

UNIVERSIDADE DE TRÁS-OS-MONTES E ALTO DOURO

**Técnicas e Tecnologias para Desenvolvimento Web na
empresa Junta – Digital Production**

Relatório de estágio de Mestrado em Multimédia

RICARDO GONÇALVES PEREIRA



Vila Real, outubro de 2018

**Técnicas e Tecnologias para Desenvolvimento Web na
empresa Junta – Digital Production**

Por

Ricardo Gonçalves Pereira

Orientador(a): Professor Doutor Emanuel Soares Peres Correia

Coorientador(a): Doutor Telmo Miguel de Oliveira Adão

Agradecimentos

A realização deste relatório de projeto marca o fim de uma importante etapa da minha vida e por isso quero agradecer a todas as pessoas que, direta ou indiretamente contribuíram de forma decisiva para a sua concretização.

Aos meus pais, que foram sem dúvida o meu grande pilar, marcando sempre presença independentemente das dificuldades, estando sempre lá para me ajudar e aconselhar. A eles devo-lhes o maior dos agradecimentos, pela luta constante para que este momento fosse possível. Sei que é o realizar de um sonho para eles, e que ao fim destes anos, de muita luta e dedicação, pode-se gritar em conjunto: Missão Cumprida!

Aos meus padrinhos, um especial agradecimento por todo o apoio, não só durante estes 5 anos, mas sim também durante a minha vida. Apesar das circunstâncias da vida nos separar, limitando-nos a pequenos momentos, vocês marcaram presença em todos eles. Esses são sem dúvida os melhores momentos que levo na bagagem para toda a vida, por isso deixo-vos um especial agradecimento por serem como uns segundos pais para mim.

A toda a minha família, avós, tios e primos, pelos conselhos dados, que serviram de exemplo para o meu crescimento pessoal e pelas palavras de encorajamento durante a minha vida académica.

Aos meus orientadores, Emanuel Peres e Telmo Adão, pela excelente orientação desde o início, pelo empenho e disponibilidade que sempre demonstraram, pela paciência, dedicação, simpatia e rigor. Sem eles não seria possível a conclusão do relatório de projeto.

A toda a equipa da Junta – Digital Production que sempre estiveram disponíveis para me ensinar, esclarecer qualquer dúvida e me incentivaram, enriquecendo ainda mais a minha aprendizagem como profissional e como pessoa.

À UTAD, a mui nobre academia e à cidade de Vila Real que me acolheram ao longo de cinco anos e me disponibilizaram todas as condições para a minha formação académica.

À família académica, Sara Carvalho, Francisco Silva e Ivo Lebreiro, que enriqueceram a minha vida académica, levando assim amizades para a vida.

À família *Panados com Pão* e amigos, António Sobrinho, Pedro Pessoa, Bernardo Silva, João Paulo, Luis Ferreira, Filipe Martins, Pedro Silva, Francisco Nunes, Manuel Pimenta e Luis Evangelista pelo vosso companheirismo e amizade, na qual partilhei momentos incríveis que se transformaram numa amizade vitalícia.

Ao meu melhor amigo, José Teixeira, ao qual tive a felicidade de poder partilhar estes 5 anos de vida académica, para além dos anos de amizade já antes vividos. Com a tua companhia e amizade tudo se tornou bastante mais fácil e sempre mais divertido. Levo na bagagem os melhores momentos, nos quais estiveste presente, por isso muito obrigado pela tua amizade pura e verdadeira.

À família lá da casa, Saulo Fonseca e Ruben Araújo, que foram os melhores colegas de casa e com quem partilhei momentos inesquecíveis.

Por fim agradeço a todas as pessoas que de algum modo estiveram presentes no meu percurso académico e que contribuíram de forma positiva.

Resumo

Com os progressos da tecnologia de consumo, tem-se assistido ao surgimento de dispositivos com diferenciados perfis de visualização - tais como computadores fixos, portáteis, *tablets* e *smartphones*, o que, por sua vez - tem diversificado a forma como os utilizadores acedem à informação, atualmente. Desta forma, derivado à variedade de tamanhos de ecrã dos dispositivos, surge a necessidade de adaptar o desenvolvimento das páginas web, disponibilizando assim esses elementos a qualquer dispositivo, de forma adaptada a cada ecrã.

Para fazer face a este desafio de adaptabilidade, várias técnicas e tecnologias de *front-end* têm vindo a ser propostas, trazendo, assim, benefícios à implementação de projetos de pequena, média ou larga escala.

É precisamente nos aspetos inerentes a este setor de mercado que este documento se centra, através da apresentação de um conjunto de trabalhos desenvolvidos em regime de estágio na empresa Junta – Digital Production, sediada em Matosinhos, que atua nas áreas de desenvolvimento e design para a web e, ainda, produção de vídeo. Será abordada a participação nas tarefas da empresa, referindo técnicas de planeamento e gestão de projetos e tecnologias usadas para a sua execução.

Palavras-chave:

Desenvolvimento de *Front-End*, Desenvolvimento de *Back-End*, Desenvolvimento *web*, Bootstrap, Sass, Laravel, Scrum, Git, SourceTree.

Abstract

With the progresses in consumer-grade technology, it has been witnessed the development of several devices with differentiated display profiles - such as fixed computers, laptops, tablets and smartphones -, which, in turn, has been diversifying the way that users access to information, currently. Thus, due to the variety of screen sizes, a need to adapt web pages development has been identified, more specifically, to uniformize contents presentation along the wide range of devices' screen.

To meet this adaptability challenge, various front-end techniques and technologies have been proposed, bringing benefits to the implementation of small-, medium- or large-scale projects.

This document focuses in the aspects related to this market sector through the presentation of a set of works developed in an internship that took place at *Junta - Digital Production* company, headquartered in Matosinhos, which operates in the areas of web development and design and also video production. The participation in company's business model is addressed along with the applied techniques regarding planning and management of projects, as well as technologies for implementation.

Keywords:

Front-End Development, Back-End Development, Web Development, Bootstrap, Sass, Laravel, Scrum, Git, SourceTree

Índice Geral

Introdução	10
1.1 Motivação	10
1.2 Objetivo do Estágio	11
1.3 Estrutura da Documento.....	11
2 Ferramentas e tecnologias de planeamento e desenvolvimento	12
2.1 Gestão de Projetos.....	12
2.1.1 Metodologias de desenvolvimento ágil.....	12
2.1.2 Scrum	12
2.1.3 Sistema de controlo de versões Git	14
2.2 Desenvolvimento de projetos.....	16
2.2.1 Design web responsivo	16
2.2.2 Frameworks de front-end.....	17
2.2.3 Sistemas de gestão de conteúdos.....	24
2.2.4 Desenvolvimento fullstack.....	27
3 Apresentação da empresa e do grupo de trabalho	30
3.1 Projetos e Tarefas.....	30
3.2 Colaboração em Projetos de Raíz.....	31
3.2.1 HiSeedTech	31
3.2.2 GoParity	34
3.2.3 Ca'dore	36
3.2.4 Norsk Titanium	40
3.3 Colaboração noutros Projetos já Implementados	43
3.3.1 Casa das Ciências - Criação da página de Sugestões	43
3.3.2 Campanha - Cantê	45
3.3.3 Direito ao Esquecimento. Consentimento Geral - Trotinete.....	48
4 Conclusão e trabalho futuro	51
Referências Bibliográficas.....	52

Índice de Figuras

Figura 1 - Passos da Sprint.....	13
Figura 2 - branches no SourceTree1.....	14
Figura 3 - Secção Working Copy do SourceTree.....	15
Figura 4 - Secção Working Copy do GitKraken.....	15
Figura 5 - Secção Working Copy do SmartGit.....	16
Figura 6 - Layout adaptado a qualquer dispositivo.....	17
Figura 7 - Sistema de Colunas Bootstrap.....	18
Figura 8 - Resoluções do Bootstrap 3.....	19
Figura 9 - Resoluções do Bootstrap 4.....	19
Figura 10 - 4 tipos de dispositivos da escala do Bootstrap.....	20
Figura 11 - Exemplo de criação e utilização de variáveis em Sass.....	21
Figura 12 - Exemplo de código em CSS e Nesting em Sass.....	21
Figura 13 - Exemplo da criação de um Partial em Sass.....	22
Figura 14 - Exemplo de Importação em Sass.....	22
Figura 15 - Exemplo de Mixin em Sass.....	23
Figura 16 - Exemplo de Extend/Inheritance em Sass.....	23
Figura 17 - Segundo exemplo de código processado em CSS.....	24
Figura 18 - Gráfico do Google Trends sobre a utilização de Wordpressl, Joomla, Drupal e October CMS.....	25
Figura 19 - Apresentação visual do Backend do October CMS.....	26
Figura 20 - Gráfico do Google Trends sobre a utilização do Laravel.....	27
Figura 21 – Exemplo de sintaxe num mecanimso de modelo e sintaxe de PHP.....	28
Figura 22 - Exemplo da arquitetura MVC.....	29
Figura 23 - Cronograma temporal dos projetos no estágio curricular.....	31
Figura 24 - Página principal do HiSeedTech em todos os dispositivos.....	32
Figura 25 - Slider da secção Our Network do HiSeedTech.....	33

Figura 26 - Secção Core Skills do HiSeedTech, versão desktop e mobile anteriormente explicados.....	34
Figura 27 - Página principal do Goparity em todos os dispositivos.....	35
Figura 28 - Exemplo de Wizard usado no formulário de perfil.....	36
Figura 29 - Página principal Ca'dore em todos os dispositivos.....	37
Figura 30 - Página Sobre Nós da Ca'dore.....	38
Figura 31 - Página de Contactos do Ca'dore.....	39
Figura 32 - Página das Faqs da Ca'dore.....	40
Figura 33 - Página principal do Norsk Titanium em 2 dispositivos.....	41
Figura 34 - Cartões das categorias Discover, Engage e Research.....	41
Figura 35 - Página da notícia em 2 dispositivos.....	42
Figura 36 - Página principal Casa das Ciências em 3 dispositivos.....	43
Figura 37 - Página de Sugestões da Casa das Ciências.....	44
Figura 38 - Página Princiapl Cantê em 3 dispositivos.....	45
Figura 39 - Backend de criação das Campanha Cantê.....	46
Figura 40 - Página das Campanhas Cantê.....	47
Figura 41 - Página de Campanha Cantê.....	48
Figura 42 - Página Principal da Trotinete em todos os dispositivos.....	49
Figura 43 - Página de Direito ao Esquecimento.....	49
Figura 44 - Página de Consentimento Geral.....	50

Introdução

A área do desenvolvimento web é composta por diversos tipos de profissionais, dos quais podem ser destacados o programador de *front-end*, *back-end* e *full stack*. Quanto ao *front-end developer*, este foca-se no desenvolvimento da *interface* visual, que se estabelece como o ponto de contacto direto como programador *web*. O desenvolvimento em *front-end* tem vindo a ganhar relevância devido à rápida taxa de atualização dos conteúdos e às nuances de apresentação dos mesmo que variam de dispositivo para dispositivos, facto que continua a verificar mesmo após o surgimento das tecnologias emergentes, como o HTML e CSS3, que tem vindo a potenciar a disponibilização de interfaces de programação de aplicações (API's) em torno destas tecnologias de tornar o processo de desenvolvimento mais efetivo e fácil de gerir. O *back-end developer* aborda o desenvolvimento no lado do servidor, tratando de toda a lógica de negócio das aplicações *web*. O desenvolvimento de *back-end* tem vindo a evoluir cada vez mais no sentido de permitir ao programador ter um maior controlo sobre cada segmento da aplicação, usando a arquitetura MVC, que permite que este possa efetuar alterações em partes individuais do projeto, sem afetar toda a arquitetura da aplicação. Neste documento será apresentado um portfólio de trabalhos realizados no estágio curricular do mestrado, ao longo dos 9 meses, não só foram desempenhadas tarefas com uma forte componente de implementação em front-end, sendo esta focada no desenvolvimento do visual de um *website*, bem como toda a interação (*user experience*), usando linguagens de programação como em HTML (linguagem de marcação), CSS (linguagem de estilo) e JavaScript (linguagem de script/programação). Foi feito, também desenvolvimento ligado ao *back-end*, cujo foco está a parte de gestão dos serviços aplicativos, sendo o responsável pela implementação da lógica de negócio, usando linguagens de programação como PHP, Java, Python, C#, entre outras. O estágio curricular decorreu nas instalações da Junta – Digital Production que é uma empresa que concentra os seus serviços no desenvolvimento e *design* para *web* e produção audiovisual. Devido ao estágio curricular, tive a oportunidade de poder desempenhar as funções de desenvolvimento *web*, tanto em *front-end* como em *back-end*, em ambiente real de produção para o mercado.

1.1 Motivação

Tendo em vista um contacto real na conceção de projetos multimédia em ambiente empresarial durante 9 meses (Outubro de 2018 até Junho de 2019), o presente relatório de projeto tem como objetivo o estudo e uso de técnicas e tecnologias de desenvolvimento *front-end* na empresa Junta – Digital Production, que focaliza os seus principais serviços em marketing digital, *branding*, *web design* e *web development*.

1.2 Objetivo do Estágio

O estágio curricular tem como principal objetivo a integração e experiência do aluno em ambiente real de produção para o mercado de trabalho, seguindo as metodologias de desenvolvimento instauradas na empresa, sendo que foram delineados vários objetivos de aprendizagem, do modo a poder na produção da empresa, sendo os seguintes:

- Compreender e aplicar a metodologia Ágil Scrum em desenvolvimento de soluções de *software* adotada como convenção pela empresa.

- Compreender e aplicar a *framework* de desenvolvimento *front-end*, Bootstrap e também o pré-processador de CSS o Sass, para a implementação dos *layouts* dos projetos.

- Estudar e usar o sistema de gestão de conteúdos October CMS para a criação de *websites* institucionais e também para comércios eletrônicos.

1.3 Estrutura do Documento

O presente relatório encontra-se dividido em 3 capítulos, para além da introdução, o segundo capítulo aborda o estado da arte, onde é apresentada uma revisão bibliográfica sobre técnicas, tecnologias e conceitos relacionados ao desenvolvimento para *web*.

No terceiro capítulo serão apresentados todos os projetos desenvolvidos, no qual será abordado o planeamento e o desenvolvimento dos mesmos, apresentando imagens inerentes ao projeto.

O quarto capítulo é referente às conclusões obtidas na realização do relatório, uma revisão geral sobre tudo o que foi feito, dificuldades, adequação das metodologias de trabalho, como a experiência obtida no uso das tecnologias para a conceção dos projetos. Por fim serão também definidas expectativas para futuros projetos.

2 Ferramentas e tecnologias de planeamento e desenvolvimento

2.1 Gestão de Projetos

Neste capítulo serão abordadas técnicas de planeamento e gestão utilizadas no desenvolvimento de projetos na empresa Junta – Digital Production.

2.1.1 Metodologias de desenvolvimento ágil

As metodologias ágeis foram criadas no sentido de acelerar e melhorar o processo de desenvolvimento, comunicação e interação das equipas envolvidas na realização de um projeto comum. Desta forma haverá uma maior organização no alcance das metas, um acompanhamento aprimorado do progresso das operações, maior colaboração e resposta rápida por parte da equipa às mudanças. Todos estes aspetos favorecem a produtividade dos programadores, logo da própria empresa também, além de, também, reduzir custos, havendo uma maior satisfação com o trabalho.

2.1.2 Scrum

As metodologias e tecnologias abordadas neste capítulo vão ao encontro da filosofia e funcionamento da empresa, sendo um aspeto bastante importante a referir. Ágil Scrum é a metodologia de desenvolvimento de *software* aplicada na empresa, esta usa repetição e incremento. Foi projetada para gerir requisitos, controlar riscos e otimizar a previsibilidade de um projeto, melhorando a comunicação entre os membros da equipa de desenvolvimento, clientes projetos e outros membros da equipa. Foi definida, formalizada e publicada como a primeira metodologia de desenvolvimento Ágil (Lei, Ganjeizadeh, Jayachandran, & Ozcan, 2017).

No Scrum, os projetos são divididos em ciclos chamados de *Sprints*.

Sprint – Representa um intervalo de tempo dentro do qual há um conjunto de atividades que devem ser executadas. A *Sprint* é composta por uma reunião de planeamento, reuniões diárias, o trabalho de desenvolvimento do projeto, uma revisão da *Sprint* e a retrospectiva da mesma. Na *Sprint* são definidas todas as tarefas a realizar para construir o projeto, cujo planeamento daí resultante irá guiar a equipa de desenvolvimento até ao resultado do produto.

Sprint Planning Meeting (Reunião de planeamento do *Sprint*) – É uma reunião na qual estão presentes o *Product Owner*, que é quem define as funcionalidades desejadas para o

produto, o *Scrum Master*, que é quem procura assegurar que a equipa cumpre e segue as práticas do Scrum, e por fim o *Scrum Team*, que é constituída pelos membros da equipa de desenvolvimento do projeto. Estes intervenientes trabalham juntos para completar o conjunto de trabalho com o qual se comprometeram conjuntamente para o *Sprint*. Nesta reunião o *Product Owner* descreve as tarefas para a equipa, que por fim essas tarefas irão dar origem ao *Sprint Backlog*.

Sprint Backlog – É uma lista de tarefas que o *Scrum Team* se compromete a fazer num *Sprint*. As funcionalidades a serem que desenvolvidas no *Sprint Backlog* são extraídas do *Product Backlog*, com base nas prioridades definidas pelo *Product Owner*.

Product Backlog – É uma lista que contém todas as funcionalidades desejados para o produto, que é definida pelo *Product Owner*. Este deve garantir uma boa organização de modo a maximizar o valor (Schwaber & Sutherland, n.d.), mas esta lista não precisa estar completa no início do projeto. Normalmente, cresce à medida que o projeto vai avançado e houver necessidade de acrescentar pontos importantes a tratar.

Daily Scrum (Reunião Diária) – Durante a *Sprint* a equipa faz reuniões diárias com o objetivo de disseminar conhecimento sobre o que foi feito no dia anterior, identificar problemas ou dificuldades na execução das tarefas, de modo a definir uma estratégia para colmatar os problemas. Contudo esta reunião não deve ser usada como uma reunião de resolução de problemas, mas sim para haja um conhecimento do que foi feito no dia anterior e haja um compromisso daquilo que vai ser tratado durante o dia. A abordagem aos problemas na execução das tarefas pode ser feita unto do *Scrum Master*, no âmbito de uma outra reunião.

Sprint Retrospective – Ocorre ao final de um *Sprint* e serve para identificar o que funcionou bem, o que pode ser melhorado e que ações serão tomadas para melhorar.

Sprint Review Meeting – Ao final de cada *Sprint* é feito uma reunião com a *Scrum Team*, o *Scrum Master* e o *Product Owner*, para a apresentação do que foi alcançado na *Sprint*.

Na Figura 1 pode-se verificar todos os passos efetuados na *Sprint*, desde a primeira reunião de planeamento do *Sprint* até à apresentação do produto.

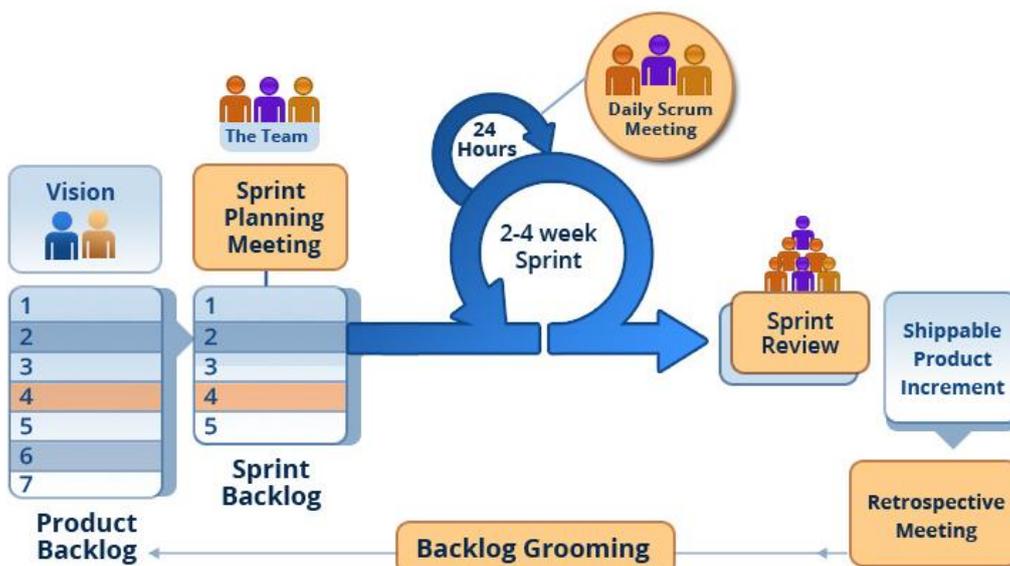


Figura 1 - Passos da Sprint. (Burk, 2017)

2.1.3 Sistema de controlo de versões Git

No desenvolvimento web em equipa, é comum o acesso ao mesmo arquivo por diferentes elementos, sendo que, para evitar problemas de edição e conflitos, usa-se o Git. Trata-se de um sistema de controlo de versões distribuídos, DVCS (*Distributed Version Control System*), ou seja, a cópia de trabalho de cada programador do código é um repositório que contém o histórico completo de todas as mudanças. Com este sistema é possível vincular de forma automática a uma edição, uma data e o elemento da equipa de desenvolvimento que submeteu a alteração. Para tal, recorre-se ao *commit*, que significa entrega ao outro lado, normalmente feito juntamente com uma descrição para o repositório. Assim, o Git irá combinar e comparar versões. Antes de usar qualquer ficheiro, deve-se verificar se há algum *pull* disponível, ou seja, se existe algum ficheiro editado por outro elemento da equipa. Sem fazer o *pull*, trazendo para a sua versão local as alterações feitas por outro membro de equipa, mais tarde irão ocorrer erros de submissão, quando for efetuado o *push*. Esses erros advêm da sobreposição de código num ficheiro, anteriormente editado, gerando assim conflitos de código. Sendo que, a boa prática seja efetuar sempre o *pull* de todos os ficheiros, de modo a trabalhar sempre na versão atual dos mesmos. Um dos principais objetivos da sua criação foi a flexibilidade, sendo este um suporte para vários tipos de fluxos de trabalho de desenvolvimento não-linear e também eficiente em projetos pequenos e grandes com compatibilidade para muitos sistemas e protocolos existentes.

SourceTree

O SourceTree é uma ferramenta gratuita, que fornece uma interface entre o utilizador e o Git, sem ter de recorrer a linhas de comando, como se pode verificar na Figura 2. Este gere de forma bastante simples os *branches*, permitindo assim a submissão de ficheiro para comparação, com um simples clique. Este informa também se temos ficheiros para fazer *push* ou para fazer *pull*.

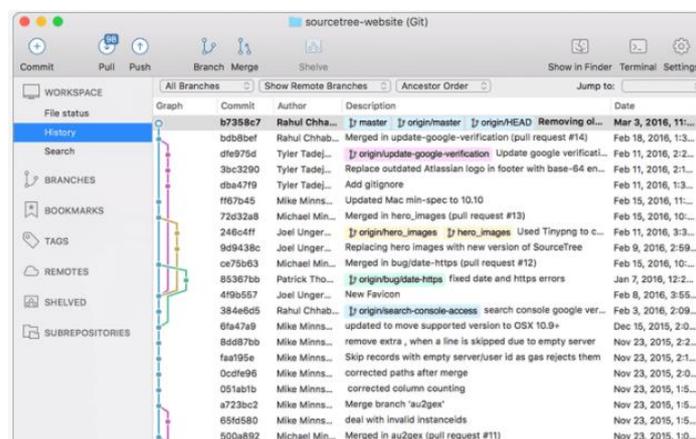


Figura 2 - branches no SourceTree

Este apresenta uma interface, onde apresenta a secção *Working Copy* que mostra a diferença entre a cópia atual e a versão no repositório Git, dando assim uma visão em tempo real dos arquivos locais para que se possa ver exatamente o que se alterou, como se pode visualizar na Figura 3

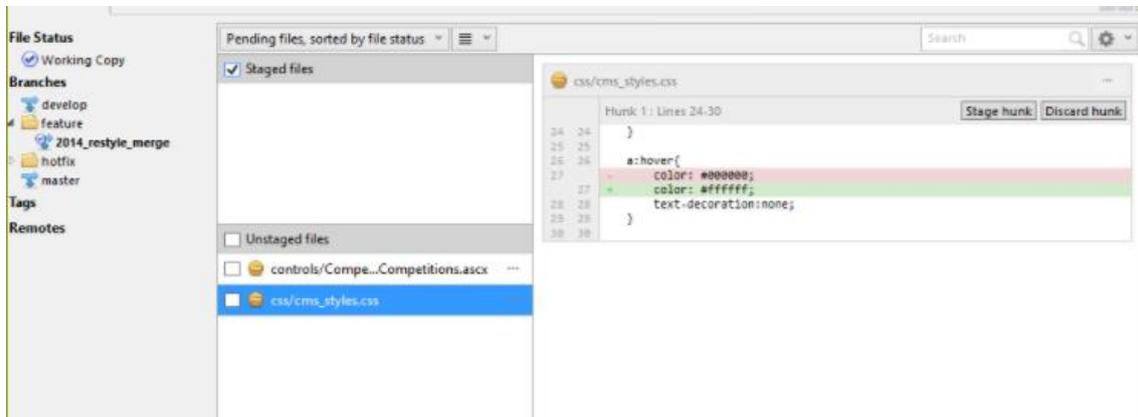


Figura 3 - Secção *Working Copy* do SourceTree

Existem outros *softwares* para o mesmo intuito que o SourceTree, como por exemplo, o GitKraken (Figura 4) ou o SmartGit (Figura 5), ambos apresentam também uma interface, que mostra as versões dos arquivos no repositório, permitindo também efetuar o *pull* e o *push* dos ficheiros.

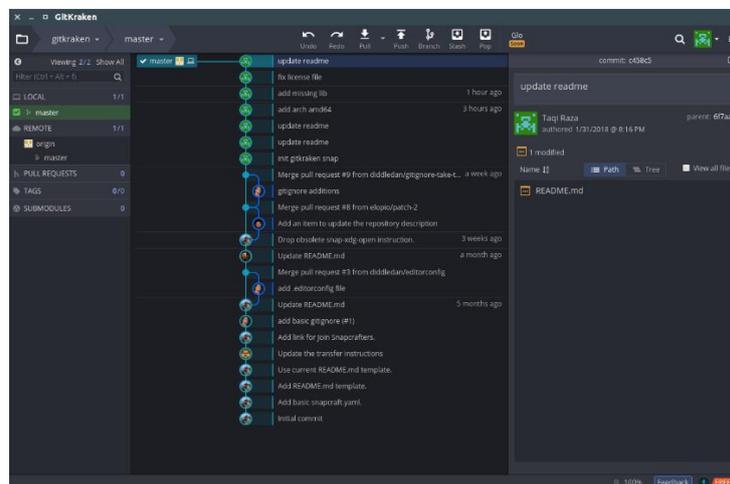


Figura 4 - Secção *Working Copy* do GitKraken

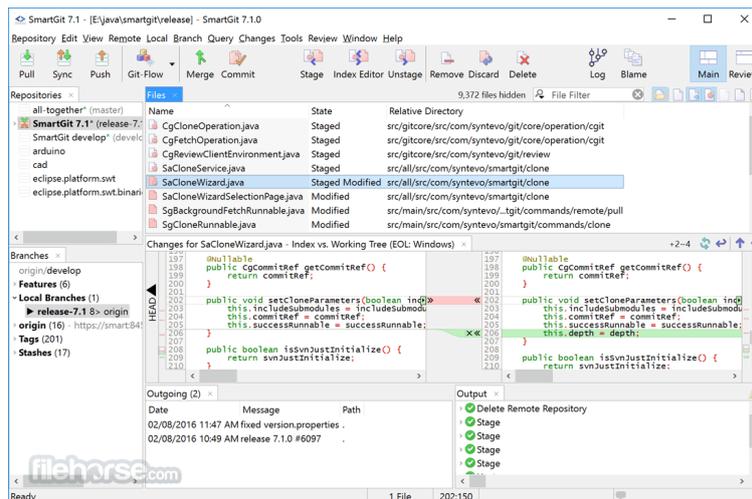


Figura 5 - Seção Working Copy do SmartGit

2.2 Desenvolvimento de projetos

Neste capítulo serão abordadas as tecnologias de utilizadas no desenvolvimento dos projetos. Serão introduzidos os conceitos de *design* web responsivo, as ferramentas utilizadas no desenvolvimento de *Front-end* e as ferramentas para o desenvolvimento em *Back-end*.

2.2.1 Design web responsivo

Na criação de páginas *online* e aplicações web as principais linguagens usadas são as primeiras são o HTML, que é um editor de hipertextos, o CSS que é um mecanismo para adicionar estilos (cores, fontes, espaçamentos, etc.) a um documento web e por fim o JavaScript, linguagem de programação implementada como dos navegadores web para que *scripts* possam ser executados do lado do cliente e interajam com o utilizador sem necessidade deste passar pelo servidor. Estas 3 linguagens permitem - por meio de formatação e *scripting* - a criação de recursos online, que podem ser acedidos na World Wide Web, através da Internet. Com a evolução tecnológica começaram a surgir inúmeros dispositivos com capacidade de navegar na web, sendo que esses mesmos apresentam dimensões distintas para os ecrãs, levando assim à necessidade de criar “mecanismos” para o desenvolvimento de *layouts* adequados a cada um. Surge assim a necessidade de um *design* web responsivo, que é um conjunto de técnicas aplicadas ao nível do layout, de modo que permite que um site se adapte a qualquer dispositivo ou largura do ecrã, como se pode verificar na Figura 6



Figura 6 - Layout adaptado a qualquer dispositivo

O Design web responsivo tem como principal objetivo adaptar o *layout* para se adequar a diferentes tamanhos de ecrã, desde desktops de monitor amplo até telefones pequenos.

2.2.2 Frameworks de front-end

Framework de front-end, pode ser definido como “pacote” que contém código pré-escrito e padronizado em arquivos e pastas, dando assim uma base para se construir aplicações, de forma flexível. Por norma, estas estruturas de front-end contêm:

- Uma grade, mecanismo que funciona como uma tabela abstrata, é responsiva e orientada a dispositivos móveis ajustando-se de acordo com a tela, quando esta muda de tamanho ou de orientação. Simplificando deste modo a organização dos elementos de design do *site*
- Estilos de fonte definidos
- Componentes pré-construídos do *site*, como painéis laterais, botões ou barras de navegação, entre outros componentes.

Bootstrap

O Bootstrap é uma estrutura livre e de código aberto para criar aplicações e sítios *web* utilizada na empresa Junta Digital – Production. Existem outras Frameworks do mesmo género do Bootstrap, como por exemplo o Foundation ou o Pure, que era usada

anteriormente com mais frequência na empresa. De acordo com o sítio *web* StackShare (“Bootstrap vs. Foundation vs. Pure”, 2018) o Bootstrap parece ser a mais popular ferramenta face aos outros candidatos. Aliás, é muito mais popular! Tem muitos mais fans, muitos mais projetos no Github e gera um número pornográfico de threads no stack overflow, em comparação com os outros 2. Por essas razões anteriormente descritas o Bootstrap é a Framework preferencial da empresa para o desenvolvimentos dos projetos da empresa, visto ser tão flexível e personalizável. Este é baseado num sistema de grelha dividido em 12 colunas que podem agrupar-se consoante a necessidade do desenvolvimento de uma página web, ou seja, pode-se dividir as colunas consoante a necessidade, como é possível confirmar na Figura 7.

.col-md-1											
.col-md-8								.col-md-4			
.col-md-4				.col-md-4				.col-md-4			
.col-md-6						.col-md-6					

Figura 7 - Sistema de Colunas Bootstrap

O sistema de grelha adotada pela framework permite adaptar-se a qualquer tipo de resolução como é possível visualizar na Figura 8 que até à 3ª versão do Bootstrap existe 4 escalas que são:

- “**Extra small devices**” - escala para telemóveis que abrange a resolução até 768 pixéis, na qual se usa classe “**.col-xs-**”;
- “**Small devices**” - para tablets em que a resolução vai dos 768 pixéis até aos 992 pixéis, usando-se a classe “**.col-sm-**”. Em seguida dos 992 pixéis até 1200 pixéis a
- “**Medium devices**” - para resoluções até 1200 pixéis, usando-se a classe “**.col-md-**”;
- “**Large devices**” - que é uma escala que começa a partir dos 1200 pixéis que é para computadores de grandes resoluções, que usa a escala “**.col-lg-**”.

	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Grid behavior	Horizontal at all times		Collapsed to start, horizontal above breakpoints	
Max container width	None (auto)	750px	970px	1170px
Class prefix	<code>.col-xs-</code>	<code>.col-sm-</code>	<code>.col-md-</code>	<code>.col-lg-</code>
# of columns	12			
Max column width	Auto	60px	78px	95px
Gutter width	30px (15px on each side of a column)			
Nestable	Yes			
Offsets	Yes			
Column ordering	Yes			

Figura 8 - Resoluções do Bootstrap 3

A partir da 4ª versão do Bootstrap há uma reestruturação das grelhas tendo sido adicionado uma nova escala de modo a abranger de forma mais eficiente os dispositivos móveis de baixa resolução, sendo que a resolução pequena é 576 pixels e não os 768 pixels anteriormente referidos. Como se pode verificar na Figura 9, para além da mudança anteriormente referida, também mudaram os “termos” para cada escala, sendo que a anterior escala “*large*” passa-se a chamar de “*Extra large*”, permitindo que na escala mais pequena, ou seja, a de 578 pixels possa assumir o termo “*Extra small*”.

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	<code>.col-</code>	<code>.col-sm-</code>	<code>.col-md-</code>	<code>.col-lg-</code>	<code>.col-xl-</code>
# of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

Figura 9 - Resoluções do Bootstrap 4

Deste modo, consegue-se abranger um aglomerado de escalas de forma a ter uma maior eficiência na adaptabilidade dos sítios *web*, como se pode ver no exemplo na Figura 10.



Figura 10 - 4 tipos de dispositivos da escala do Bootstrap

Sass

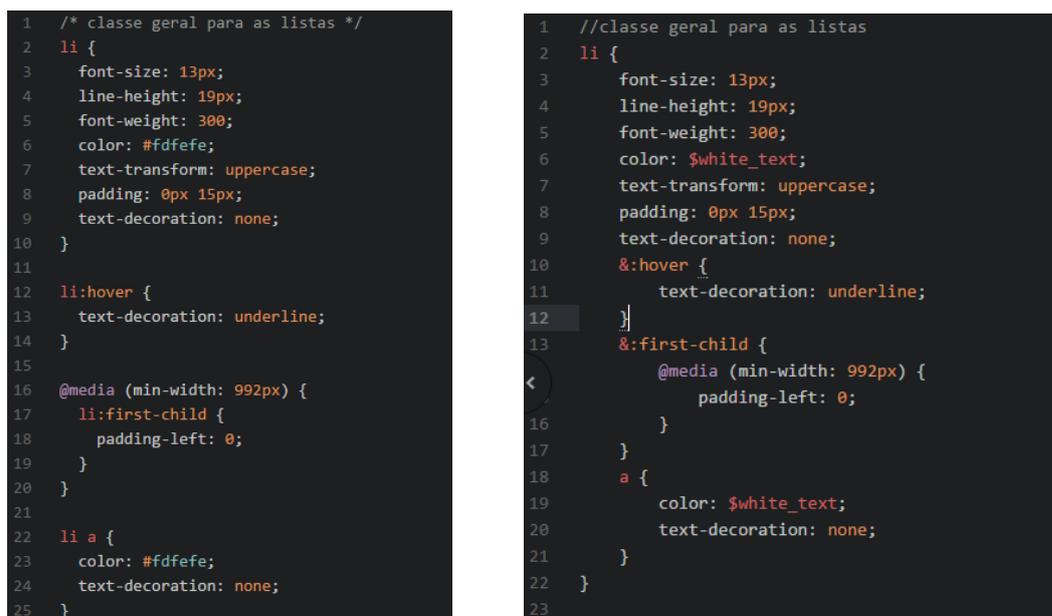
O Sass é um pré-processador que compila a sintaxe usada para CSS, que tem como principal função ajudar a escrever código reutilizável, sustentável e extensível em CSS. Os pré-processadores surgem com a necessidade de “retirar” peso no carregamento dos sítios *web*, especialmente em projetos muito longos que recorrem a extensos excertos de especificação CSS que, por sua vez, tornam os sítios web mais lentos. O Sass trouxe muitas novidades e uma nova forma de definir *layout* com a sintaxe de CSS, usando **Variables**, de modo a poder inicialmente na criação de um projeto se ponderar cores a utilizar, tamanhos de letras, fontes, e outros dados que sejam utilizados várias vezes, desta forma guardamos esses dados em variáveis de modo a poder reutilizar esse código, o que traz uma melhor experiência de definição de interfaces para o utilizador, promovendo a adoção de boas práticas. Na Figura 11 temos um exemplo da utilização de **Variables** em Sass.

```
1 //Variáveis para CSS
2
3 //Cores
4 $azul_escuro: #1d1f33;
5 $azul_claro:#0054a5;
6
7 //Tamanhos de Letras
8 $titulos: 20px;
9 $texto: 13px;
10
11 //Código CSS
12
13 //classes para os títulos e texto de uma determinada página usando as variáveis anteriormente definidas
14 .titulos_pagina_principal{
15   color: $azul_claro;
16   font-size: $titulos;
17   line-height: 20px;
18 }
19
20 .textos_pagina_principal{
21   color: $azul_escuro;
22   font-size: $texto;
23   line-height: 15px;
24 }
25
```

Figura 11 - Exemplo de criação e utilização de variáveis em Sass

Neste exemplo, definem-se variáveis para cores e para tamanhos de letra, que mais tarde foram usados em classes para títulos e para texto.

Uma grande novidade que o Sass trouxe foi o ***Nesting***, possibilitando ao utilizador criar uma hierarquia no código CSS. Desta forma, torna-se a especificação ainda mais organizado e legível, o que antes não era possível, uma vez que a definição CSS requeria sempre que criar as combinações em linhas separadas. Nos seguintes exemplos, na Figura 12, tem-se a comparação entre codificação em CSS e o mesmo código em Sass.



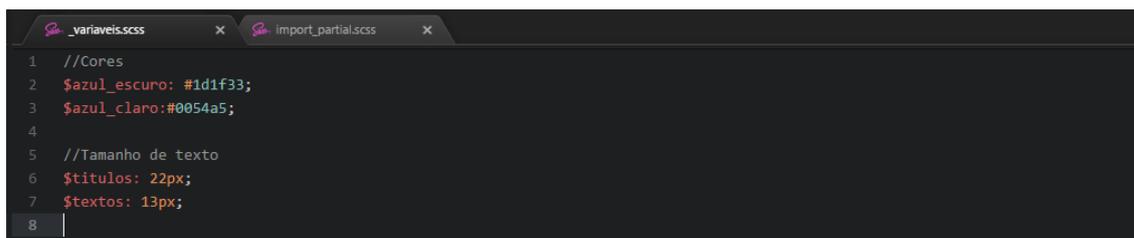
```
1 /* classe geral para as listas */
2 li {
3   font-size: 13px;
4   line-height: 19px;
5   font-weight: 300;
6   color: #fdfeff;
7   text-transform: uppercase;
8   padding: 0px 15px;
9   text-decoration: none;
10 }
11
12 li:hover {
13   text-decoration: underline;
14 }
15
16 @media (min-width: 992px) {
17   li:first-child {
18     padding-left: 0;
19   }
20 }
21
22 li a {
23   color: #fdfeff;
24   text-decoration: none;
25 }
```

```
1 //classe geral para as listas
2 li {
3   font-size: 13px;
4   line-height: 19px;
5   font-weight: 300;
6   color: $white_text;
7   text-transform: uppercase;
8   padding: 0px 15px;
9   text-decoration: none;
10   &:hover {
11     text-decoration: underline;
12   }
13   &:first-child {
14     @media (min-width: 992px) {
15       padding-left: 0;
16     }
17   }
18   a {
19     color: $white_text;
20     text-decoration: none;
21   }
22 }
23 }
```

Figura 12 - Exemplo de código em CSS e Nesting em Sass

Nestes dois exemplos pode-se ver que a diferença de estrutura de sintaxe do Sass para o CSS, tornando o código mais organizado e mais facilmente legível, proporcionando uma melhor experiência para o utilizador.

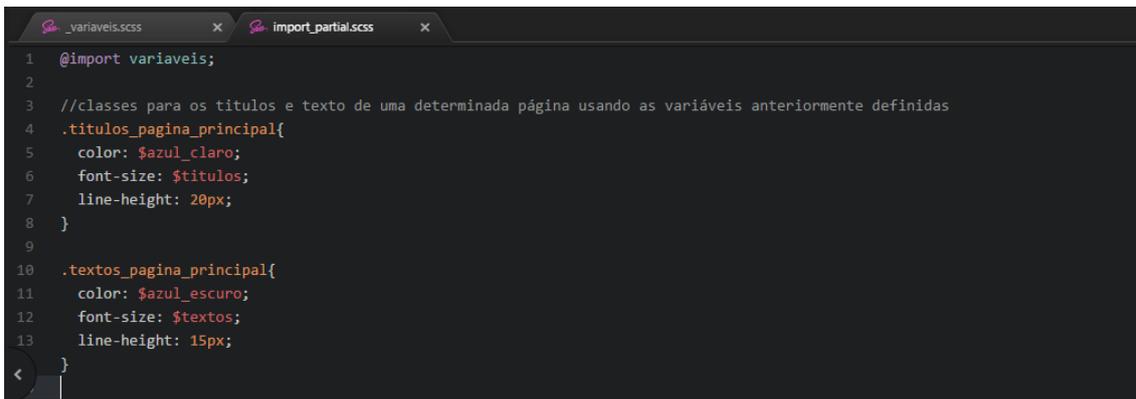
Outra novidade são os ***Partials*** e o ***Import***, que uso em conjuntos destes dois permitem a criação de outros ficheiros Sass que mais tarde podem ser incluídos noutros ficheiros, de modo a poder organizar melhor o código e tornar mais leve o carregamento das páginas do sítio *web*. Na Figura 13 seguinte são criadas as variáveis num ficheiro a parte chamado `_variaveis.scss`, que mais tarde serão importadas noutros ficheiros.



```
1 //Cores
2 $azul_escuro: #1d1f33;
3 $azul_claro:#0054a5;
4
5 //Tamanho de texto
6 $titulos: 22px;
7 $textos: 13px;
8
```

Figura 13 - Exemplo da criação de um Partial em Sass

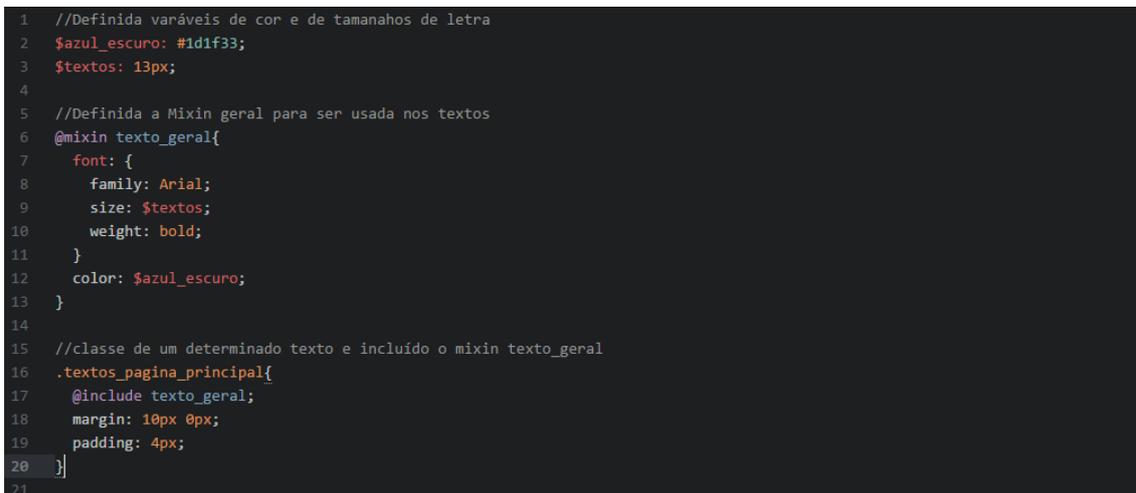
Por fim, no ficheiro **import_partial.scss** importa-se o ficheiro **_variaveis.scss**, como se pode verificar na seguinte Figura 14, permitindo assim que as variáveis possam ser reutilizadas, tornando assim o código muito mais leve e organizado. Assim, através da gestão de recursos, que são carregados apenas quando necessários, otimiza-se o carregamento dos sítios *web*.



```
1 @import variaveis;
2
3 //classes para os titulos e texto de uma determinada página usando as variáveis anteriormente definidas
4 .titulos_pagina_principal{
5   color: $azul_claro;
6   font-size: $titulos;
7   line-height: 20px;
8 }
9
10 .textos_pagina_principal{
11   color: $azul_escuro;
12   font-size: $textos;
13   line-height: 15px;
14 }
```

Figura 14 - Exemplo de Importação em Sass

Outro recurso bastante interessante que contribui para a redução e organização de código é a **Mixin**, que consiste num bloco que permite definir estilos para serem reutilizados por outras classes CSS. Na figura 15 temos um exemplo do uso do **Mixin**.



```
1 //Definida variáveis de cor e de tamanhos de letra
2 $azul_escuro: #1d1f33;
3 $textos: 13px;
4
5 //Definida a Mixin geral para ser usada nos textos
6 @mixin texto_geral{
7   font: {
8     family: Arial;
9     size: $textos;
10    weight: bold;
11  }
12  color: $azul_escuro;
13 }
14
15 //classe de um determinado texto e incluído o mixin texto_geral
16 .textos_pagina_principal{
17   @include texto_geral;
18   margin: 10px 0px;
19   padding: 4px;
20 }
21
```

Figura 15 - Exemplo de Mixin em Sass

Pode-se verificar na imagem que foram definidas variáveis de cor e de tamanhos utilizados no **Mixin**, que posteriormente foram incluídas na classe “.textos_paginas_principal”, adquirindo assim as características do **Mixin** “texto_geral”.

Por fim temos o *Extend/Inheritance* em que o conceito é parecido ao *Mixin*, mas em vez de adquirir os estilos da classe definida no *Mixin*, neste caso usa-se uma classe geral e faz-se *@extend* dessa mesma classe geral, fazendo com que a nova classe adquira as características da geral, como se pode ver no seguinte exemplo com a Figura 16. Este conceito é análogo aquilo que acontece na programação orientada a objetos, quando se recorre a mecanismos de herança ou extensão.

```
1 //Definida variáveis de cor e de tamanhos de letra
2 $azul_escuro: #1d1f33;
3 $textos: 13px;
4
5 //Definida uma classe geral para o texto
6 .texto_geral {
7     font: {
8         family: Arial;
9         size: $textos;
10        weight: bold;
11    }
12    color: $azul_escuro;
13 }
14
15 //Fez-se @extend do texto_geral para o Parágrafos (p) e para as listas (ul, ol)
16 p {
17     @extend .texto_geral;
18     border: 1px solid $azul_escuro;
19 }
20
21 ul, ol {
22     @extend .texto_geral;
23     text-transform: uppercase;
24 }
25
```

Figura 16 - Exemplo de Extend/Inheritance em Sass

Como se pode verificar na imagem, foram definidas variáveis de cor e de tamanho, criou-se uma classe geral que fora estendida para os parágrafos (*p*), e para as listas (*ul* e *ol*), fazendo com que estes adquirissem as características da classe geral. Por fim, quando este é processado para CSS, à classe geral são adicionadas o parágrafo e as listas, ficando assim com esta formatação mais curta, mais organizada e mais leve, visível na Figura 17.

```
1 .texto_geral, p, ul, ol {
2     font-family: Arial;
3     font-size: 13px;
4     font-weight: bold;
5     color: #1d1f33;
6 }
7
8 p {
9     border: 1px solid #1d1f33;
10 }
11
12 ul, ol {
13     text-transform: uppercase;
14 }
```

Figura 17 – Segundo exemplo de código processado em CSS

2.2.3 Sistemas de gestão de conteúdos

Um CMS ou *Content Management System* é uma aplicação de *software* ou um conjunto de programas relacionados que são usados para criar e gerir conteúdos digitais. Os CMS são normalmente usados para gerir conteúdo empresarial (ECM/ECMS) ou para gerir conteúdo da *web* (WCM).

Um ECMS ou *Enterprise Content Management System* é uma aplicação criada para atender às necessidades de uma organização de grande escala, fornecendo um sistema de gestão de conteúdo aos processos da empresa, com ferramentas e recursos para gerir, armazenar e entregar conteúdo e documentos enquadrados no contexto organizacional.

Um WCM ou *Web Content Management System* é uma aplicação da *web* que fornece recursos para vários utilizadores com diferentes níveis de permissão para gerir o conteúdo *web*, sem a necessidade de ter conhecimento em HTML. Deste modo o WCM fornece um painel de administração que possibilita criar, editar, publicar páginas *web* e artigos.

Existem vários tipos de CMS no mercado, sendo que um dos mais utilizados é o WordPress devido à sua facilidade de utilização, que não requer instalação, sendo o site hospedado, tendo disponível vários *plugins* já instalados e ainda atualizações e segurança gratuita. Outra CMS bastante usada no mercado de trabalho é o Joomla, que é grátis e *open-source*, baseado no modelo MVC que pode ser usado de forma independente do CMS, sendo poderoso para criação de aplicações *online*. Outra solução do mesmo género bastante forte no mercado é o Drupal que permite criar e gerir aplicações *web*, podendo melhorá-las e adaptá-las rapidamente à comunidade Drupal. O CMS utilizado na empresa Junta – Digital Production é o October CMS, que é baseado em Laravel, este é bastante recente com pouco mais de 4 anos sendo que a sua comunidade é mais pequena que as anteriormente referidas. Tal como se pode verificar na Figura 18, baseado numa pesquisa a partir do *Google Trends*, o wordpress está no topo da lista de CMS usadas pelo mercado.

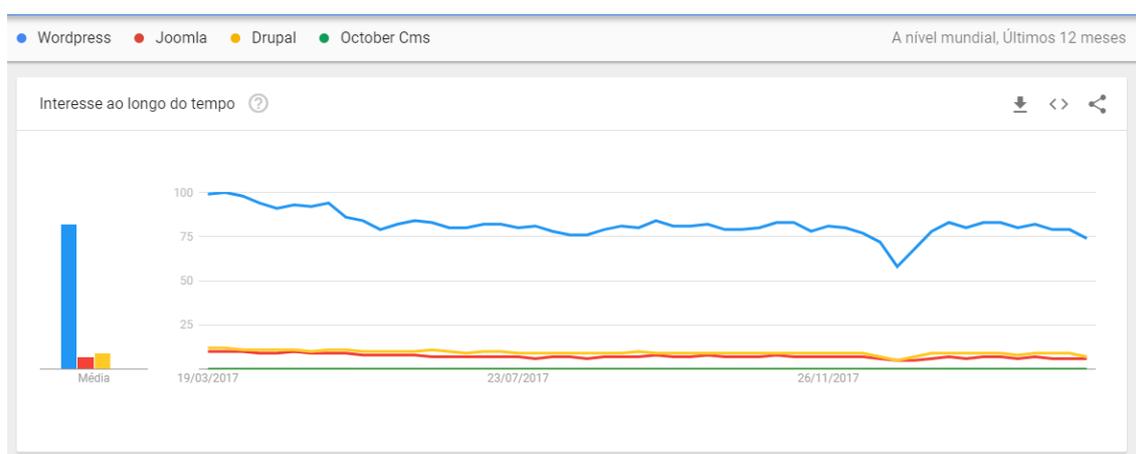


Figura 18 - Gráfico do Google Trends sobre a utilização de Wordpress, Joomla, Drupal e October CMS

October CMS

O October CMS é um sistema de gestão de conteúdo leve baseado na linguagem de programação PHP e na *framework* de aplicações para a web, Laravel. Este suporta o MySQL, o SQLite e o PostgreSQL para a base de dados do *back-end*. O objetivo deste é tornar o fluxo de trabalho de desenvolvimento web mais fácil, sendo que a sua aprendizagem é simples e rápida. É uma ferramenta escalável e extensível através do seu sistema de *plugins*, permitindo, assim, a fácil criação de interfaces administrativas de *back-end*. Tal como o Wordpress, o October suporta o uso de temas, quem definem o *design* do sítio web. Alterando o tema, mudar-se-á o aspeto ou aparência do *front-end* da página. Comparativamente a outro CMS, como por exemplo, o Wordpress, este é usado por cerca de 27% da *web*. Segundo um estudo comparativo entre o Wordpress e o October CMS, afirmando que mais de 18 milhões de *sites* foram criados a partir da CMS Wordpress e cerca de 13 mil criados com base no October CMS até à data que foi feita a comparação pela Leader Internet em 2017 (“October CMS Vs Wordpress”, (2018)). Ao contrário do *backend* do Wordpress que nem sempre é muito óbvio para onde se deve seguir, o October tem um *backend* bastante fácil de navegar. A apresentação visual do *backend* tem layout atrativo e responsivo, como pode visualizar na Figura 19.

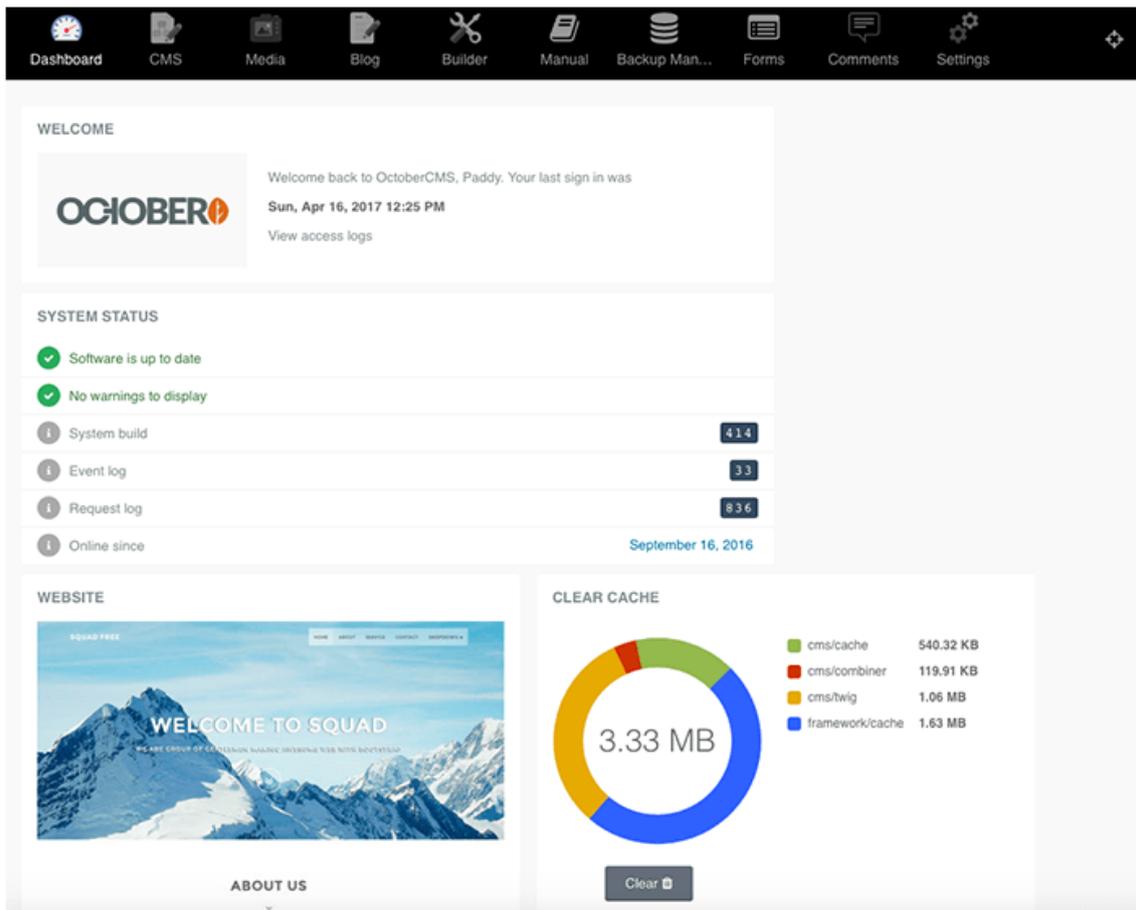


Figura 19 - Apresentação visual do Backend do October CMS

Este apresenta um código MVC limpo e bastante fácil de estender, facilita também criar o seu próprio plugin, apresentando documentação explicativa. O October é extensível, construído em Laravel, em estrutura AJAX, apresenta uma interface de limpa para administrador, páginas construídas com componentes reutilizáveis e o seu processo de aprendizagem é bastante rápido. Segundo a comparação feita pelo *site* Leader Internet, no ponto 10 sobre o desempenho das CMS's, o October é bastante mais rápido, pois este como é criado com base em Laravel, o seu nível de *performance* é bastante melhor. Noutro ponto de comparação o *site* Leader Internet aborda a questão da segurança e conclui que o October, apenas permite o acesso ao index.php e todos os plugins passam por um processo de aprovação, enquanto no Wordpress qualquer ficheiro PHP pode ser executado, sem qualquer nenhum processo de aprovação dos plugins. No final em termos de quota de mercado o Wordpress apresenta uma clara vantagem, uma boa parte derivada ao tempo que está no mercado comparativamente ao October CMS que é bastante mais recente, fazendo com que tenha maior vantagem em termos de suporte, *plugins* e popularidade, o que para a maioria das pessoas sé fator decisivo na escolha do uso de CMS. Na criação de sítios web de portfólio simples ou apresentação de uma empresa pequena, o Wordpress torna-se a escolha mais óbvia, mas no caso da construção de uma aplicação que exija maior personalização do que a instalação base do CMS, o October CMS oferece uma maior flexibilidade no desenvolvimento de sítios web e aplicações com recursos avançados de maneira estruturada.

2.2.4 Desenvolvimento fullstack

Laravel

Laravel é uma framework de desenvolvimento rápido, livre e de código aberto em PHP, tem como principal objetivo permitir trabalhar de forma estruturada e rápida. O Laravel é das frameworks mais populares da atualidade, possuindo uma equipa de desenvolvimento ativa e competente, tendo também uma comunidade bastante grande e uma grande aceitação no mercado, tal como se pode verificar na Figura 20, que comprova o crescimento de popularidade do Laravel, através do Google Trends.

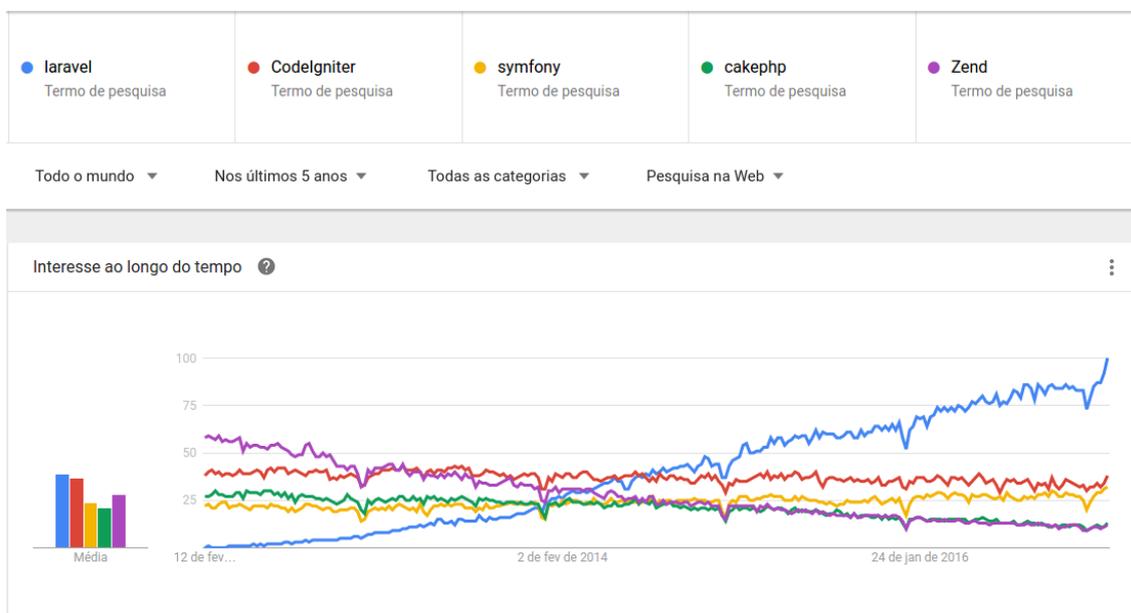


Figura 20 - Gráfico do Google Trends sobre a utilização do Laravel

O que faz do Laravel uma ferramenta poderosa é o facto de ser tão diversificado e ao mesmo tempo ser simples manuseamento. Eis alguns exemplos de recursos do Laravel:

- **Composer** – é usado para gerir dependências de forma fácil e gerir pacotes de terceiros nas aplicações.
- **Sistema de rotas** – servem para fazer um mapeamento do URL digitado no navegador para alguma ação dentro da aplicação.
- **Blade – Sistema de templates** – é um compilador de *templates* do Laravel. A diferença deste para os outros *templates* é a sua flexibilidade, não restringindo o uso de PHP misturado na *syntaxe* do *template*. O grande objetivo do Blade é reduzir a quantidade de código PHP

inserido no meio do HTML e aumentar a reutilização, disponibilizando uma serie de diretivas que são inseridas junto ao código HTML.

- **Eloquent e QueryBuilder** – O Eloquent é o *Object-Relational Mapping* (ORM que é uma Framework que permite trabalhar com código SQL em qualquer altura) padrão do Laravel que através de uma classe “*Model*” interage com tabelas da base de dados. Permite desta forma, a consulta de dados com operações CrUD (*create-update-delete*). A grande vantagem é poder implementar aplicações sem ter que escrever instruções SQL embebidas no código.

- **QueryBuilder** – é um construtor de queries, que permite executar a maioria das operações de base de dados.

- **Artisan Console** – O Artisan é uma interface em linha de comandos que o desenvolvimento das aplicações.

Outra *framework*, com bastante interesse no mercado de trabalho é o CodeIgniter, baseado em PHP, esta *framework* é um kit de ferramentas para a criação de aplicações *web*. Este apresenta uma biblioteca completa para as principais tarefas de desenvolvimento *web*, tais como, aceder à base de dados, enviar e-mails, validar dados de formulários, manter sessões, entre outros. O CodeIgniter não requer um mecanismo de modelo, embora este venha com um analisador de modelo simples que pode ser usado opcionalmente. Na Figura 21, é apresentado um exemplo na sintaxe usada num mecanismo de modelo e o mesmo exemplo na sintaxe de PHP.

```
<ul>
{foreach from=$addressbook item="name"}
    <li>{$name}</li>
{/foreach}
</ul>
```

```
<ul>
<?php foreach ($addressbook as $name):?>
    <li>?={$name}?</li>
<?php endforeach; ?>
</ul>
```

Figura 21 – Exemplo de sintaxe num mecanismo de modelo e sintaxe de PHP

O CodeIgniter foi criado com os seguintes objetivos:

- Para instanciação dinâmica, sendo que os componentes são carregados e as rotinas executadas somente quando solicitadas, em vez de globalmente;

- Para acoplamento solto. Acoplamento é o grau em que os componentes de um sistema dependem uns dos outros. Quanto menos componentes dependem um do outro, mais reutilizável e flexível o sistema se torna;

- Singularidade de componentes, ou seja, cada classe e suas funções são altamente autónomas para permitir a máxima reutilização.

As *frameworks* anteriormente referenciadas são baseadas na arquitetura MVC, *model-view-controller*, que consiste em relacionar eficientemente a interface do utilizador com os

modelos de dados subjacentes. Este modelo propõe 3 componentes a serem usados, como se pode visualizar também na Figura 22:

Model – representa a estrutura lógica dos dados numa aplicação de *software*. Este componente não contém nenhuma informação sobre a interface do utilizador. Este gere diretamente os dados e a lógica da aplicação.

View - pode ser qualquer representação de saída de informações para a interface do utilizador, como um gráfico ou um diagrama.

Controller – representa a componente que, de certo modo, ligam o **Model** à **View**.

As vantagens do uso do Modelo MVC são:

- Processo de desenvolvimento mais rápido, permitindo que numa equipa de desenvolvimento possa ter 2 programadores, um a trabalhar na visão (*view*) e outro no controlador (*controller*), aumentando desta forma a produtividade da equipa.
- A possibilidade de criar várias visualizações para o mesmo modelo (*model*), evitando assim a duplicação de código.

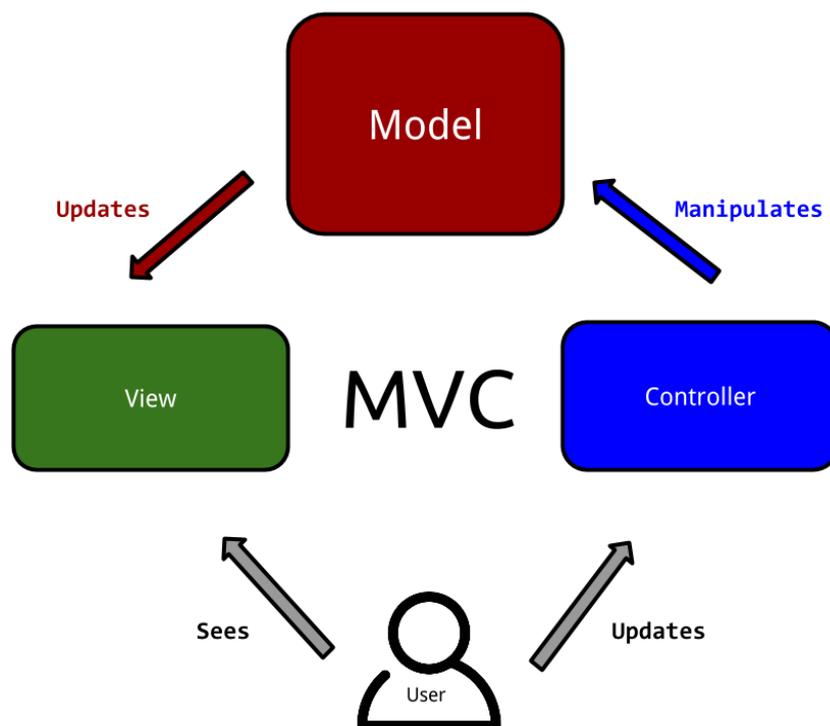


Figura 22- Exemplo da arquitetura MVC (Singh, 2015)

3 Apresentação da empresa e do grupo de trabalho

A empresa Junta – Digital Production, fundada em 2013, situa-se em Matosinhos e disponibiliza serviços de Marketing Digital e de produção de Conteúdo. A empresa dá, naturalmente primazia aos interesses do cliente e centra a gestão no modelo de negócio. A empresa, inicialmente, era apenas centrada na produção audiovisual. A partir de 2016 um novo departamento focado no desenvolvimento de aplicações web é criado, motivado, sobretudo, pela colaboração de um novo parceiro. Desde então a empresa cresceu tanto no leque de ofertas de produtos para o mercado, oferecendo serviços na criação de conteúdo digital para Web marketing e gestão social media (vídeo, fotografia, design, portfólios e catálogos online e websites), como também cresceu no número de funcionários, que já inclui, criativos e programadores para a web.

3.1 Projetos e Tarefas

Durante o estágio curricular houve colaboração em variadíssimos projetos, alguns de raiz, outros já implementados. O seguinte cronograma apresenta 7 projetos, como é possível visualizar na Figura 23 e este representa o espaço temporal de cada projeto no decorrer dos 9 meses do estágio curricular. Os projetos em questão com as tarefas impostas para cada um são os seguintes:

- HiseedTech
 - *Slider* para os sócios da empresas.
 - Criação da secção “Core Skills”.
- GoParity
 - *Slider* do *banner*.
 - Formulário de registo de utilizadores.
- Ca’dore
 - Página “Sobre Nós”.
 - Página “Contactos”.
 - Página “Faqs”.
- Norks Titanium
 - Página Principal.
 - Página da Notícia.
- Casa das Ciências
 - Página “Sugestões”.
 - Lógica da página “Sugestões”.
- Cantê
 - Lógica de criação de campanhas.
- Trotinete
 - Lógica do formulário de direito ao esquecimento.
 - Lógica do formulário de consentimento geral.

Projeto	Out	Nov	Dez	Jan	Fev	Mar	Abr	Mai	Jun
HiSeedTech		■							
GoParity			■	■					
Ca'dore				■					
Norsk Tit.				■	■				
Casa das C.					■	■			
Cantê						■	■		
Trotinete								■	■

Figura 23 – Cronograma temporal dos projetos no estágio curricular

Com a execução das tarefas foram adquiridos inúmeros conhecimentos sobre a concepção de um sítio *web*, sendo que os mais sonantes, são nomeadamente a adaptabilidade das aplicações aos diferentes formatos de ecrã, sendo que no início do estágio fora uma das dificuldades encontradas na colaboração nos primeiros projetos. Outros conhecimentos adquiridos foram a criação dos modelos de dados para a criação de um produto. Para estas tarefas as maiores dificuldades foram entender a função de um *Model*, de um *controller*, de um *component* e por fim como relacioná-los de modo a trazer os dados para a página que pertencem.

3.2 Colaboração em Projetos de Raíz

Nas próximas secções serão abordados os projetos desenvolvidos em colaboração com a equipa de desenvolvimento da empresa, especificando técnicas e tecnologias usadas para a conceção dos mesmos.

3.2.1 HiSeedTech

O HiSeedTech é uma associação sem fins lucrativos fundada por empresas privadas que se juntaram com o objetivo de suprir uma falha de mercado, resultante da escassez de iniciativas com um foco específico no apoio à translação de tecnologias desenvolvidas por investigadores para o mercado. Ao todo, a associação é constituída por 20 companhias de vários setores de atividade e que operam em diferentes locais no país, como o Banco Popular, a BlueClinical, a CIN, a Clarke, a Modet & Company, a Costa Verde, a Diergy, a Efacec, o Esporão, a everis, a Galp Energia, a Hovione Capital, a Intercapital, a Lameirinho, a Promotor, a PwC, a Renova, a SNAP!, a TauCapital e a Tecnimed.

Para este sítio *web* foram precisos implementar 5 páginas. A página principal *Home*, que contém informação sobre o projeto em si, sobre os programas, novidades, eventos e empresas associadas ao projeto (ver Figura 23). A página *At a Glance*, que apresenta um esquema do projeto, refere também as empresas associadas e fala também sobre as pessoas que integram a estrutura e a equipa deste mesmo. Página *Programs* foca-se nos projetos projetos. A página *News & Events*, contém informações e eventos, aos quais o projeto se encontra associado. Por fim, a páginas dos contactos, na qual se pode enviar uma mensagem por correio eletrónico, permite ter acesso à morada e às páginas das redes sociais.

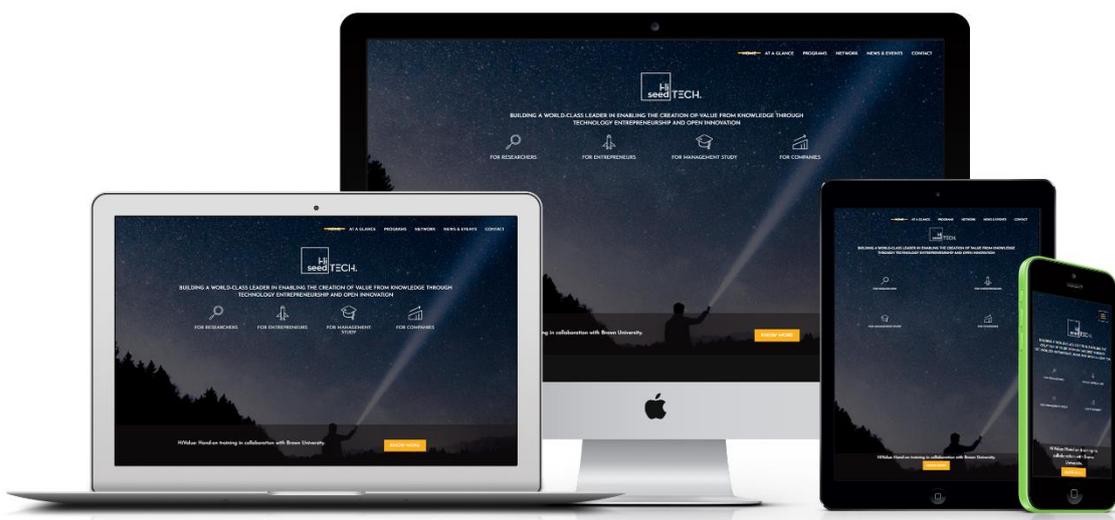


Figura 24 - Página principal do HiSeedTech em todos os dispositivos

O sítio web do HiSeedTech foi construído na plataforma October CMS com recurso ao Bootstrap ao nível de *front-end*. Os principais desafios do projeto encontravam-se na página principal, começando pelo *Header*, no qual o utilizador poderia visualizar um *slider* com notícias recentes, possibilitando aceder diretamente à notícia, a partir do botão *Know More*. Para esse *slider* usou-se o *Owl Carousel 2*, que fornece um *slider* responsivo, rápido de implementar e de uso livre. Para a secção do *Our Network*, o desafio estava em fazer um *slider* que apresentasse até um máximo de 20 sócios ou patrocinadores da empresa, e que a partir desse número possível se possibilitasse a apresentação de entidades adicionais. Contudo também era preciso fazer um filtro em tempo real que interagisse com o *slider*, reagrupando desse modo os elementos. Para este desafio usou-se o *Slick Carousel*, que permitiu que fosse possível ter um *slider* com mais do que uma coluna de altura, possibilitando a apresentação de 20 elementos.

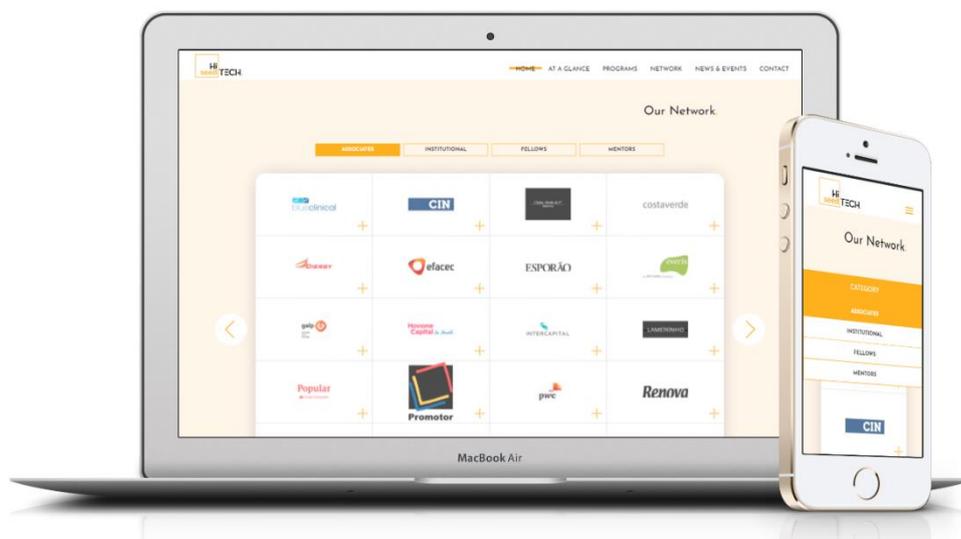


Figura 25 - Slider da secção Our Network do HiSeedTech

Existiram exceções no projeto quanto ao *responsive*, nomeadamente nas secções que houve a necessidade de criar para a mesma, 2 secções diferentes, uma para computador e outra para dispositivos móveis. Quanto à secção do *Core Skills*, até aos 1024px, organizou-se os elementos de forma a apresentar os primeiros em pares (ver Figura 25), tendo como exceção o ultimo elemento, visto que eram 5 e seria impossível dividir o espaço de forma igual, então esse ocupou o espaço na totalidade. A partir dos 1024 px esta secção desaparece, usando o *display:none*, e aparece a outra para computador na qual criou-se uma imagem com os elementos organizados da forma desejada pelo cliente. Na figura 25 pode-se verificar o resultado obtido.

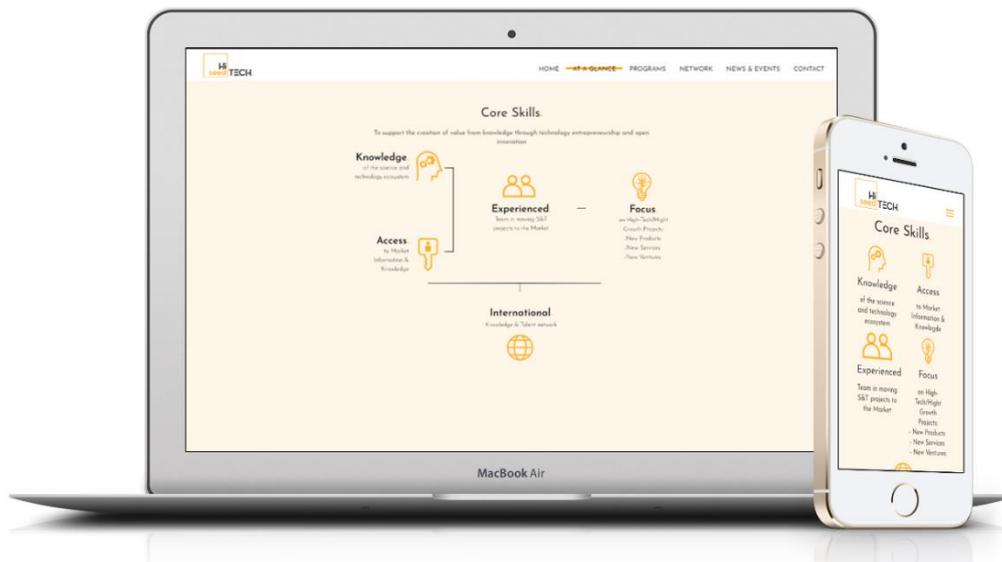


Figura 26 - Seção Core Skills do HiSeedTech, versão desktop e mobile anteriormente explicados

3.2.2 GoParity

A GoParity é a primeira plataforma portuguesa de *crowdfunding*, que oferece oportunidades de investimento sustentável. Permite que cidadãos e empresas invistam os seus fundos, na qual a plataforma garante empréstimos com garantia, rentabilidades atrativas e investimentos sustentáveis. Existem três tipos de utilizadores desta plataforma: os promotores, que são os utilizadores que vão promover os seus projetos de energia sustentável, obtendo assim um empréstimo por parte do GoParity; os investidores, que são utilizadores que investem nos projetos de modo a conseguir rentabilizar o seu investimento; e os instaladores, que são utilizadores que obtêm financiamento do GoParity para então, poder instalar produtos próprios ou em regime de revenda.

O *website* foi construído na plataforma October CMS, usando Bootstrap ao nível de *front-end*. O principal objetivo do projeto foi a reconstrução visual do *website*, implementando alterações na página principal, na página de registo de utilizador e na página de *login*. Para a página principal criou-se um slider para o header, recriou-se os cartões dos projetos, tornando o visual mais atrativo e limpo. Alterou-se a apresentação dos parceiros utilizando o modal do Bootstrap e as notícias utilizando um *slider* na qual o utilizador pode interagir. Para o registo de utilizadores – investidores, promotores e instaladores – recorreu-se a um sistema de formulários, tendo, para cada tipo, um conjunto de opções associado. Para a criação desses mesmos formulários utilizou-se o *plugin jQuery step wizard*, que tem compatibilidade com o Bootstrap. Este serve para criar formulários com etapas de registos.

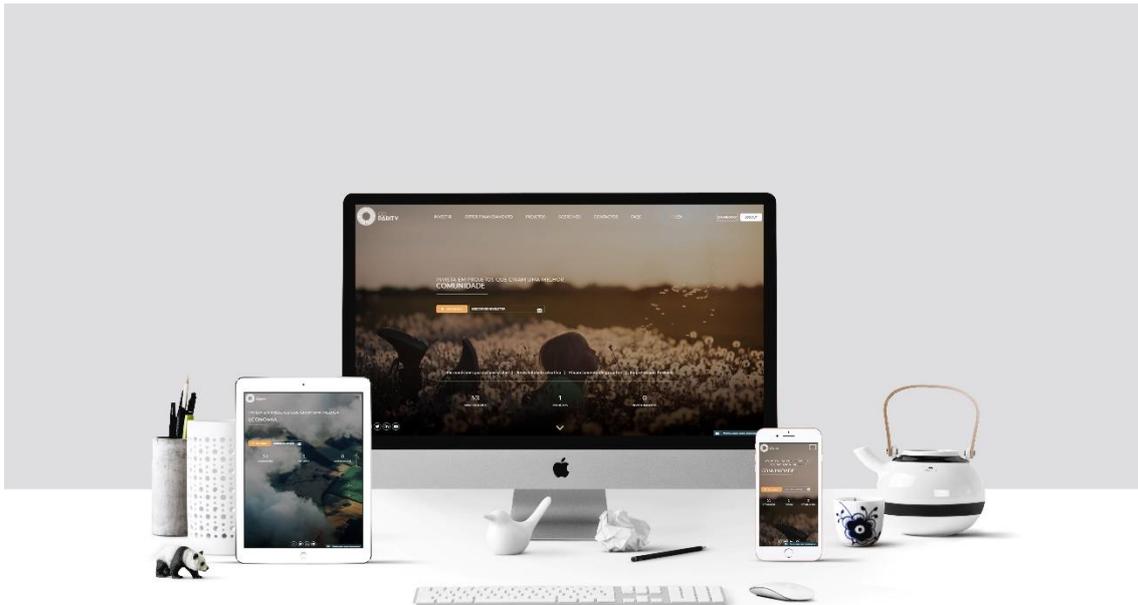


Figura 27 - Página principal do Goparity em todos os dispositivos

Para o *slider* do *banner*, da Figura 26, recorreu-se ao Owl Carousel 2, utilizando um modelo base denominado de carousel Autoplay, que possibilita a alteração do estado do slider de forma automática. Adaptando-o ao objetivo do site, este não interage com o utilizador. Os controlos de interação encontram-se embutidos no próprio slide. Quanto ao registo de utilizadores, investidores e promotores, utilizou-se o Smart Wizard, que é um *plugin* customizável e bastante flexível, usado com Bootstrap. Este permite implementar um registo, estando dividido por passos. Primeiro, foi criado um formulário de registo personalizado para cada tipo de utilizador, utilizando a mesma base de construção, visual e interatividade. Este varia mediante a escolha de cada utilizador, afetando assim o comportamento do formulário de registo. Um exemplo visual de um dos Wizards que a plataforma dispõe é o de perfil de investidor, como é possível visualizar na Figura 27. O registo é feito passo a passo, seguindo um critério de obrigatoriedade no preenchimento dos campos que só permite ao requerente (utilizador) progredir quando esse critério é satisfeito.

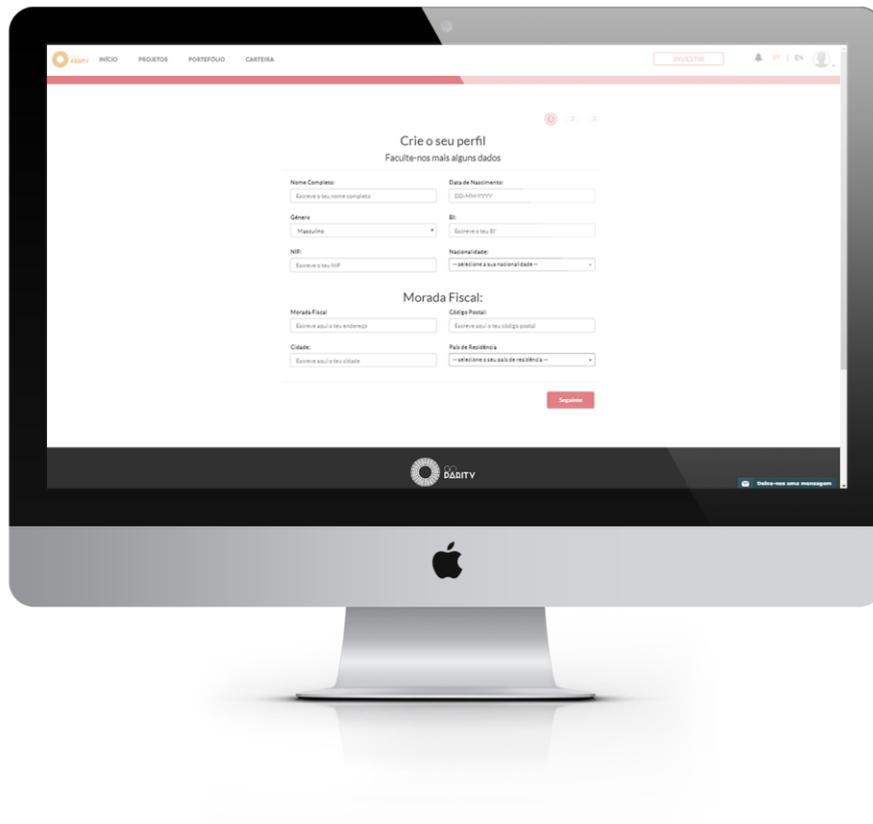


Figura 28 - Exemplo de Wizard usado no formulário de perfil

3.2.3 Ca'dore

Ca'dore é uma marca de joias criada por Patrícia d'Orey, formada em Gestão Hoteleira e com vários cursos de Pintura. Patrícia decide, enveredar pela área do design criando a marca Ca'dore, na qual tenta transmitir o seu carácter dinâmico e espírito ousado.

Para o *website* da Ca'dore foram criadas 5 páginas principais: *Loja*, na qual o utilizador tem acesso aos produtos da loja; *Sobre Nós*, que tem como objetivo informar acerca da marca e a criadora; *Press*, na qual o utilizador poder consultar revistas em que os produtos da Ca'dore tenham sido expostos. *Contactos*, na qual tem acesso à morada da loja, aos *links* para as redes sociais e onde se pode, também, entrar em contacto com a criadora enviando uma mensagem; por último a página das *Faqs*, oferece apoio para as questões mais comuns alusivas a, por exemplo, trocas, devoluções e portes de envio dos produtos. Outras páginas criadas para o projeto incluem a dos produtos, em que o utilizador pode visualizar cada produto individualmente e proceder ao processo de compra, a das compras de produtos

compra de produtos, as de registo e login, dado que um utilizador terá que estar registado e autenticado para proceder a uma compra.

Este projeto foi construído na plataforma October CMS, usando Bootstrap e Sass como linguagens de *front-end*.

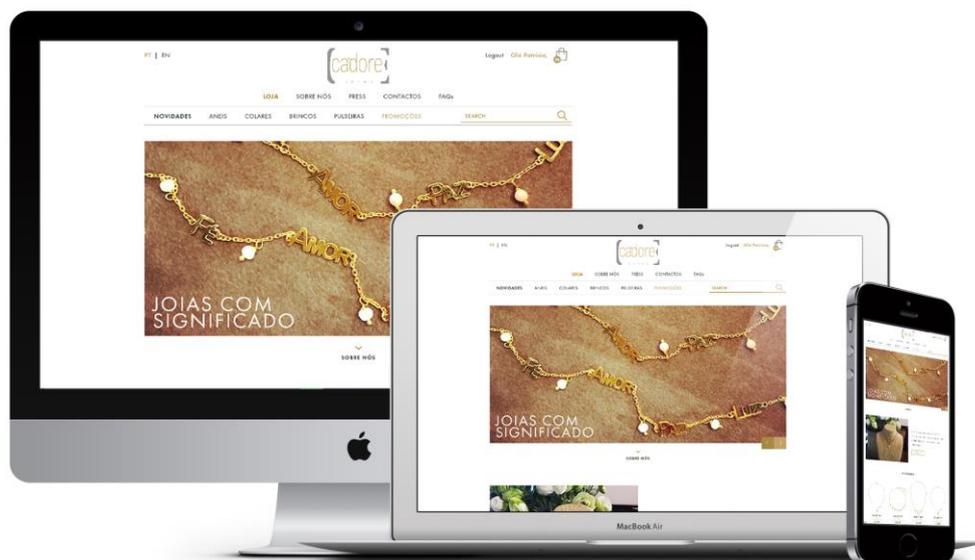


Figura 29 - Página principal Ca'dore em todos os dispositivos

Os desafios propostos incluíram a criação das páginas “Sobre Nós”, “Contactos” e “Faqs”. Para a página “Sobre Nós”, o objetivo era organizar a informação ao mesmo nível de cada imagem representante, – imagens sobre o espaço e a criadora - de modo a que a imagem ocupasse 50% do espaço e os restantes 50% fosse reservado para texto. Neste caso, um sistema de 6 grelhas para cada conjunto de imagens foi configurado, utilizando as respetivas funcionalidades do Bootstrap. Como a página iria abranger ter 3 secções diferentes, optou-se por implementar alternância de posições entre texto e imagem, ou seja, imagem à esquerda e texto à direita e, vice-versa na sequência, cujo a lógica se repetia ao longo das 3 iterações. Para o caso do dispositivo móvel escolheu-se um padrão igual para todos os elementos, as imagens e o texto teriam de ocupar 100% do espaço, pois ao lado uns dos outros não seria fácil perceber a informação. Neste caso a ordem foi sempre a mesma, a imagem viria sempre antes do texto nos 3 casos, como é apresentado na Figura 29, que demonstra a página em ecrã de computador e em dispositivo móvel.

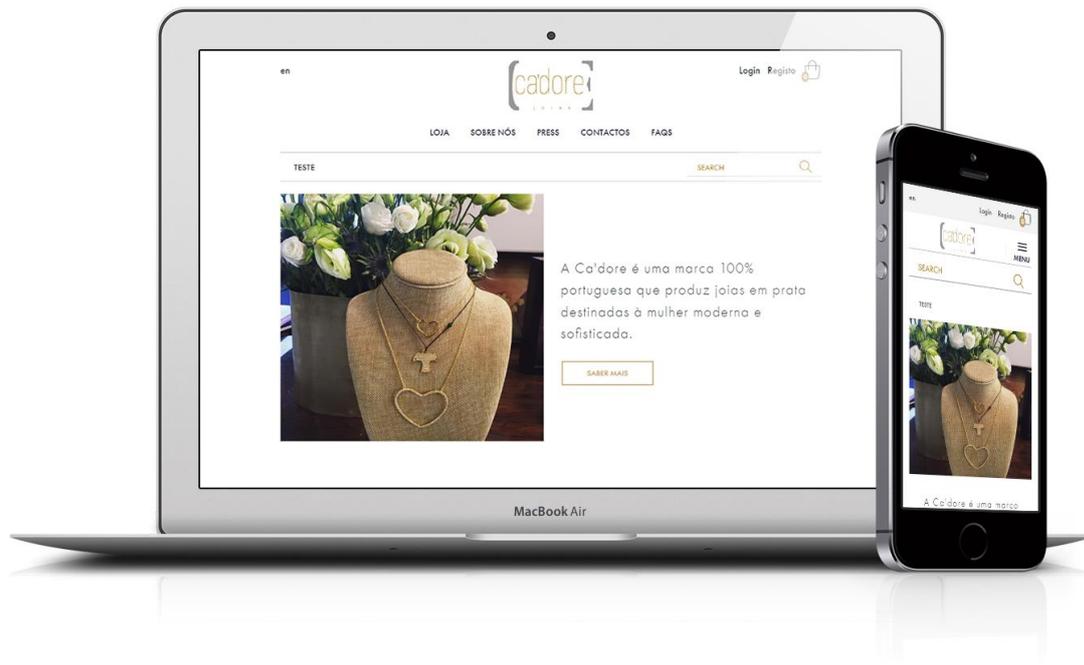


Figura 30 - Página Sobre Nós da Ca'dore

Para a página de “Contactos” era pedido que tivesse como informação a morada da loja, a imagem do mapa do sítio preciso de onde se situava a loja, a possibilidade de poder aceder às páginas das redes sociais da loja e, ainda, um formulário de envio de e-mail. Este, como se poderá verificar na Figura 30, os campos que o utilizador preencherá, à exceção da mensagem, todos ocupam metade da largura do formulário. Cada um com o respetivo *padding* atribuído de forma automática pelo Bootstrap (15px), distanciando-os para que não fiquem colados e ocupem 100% do espaço. Para o caso do dispositivo móvel, os *inputs* ocupam 100% do espaço visto que a largura é bastante pequena e seria quase ilegível ter 2 campos na mesma linha num dispositivo móvel. Sendo assim todos ocupam 100% do comprimento, neste caso, as 12 grelhas do Bootstrap. No funcionamento do formulário usou-se a biblioteca *Material Design* que foi criada para ser usada com Bootstrap. Nesta biblioteca utilizou-se os formulários com animação do *placeholder* dos *inputs*, que ao serem selecionados este sai do campo e passa para título do mesmo, dando uma melhor interatividade.

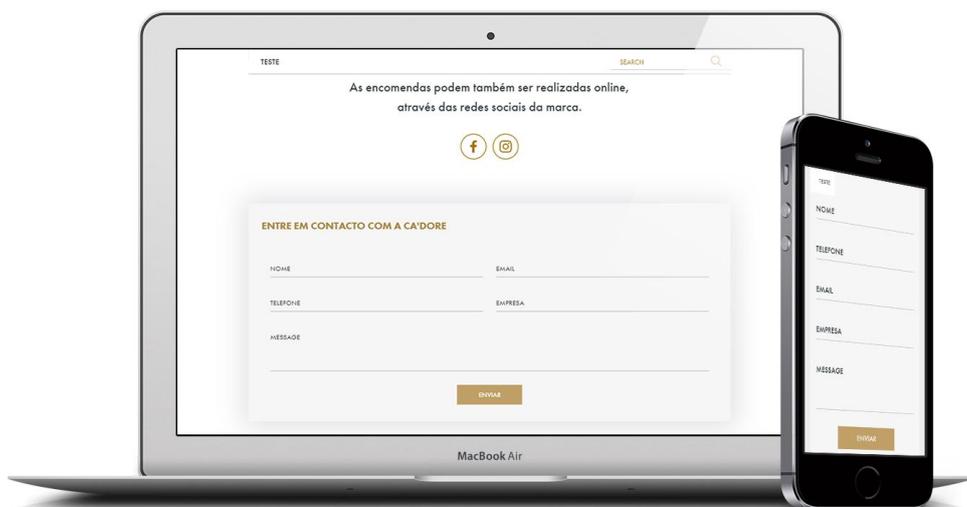


Figura 31 - Página de Contactos do Ca'dore

Por fim para a página das “Faqs” utilizou-se a função *Collapse* do Bootstrap, para organizar as perguntas e as respostas, sendo inicialmente apresentada a vista com as perguntas colapsadas, que acabam por vislumbrar as respetivas respostas, através do clique em cada tópico. Este tipo de dinâmica vista proporcionar uma melhor interação do utilizador com a plataforma. Na Figura 31 tem a demonstração visual da página com uma das perguntas já seleccionada e a respetiva resposta, funcionando de igual modo em todos os dispositivos.

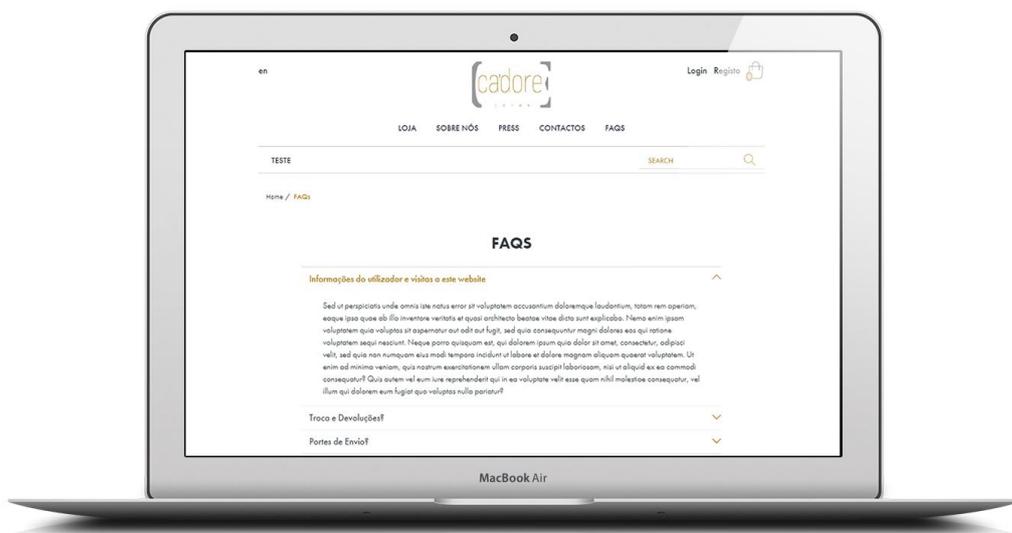


Figura 32 - Página das Faqs da Ca'dore

3.2.4 Norsk Titanium

A Norsk Titanium é uma empresa que se dedica à produção de estruturas de titânio para os mais diversos setores, incluindo o aeroespacial, da defesa, do petróleo e do gás, marítimo, entre outros. Fundou-se em 2014, na cidade norueguesa de Honefoss e possui, atualmente, escritórios em Oslo (Noruega) - repartição de vendas - e Nova Iorque (Estados Unidos da América).

Para este trabalho, o objetivo foi criar um pequeno *website* informativo sobre a empresa, com o intuito de divulgar ou apresentar artigos curtos, agrupados em 3 categorias: “Discover”, “Engage” e “Research”. Foram assim criadas 5 páginas:

- Principal, em que se apresenta 3 artigos de cada categoria;
- Discover, que tem os artigos listados desta mesma categoria;
- Engage, que tem a mesma estrutura da página Discover, mudando só a categoria dos artigos listados;
- Research, que tem a mesma estrutura das anteriores;
- Pesquisa, que tem como finalidade a pesquisa de artigos.

Este projeto foi desenvolvido na plataforma October CMS, usando Bootstrap e Sass como linguagens de *front-end*.



Figura 33 - Página principal do Norsk Titanium em 2 dispositivos

Todo o *site* foi construído no *layout* base do Bootstrap, utilizando o *container* situa a informação exatamente dentro das grelhas bases do Bootstrap sem nunca poder ocupar a totalidade do ecrã. Já o *container-fluid* permite que a informação ocupe toda a largura do ecrã. Na página principal é apresentado um *slider* informativo, feito com o *Owl Carousel*, do mesmo modo que em outros projetos anteriormente referidos. Em seguida, a página apresenta cartões informativos, cuja ação de clique reencaminha para páginas de notícias associados. Existem 3 categorias para os cartões, que são o “*Discover*”, “*Engage*” e “*Research*”, apresentados de formas distintas, os primeiros cartões são apresentados com título e descrição, enquanto os cartões de segunda fila, são apresentados com um fundo de dois tons de cinza na mesma com título e descrição; para a terceira fila de cartões a estrutura é exatamente a mesma que a primeira, apenas com a diferença que estes apresentam uma palavra-chave referente à notícia. A ordem das categorias é indiferente, pois os cartões adaptam-se mediante a fila que irão ocupar, apresentando sempre um máximo de 3 cartões na primeira e terceira fila e 2 cartões na segunda fila. Como é possível verificar na Figura 33, a categoria que é apresentada por uma faixa de cor diferente dos fundos dos cartões, alinhada à esquerda, refere à que categoria os cartões pertencem.

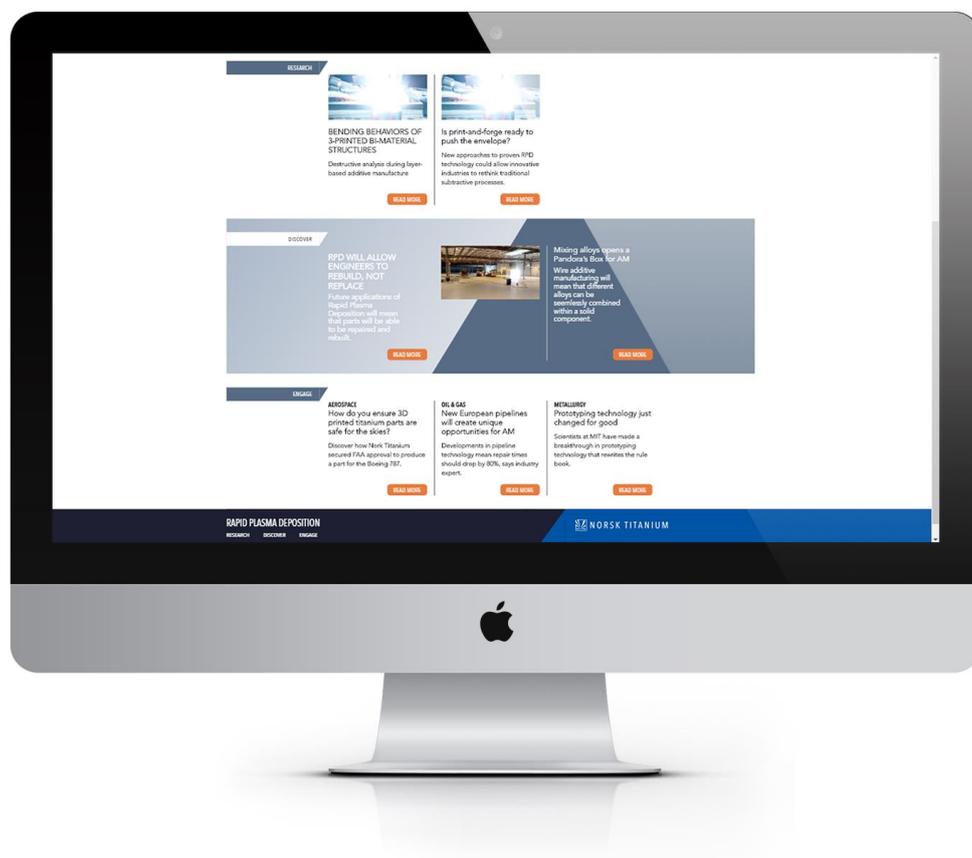


Figura 34 - Cartões das categorias Discover, Engage e Research

Para além da página principal, existem 3 páginas referentes a que cada categoria no qual as notícias são apresentadas da mesma forma que na principal, mas os cartões são todos iguais, apresentando imagem, palavra-chave, título e descrição. Existe também uma página de pesquisa, à qual se tem acesso através da *navbar*, que apresenta cartões de notícias mediante a solicitação do utilizador, organizados da mesma forma nas páginas por categorias. Por fim, a página referente à notícia escolhida, apresenta o texto sobre a notícia e cartões de notícias da mesma categoria, possibilitando o utilizador de poder entrar noutras notícias. Para plataformas móveis a informação é organizada de forma diferente, visto que o tamanho do ecrã difere bastante. Em primeiro lugar é apresentada a notícia com o texto e em seguida os cartões de outras notícias. Todos estes cartões, seja em *desktop* ou em *mobile*, não apresentam imagem, apenas o título e o excerto, como é possível visualizar na Figura 34 que demonstra as duas versões anteriormente descritas.

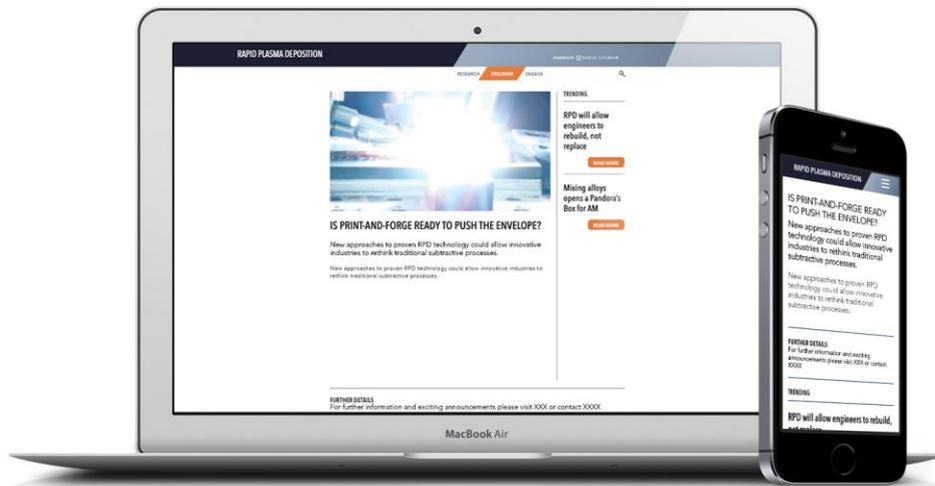


Figura 35 - Página da notícia em 2 dispositivos

3.3 Colaboração noutros Projetos já Implementados

3.3.1 Casa das Ciências - Criação da página de Sugestões

A Casa das Ciências é um portal de base colaborativa que recolhe, valida e divulga materiais digitais para servir os professores de ciências dos diferentes níveis de ensino. Esta plataforma foi criada para divulgar o trabalho de todos os professores e investigadores que tenham desenvolvido soluções digitais com o propósito do ensino na área de ciências, sendo todo o material disponibilizado, de acesso direto e sem qualquer tipo de custo para o utilizador. A plataforma está dividida em quatro partes, a primeira tem os recursos educativos digitais das disciplinas de Biologia, Geologia, Física, Química, Matemática e Introdução às ciências, disponibilizando *powerpoints*, vídeos e animações, simulações interativas e documentos. A segunda parte refere-se à WikiCiências, que é a Enciclopédia digital com conceitos de ciência. A terceira parte é o Banco de Imagens, que é uma galeria de imagem de interesse científico e didático. Por fim, a última parte que é a Revista de Ciência Elementar com artigos de opinião científica e didática, entrevistas, biografias de cientistas e sugestões de recursos digitais e imagens. Esta plataforma foi criada tendo como base de construção o October CMS. Para o desenvolvimento de *front-end* foi utilizado a framework Bootstrap. Na Figura 35, é apresentada a plataforma nos diferentes dispositivos.

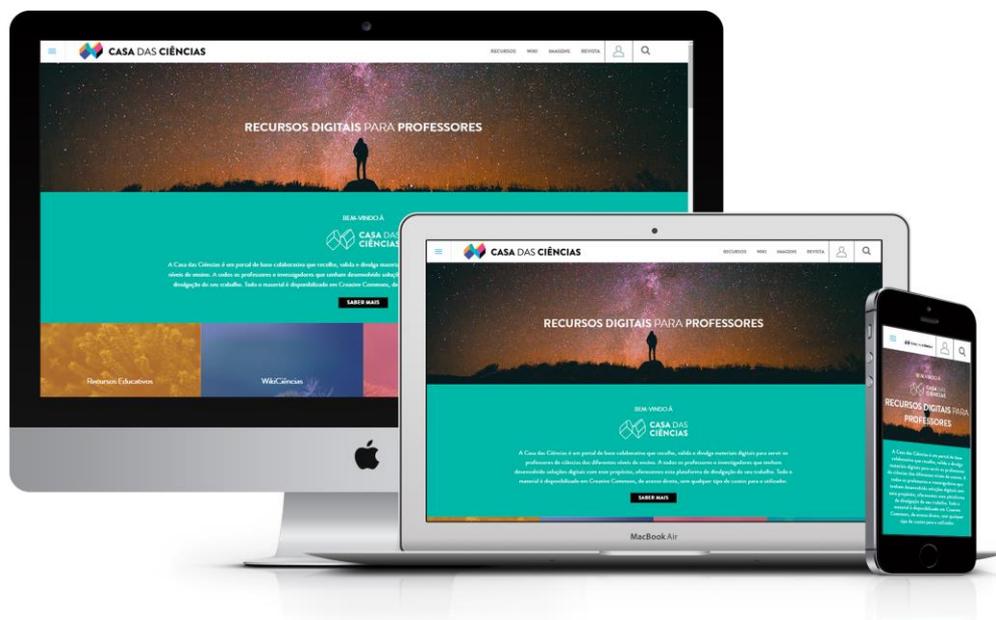


Figura 36 - Página principal Casa das Ciências em 3 dispositivos

O desafio proposto na criação da página das sugestões passou por equipar o gestor da plataforma - que é o criador do conteúdo que é apresentado na plataforma - com ferramentas que lhe permitisse definir os recursos que eram sugeridos, mediante a categoria, ano e ciclo, a partir de um painel de controlo. Nessa página os recursos teriam de ser apresentados mediante a categoria, em primeiro lugar, para depois serem mostrados por grupos organizados em ano e ciclo. Para tal, criou-se os componentes que retornam os recursos mediante a categoria - verificando, neste caso, se são recursos de sugestão -, bem como ano e ciclo. Reutilizando-se o Model de criação dos recursos, acrescenta-se simplesmente os campos de sugestão e de ciclo para fiquem, assim definidos. Em seguida, acendendo à página de sugestões, realiza-se a chamada ao componente que irá desempenhar a sua função previamente definidas. Na figura 36 pode-se verificar que a página tem um sistema de filtragem por categorias, que são apresentadas de forma dinâmica, ou seja, o gestor no painel de controlo pode criar e remover as categorias mediante a necessidade de forma responsiva, pois o componente também já integra essa função, de listar as categorias para a filtragem. Escolhendo umas das categorias é chamado um método único que serve de igual modo para todas as outras: simplesmente recebe a informação de qual categoria, realiza consultas aos recursos e, por fim executa um método designada de *Render Partial*, uma vez que a página está dividida em várias partes, chamadas de *partials* (um *partial* com a barra de navegação, outro com o sistema de filtragem e os recursos e por fim outro para o *footer*). Esta função recarrega o *partial* definido no componente, de modo a que a página tenha que recarregar na totalidade, mas só a parte dos recursos da categoria escolhida. Por fim, os anos e os ciclos por baixo das categorias são hiperligações que levam o utilizador diretamente para as respetivas secções, de modo a evitar a obrigatoriedade do deslize na página para chegar ao ano/ciclo. Em função deste aspeto, procurou-se a melhoria da interatividade.

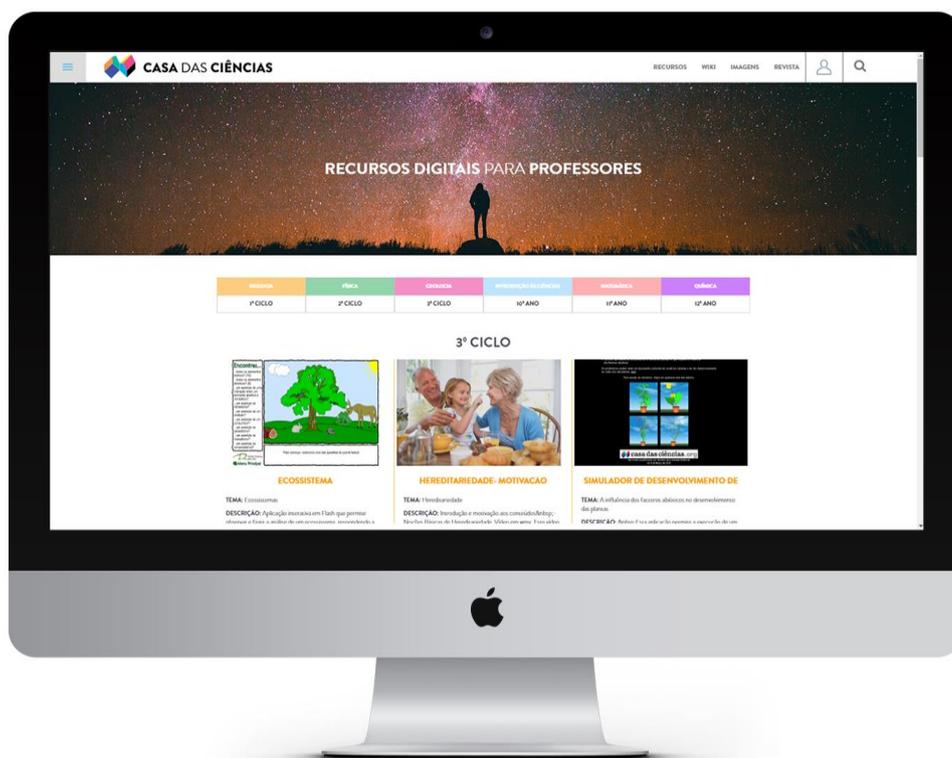


Figura 37 - Página de Sugestões da Casa das Ciências

3.3.2 Campanha - Cantê

A marca Cantê foi fundada em 2015 a partir de duas amigas que tinham o objetivo comum de ter uma loja e atelier de uma marca de praia. Com o crescimento substancial das vendas e da procura dos produtos, tiveram que aumentar o espaço, tendo também que criar uma loja *online*, de modo a poder apresentar todos os produtos produzidos e alcançar um público-alvo ainda maior. A loja *online* é dividida em várias secções, a secção My Cantê que apresenta os biquínis de praia, My First Cantê que apresenta roupa para crianças e por fim My Intimate Cantê que tem produtos de roupas interior feminina. Existem outras secções como as “novidades”, na qual são apresentados os novos produtos. Há, ainda, a possibilidade de aceder a outro sítio *web* de venda de roupa de verão, mas para homem, neste caso a Molkot. O sítio *web* da Cantê tem como base de construção o October CMS. Para o front-end foi utilizado a framework Bootstrap. Como se pode verificar na Figura 37, o sítio *web* é completamente a responsivo a qualquer dispositivo.

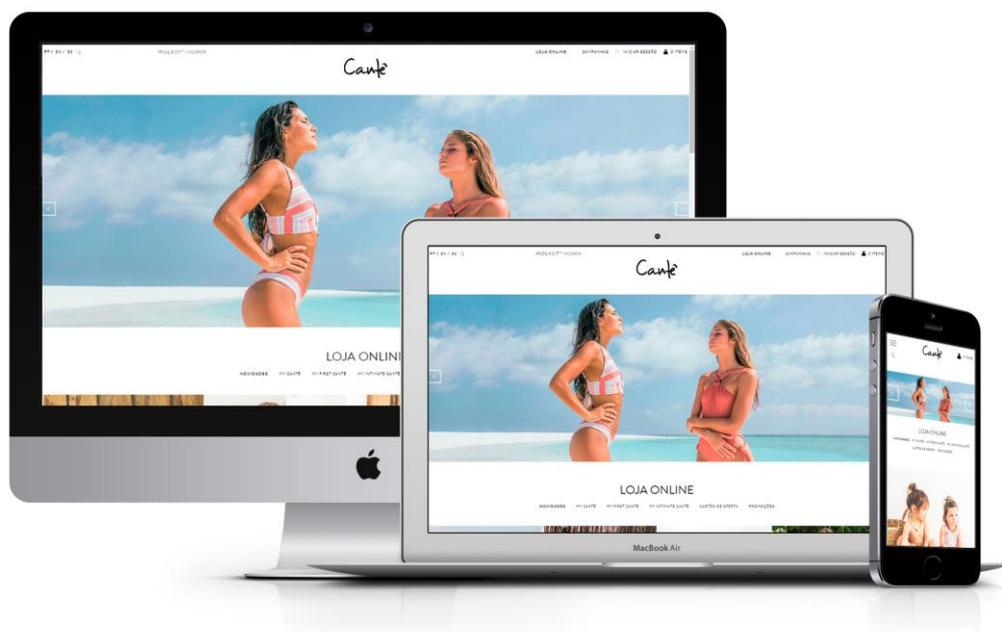


Figura 38 - Página Principál Cantê em 3 dispositivos

O trabalho realizado responde ao desafio da criação de mecanismos adaptados ao modelo de negócio da Cantê, de modo a que a pessoa responsável pela gestão do sítio *web* possa criar as campanhas de forma dinâmica a partir do painel de controlo. Para tal, criou-se um componente que trata de toda a lógica, sendo esse componente chamado a partir da página em questão. Definindo uma categoria, que passada através do *URL* que será comparado ao *slug* das categorias. Assim o componente terá um ponto de referência na procura, o objetivo cujo categoria seja igual ao valor trazido pelo *slug*. A partir do *Fields.yaml* cria-se toda a usabilidade que o utilizador poderá ter a partir do painel de controlo e fazendo assim a ligação com a base de dados. No final o resultado visual da criação do *Model* é gerido a partir do controlador que é quem faz toda a ligação. O resultado daquilo que o utilizador encontrará no painel de controlo para poder criar as suas campanhas de forma dinâmica pode ser visto na Figura 38.

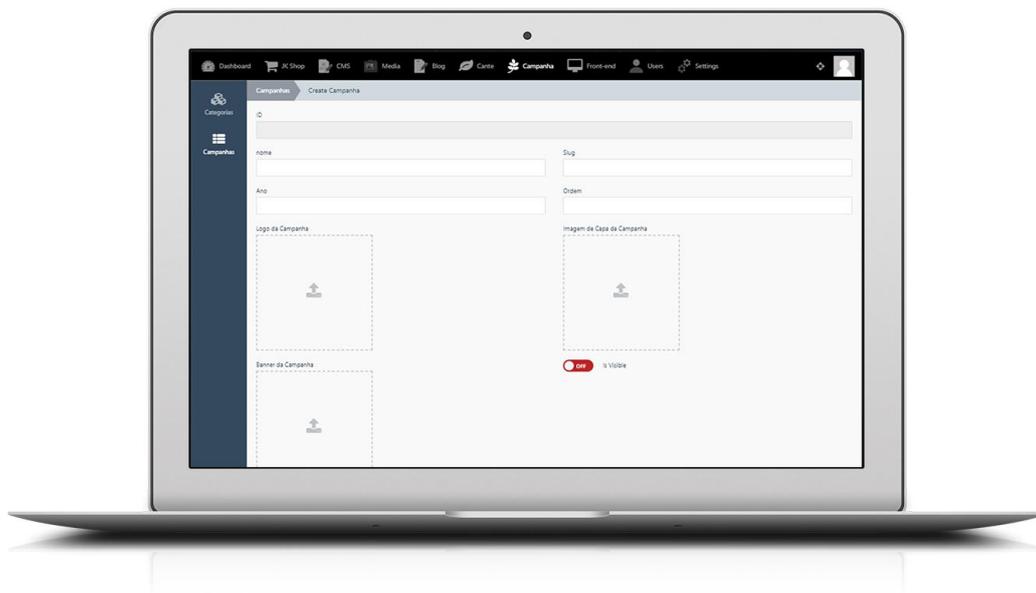


Figura 39 - Backend de criação das Campanha Cantê

Após a criação da campanha, esta pode ser visualizada numa das três opções de categoria My Cantê, My Intimate Cantê ou My First Cantê. Uma vez dentro de uma dessas páginas poderá visualizar as campanhas em vigor, anteriormente criadas a partir do painel de controlo. A Figura 39 apresenta a página de uma destas categorias que, neste caso, corresponde às campanhas My Cantê.



Figura 40 - Página das Campanhas Cantê

Dentro da devida secção do sítio web, o utilizador poderá visualizar os produtos que a nova campanha apresenta, dispostas por imagens anteriormente e parametrizadas pelo gestor, a partir do painel de controlo, como é possível verificar na Figura 40.

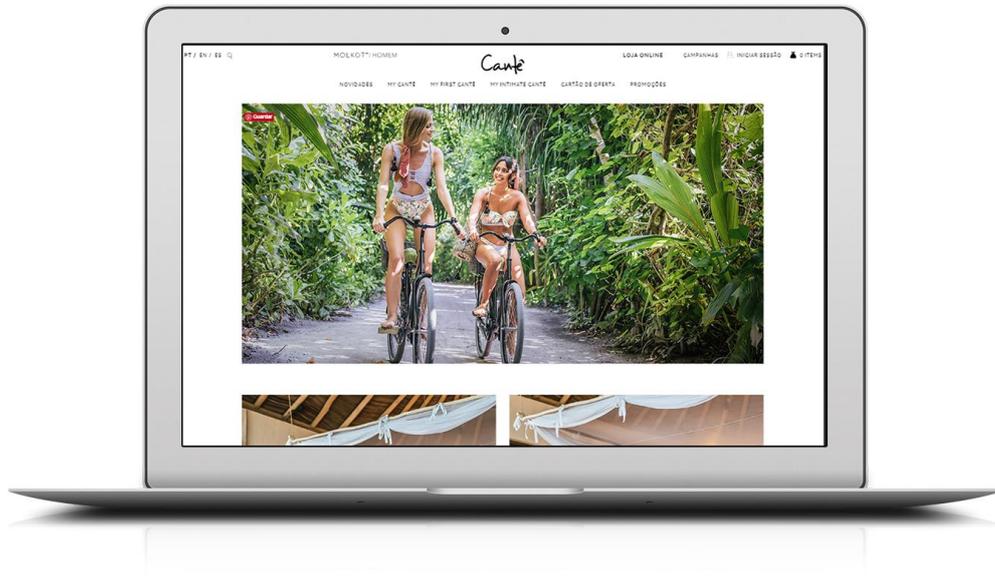


Figura 41 - Página de Campanha Cantê

3.3.3 Direito ao Esquecimento. Consentimento Geral - Trotinete

A Trotinete nasceu há 25 anos como um projeto da fundadora Matilde Vasconcelos, Designer de formação. A plataforma surge com o objetivo de poder expandir o negócio, podendo também apresentar o catálogo dos produtos e possibilitar a venda dos mesmos *online*. Na Figura 41 poderá ser visualizada a página principal, que tem como navegação a “Marca” que fala e explica a própria marca, os “Artigos” ao qual poderá pesquisar todos os artigos presentes na plataforma, a “Coleção” para poder procurar produtos de uma determinada categoria (Colégio, Casual, Desporto, Natação ou Infantilário), a página de orçamento que permite escolher produtos e ter uma noção do valor final. A plataforma foi desenvolvida, como base de construção o October CMS e para o *front-end* a *framework* Bootstrap.



Figura 42 - Página Principal da Trotinete em todos os dispositivos

Foi solicitada a criação de uma página para o direito ao esquecimento na qual o utilizador, preenchendo o formulário, pede para que os dados pessoais sejam removidos da base de dados da plataforma. Foi então criado um componente que recebe os dados enviados a partir do formulário e adiciona esse utilizador a uma tabela que o gestor da plataforma terá acesso a partir do painel de controlo, através do modelo criado para receber os dados daquela tabela. Na seguinte Figura 42, pode-se visualizar o formulário de direito ao esquecimento.



Figura 43 - Página de Direito ao Esquecimento

Quanto à página de “Consentimento Geral”, o conceito é o mesmo que o da página de “Direito ao Esquecimento”, mas com objetivos opostos. Para poder continuar a efetuar encomendas, receber produtos, serviços, campanhas, eventos e informações da plataforma, o utilizador terá de preencher o formulário inerente ao consentimento. Uma vez preenchido, este utilizador será atribuído a uma tabela que o gestor da plataforma poderá visualizar a partir do painel de controlo. Na figura 43 pode-se visualizar o formulário do Consentimento Geral.

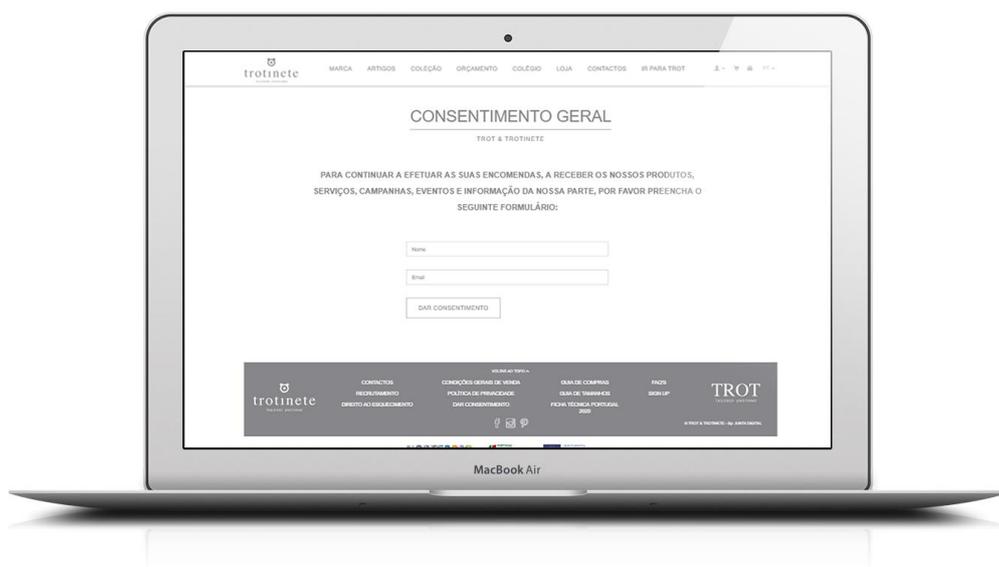


Figura 44 - Página de Consentimento Geral

4 Conclusão e trabalho futuro

No início do documento abordou-se a importância do uso de metodologias Ágil no desenvolvimento de projetos multimídia. O recurso a estas metodologias permite abordar o surgimento de problemas no desenvolvimento dos projetos, esta proporciona o trabalho de equipa e um maior controlo na distribuição das tarefas aos membros da equipa.

Para integrar os trabalhos junto da equipa da Junta – Digital Production foi necessário estudar os conceitos de *design* responsivo, juntando também a esse estudo, as frameworks de desenvolvimento de *front-end*. Aprofundou-se o estudo da Framework Bootstrap e do Pré-processador Sass para a implementação dos *layouts*. Foi necessário também um estudo sobre o October CMS, usado nos projetos da Junta – Digital Production. Graças a esse estudo, foram cumpridos os objetivos iniciais do estágio, em termos de integração e ambientação com as ferramentas de trabalho, permitindo assim a integração na equipa de trabalho e realização dos vários projetos para a *web*.

Até a integração com a equipa nos projetos de desenvolvimento, foram precisos 2 meses de intrusão com as ferramentas, principalmente com o uso do Bootstrap. Os projetos HiSeedTech, GoParity, Ca'dore e Norks Titanium permitiram, sem dúvida, uma grande evolução e capacidade de desenvolvimento de *front-end*. Em seguida, tendo os conhecimentos mais cimentados quanto ao uso das ferramentas de desenvolvimento de *front-end*, foram introduzidas as ferramentas de desenvolvimento de *back-end*, de forma evolutiva e gradual, aumentando o nível de dificuldade das tarefas impostas. Os projetos Casa das Ciências, Cantê e Trotinete permitiram-me ultrapassar barreiras importantes na área do desenvolvimento em *back-end*. Atualmente PHP e Laravel são linguagens de programação usadas com bastante mais frequência e sempre de forma evolutiva, sendo o próximo desafio sempre maior do que o ultrapassado.

Em termos pessoais foram ultrapassadas todas as expectativas, o que leva a crer que foi atingido o nível de amadurecimento necessário para a produção de soluções para o mercado real, em contexto de ambiente empresarial. No sentido de continuar a explorar o desenvolvimento das capacidades, tanto a nível de *front-end* como de *back-end*, reconhece-se como necessário estudar frameworks adicionais, nomeadamente de JavaScript(Tais como Vue.js, framework compatível com Otctober CMS, Angular JS e React JS, sendo estas bastante procuradas no mercado de trabalho).

Referências Bibliográficas

Arthur, A.P., da Silva (2014). Design Responsivo: Técnicas, Frameworks e Ferramentas. Obtido de: <http://bsi.uniriotec.br/tcc/textos/201412Almeida.pdf>. Consultado a 20 de Maio de 2018.

What is Git. (2018). Obtido de: <https://www.atlassian.com/git/tutorials/what-is-git>. Consultado a 19 de Setembro de 2018.

Bilal Cinarli (2014). Na Introduction to CSS Pre-Processors: SASS, LESS and Stylus. Obtido de: <https://htmlmag.com/article/an-introduction-to-css-preprocessors-sass-less-stylus>. Consultado a 20 de Junho de 2018.

Bootstrap Documentation (2018). Grid System. Obtido de: <https://getbootstrap.com/docs/4.0/about/overview/>. Consultado a 22 de Novembro de 2018.

Bootstrap vs. Foundation vs. Pure. (2018). Obtido de: <https://stackshare.io/stackups/bootstrap-vs-foundation-vs-pure>. Consultado a 7 de Novembro de 2018.

Six Benefits of Using MVC Model for Effective Web Application Development. (2018). Obtido de: <https://www.brainvire.com/six-benefits-of-using-mvc-model-for-effective-web-application-development/>. Consultado a 6 de Setembro de 2018.

Burk, J. (2017). Obtido de: <https://blog.exsilio.com/wp-content/uploads/2017/09/scrum.png>. Consultado a 2 de Julho de 2018.

CodeIgniter at a Glance. (2018). Obtido de: https://www.codeigniter.com/user_guide/overview/index.html. Consultado a 16 de Setembro de 2018.

What is a Content Management System (CMS)? (2018). Obtido de: <https://www.comentum.com/what-is-cms-content-management-system.html>. Consultado a 21 de Setembro de 2018.

Daniel Viana (2017). O que é front-end e back-end? Obtido de: <https://www.treinaweb.com.br/blog/o-que-e-front-end-e-back-end/>. Consultado a 23 de Setembro de 2018.

SCRUM. (2014). Obtido de: <https://www.desenvolvimentoagil.com.br/scrum/>. Consultado a 12 de Junho de 2018.

Eva Hard, Paul Kapellari, Steven Luong, Norbert Spot (2011). Responsive Web Design. Obtido de: <https://courses.isds.tugraz.at/iaweb/surveys/ws2011/g3-survey-resp-web-design.pdf>. Consultado a 21 de Maio de 2018.

O que você precisa saber sobre metodologias ágeis de desenvolvimento. (2017). Obtido de: <https://gaea.com.br/o-que-voce-precisa-saber-sobre-metodologias-ageis-de-desenvolvimento/>. Consultado a 12 de Maio de 2018.

Sobre Controlo de Versão. (2018). Obtido de: <https://git-scm.com/book/pt-br/v2/Come%C3%A7ando-Sobre-Controle-de-Vers%C3%A3o>. Consultado a 23 de Setembro de 2018.

Jamie Donnelly (2015). 5 reasons to use SourceTree for Git. Obtido de: <https://sagittarius.agency/blog/5-reasons-to-use-sourcetree-for-git>. Consultado a 22 de Maio de 2018.

João Roberto (2017). O que é Laravel? Porquê usá-lo? Obtido de: <https://medium.com/joaorobertopb/o-que-%C3%A9-laravel-porque-us%C3%A1-lo-955c95d2453d>. Consultado a 8 de Setembro de 2018.

Laurence Bradford (2017). Learn About Front-end Framework in Web Development. Obtido de: <https://www.thebalancecareers.com/what-is-a-front-end-framework-and-why-use-one-2071948>. Consultado a 5 de Maio de 2018.

October CMS Vs Wordpress. (2017). Obtido de: <https://leaderinternet.com/blog/october-cms-vs-wordpress>. Consultado a 6 de Abril de 2018.

Lei, H., Ganjezadeh, F., Jayachandran, P. K., & Ozcanm, P. (2017). A statistical analysis of the effects of Scrum and Kanban on software development projects. *Robotics and Computer-Integrated Manufacturing*, 43. 59-67. Obtido de: <https://doi.org/10.1016/j.rcim.2015.12.001>. Consultado a 5 de Maio de 2018.

Nick Salloum (2014). Introducing OctoberCMS – a Laravel – based CMS. Obtido de: <https://www.sitepoint.com/introducing-octobercms-laravel-based-cms/>. Consultado a 4 de Agosto 2018.

Introduction. (2018). Obtido de: <https://octobercms.com/docs/cms/themes#introduction>

October CMS Vs Wordpress. (17 de Abril de 2017). Obtido de: <https://leaderinternet.com/blog/october-cms-vs-wordpress>. Consultado a 9 de Agosto 2018.

Sass Basics. (2018). Obtido de: <http://sass-lang.com/guide> Consultado a 20 de Agosto 2018

Schwaber, K., & Sutherland, J. (n.d.). The Definitive Guide to Scrum: The Rules of the Game. Obtido de: <https://www.scrum.org/resources/scrum-guide>. Consultado a 10 de Agosto 2018.

Singh, A. (2015). Obtido de: <http://blogs.quovantis.com/wp-content/uploads/2015/03/MVC1.png>. Consultado a 11 de Maio 2018.

Simplicity and power in a beautiful Git GUI. (2018). Obtido de: <https://www.sourcetreeapp.com/>. Consultado a 3 de Agosto 2018.

Content Management System (CMS). (2018). Obtido de:

<http://searchcontentmanagement.techtarget.com/definition/content-management-system-CMS>. Consultado a 18 de Agosto 2018.

Model-View-Controller (MVC). (2018). Obtido de:

<https://whatis.techtarget.com/definition/model-view-controller-MVC>. Consultado a 12 de Julho 2018.

Bootstrap Grid System. (2018). Obtido de: <https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/bootstrap-grid-system.php>. Consultado a 15 de Agosto 2018.

Wendell Adriel (2016). Sass vs. LESS vs. Stylus: Batalha dos Pré-processadores. Obtido de: <https://tableless.com.br/sass-vs-less-vs-stylus-batalha-dos-pre-processadores/>. Consultado a 14 de Abril 2018.