]. Available: https://api.jquery.com/category/selectors/.

[30] j. F. -. jquery.org, “.animate() | jQuery API Documentation,” [Online]. Available: http://api.jquery.com/animate/.

[31] WOOTHEMES, “Flexslider by WooThemes,” [Online]. Available: http://www.woothemes.com/flexslider/.

[32] J. Sorgalla, “jCarousel - Riding carousels with jQuery,” [Online]. Available: http://sorgalla.com/jcarousel/.

[33] M. Lou, “Blueprint: Google Grid Gallery,” [Online]. Available:

http://tympanus.net/codrops/2014/03/21/google-grid-gallery/.

[34] D. DeSandro, “Masonry,” [Online]. Available: http://masonry.desandro.com/.

[35] 2.-2. M. D. N. a. i. contributor, “Using data attributes - Web developer guides | MDN,” [Online]. Available: https://developer.mozilla.org/en/docs/Web/Guide/HTML/Using_data_attributes.

[36] P. Botelho, “A Collection of Page Transitions,” [Online]. Available: http://tympanus.net/codrops/2013/05/07/a-collection-of-page-transitions/.

[37] A. Lumsden, “Create a Parallax Scrolling Website Using Stellar.js - Tuts+ Web Design Tutorial,” [Online]. Available: http://webdesign.tutsplus.com/tutorials/create-a-parallax-scrolling-website-using-stellar-js--webdesign-7307.

[38] M. Dalgleish, “Stellar.js,” [Online]. Available: http://markdalgleish.com/projects/stellar.js/.

[39] A. Lumsden, “Create a Parallax Website using Stellar.js,” [Online]. Available: http://webdesigntutsplus.s3.amazonaws.com/tuts/338_parallax/src/index.html.

[40] j. F. -. jquery.org, “jQuery Mousewheel | jQuery Plugin Registry,” [Online]. Available: https://plugins.jquery.com/mousewheel/.

[41] “Jquery Parallax Scrolling Tutorial,” [Online]. Available: http://themeforest.s3.amazonaws.com/116_parallax/tutorial-source-files/tut-index.html.

[42] A. Flesler, “flesler/jquery.scrollTo · GitHub,” [Online]. Available: https://github.com/flesler/jquery.scrollTo.

[43] W3Schools, “HTML5 Web Storage,” [Online]. Available: http://www.w3schools.com/html/html5_webstorage.asp.

[44] M. D. Network, “HTTP cookies - HTTP | MDN,” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies.

[45] K. Limited, “MixItUp,” [Online]. Available: view-source:https://mixitup.kunkalabs.com/.

[46] B. Pratdesaba, “jQuery.animateSprite 1.3.4 | sprite animations made simple,” [Online]. Available: http://blaiprat.github.io/jquery.animateSprite/.

[47] V. Beal, “What is Search Engine Optimization (SEO)? Webopedia,” [Online]. Available: http://www.webopedia.com/TERM/S/SEO.html.

[48] Google, “Search Engine Optimization,” 2010. [Online]. Available: http://static.googleusercontent.com/media/www.google.com/pt-PT//webmasters/docs/search-engine-optimization-starter-guide.pdf.

[49] SEOmoz, “Title Tag - Learn SEO - Moz,” [Online]. Available: http://moz.com/learn/seo/title-tag.

[50] SEOmoz, “Meta Description Tag - Learn SEO - Moz,” [Online]. Available: http://moz.com/learn/seo/meta-description.

[51] T. Belem, “Aprendendo URLs amigáveis (Friendly URLs) - Thiago Belem / Blog,” [Online]. Available: http://blog.thiagobelem.net/aprendendo-urls-amigaveis/.

[52] SEOmoz, “SEO Tools and Search Engine Services - The Beginners Guide to SEO - Moz,” [Online]. Available: https://moz.com/beginners-guide-to-seo/search-engine-tools-and-services.

[53] G. Inc, “Search Console - Página Inicial,” [Online]. Available: https://www.google.com/webmasters/tools/home?hl=pt-PT.

[54] SEOmoz, “Canonicalization and the Canonical Tag - Learn SEO - Moz,” [Online]. Available: https://moz.com/learn/seo/canonicalization.

[55] SEOmoz, “Robots.txt and Meta Robots - SEO Best Practices - Moz,” [Online]. Available: https://moz.com/learn/seo/robotstxt.

[56] M. Purtell, “In 2014, How Important is an H1 Tag for SEO? | SEJ,” [Online]. Available: https://www.searchenginejournal.com/in-2014-how-important-is-an-h1-tag-for-seo/.

[57] Woorank, “How to optimize page load time | SEO Guides from WooRank | WooRank.com | WooRank.com,” [Online]. Available: https://www.woorank.com/en/p/headings-tags.

[58] T. Otwell, “Routing - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/routing.

[59] T. Otwell, “Controllers - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/controllers#controller-filters.

[60] T. Otwell, “Controllers - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: https://laravel.com/docs/4.2/controllers#restful-resource-controllers.

[61] T. Otwell, “Controllers - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: https://laravel.com/docs/4.2/controllers#implicit-controllers.

[62] T. v. d. Broek, *Object Relational Mappings,* 2007.

[63] T. Otwell, “Eloquent ORM - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/eloquent.

[64] “P of EAA: Active Record,” [Online]. Available: http://www.martinfowler.com/eaaCatalog/activeRecord.html.

[65] T. Otwell, “Eloquent ORM - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/eloquent#eager-loading.

[66] T. Otwell, “Query Builder - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/queries.

[67] “PHP: PDO - Manual,” [Online]. Available: http://php.net/manual/en/book.pdo.php.

[68] “PHP: SQL Injection - Manual,” [Online]. Available: http://php.net/manual/en/security.database.sql-injection.php.

[69] N. P. a. D. Mazières, “A Future-Adaptable Password Scheme,” 1999.

[70] T. Otwell, “Laravel - The PHP framework for web artisans.,” [Online]. Available: http://laravel.com/docs/4.2/security#authenticating-users.

[71] T. Otwell, “HTTP Routing - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/master/routing#csrf-protection.

[72] T. Otwell, “Security - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/security#protecting-routes.

[73] “What is Cross-site Scripting and How Can You Fix it?,” [Online]. Available: http://www.acunetix.com/websitesecurity/cross-site-scripting/.

[74] T. Otwell, “Templates - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: http://laravel.com/docs/4.2/templates#other-blade-control-structures.

[75] “Powerful component based mailing library for PHP - Swift Mailer,” [Online]. Available: http://swiftmailer.org/.

[76] Rackspace, “Transactional Email API Service for Developers by Rackspace - Mailgun,” [Online]. Available: https://www.mailgun.com/.

[77] Mandrill, “Transactional Email from MailChimp - Mandrill,” [Online]. Available: https://www.mandrill.com/.

[78] “PHP: mail - Manual,” [Online]. Available: http://php.net/manual/en/function.mail.php.

[79] T. Otwell, “Localization - Laravel - The PHP Framework For Web Artisans,” [Online]. Available: https://laravel.com/docs/4.2/localization#introduction.

[80] C. -. F. Knabben, “CKEditor.com | The best web text editor for everyone,” [Online]. Available: http://ckeditor.com/.

[81] Gantt.com, “What is a Gantt Chart? Gantt Chart Information, history and Software,” [Online]. Available: http://www.gantt.com/.

[82] U. Dinamenta, “Editable JavaScript Gantt Chart - dhtmlxGantt,” [Online]. Available: http://dhtmlx.com/docs/products/dhtmlxGantt/.

[83] U. Dinamenta, “load Gantt Docs,” [Online]. Available: http://docs.dhtmlx.com/gantt/api__gantt_load.html.

[84] U. Dinamenta, “Gantt API Gantt Docs,” [Online]. Available: http://docs.dhtmlx.com/gantt/api__refs__gantt.html.

[85] U. Dinamenta, “Connector object DHTMLX Docs,” [Online]. Available: http://docs.dhtmlx.com/connector__php__connector_object_methods.html#rendersql.

[86] U. Dinamenta, “PHP Connector API DHTMLX Docs,” [Online]. Available: http://docs.dhtmlx.com/connector__php__reference.html.

[87] U. Dinamenta, “DataProcessor DHTMLX Docs,” [Online]. Available: http://docs.dhtmlx.com/dataprocessor__index.html.

[88] U. Dinamenta, “onBeforeTaskAdd Gantt Docs,” [Online]. Available: http://docs.dhtmlx.com/gantt/api__gantt_onbeforetaskadd_event.html.

[89] SetCronJob, “Online CronJobs - Reliable Web Cron service - SetCronJob,” [Online]. Available: https://www.setcronjob.com/.

[90] Google, “Website Oficial do Google Analytics - Análise da Web e Relatórios – Google Analytics,” [Online]. Available: http://www.google.com/analytics/index.html.

[91] D. S. A. P. Deepika Verma, “Google Analytics for Robust Website Analytics,” San José State University.

[92] Google, “OpenID Connect (OAuth 2.0 for Login) - Google Accounts Authentication and Authorization; Google Developers,” [Online]. Available: https://developers.google.com/accounts/docs/OpenIDConnect#consentpageexperience.

[93] Google, “Using OAuth 2.0 to Access Google APIs - Google Accounts Authentication and Authorization; Google Developers,” [Online]. Available: https://developers.google.com/accounts/docs/OAuth2.

[94] Google, “Using OAuth 2.0 for Server to Server Applications - Google Accounts Authentication and Authorization; Google Developers,” [Online]. Available: https://developers.google.com/accounts/docs/OAuth2ServiceAccount.

[95] Google, “Google Developers Console,” [Online]. Available: https://console.developers.google.com/project.

[96] Google, “Hello Analytics API: PHP quickstart for service accounts | Analytics Core Reporting API | Google Developers,” [Online]. Available: https://developers.google.com/analytics/devguides/reporting/core/v3/quickstart/service-php#clientId.

[97] Google, “Installation | API Client Library for PHP (Beta) | Google Developers,” [Online]. Available: https://developers.google.com/api-client-library/php/start/installation.

[98] N. A. &. J. Boggiano, “Composer,” [Online]. Available: https://getcomposer.org/.

[99] Google, “What Is The Management API - Overview | Analytics Management API | Google Developers,” [Online]. Available: https://developers.google.com/analytics/devguides/config/mgmt/v3/.

[100] Google, “What Is The Core Reporting API - Overview - Google Analytics; Google Developers,” [Online]. Available: https://developers.google.com/analytics/devguides/reporting/core/v3/.

[101] Google, “Core Reporting API - Developer Guide - Google Analytics; Google Developers,” [Online]. Available: https://developers.google.com/analytics/devguides/reporting/core/v3/coreDevguide.

[102] Google, “Core Reporting API - Reference Guide | Analytics Core Reporting API | Google Developers,” [Online]. Available: https://developers.google.com/analytics/devguides/reporting/core/v3/reference.

[103] j. F. -. jquery.org, “Accordion | jQuery UI,” [Online]. Available: http://jqueryui.com/accordion/.

[104] C. W. Store, “Window Resizer - Chrome Web Store,” [Online]. Available: https://chrome.google.com/webstore/detail/window-resizer/kkelicaakdanhinjdeammmilcgefonfh?hl=en.

[105] “Screenfly / Test Your Website at Different Screen Resolutions,” [Online]. Available: http://quirktools.com/screenfly/.

[106] H. V. A. O. Shauvik Roy Choudhary, “WEBDIFF: Automated Identification of Cross-browser Issues in Web Applications”.

[107] N. N. Group, “Why You Only Need to Test with 5 Users,” [Online]. Available: http://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/.

[108] G. Developers, “Page Speed Insights,” [Online]. Available: https://developers.google.com/speed/docs/insights/about.

[109] I. Grigorik, “Optimizing encoding and transfer size of text-based assets — Web Fundamentals,” [Online]. Available: https://developers.google.com/web/fundamentals/performance/optimizing-content-efficiency/optimize-encoding-and-transfer#minification-preprocessing--context-specific-optimizations.

[110] “YUI Compressor,” [Online]. Available: http://yui.github.io/yuicompressor/.

[111] “JSMIN, The JavaScript Minifier,” [Online]. Available:

http://www.crockford.com/javascript/jsmin.html.

[112] “cssmin.js / Stoyan's phpied.com,” [Online]. Available: http://www.phpied.com/cssmin-js.

[113] “Optimize Images | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/OptimizeImages.

[114] “Enable Compression | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/EnableCompression.

[115] “Remove Render-Blocking JavaScript | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/BlockingJS.

[116] “Improve Server Response Time | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/Server.

[117] “Reduce the size of the above-the-fold content | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/PrioritizeVisibleContent.

[118] “Configure the Viewport | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/ConfigureViewport.

[119] “Size Content to Viewport | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/SizeContentToViewport.

[120] “Size Tap Targets Appropriately | PageSpeed Insights | Google Developers,” [Online]. Available: https://developers.google.com/speed/docs/insights/SizeTapTargetsAppropriately.

Anexo 1 - Template do Questionário para os Testes de Usabilidade

---

Antes de mais, muito obrigado pela sua disponibilidade para realizar uma sessão que tem como objectivo testar o *website* da Pearlmaster e o *website* da Sunaitec, num computador e num telemóvel.

Como já foi referido, nesta sessão irá testar dois *websites* diferentes, o Pearlmaster e o Sunaitec, onde poderá navegar e efectuar determinadas ações que lhe serão pedidas. Estes *websites* foram desenvolvidos durante o estágio na empresa The Silver Factory.

Toda a informação criada durante esta sessão será recolhida, analisada e anexada a um relatório de estágio desenvolvido no âmbito do mestrado de Engenharia Informática e Computação Móvel na Escola Superior de Tecnologia e Gestão de Leiria.

Esta sessão está dividida em três partes. A primeira parte é composta por um questionário com questões pessoais e questões sobre a sua experiência com o manuseamento de equipamentos tecnológicos. A segunda parte é constituída por dois testes práticos (computador e telemóvel) para cada *website*, e a terceira e última parte é constituída por um questionário acerca da sua experiência ao realizar os testes práticos.

Lembro que toda a informação gerada nas sessões será mantida anónima.

Parte 1

Questionário

Este questionário é composto por dois grupos de questões. Boa sorte!

1. Informação Pessoal

(selecione apenas uma única resposta com um círculo à volta da letra da resposta)

a. Qual a sua Idade?

i. < 18

ii. 18 < 45

iii. 45 < 65

iv. 65 <

b. Sexo

i. Masculino

ii. Feminino

c. Habilitações Literárias

i. Não frequentou nenhum ano

ii. 1º ciclo do ensino básico (4º ano)

iii. 2º ciclo do ensino básico (6º ano)

iv. 3º ciclo do ensino básico (9º ano)

v. Ensino Secundário (12º ano)

vi. Ensino Superior

2. Experiência com tecnologia

(consoante ser verdadeiro (V) ou falso (F), selecione apenas uma única resposta com um círculo à volta da letra da resposta)

a. Possui um computador pessoal ou um dispositivo móvel com interação táctil (por toque e gestos) com ligação à internet.

i. V

ii. F

Se selecionou Falso à afirmação anterior não responda às próximas questões deste Grupo 2.

b. Tem facilidade de interagir com o seu computador pessoal.

i. V

ii. F

c. Tem facilidade de interagir com o seu dispositivo móvel com interação táctil.

i. V

ii. F

d. Tem facilidade em aceder e navegar em websites com o seu computador pessoal.

i. V

ii. F

e. Tem facilidade em aceder e navegar em *websites* com o seu dispositivo móvel com interação táctil.

i. V

ii. F

f. Área de Interesse quando usa o navegador do seu computador ou dispositivo móvel.

(selecione uma ou mais áreas de interesse ao qual costuma pesquisar e navegar, com um círculo à volta da letra)

a. Desporto (A Bola, Record, etc)

b. Lazer (Redes Sociais, Jogos Online)

c. Lojas Online (eBay, Amazon, OLX, etc)

d. Comunicação e Informação (Jornais Online)

e. Outros

Parte 2

Questionário

Teste Prático – Pearlmaster (Computador)

O objectivo deste teste prático é de avaliar a usabilidade do *website* num computador. Este teste prático é composto por N tarefas. Execute cada uma das tarefas e de seguida assinale o nível de dificuldade que sentiu ao executar cada uma das tarefas.

O *website* da Pearlmaster (http://www.pearlmaster.pt/) já se encontra previamente aberto no *browser* do computador.

(selecione apenas uma única resposta com um círculo à volta da letra da resposta)

1. Na página inicial, altere o idioma do *website* para "Espanhol".

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

2. Na página inicial, selecione uma rede social.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

3. Na página inicial, identifique o *slider*.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

4. Na página inicial, no *slider*, navegue para o segundo elemento.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

5. No menu principal, aceda à página do Portfólio.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

6. Na página do Portfólio, dentro da categoria “Automóvel”, altere para a subcategoria “funcionais”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

7. Navegue para o terceiro elemento da galeria.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

8. Volte à página inicial.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

9. No menu principal, aceda à página dos “Contactos”

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

10. Preencha o formulário e envie-o com sucesso.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

11. Identifique a empresa que desenvolveu o *website*.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

Teste Prático – Pearlmaster (Dispositivo Móvel)

O objectivo deste teste prático é de avaliar a usabilidade do *website* num dispositivo móvel. Este teste prático é composto por N tarefas. Execute cada uma das tarefas e de seguida assinale o nível de dificuldade que sentiu ao executar cada uma das tarefas.

O *website* da Pearlmaster (http://www.pearlmaster.pt/) já se encontra previamente aberto no browser do telemóvel.

(selecione apenas uma única resposta com um círculo à volta da letra da resposta)

1. Na página inicial, altere o idioma do *website* para "Espanhol".

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

2. Na página inicial, selecione uma rede social.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

3. Na página inicial, identifique o *slider*.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

4. Na página inicial, no *slider*, navegue para o segundo elemento.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

5. No menu da página inicial, aceda à página do Portfólio

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

6. Na página do Portfólio, dentro da categoria “Automóvel”, altere para a subcategoria “funcionais”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

7. Navegue para o terceiro elemento da galeria.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

8. Volte à página inicial.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

9. No menu principal, aceda à página dos “Contactos”

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

10. Preenche o formulário e envie-o com sucesso.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

11. Identifique a empresa que desenvolveu o *website*.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

12. Identifique o botão da pesquisa, e faça uma pesquisa pela página “Consultores Externos”

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

Teste Prático – Sunaitec (Computador)

O objectivo deste teste prático é de avaliar a usabilidade do *website* num computador. Este teste prático é composto por N tarefas. Execute cada uma das tarefas e de seguida assinale o nível de dificuldade que sentiu ao executar cada uma das tarefas.

O *website* da Sunaitec (http://www.sunaitec.pt/) já se encontra previamente aberto no browser do computador.

(selecione apenas uma única resposta com um círculo à volta da letra da resposta)

1. No menu do topo, altere o idioma do *website* para "Espanhol".

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

2. No menu do topo, selecione uma rede social.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

3. Na primeira secção, identifique o primeiro *slider*.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

4. Nesse *slider*, navegue para o segundo elemento.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

5. Navegue para a secção da “Sunaitec”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

6. Selecione o Botão “Veja o nosso Vídeo”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

7. Navegue para a secção “Vantagens”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

8. Visualize a informação referente à “Integração Arquitetónica”.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

9. Navegue para a secção do “Portfólio”.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

10. Selecione o primeiro elemento da secção do “Portfólio”.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

11. Navegue para a segunda foto.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

12. Navegue para a secção de “Contactos”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

13. Selecione o formulário de “Instaladores”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

14. Preencha o formulário, e envie os dados.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

15. Identifique a empresa que desenvolveu o *website*.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

16. Volte para o topo do *website*.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

Teste Prático – Sunaitec (Dispositivo Móvel)

O objectivo deste teste prático é de avaliar a usabilidade do *website* num dispositivo móvel. Este teste prático é composto por N tarefas. Execute cada uma das tarefas e de seguida assinale o nível de dificuldade que sentiu ao executar cada uma das tarefas.

O *website* da Sunaitec (http://www.sunaitec.pt/) já se encontra previamente aberto no browser do telemóvel.

(selecione apenas uma única resposta com um círculo à volta da letra da resposta)

1. No menu do topo, altere o idioma do *website* para "Espanhol".

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

2. No menu do topo, selecione uma rede social.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

3. Na primeira secção, identifique o primeiro *slider*.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

4. Nesse *slider*, navegue para o segundo elemento.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

5. Navegue para a secção da “Sunaitec”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

6. Selecione o Botão “Veja o nosso Vídeo”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

7. Navegue para a secção “Vantagens”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

8. Visualize a informação referente à “Integração Arquitetónica”.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

9. Navegue para a secção do “Portfólio”.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

10. Selecione o primeiro elemento da secção do “Portfólio”.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

11. Navegue para a segunda foto.

a. Muito fácil

b. Fácil

c. Difícil

d. Muito Difícil

e. Impossível

12. Navegue para a secção de “Contactos”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

13. Selecione o formulário de “Instaladores”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

14. Preencha o formulário, e envie os dados.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

15. Identifique a empresa que desenvolveu o *website*.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

16. Volte para o topo do *website*.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

17. Identifique o botão da pesquisa, e faça uma pesquisa pela página “Contactos”.

    a. Muito fácil

    b. Fácil

    c. Difícil

    d. Muito Difícil

    e. Impossível

Parte 3

Questionário

Este questionário é relativo à sua experiência obtida após a realização dos dois testes práticos para cada *website*. Boa sorte!

1. Experiência Obtida

(selecione apenas uma única resposta com um círculo à volta da letra da resposta)

a. De uma forma geral, considerou intuitiva a forma de navegar nos *websites*?

i. Sim

ii. Não

b. Teve oportunidade de navegar o mesmo *website* num computador e num dispositivo móvel. Onde achou mais fácil de manusear e de efectuar as várias tarefas propostas?

i. Computador

ii. Dispositivo Móvel

Porquê?

_____

_____

_____

c. Teve também a oportunidade de navegar em dois tipos diferentes de *website*, o da Pearlmaster (*multi page*) com vários menus que redirecionavam para outras páginas, e o da Sunaitec (*single page*), que era constituído por uma única página.

Qual deles achou mais fácil de manusear e de efectuar as várias tarefas propostas?

i. Pearlmaster

ii. Sunaitec

Porquê?

_____

_____

_____

d. Tem alguma sugestão a fazer para melhorar o aspecto ou forma de interagir com os *websites* de forma a facilitar a sua utilização?

_____

_____

_____

Anexo 2 – Configurações no ficheiro .htacess do servidor Apache

```
<IfModule mod_gzip.c>
mod_gzip_on Yes
mod_gzip_dechunk Yes
mod_gzip_item_include file .(html?|txt|css|js|php|pl)$
mod_gzip_item_include handler ^cgi-script$
mod_gzip_item_include mime ^text/.*
mod_gzip_item_include mime ^application/x-javascript.*
mod_gzip_item_exclude mime ^image/.*
mod_gzip_item_exclude rspheader ^Content-Encoding:.*gzip.*
</IfModule>

# compress text, html, javascript, css, xml:
AddOutputFilterByType DEFLATE text/plain
AddOutputFilterByType DEFLATE text/html
AddOutputFilterByType DEFLATE text/xml
AddOutputFilterByType DEFLATE text/css
AddOutputFilterByType DEFLATE application/xml
AddOutputFilterByType DEFLATE application/xhtml+xml
AddOutputFilterByType DEFLATE application/rss+xml
AddOutputFilterByType DEFLATE application/javascript
AddOutputFilterByType DEFLATE application/x-javascript

<IfModule mod_expires.c>
ExpiresActive On
ExpiresByType image/jpg "access 1 week"
ExpiresByType image/jpeg "access 1 week"
ExpiresByType image/gif "access 1 week"
ExpiresByType image/png "access 1 week"
ExpiresByType text/css "access 10 minutes"
ExpiresByType text/html "access 1 second"
ExpiresByType application/pdf "access 1 week"
ExpiresByType text/x-javascript "access 1 week"
ExpiresByType application/x-shockwave-flash "access 1 week"
ExpiresByType image/x-icon "access 1 week"
ExpiresDefault "access 2 hours"
</IfModule>
```