Universidade do Minho
Escola de Engenharia

Tiago Martins Teles

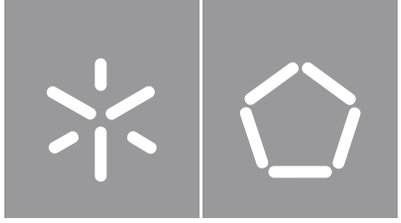
Geração de Lojas Online Orientada Para Uma Framework – Funcionalidade de Gestão de Utilizadores

janeiro de 2023

Geração de Lojas Online Orientada Para Uma Framework – Funcionalidade de Gestão de Utilizadores

Tiago Martins Teles

UMinho | 2023



Universidade do Minho
Escola de Engenharia

Tiago Martins Teles

Geração de Lojas Online Orientada Para Uma Framework – Funcionalidade de Gestão de Utilizadores

Dissertação de Mestrado
Mestrado Integrado em Engenharia de Telecomunicações e Informática

Trabalho efetuado sob a orientação do
Professor Doutor José Manuel Tavares Vieira Cabral e Professor Doutor Sérgio Adriano Fernandes Lopes

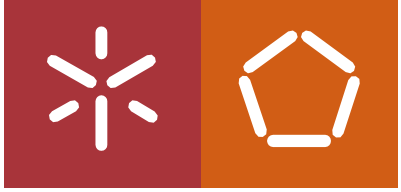
janeiro de 2023

Universidade do Minho
Escola de Engenharia

Tiago Martins Teles

Geração de Lojas Online Orientada Para Uma Framework – Funcionalidade de Gestão de Utilizadores

Janeiro de 2023



Universidade do Minho
Escola de Engenharia

Tiago Martins Teles

Geração de Lojas Online Orientada Para Uma Framework – Funcionalidade de Gestão de Utilizadores

Dissertação de Mestrado
Mestrado Integrado em Engenharia de Telecomunicações e Informática

Trabalho efetuado sob a orientação do
Professor Doutor Sérgio Lopes
Professor Doutor José Cabral

AGRADECIMENTOS

Numa jornada da minha vida, que me trouxe alegrias e tristezas, recompensas e sacrifícios, conhecimento e experiências únicas, chego agora à reta final e com o coração quente relembro quem me apoiou e fez crescer nestes últimos anos.

Em primeiro lugar, os meus pais, que me deram a oportunidade de obter todas as ferramentas necessárias para o meu sucesso académico e que sempre demonstraram um apoio incondicional ao longo deste percurso.

À minha namorada, o meu sincero obrigado, pelo teu apoio e amizade, pelos discursos de motivação, pela paciência sem fim, por me segurares nos piores momentos e festejares comigo as minhas conquistas. Não há palavras que consigam descrever o quanto tu me ajudas a ser uma pessoa melhor. Obrigado por seres assim.

Aos meus amigos próximos, que fizeram estes anos de universidade inesquecíveis. A todos um obrigado.

Agradeço aos orientadores Sérgio Lopes e José Cabral, por terem me aceite como orientando, pela disponibilidade de tempo e por todas sugestões que elevaram o meu trabalho.

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Geração de Lojas Online Orientada Para Uma Framework – Funcionalidade de Gestão de Utilizadores

RESUMO: O desenvolvimento de todo o código necessário para uma loja *online* é um trabalho demorado e complexo. Todas as lojas *online* têm diferentes exigências e condições, no entanto, existem pontos comuns entre vários tipos de loja. Supondo a possibilidade de extrair esses aspetos comuns, seria possível a criação de um esqueleto de código com todos os componentes básicos, sobre os quais um programador poderia acrescentar e moldar a loja *online*, reduzindo assim o tempo necessário para a desenvolver.

A elaboração desta dissertação começou pela criação de uma loja *online* que suportasse um negócio fictício de venda de bicicletas e produtos relacionados. Este passo serviu para identificar e avaliar quais seriam os parâmetros necessários especificar, e em que sentido fariam diferença na construção de um website de vendas bem como, distinguir os componentes comuns e as suas relações.

A presente dissertação tem como propósito principal o desenvolvimento de uma ferramenta para a criação de lojas *online*. Esta ferramenta pretende ser usada por programadores para produzir o código padrão que é comum a qualquer implementação de uma solução de loja *online*. Através da recolha de parâmetros sobre a loja *online* a ser criada, são construídos os ficheiros necessários para a implementação do site.

A aplicação concebida focou-se na utilização de apenas alguns dos componentes básicos, nomeadamente, “Utilizadores” e “Categorias” no *back-end*, “Registo e Login” e “Perfil” no *front-end*, mostrando assim ser possível usar a *framework* desenvolvida para uma implementação parcial de uma loja *online*. A abordagem modular permite expandir a aplicação, criando e adicionando outros componentes à estrutura parcial existente.

PALAVRAS-CHAVE: Comércio Eletrónico, Plataforma E-Commerce, Loja Online

Framework Oriented Online Store Generation – User Management Functionality

ABSTRACT: Developing all the code necessary for an online store is a complex and time-consuming task. Every online store has different demands and conditions, however, there are common aspects to various types of stores. Assuming the possibility of extracting those common aspects, it would be possible to create a code skeleton with all the basic components, over which a programmer could add or mold an online store, reducing the development time.

The elaboration of this dissertation began with the creation of an online store that would support a fictitious business selling bicycles and related products. This step served to identify and evaluate which parameters needed to be specified, and in what sense they would make a difference in the construction of a sales website, as well as to distinguish common components and their relationships.

The main purpose of this dissertation is to develop a tool for creating online stores. This tool is intended to be used by programmers to produce the “boilerplate” code that is common to any implementation of an online store solution. By collecting parameters about the online store to be created, the necessary files for the “deploying” of the site are generated.

The conceived application focused on the use of only some of the basic components, namely, “Users” and “Categories” in the back-end, “Registration and Login” and “Profile” in the front-end, thus proving that a partial implementation of an online store is possible. The modular approach allows expanding the application by creating and adding other components to the existing partial structure.

KEYWORDS: E-Commerce, E-Commerce Platform, Online Store

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos e Siglas

API	Application Programming Interface
CMS	Content Management System
CSS	Cascading Style Sheets
EDI	Electronic Date Interchange
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JWT	JSON Web Token
MERN	MongoDB, ExpressJS, React e NodeJS
SaaS	Software as a Service
SEO	Search Engine Optimization
URI	Uniform Resource Identifier
URL	Unified Resource Locator

1. INTRODUÇÃO

Serve como objetivo deste presente capítulo apresentar o enquadramento e relevância do tema escolhido para esta dissertação, os objetivos que levaram à realização deste trabalho, bem como a estrutura do mesmo.

1.1. Enquadramento e Relevância do Tema

No mundo atual do comércio é cada vez mais comum existirem empresas a operar em simultâneo em lojas físicas e *online*, ou somente com base em lojas *online (e-commerce).* De acordo com Kalia, Kaur & Singh (2017:05), "*se organizações ou negócios não têm um website, é como se elas não existissem. Com mais pessoas a usar motores de busca para encontrar informação agora, mais do que nunca, é critico que organizações tenham uma presença forte na web*".

São inúmeras as vantagens que advêm da adoção de uma loja *online*, quer seja para pequenas empresas ou para grandes negócios, com o acesso a uma maior base de clientes, melhor gestão de inventários e maior exposição dos produtos. Na perspetiva do cliente, este pode ter acesso a uma maior oferta e variedade de produtos, uma fácil comparação entre eles, bem como usufruir da possibilidade de compra em qualquer instante.

Seja para comprar, procurar ou comparar, uma boa loja *online* tem de conseguir suportar um leque de funcionalidades para que a experiência, quer do cliente quer do vendedor, seja o mais apelativa possível, permitindo prolongar a “vida útil” da loja. Com esta premissa em mente, o desenvolvimento de referida loja seria bastante focado na personalização da mesma.

Um dos problemas mais presentes na criação de uma loja *online* consiste na falta de eficiência com que estas lojas são desenvolvidas, pois o desenvolvimento de raiz das mesmas é um processo complexo e demorado, consumindo muito tempo e recursos. Dado este panorama, a utilização de plataformas ou ferramentas cujo objetivo é a criação de *websites*

de uma forma mais "automática", e menos demorada, acaba por ser uma das soluções encontradas por muitas empresas.

Dada a necessidade do mercado, o papel das plataformas de *e-commerce* tem vindo a crescer cada vez mais, sendo já possível encontrar implementações deste tipo de ferramentas, com vários níveis de complexidade.

Todas as lojas *online* têm diferentes exigências e requisitos. No entanto, existem pontos comuns entre os vários tipos de loja, que podem ser extraídos em módulos. Por exemplo, páginas individuais de produtos ou serviços, páginas do processo de pagamento e carrinho de compras. São estes módulos a base de oferta de uma plataforma para a criação de lojas *online*, porém também existem aspetos mais técnicos, como um sistema seguro de pagamentos ou um sistema de contas de utilizadores e gestão de sessões.

1.2. Objetivos

Como referido na secção anterior, as plataformas para criação de lojas *online* são cada vez mais utilizadas e necessárias no mercado atual, sendo por isso importante a sua melhoria e aperfeiçoamento. Desta forma, o tema deste trabalho é a criação de uma aplicação de criação de lojas *online* com intuito de desenvolver uma solução que fornece o esqueleto em código de uma loja, sobre o qual um programador poderá adicionar ou modificar o código de acordo com os requisitos do cliente para o qual a loja está a ser desenvolvida.

No âmbito da realização deste trabalho, foi inicialmente desenvolvida uma loja *online* para servir um negócio fictício de venda de bicicletas e produtos relacionados. Este exercício de construção do *website* de raiz serviu para fazer um levantamento de parâmetros essenciais para criação de uma loja *online*, tais como: nome da loja, slogan, esquema de cores, links para redes sociais, entre outros. Para além destes parâmetros, serviu também para estruturar as funcionalidades constituintes de uma boa loja como, por exemplo: a filtragem do catálogo de produtos, a lista de produtos favoritos, o histórico de encomendas ou a funcionalidade de guardar carrinhos para futura compra. Através desta análise dos parâmetros e funcionalidades importantes numa loja *online*, foi possível mapear os módulos transversais que podem ser utilizados em qualquer loja. São estes módulos a base da criação de uma plataforma de *e-commerce*.

É objetivo desta dissertação a conceção e desenvolvimento da *framework,* que define e gera o código dos módulos comuns a todas as lojas *online.* O trabalho desenvolvido consiste apenas na implementação parcial desta *framework,* com enfâse na vertente de gestão de utilizadores, procurando desenvolver uma forma eficaz e segura de implementar uma solução de criação de utilizadores e dos seus perfis, controlar as suas sessões dentro do *website.*

Para o desenvolvimento desta dissertação, foi definida a seguinte lista de objetivos a serem cumpridos:

a. criação de uma loja *online*;
b. levantamento dos parâmetros;
c. esquematização dos componentes a serem gerados pela *framework*;
d. desenvolvimento parcial da *framework*.

1.3. Estrutura da Dissertação

A presente dissertação está estruturada em cinco capítulos, sendo que cada um deles representa uma fase do desenvolvimento do projeto, contendo informação relevante à fase em questão.

Este primeiro capítulo contém uma breve introdução ao tema, o seu enquadramento, os objetivos e a estrutura da dissertação.

No segundo capítulo, são apresentados os conceitos fundamentais relativos ao *e-commerce* e as suas vertentes, lojas *online* e as diretrizes do seu desenvolvimento. Ainda no segundo capítulo, são apresentados, de forma sucinta, cinco plataformas já existentes que têm o mesmo objetivo da *framework* a desenvolver: a criação automática de lojas *online*. Para finalizar o capítulo do Estado da Arte, são comparadas as diversas soluções com recurso a uma tabela que apresenta vários parâmetros considerados importantes aquando da escolha de qual plataforma usar.

O terceiro capítulo é dedicado à explicação do desenvolvimento da loja *online*, começando pela sua arquitetura geral, bem como os diversos componentes que a constituem.

O quarto capítulo apresenta a estrutura da *framework* concebida, a linha de pensamento por detrás das decisões tomadas no seu desenvolvimento e o processo de implementação da aplicação de criação de lojas *online.*

O quinto capítulo é constituído pela conclusão desta dissertação, bem como sugestões para trabalhos futuros.

2. ESTADO DA ARTE

O conteúdo deste capítulo visa abordar os conceitos relevantes para o desenvolvimento da dissertação, começando por introduzir o tema do comércio eletrónico ou *e-commerce*, vantagens e desvantagens das lojas *online*, uma lista de algumas ferramentas para criação de lojas *online* já existentes no mercado, bem como uma comparação entre elas.

2.1. Comércio Eletrónico

2.1.1. Conceito do Comércio Eletrónico

Comércio eletrónico, também designado como *e-commerce*, é definido por Khoshnampour & Nosrati (2011:94) como "*comprar e vender produtos ou serviços na internet ou noutras redes*". O comércio eletrónico foi estabelecido na década de 1960, quando as organizações empresariais implementaram o uso do *Electronic Date Interchange* (EDI) como uma ferramenta para trocar dados com outras organizações empresariais, sendo que em 1979 a partilha de documentos por meio de redes eletrónicas para empresas foi estabelecido universalmente (Taher, G., 2021).

Segundo Nanehkaran (2013), o comércio eletrónico é um poderoso conceito e processo que mudou fundamentalmente a corrente da vida humana. Este autor define *e-commerce* como "*a interação entre sistemas de comunicação, sistemas de gestão de dados e segurança, que permite a troca de informações comerciais relacionadas à venda de produtos ou serviços (…)*" (2013:190). Já Niranjanamurthy, M., et al., (2013) definem comércio eletrónico simplesmente como a atividade económica que ocorre *online*.

Taher, G. (2021:154) afirma que "*o comércio eletrónico desempenha um papel essencial no avanço da tecnologia da informação, bem como da comunicação*". Segundo o mesmo autor, as melhorias na Internet, juntamente com os avanços na tecnologia da informação e o progresso na logística e nas entregas, permitiram que quase todas as empresas comprassem, vendessem e se relacionassem numa escala global, levando a um inesperado interesse pelo comércio eletrónico. Da mesma forma, Dragomirov, N. (2020) explica que o *e-commerce*

atualmente vê mais apoiantes e utilizadores devido ao aumento do acesso à Internet e ao maior número de dispositivos móveis para efetuar compras *online*.

O comércio eletrónico tornou-se uma prioridade e abriu as portas para as empresas, fornecendo a estas oportunidades para melhorar e avançar na posição de mercado. É possível afirmar que o "*comércio eletrónico tornou-se uma dinâmica fundamental da economia ao longo dos anos (…) permitindo que as empresas alargassem a sua carteira de clientes, bem como a sua capacidade de permitir que os seus consumidores comprem conforme sua conveniência*" (Taher, 2021:155).

2.1.2. Vantagens e Desvantagens

O comércio eletrónico tornou-se um ambiente imperativo para os negócios, e as oportunidades que este oferece são tão grandes que não há como voltar atrás (dos Santos, V., et al., 2017). As lojas *online* proporcionam inúmeros benefícios, quer para consumidores, quer para as empresas. Apresentam-se, abaixo, algumas das vantagens do e-*commerce* para os consumidores.

a. Compras 24/7

Através do *e-commerce*, é possível a venda a qualquer hora, durante todo o ano. Utilizadores e clientes podem visitar *websites*, pesquisar por produtos e serviços, bem como determinar os seus pedidos de acordo com as suas necessidades (Nanehkaran, Y. A., 2013; Niranjanamurthy et al., 2013; Taher, G., 2021; Khan A., 2016).

b. Rapidez

Com a vantagem de ser rápido, o *e-commerce* facilitou o processo de compra e venda, bem como a fácil localização de produtos. Com o comércio eletrónico não existem filas de espera, sendo esta uma das conveniências mais populares entre clientes (Taher, G., 2021; Khan A., 2016, Niranjanamurthy et al., 2013).

c. Acesso à Informação

O facto de os consumidores terem acesso constante ao comércio eletrónico permite que os mesmos encontrem informações relacionadas com os produtos e serviços que desejam, comparem custos e benefícios e, eventualmente, avaliem seu valor antes de realizar a compra (Taher, G., 2021). Este acesso à informação possibilita aos consumidores uma maior oportunidade de comparar diferentes preços e recursos, comparando as escolhas num sistema convencional ou físico que podem ser muito difíceis, dado que visitar todas as lojas e pedir preços pode ser desgastante para os clientes (Taher, G., 2021). De acordo com Khan A. (2016:19) "*além de comprar e vender, muitas pessoas usam a Internet como fonte de informação para comparar preços ou ver os produtos mais recentes em oferta antes de fazer uma compra online ou numa loja tradicional*".

d. Variedade de Opções

O comércio eletrónico proporciona aos consumidores uma enorme variedade de produtos e serviços fazendo com que estes tenham várias opções de escolha, sendo que grande parte delas seriam impossíveis através do comércio tradicional caracterizado pelo stock limitado (Taher, G., 2021). Ambos Niranjanamurthy et al. (2013) e Khan A. (2016) colocam a facilidade de trocar de fornecedor de produtos ou serviços, por insatisfação ou por não cumprir certas exigências, sem haver necessidade de deslocação física, como benefício do *e-commerce*.

e. Acessibilidade

As lojas *online* permitem disponibilizar a clientes artigos e serviços que não são possíveis de adquirir em lojas físicas próximas do consumidor como, por exemplo, a venda de produtos tradicionais de uma certa região que, sem uma loja *online*, seriam apenas possíveis de adquirir deslocando-se à tal região. Pelas palavras de Khan (2016:21), este benefício verifica-se quando "*o cliente pode comprar um produto que não está*

disponível no mercado local ou nacional, o que dá ao cliente uma gama mais ampla de acesso a produtos do que antes".

Para além dos benefícios apresentados para os consumidores, o *e-commerce* também apresenta benefícios para as empresas ou negócios.

a. Sem Limitações Geográficas

De acordo com Taher, G. (2021), as lojas convencionais impõem limitações aos vendedores, quer seja o cliente que efetua a deslocação maior para poder adquirir um item que não existe à venda perto de si, quer seja um negócio que pretende disponibilizar os seus serviços ou produtos numa outra localização, estão sempre associados custos e esforços extras. Através do comércio eletrónico, as empresas podem ampliar o seu negócio para mercados nacionais e internacionais, estando desta forma não limitados a geografias ou distâncias.

b. Redução de Custos

Negócios que utilizam o *e-commerce*, por norma, apresentam custos operacionais significativamente menores, em comparação com as lojas físicas, pois o número de trabalhadores pode ser bastante reduzido, e a logística de inventário simplificada (Taher, G., 2021). Como não há necessidade de uma loja física, as empresas que operam a partir do comércio eletrónico conseguem economizar em várias despesas gerais (Niranjanamurthy et al., 2013).

c. Mais Informação Sobre Clientes

"*Os vendedores online recolhem uma grande quantidade de dados dos consumidores para garantir que estejam a direcionar o seu negócio ao público certo*" (Taher, G., 2021). A partir desta recolha intensiva de dados, as empresas têm mais informação e poder de decisão a partir da análise dos mesmos, especialmente quando a empresa prepara estratégias de marketing e promoção dos seus produtos ou serviços para aumentar o seu potencial de venda.

d. Fácil Iniciação e Gestão do Negócio

A redução da complexidade de criar um negócio é notável quando se opta pelo comércio *online* (Taher, G., 2021; Niranjanamurthy et al., 2013). Sem preocupações relacionadas com a localização, gestão de um inventário local, funcionários e horários de funcionamento, uma loja *online* é atualmente uma solução desejada para muitos negócios.

Como qualquer outro negócio onde sempre há altos e baixos, apesar das suas vantagens, o comércio eletrónico também tem as desvantagens, quer para o consumidor quer para as empresas. Algumas das desvantagens para o consumidor são apresentadas de seguida.

a. Websites Fraudulentos

Com a acrescida facilidade de estabelecer uma loja *online*, aumenta também a facilidade de criar negócios fictícios, que têm como objetivo burlar os consumidores que efetuarem compras nessas lojas, sendo esta uma desvantagem sentida tanto pelos consumidores, como também pelas empresas (Niranjanamurthy et al., 2013).

b. Falta de Garantia de Qualidade

Não sendo possível inspecionar o produto fisicamente, o consumidor fica à merce das provas de autenticidade e qualidade que são fornecidas no *website*, sejam estas, fotografias, vídeos ou documentos.

c. Falta de Toque Pessoal

Muitos consumidores referem a falta de toque pessoal como uma das desvantagens do comércio eletrónico, dado que consumidores apreciam a experiência íntima de ir fisicamente à loja e se relacionar com os vendedores (Taher, G., 2021; Niranjanamurthy et al., (2013).

d. Atraso na Entrega

A espera é um dos fatores que pode ser considerado uma desvantagem do *e-commerce*, dado que comprar *online* significa esperar algum tempo para ter a sua compra em mãos (Taher, G., 2021).

e. Serviços ao Consumidor Restritos

Caso um consumidor tenha alguma dúvida sobre determinado item numa loja física, terá sempre alguém que trabalha na loja pronto a ajudar. Por outro lado, o atendimento ao cliente em sites de comércio eletrónico pode ser restrito, dado que o *website* pode oferecer serviços apenas num horário de trabalho específico e, às vezes, uma ligação para a divisão de serviços ao consumidor pode colocar o consumidor em espera por um longo tempo (Khurana, 2019).

Mencionadas as desvantagens para o consumidor, são enumeradas agora algumas das desvantagens para as empresas.

a. Dificuldade na retenção de clientes

Como há uma interação mínima direta do cliente com a empresa, a fidelização do cliente nem sempre é garantida. Existem vários motores de busca e sites de comparação de compras que ajudam os consumidores a localizar os melhores preços, colocando assim uma desvantagem para os negócios e empresas (Niranjanamurthy et al., 2013).

b. Desenvolvimento de Software

O software está constantemente a ser atualizado e modificado, o que pode provocar limitações às empresas *online*. Por exemplo, é necessário que as empresas atualizem constantemente o *software* e *hardware* para suportar o desenvolvimento das que utilizam (Taher, G., 2021).

c. Alta Dependência no *Website*

Devido à natural dependência de uma ligação à *Internet*, que surge da adoção do modelo de comércio *online*, quando esta falha, todas as potenciais vendas e provavelmente a parte logística ficam estagnadas. Para mitigar este problema as empresas contratam, por norma, um serviço de alojamento de *websites* com medidas para que o tempo que a loja esteja *offline* seja mínimo (Niranjanamurthy et al., 2013.)

2.1.3. Atributos de uma Loja *Online*

Para alguns negócios, abrir uma loja eletrónica pode significar apenas uma diversão ou um método de promoção da sua loja física, mas para outros o comércio eletrónico é uma solução de sucesso, pela qual podem obter lucros muito grandes (Lixandroiu, R., & Maican, C., 2015).

As variáveis de sistema de um *website*, tais como *website design*, fiabilidade, facilidade de acesso e conveniência são os fatores primários usados pelo consumidor para analisar o desempenho de uma loja *online*, sendo que a informação e a segurança podem levar os consumidores a ganhar maior satisfação ao aceder à mesma. Tendo estes fatores em conta, uma loja *online* deverá providenciar informação atualizada, correta, útil e completa (Lin, 2007).

Abdullah, E. et al. (2021) afirmam que a usabilidade e a acessibilidade são dos aspetos mais importantes que as empresas que oferecem produtos de *software* devem ter em conta, usando como definição de usabilidade "*um atributo de qualidade que avalia a facilidade de uso das interfaces de utilizador*". De acordo com os mesmos autores, dentro dos aspetos técnicos, é possível enumerar um grande número de pontos cruciais para um bom desempenho do *website*, entre eles encontramos a) complexidade da aplicação, b) legibilidade do código, c) versatilidade do código e d) portabilidade do código.

Para que todos estes aspetos estejam presentes na loja criada, os programadores recorrem regularmente a esquemas das interações dos utilizadores com o *website* e os diversos cenários que podem ser apresentados, colocando-se na perspetiva do consumidor.

Muitos destes atributos podem ser considerados essenciais para qualquer tipo de loja *online*. Por isso, de maneira a fazer um levantamento dos atributos constituintes de um bom *website,* e como eles se relacionam com as diversas ferramentas de *e-commerce*, é possível enumerar os seguintes.

2.1.3.1. Usabilidade

A usabilidade refere-se à eficácia, eficiência e satisfação na experiência do utilizador na loja (Kotian, H., & Meshram, B. B., 2017, January). É refletida na simplicidade do *design*, fácil compreensão da informação apresentada, e uma navegação descomplicada pelos conteúdos da loja.

2.1.3.2. Confiabilidade

A confiabilidade refere-se à veracidade do conteúdo exposto e à segurança nos sistemas implementados. Funcionalidades como um sistema de classificações e comentários contribui para aumentar a confiança do consumidor, sendo que métodos de pagamentos fiáveis conhecidos e globalmente usados são também pontos fortes no aumento da confiabilidade.

2.1.3.3. Conveniência de Acesso

A conveniência de acesso traduz-se na facilidade que o utilizador tem em aceder ao *website*, a rapidez com que este carrega e o suporte oferecido ao cliente. A escolha de uma boa hospedagem e a implementação de um sistema de apoio ao cliente são fatores importantes para garantir a satisfação do cliente ao usar o *website.*

2.2. Plataformas de *E-commerce*

2.2.1. Conceito de Plataforma de *E-commerce*

Dragomirov, N. (2020:251) define uma plataforma de *e-commerce* como *"produtos de informação que dão aos vendedores soluções para apresentar os seus produtos e serviços, no espaço digital e, se houver interesse manifestado pelo comprador, apoiar o processo de venda,*

finalizar a venda e a transação". Este autor acrescenta também que as plataformas de *e-commerce* são responsáveis pela automação de muitas funções de comércio e atendimento ao cliente, bem como proporcionar uma infraestrutura de suporte adequada para a gestão das atividades logísticas, de forma a garantir a movimentação dos fluxos de materiais na cadeia de fornecimento (Dragomirov, N.,2020).

De acordo com Lixandroiu, R., & Maican, C. (2015:54), "*para a instalação da plataforma de e-commerce, são necessários esforços de parametrização, importação de dados e produtos, design de templates de configuração, formas de pagamento, transporte, entre outros*". Na secção anterior discutiu-se quais os atributos a serem ponderados no desenvolvimento de uma loja *online*, são estes aspetos, que uma plataforma de *e-commerce* também usa para elaborar soluções e definir quais informações de configuração são necessárias requerer do cliente, com o objetivo de criar uma loja *online* personalizada, mas funcional.

Segundo o estudo realizado por Afolabi, A. O., et al. (2022), os principais benefícios associados com o uso de uma plataforma de *e-commerce,* passam pela facilidade de uso, rapidez na criação de um *website,* preços acessíveis bem como, não ser necessário conhecimentos de programação para usar as mesmas.

Contudo, este tipo de soluções também apresentam as suas limitações, sobretudo relacionadas com a personalização da loja *online*. Isto acontece porque no uso destas plataformas, a necessidade da generalização do código base, implica a uniformização do aspeto e das funcionalidades das lojas.

2.2.2. Tipos de *Frameworks* para *E-commerce*

É possível enumerar três tipos gerais de *frameworks* usadas em *e-commerce*, sendo que cada uma é diferente, tendo as suas vantagens e desvantagens. Estas *frameworks* diferenciam-se de acordo com o objetivo para que são usadas, bem como os recursos disponíveis para serem investidos.

2.2.2.1. *Software as a Service*

Software as a Service (SaaS) descreve um modelo de subscrição onde estão incluídos o *software*, hospedagem do servidor, manutenção do *website* e atualizações futuras de

software. Este modelo traz vantagens em termos de rapidez na obtenção de um *software* pronto para entrar no mercado, sem ter preocupações com custos de segurança, manutenção e hospedagem da plataforma desenhada.

Optar por esta fórmula para desenvolver uma loja *online* poderá ter limitações nos campos de customização do aspeto da página, bem como ficar restrito à lista de funcionalidades apresentado pelo vendedor.

Esta é a solução indicada para pequenos negócios, que tenham recursos limitados, quer monetários quer humanos, mais especificamente programadores dedicados a criar e manter um *software* personalizado.

2.2.2.2. *Open Source*

O uso de software *open source* consiste em usar código desenvolvido, de forma gratuita, por terceiros, e desenvolver a partir desse código base uma solução com todas as especificações desejadas pelo utilizador. Este estilo de *framework* traz um nível de customização superior a qualquer outro tipo de *framework,* e não tem custo inicial associado, uma vez que toda a gente tem acesso ao código.

Neste caso, não são fornecidos serviços de *hosting*, manutenção ou segurança, ficando a cargo do utilizador contratar programadores para tal efeito, bem como assegurar-se de uma solução independente para fazer *hosting* do *website* e servidores.

Para um negócio que possui uma equipa de programadores ou recursos para estabelecer uma *framework,* e cuja solução final requer diversas funcionalidades como, por exemplo integrações com sistemas já existentes, a *framework* baseada em *software open source* é provavelmente a mais adequada

2.2.2.3. *Headless Commerce*

O conceito de *headless commerce* refere-se à separação entre *back-end* e *front-end*, proporcionando ao utilizador a escolha de qualquer tecnologia para desenvolver o *front-end*. Em cenário industrial, este modelo proporciona também o melhor ambiente para implementar um esquema multi-vendedor.

Com esta separação, é possível modificar o *back-end* e o *front-end,* independentemente um do outro, e facilitar assim melhorias e atualizações sem necessidade de revisão de toda a *stack* de *software*.

Esta forma de separação requer uma equipa de programadores com conhecimentos avançados, não só para construir a solução de *e-commerce*, mas também para a manter e melhorar. Os custos de manutenção podem ser grandes, devido à individualização dos componentes.

2.2.3. Tipos de Plataformas de *E-Commerce*

2.2.3.1. Sistema de Gestão de Conteúdos

De acordo com os autores Suh, P., Ellis, J., & Thiemecke, D. (2002), um sistema de gestão de conteúdos, ou CMS, do inglês *Content Management System*, é um tipo de plataforma desenhado com o intuito de facilitar todo o processo de criação e manutenção de *websites,* desde a disposição do conteúdo na página, até às soluções para colocar o *website online*.

Como vantagens da utilização de um CMS, é possível incluir o acrescido à simplificação dos processos de manutenção, a rentabilidade, bem como, a capacidade de integração de *plugins* ou extensões para aumentar o catálogo de funcionalidades.

Tipicamente, a utilização deste tipo de plataforma não requer conhecimentos profundos de programação, fazendo uso de extensivas interfaces gráficas para proporcionar, ao utilizador, o maior controlo e poder de decisão sobre o funcionamento do *website.* Contudo, é dado ao utilizador a opção de editar diretamente o código.

Uma solução com o nível de personalização e controlo que um CMS oferece apresenta algumas desvantagens, nomeadamente o tempo necessário para configurar, instalar e fazer manutenção. Os custos associados a *plugins*, extensões e hospedagem também é outra das desvantagens deste tipo de plataforma de *e-commerce*. É de referir que, resultante da liberdade na manipulação dos elementos individuais do *website,* acresce a responsabilidade do utilizador em garantir a segurança do mesmo.

2.2.3.2. Construtor de Websites

Um construtor de *websites* apresenta-se como uma alternativa ao CMS, oferecendo uma utilização mais simplificada. Esta simplificação é conseguida através de restrições ao *layout* do *website,* dado existir uma limitação de opções através de um catálogo já predefinido pela plataforma. Outros dois fatores que simplificam a experiência do utilizador é a integração automática de funcionalidades populares no mercado, tais como SEO (*Search Engine Optimization)*, ferramentas de marketing, medidores de desempenho, entre outros, bem como a possibilidade de hospedagem do *website*. Este tipo de soluções permite guiar o utilizador através das diversas etapas na elaboração e publicação de um *website,* sem que este seja sobrecarregado com dezenas de opções de configuração não essenciais para o mesmo.

As vantagens associadas a este tipo de solução passam pela rapidez em obter e colocar um *website online*, a experiência simples e de fácil compreensão por parte do utilizador, e o facto de não ser obrigatório conhecimento em programação. Todas estas vantagens tornam esta solução ideal para iniciantes ou, como mencionado anteriormente, para utilizadores que não possuem conhecimento nem experiência na área da programação.

É comum este tipo de plataformas restringir o acesso ao código, e fazer uso de *software* proprietário e modelos predefinidos para assegurar a criação de um *website* seguro e funcional. Esta restrição ao código pode dificultar a migração para outro sistema ou plataforma, tornando o processo mais complexo e demorado.

A reduzida capacidade de personalização no *design* do *website* tanto pode ser vista como vantagem ou desvantagem. Com um número elevado de *websites* criados através do mesmo modelo, mais difícil será de destacar uma loja única ou diferenciada das restantes. Além disso, o facto de construtores de *websites* serem sistemas fechados fazem com que o *website* criado nunca seja verdadeiramente do utilizador, dado que este fica dependente de todos os serviços dados pela plataforma.

2.2.4. Ferramentas Existentes

Existe um vasto catálogo de opções quando falamos da criação de sites de comércio eletrónico, sejam estes *scripts* simples, plataformas com estruturas predefinidas, ou até

soluções específicas desenhadas de raiz para acomodar as necessidades do negócio. A chave é perceber quais são os requisitos da loja a desenvolver, e entender qual a opção mais apropriada.

Com isto em mente, foram testadas quatro ferramentas disponíveis de forma gratuita, por tempo limitado ou permanente. Em baixo, é descrito, de uma maneira sucinta, as vantagens, desvantagens e elementos únicos de cada plataforma de *e-commerce* analisada.

2.2.4.1. Shopify

A Shopify (Shopify, 2022) é uma plataforma SaaS das mais usadas no mercado atual (BuiltWith, 2022), que proporciona a oportunidade de criar soluções de *e-commerce* sem conhecimentos de programação. Baseada em planos de subscrições mensais, estes planos incluem *hosting*, suporte para *add-ons* e alguns métodos de pagamento. Custos extra estão normalmente associados a extensões pagas, domínios personalizados e métodos de pagamento adicionais. A Shopify, ao contrário da maioria das outras soluções estudadas, cobra taxas por transação efetuada, exceto quando esta é feita a partir do sistema de pagamentos prioritário da Shopify, o Shopify Payments.

O fator personalização na Shopify é extremamente completo, sem que se torne demasiado complexo para os utilizadores novatos. Oferece temas por defeito com *layouts* editáveis, sendo possível alterar com grande detalhe os diversos componentes da loja. Existe também a opção de adquirir um tema pago a partir do seu Marketplace (Shopify, 2022), ou até a partir de terceiros que desenvolvem *layouts* para o Shopify como, por exemplo, ThemeForest (ThemeForest, 2022). Existe também um sistema de suporte para dúvidas, quer sobre o *software* da Shopify, quer para as suas extensões.

Percebe-se que a plataforma tem o objetivo de criar soluções que sejam facilmente implementadas e mantidas por clientes que não tenham experiência com programação, ou a hipótese de contratar um programador. Com as funcionalidades de integração com sites de terceiros e até lojas físicas, torna a Shopify numa boa escolha, não só para pequenos negócios sem grandes recursos, mas também para negócios de grande escala já com uma rede de lojas físicas.

2.2.4.2. osCommerce

O osCommerce é a solução *open-source* para *e-commerce* criada por Harald Ponce de Leon, podendo executar sem custos num servidor com as ferramentas de PHP (*Hipertext Preprocessor*) e *MySQL*. É preciso ter em conta que não é oferecido qualquer tipo de *hosting*, domínio ou escolha de temas para o *website*.

Sendo o seu código público, existe uma grande comunidade de programadores e outros interessados, que contribuem regularmente para melhorar a plataforma. É através do *marketplace* que os utilizadores podem melhorar a solução base com novos temas, métodos de pagamento, estatísticas, gestão de negócio, entre outros.

O osCommerce, sendo uma solução que já existe há bastante tempo, apresenta dificuldades na escalabilidade, que se refletem na possível lentidão no *website*.

2.2.4.3. PrestaShop

A Prestashop (PrestaShop, 2023) apresenta-se como uma solução grátis para a criação de uma loja *online,* personalizada ao gosto do cliente, com cerca de 300 mil *websites* ativos a usar esta ferramenta (PrestaShop, 2023). A versão experimentada utilizada para esta análise não oferece serviços de hospedagem do *website*, sendo esta uma funcionalidade paga, sendo que, e de forma a analisar esta plataforma, experimentou-se a versão sem hospedagem de *website*.

A ferramenta oferece uma boa coleção de funcionalidades, como marketing e SEO, controlo de inventários e catalogação de produtos, e diversos métodos de pagamento. Possui também uma vasta lista de *add-ons* no seu Marketplace (PrestaShop, 2023), sendo a grande maioria catalogados a preços bastante significativos. A oferta de *add-ons* sem custos é geralmente composta por módulos desenvolvidos pela própria Prestashop*,* de forma a adicionar funcionalidades mais comuns à loja, sem confundir os novos utilizadores e a sua experiência na plataforma.

A loja criada tem como base um único tema predefinido pela Prestashop, sendo este alterável através do portal da Prestashop*,* ou editando diretamente os ficheiros HTML e CSS. Temas extra podem ser comprados no *Marketplace,* e oferecem diferentes níveis de personalização.

A estrutura do *backoffice* contém todas as ferramentas que um vendedor precisaria para iniciar um negócio, pecando apenas na dificuldade em encontrar certas funcionalidades escondidas em diversas camadas de menus. De notar que o suporte personalizado é pago, isto é, a Prestashop oferece documentação e fóruns de comunidade sem custos, mas para ter ajuda de um membro da equipa na construção e personalização da loja, é apenas possível através de um plano de suporte anual pago.

Em suma, a Prestashop é uma solução viável para criar a loja *online* de um negócio que, apesar de ser grátis de instalar, poderá ter um custo elevado quando somados os custos de hospedagem, temas extra e módulos adquiridos.

2.2.4.4. WooCommerce

Um *plugin open source,* famoso para *websites* construídos com recurso à ferramenta WordPress*,* é o WooCommerce (WooCommerce, 2022), que transforma um *website* de conteúdos numa loja *online* completa e pronta a ser usada. A criação do *website* e o uso do *plugin* são gratuitos, mas a hospedagem da loja fica da responsabilidade do cliente, de onde podem advir custos conforme o serviço que for escolhido. A estrutura de funcionamento é muito parecida com as ferramentas referidas anteriormente, onde existe a estrutura base que é gratuita e, mediante as necessidades e requisitos da loja que se deseja criar, é possível fazer usufruto de um leque extensivo de *add-ons,* pagos e gratuitos.

Um grande ponto de referência do WooCommerce é a sua imensa lista de temas, sendo que estes variam desde o mais simplista com apenas algumas variáveis de caracterização, até ao mais complexo com várias camadas de personalização. A maioria dos temas são facilmente instalados através do próprio catálogo da WooCommerce, mas existe suporte para instalar temas desenvolvidos por terceiros.

Esta já é uma solução que se diferencia das demais, pois requer um bom nível de conhecimento do *workspace* da Wordpress, nomeadamente como este é programado e estruturado. A curva de aprendizagem é bastante acentuada, principalmente se o *website* for criado de raiz através da Wordpress, e só depois acionar o *plugin* WooCommerce. Dado estes factos, esta será uma ferramenta mais indicada para programadores com alguma experiência.

2.2.4.5. Wix

Uma plataforma para a construção de *websites* que oferece uma forma simples de criar um *website,* sem a necessidade de conhecimentos de programação. Na Wix é possível criar qualquer tipo de *website*, incluindo lojas *online*. É nesta vertente de *e-commerce* que a ferramenta foi analisada (Wix, 2023).

A experiência de uso da plataforma Wix começa com a escolha do tipo de *website, e* de um *layout* de entre os que são disponibilizados pela equipa da Wix. São também selecionadas as funcionalidades que o *website* irá ter, neste caso, funções de *e-commerce,* bem como um plano de subscrição.

A interface para edição do *design* da loja é do estilo *drag-and-drop*, onde é possível usufruir de um nível de personalização bem mais profundo que nas restantes ferramentas analisadas anteriormente, sendo o utilizador capaz de modificar o posicionamento, dimensões e comportamento de cada elemento na página, individualmente.

A gestão do conteúdo através do *backoffice* na plataforma Wix é simples e básica, e isto apresenta vantagens e desvantagens, já que facilmente se adicionam novos produtos. No entanto, não existe hierarquia na categorização dos mesmos, sendo assim impossível criar subcategorias. A gestão de encomendas é também clara, com funcionalidades para tratamento de grandes volumes de encomendas ao mesmo tempo. A Wix possui também um catálogo onde é permitido comprar *add-ons* para ampliar e melhorar a experiência de uso (Wix, 2023).

Apesar de larga coleção de temas disponíveis, a Wix não permite a troca de temas após finalizar o processo de criação, sendo que, para tal efeito, é necessário reconstruir o *website* desde o início. Devido ao elevado número de detalhes com os quais o utilizador pode interagir, é provável que este gaste demasiado tempo com pormenores que não influenciarão o desempenho da loja em modo algum.

A maior desvantagem da ferramenta Wix é o seu desempenho abaixo da média, quer na frente de loja, quer no *backoffice*, sendo que os tempos de carregamento são consideravelmente superiores que a competição.

2.2.5. Comparação entre Ferramentas Existentes

Baseado nas análises das várias ferramentas disponíveis que cumprem o mesmo objetivo que a *framework* a ser desenvolvida nesta dissertação, construiu-se a Tabela 1 de forma a sumariar a comparação descrita nos últimos parágrafos.

Para esta avaliação, foram escolhidos certos parâmetros que representam os requisitos essenciais e qualidades mais procuradas numa solução para criação de uma loja *online*. Os parâmetros foram a) o preço ou cobrança pelo serviço, quais os modelos de monetização e outros custos associados, b) inclusão do serviço de hospedagem, c) o *design* e a sua atratividade, d) experiência de interação e facilidade de uso, e) escalabilidade da solução gerada, f) métodos de pagamento disponíveis, g) quão completo é o *backoffice*, h) *layouts* e manipulação ou personalização destes, i) níveis de conhecimento de programação necessários para ser usada e, por fim, j) manutenção.

Tabela 1 - Tabela Comparativa entre Ferramentas

	Shopify	**WooComerce**	**PrestaShop**	**OsCommerce**	**Wix**
Preço	Subscrição	Grátis	Grátis	Grátis	Subscrição
Hospedagem	Sim	Não	Não	Não	Sim
Design	Muito bom	Bom	Bom	Bom	Muito bom
Interação	Muito bom	Bom	Bom	Bom	Muito bom
Métodos de Pagamento	Excelente	Excelente	Excelente	Excelente	Excelente
BackOffice	Muito bom	Médio	Bom	Bom	Muito bom
Escalabilidade	Muito boa	Boa	Boa	Fraca	Fraca
Conhecimentos programação	Não	Sim	Sim	Sim	Não
Temas/Layout	Pagos. Editáveis.	Gratuitos. Editáveis.	Gratuitos. Editáveis.	Gratuito. Não editável.	Gratuitos. Editáveis.
Manutenção	Atualizações automáticas	Atualizações manuais	Atualizações manuais	Atualizações manuais	Atualizações automáticas

Atendendo aos aspetos comuns entre todas as plataformas analisadas, percebe-se que a filosofia aplicada no desenvolvimento das soluções é influenciada pelo meio em que estes se inserem, isto é, num ambiente onde a personalização é posta como uma elevada prioridade e, ao mesmo tempo, a minimização de custos é vista como uma mais-valia. A abordagem mais comum é a de assegurar uma boa base, e oferecer a oportunidade de adquirir/adicionar funcionalidades para melhor moldar a loja aos requisitos dos clientes.

Podemos concluir, tal como refere Lixandroiu, R., & Maican, C. (2015), que uma boa plataforma de *e-commerce* permite adicionar ferramentas para ajudar o seu negócio a crescer, reduzir significativamente o processo de manutenção e simplificar as tarefas de administração.

3. LOJA ONLINE

No presente capítulo, são explorados os requisitos de uma loja *online*, a arquitetura que esta poderá seguir, e a interação do utilizador quer como cliente, quer como administrador da loja. O propósito da criação desta loja *online* consiste em fazer um levantamento da arquitetura e componentes necessários numa loja *online*, de forma a, posteriormente, esta estrutura e componentes serem traduzidos no desenvolvimento da *framework*. Por outras palavras, o *website* desenvolvido para esta dissertação serve como base para a criação da *framework*.

A loja descrita neste capítulo foi idealizada para servir um negócio fictício de venda de bicicletas e produtos relacionados, contando com um *frontoffice* para consulta de produtos e realização de encomendas por parte dos clientes, um *backoffice* composto por ferramentas de gestão, e um *back-end* de suporte à loja. Foi apenas implementado um método de pagamento, *Paypal,* por este ser capaz de integrar tanto pagamentos por cartão de crédito, como de débito, para além de permitir usar dinheiro da conta *Paypal*. Após realizada uma encomenda, o envio da mesma fica a cargo do administrador da loja, pois a implementação da loja não abrange logística de entrega de encomendas.

3.1. Requisitos

Para o funcionamento de um *website* de *e-commerce* é requerida uma lista de funcionalidades básicas. De maneira a estudar quais são, a sua importância e como se relacionam, foram visitadas diversas lojas *online,* concluindo ser necessário os seguintes requisitos funcionais.

3.3.1. Frontoffice

a) Registo, *login* e *logout* de utilizadores;
b) Consulta e edição de informação pessoal;
c) Consulta do catálogo de produtos;
d) Filtragem de produtos mediante categoria ou nome;

e) Consulta de detalhes sobre um produto;
f) Marcar produto como favorito;
g) Criação de carrinhos de compras;
h) Adição e remoção de produtos do carrinho de compras;
i) Efetuar encomenda dos produtos no carrinho;
j) Consulta do histórico de encomendas;
k) Consulta de detalhes da encomenda;

3.3.2. Backoffice

a) Adicionar, remover e editar contas de utilizador;
b) Adicionar, remover e editar produtos no catálogo;
c) Adicionar, remover e editar categorias de produtos;
d) Adicionar, remover e editar encomendas;
e) Associar produtos a marcas e categorias;
f) Associar produtos semelhantes;
g) Marcar produtos como novidade ou em promoção;
h) Emissão de faturas;

3.2. Abordagem

A estruturação e desenvolvimento da loja *online* foi feita recorrendo ao *stack* de tecnologias MERN (*MongoDB, ExpressJS, React* e *NodeJS*). Foi utilizado *NodeJS* de forma a permitir executar *Javascript* em servidores *back-end*, e *ExpressJS* para permitir a configuração de um servidor HTTP. Foi também usado *MongoDB* como base de dados para suporte à loja e, por fim, *React* para desenvolver o *front-end* da loja. O fluxo de informação e o estado da aplicação no *front-end* são geridos com recurso ao *Redux*, uma biblioteca para auxiliar no controlo de variáveis e mudanças das mesmas, ao longo do uso da loja. Abaixo, na figura 3, é apresentado o esquema da arquitetura usada na loja *online*, com as diversas relações entre as tecnologias.

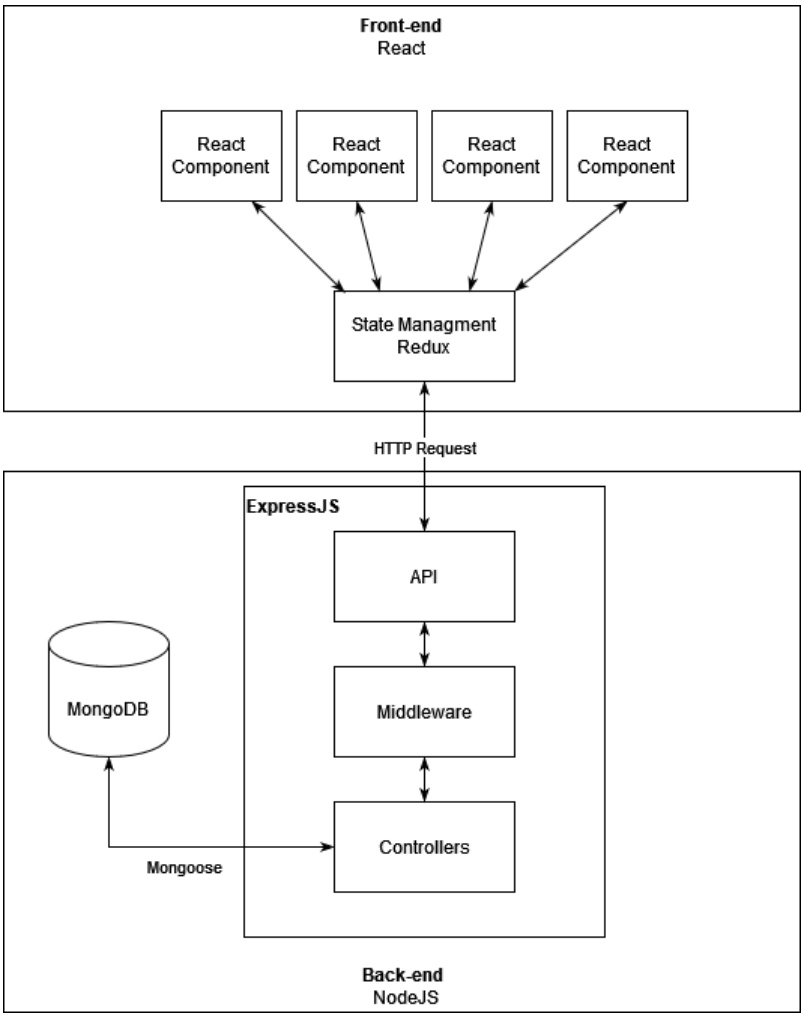


Figura 1 - Esquema da Arquitetura Geral da Loja *Online*

É possível desenvolver o *back-end* de duas formas, sendo que uma delas teria dois servidores HTTP separados, um para pedidos do cliente - como consulta de produtos, pagamentos e encomendas, e outro para os pedidos do administrador (*backoffice*), permitindo edição de produtos, contas e encomendas. Apesar da distinção entre servidores, eles partilhariam a base de dados e serviriam um só *front-end,* que englobaria quer a frente de loja quer as páginas de *backoffice.*

A segunda hipótese de desenvolvimento do *back-end*, e a escolhida na programação da loja desta dissertação, é um único servidor que responde a pedidos de todos os utilizadores, sendo os diferentes níveis de acesso (cliente ou administrador) configurados através de *middleware*. Deste modo, toda a informação relativa à loja é guardada numa única base de dados.

Para uma melhor compreensão desta abordagem, é necessário estabelecer os papeis dos dois tipos diferentes de utilizadores, o cliente comum e o administrador da loja.

O *design* da loja permite, como a maioria das lojas já existentes na *Internet*, que qualquer pessoa consulte o catálogo de produtos, os detalhes dos mesmos, e a criação de um carrinho de compras. O registo do utilizador através da criação de uma conta no *website* apenas é necessário quando este desejar proceder à encomenda dos produtos, permitindo assim facilmente consultar o estado da encomenda no futuro, pois ficará associada a uma conta de utilizador. A diferença entre um utilizador e um visitante é apenas o facto de a pessoa ter uma sessão iniciada ou não.

Por outro lado, um administrador da loja é alguém que já terá uma conta criada por omissão, sendo apenas fornecidas as credenciais para que este possa iniciar a sessão no *website*. Aquando da autenticação do utilizador, se este estiver indicado como conta de administrador, será permitido o acesso ao painel de *backoffice*, que contém todas as ferramentas e funcionalidades para a gestão e manutenção, quer da loja quer dos seus clientes. De notar que um utilizador com permissões de administrador tem acesso às ferramentas de gestão em adição a quase todas as funcionalidades do consumidor comum, sendo possível, desta forma, consultar produtos e os seus detalhes, mas não lhe é permitido efetuar a encomenda dos mesmos.

Os diagramas de casos de uso apresentados nas figuras 2 e 3, esquematizam as interações que os diversos tipos de utilizadores têm com as funcionalidades da loja.

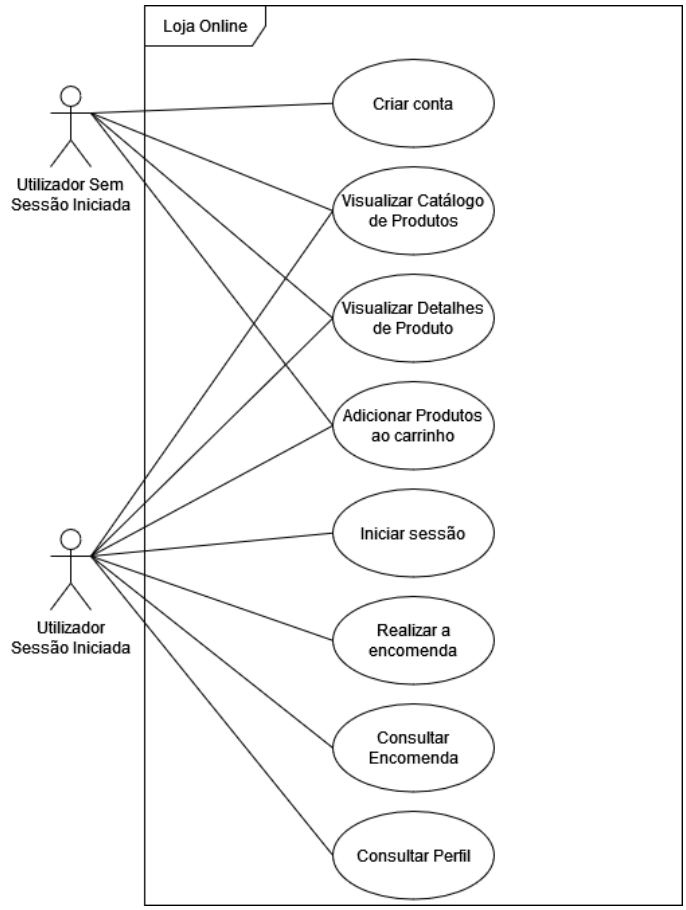


Figura 2 - Diagrama Caso de Uso *Frontoffice*.

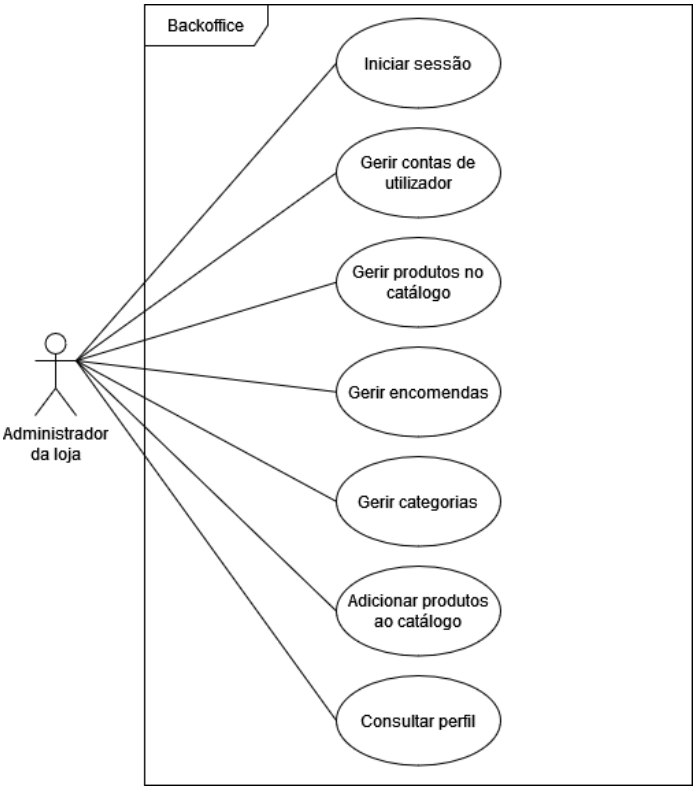


Figura 3 - Diagrama Caso de Uso *Backoffice*.

3.3. Arquitetura

Tendo em conta os requisitos levantados na secção 3.1, e com o intuito de uma implementação modular, na qual a *framework* abordada no capítulo 4 será baseada, as diversas partes da loja *online* foram divididas em componentes.

3.3.1. Componentes *Front-end Frontoffice*

3.3.1.1. Autenticação

A Autenticação é um sistema com *login*, *logout* e registo de utilizadores, sendo composto por uma página com um formulário de registo na primeira utilização. Este formulário é necessário na criação de conta de utilizador, onde são recolhidos dados como nome, email, morada entre outras informações pertinentes para identificação, utilização do site, e simplificação dos processos de compra. Para proceder ao *login,* são apenas requeridas as credenciais de *email* e *password*.

3.3.1.2. Perfil

O componente Perfil consiste numa página onde são apresentados os dados pessoais do utilizador, com sessão iniciada, sendo possível a edição dos mesmos. Está também integrada na página de perfil de utilizador, a funcionalidade de histórico de encomendas, onde é permitido consultar o estado das encomendas efetuadas, sendo que o estado das mesmas é atualizado conforme a sua evolução (pagamento pendente, pago, enviada, recebida).

3.3.1.3. Catálogo de Produtos

Apresentação de todos os produtos vendidos pela loja, bem como opções para filtrar resultados, mediante parâmetros predefinidos (preço, categorias, marcas, etc). Os produtos estão representados através de uma lista de cartas, isto é, componentes mais pequenos constituídos pela fotografia do produto, o seu nome, categoria a que pertence e um botão para saber mais detalhes.

3.3.1.4. Carrinho de Compras

Esta é das principais funcionalidades da loja, com capacidade de registar múltiplos produtos e as suas quantidades, para que o utilizador possa proceder à compra dos mesmos, numa só transação. É possível adicionar itens ao carrinho com um simples clique de um botão na página do produto. Já a página dedicada à apresentação do carrinho coloca ao dispor do utilizador ações para modificar o número de unidades a adquirir de um certo produto, para remover por completo o artigo e consultar o valor total dos itens e custos extras.

3.3.1.5. Registo de Encomendas

Capacidade de efetuar uma compra através de três passos, começando pela confirmação da lista de itens a comprar, seguido do pagamento do valor total incluindo taxas e, por fim, o preenchimento dos dados para envio caso necessário.

Na figura 4 está representado a interface do componente Catálogo de Produtos, onde é possível ver como são apresentados os produtos ao utilizador e qual a informação que está disponível.

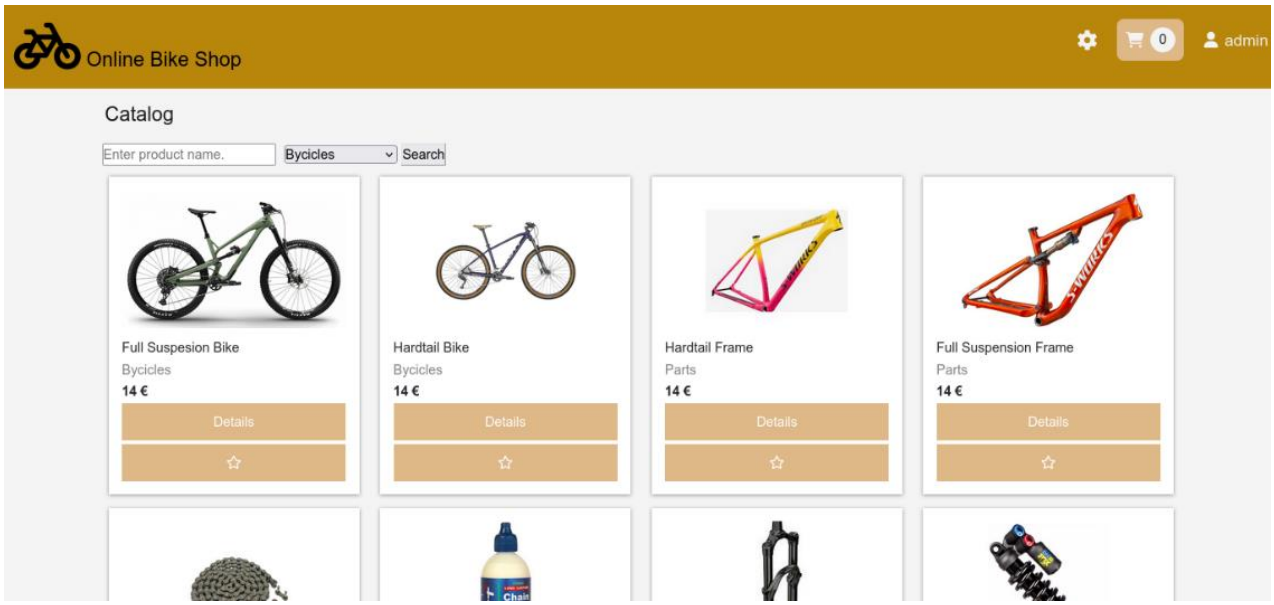


Figura 4 - Interface do Componente Catálogo de Produtos

3.3.2. Componentes *Front-end Backoffice*

3.3.2.1. Gestão do Catálogo

A Gestão do Catálogo de Produtos é um componente de *backoffice* desenhado para ser usado por um gestor da loja, para que este possa manipular o catálogo apresentado aos consumidores. É apresentada uma lista de todos os produtos correntemente vendidos na loja, sobre os quais é possível executar um leque de ações, desde editar a informação do produto, eliminá-lo do catálogo, ou usá-lo como base para criar um novo produto.

3.3.2.2. Gestão de Encomendas

A Gestão de Encomendas é um componente de *backoffice* focado no controlo de encomendas. Em termos de interface, conta com a lista de encomendas efetuadas pelos consumidores, permitindo que esta seja filtrada por estado da encomenda. Sempre que o gestor assim o desejar, pode individualmente consultar e/ou atualizar os detalhes de uma encomenda, ou até mesmo cancelar por completo a mesma. Para todas estas ações existem botões dedicados para facilitar a interação e aumentar a rapidez e fluidez do controlo do *website*.

3.3.2.3. Gestão de Utilizadores

A Gestão de Utilizadores consiste num componente de *backoffice*, sendo este constituído por uma tabela para facilitar a visualização da lista de clientes registados no *website*. Podem ser pesquisados nomes específicos de clientes para facilitar a manutenção dos perfis e deteção de potenciais problemas com um utilizador específico. É neste componente que podem ser criadas novas contas com acesso administrativo, algo que não é possível usando o componente de autenticação acima descrito.

3.3.2.4. Gestão de Categorias

A Gestão de Categorias é o componente responsável pela logística da árvore de categorias em que os produtos se baseiam. Este componente permite adicionar uma categoria, com nome e descrição, ficando esta disponível para ser escolhida aquando da criação do produto. Para configurar uma subcategoria basta atribuir-lhe uma categoria-mãe. Editar e remover categorias existentes é uma tarefa fácil usando os botões disponíveis para essas ações.

É apresentado na figura 5, um exemplo da interface de um dos componentes do *backoffice*, nomeadamente, a página de gestão de produtos.

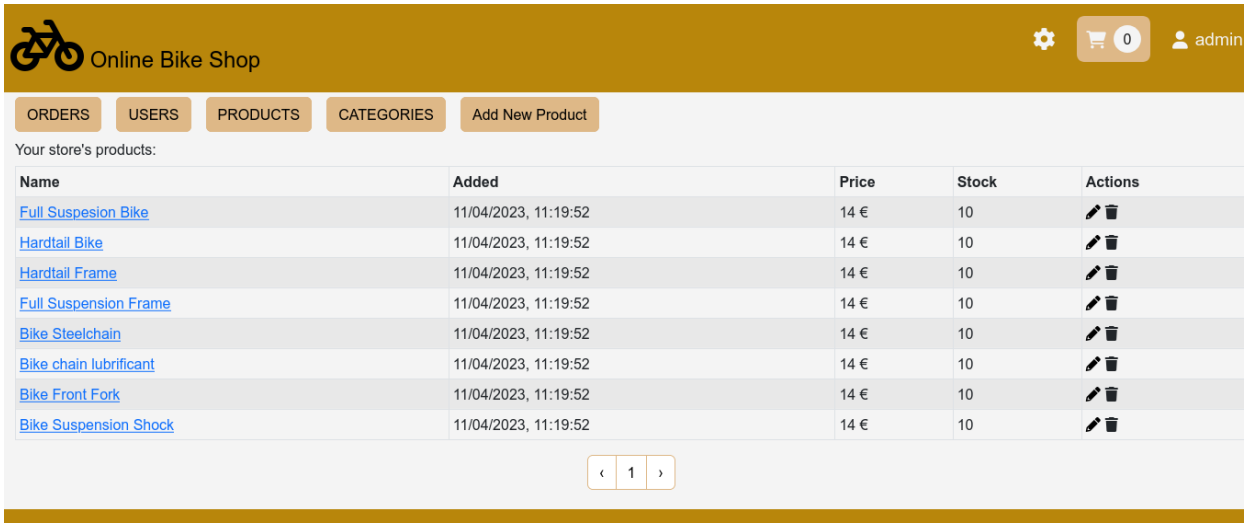


Figura 5 - Interface do Componente Gestão de Catálogo

3.3.3. Componentes *Back-end*

3.3.3.1. Utilizadores e Autenticação

Este componente da estrutura do *back-end* é desenhado para suportar todas as ações relacionadas com gestão de utilizadores e com o sistema de autenticação dos mesmos. Sendo cada utilizador representado numa base de dados pelo seu nome, endereço de *email*, *password* (encriptada) e nível de permissão. Na presente implementação, apenas existem dois níveis de permissões, administrador e não administrador. Após a verificação com sucesso das credenciais do utilizador, é gerado um *Json Web Token*, que passa a servir de método de autenticação quando há

comunicação com o servidor. Este *token* tem uma data de expiração predefinida pelo servidor. O servidor está configurado com os *endpoints* listados na Tabela 2.

Tabela 2 - *Endpoints* do Componente Utilizadores

URL	Método	Acesso	Pedido	Resposta
.../users/register	POST	Publico	Credenciais da conta a criar	Conta criada
.../users/login	GET	Publico	Credenciais da conta	Token de sessão
.../users/	GET	Admin		Lista dos utilizadores registados
.../users/:id	GET	Privado	ID do utilizador	Perfil do utilizador incluindo histórico de encomendas

3.3.3.2. Produtos

Componente responsável por coordenar todos os produtos, sendo este composto por ferramentas de forma a permitir consultar, adicionar, remover e alterar o catálogo de produtos existentes na loja. Um produto tem como suas propriedades um nome, uma descrição, o preço por unidade, a quantidade de *stock* disponível e uma coleção de imagens do produto. A quantidade de produtos que são facultados por pedido é limitado ao tamanho de página, ou ao número de produtos existentes que cumprem os filtros inseridos. A consulta dos produtos ou dos seus detalhes não carece de autenticação. Por outro lado, a sua manipulação quer seja adicionar, remover ou alterar um existente é apenas possível com nível de acesso de administrador. Os *endpoints* constituintes deste componente estão listados na Tabela 3.

Tabela 3 - *Endpoints* do Componente Produtos

URL	Método	Acesso	Pedido	Resposta
.../api/products/	GET	Publico	Número da página	Lista de produtos
.../api/products/	POST	Admin		Produto criado
.../api/products/:id	GET	Publico	Id do produto	Detalhes do produto
.../api/products/:id	DELETE	Admin	Id do produto	
.../api/products/:id	PUT	Admin	Id do produto	Produto atualizado

3.3.3.3. Encomendas

O componente de encomendas é bastante dependente dos restantes descritos anteriormente, dado que depende da informação guardada nos módulos anteriores, seja detalhes de produto como o *stock*, ou dados de utilizador a quem as encomendas pertençam. Na criação de uma encomenda, esta é sempre iniciada sem detalhes de pagamento nem de entrega. Por outras palavras, é considerada por pagar e por entregar. Após a interação com sucesso no *gateway* da *Paypal*, a encomenda é atualizada com os dados da transação executada, e considerada como paga. O campo de entrega é manualmente atualizado pelo administrador aquando da verificação de chegada da encomenda ao destino através de terceiros. Os *endpoints* deste componente estão apresentados na Tabela 4.

Tabela 4 - *Endpoints* do Componente Encomendas

URL	Método	Acesso	Pedido	Resposta
.../api/orders/	GET	Admin	Número de página	Lista de todas as encomendas
.../api/orders/	POST	Privado		Encomenda criada
.../api/orders/:id	GET	Privado	ID da encomenda	Detalhes da encomenda
.../api/orders/:id/payment	POST	Privado	ID da encomenda	Encomenda com pagamento atualizado
.../api/orders/:id/delivery	POST	Privado	ID da encomenda	Encomenda com a entrega atualizada

3.3.3.4. Categorias

O componente de categorias tem como função mapear hierarquicamente as categorias em que os produtos se podem inserir. As categorias são compostas por nome, descrição e, se assim for o caso, uma categoria-mãe. Para facilitar a leitura e a navegação das subcategorias, aquando do envio da lista de categorias para o *front-end*, esta vai esquematizada em árvore. É possível consultar, adicionar, editar e remover categorias através dos *endpoints* especificados na Tabela 5.

Tabela 5 - *Endpoints* do Componente Categorias

URL	Método	Acesso	Pedido	Resposta
.../api/categories/	GET	Publico		Lista das categorias
.../api/categories/	POST	Admin	Dados da nova categoria	Categoria criada
.../api/categories/	DELETE	Admin	Nome da categoria	
.../api/categories/	PUT	Admin	Novos dados	Categoria Atualizada

3.4. Resumo

No presente capítulo foram apresentadas, a abordagem tomada na conceção da loja *online,* foi feito um levantamento dos requisitos da mesma e explorada a arquitetura em componentes, tendo sido descrito o contexto e a função de cada um deles. Consideram o conteúdo abordado, é possível afirmar que os objetivos iniciais do capítulo foram cumpridos.

4. FRAMEWORK

No capítulo que se segue, é descrita a estrutura e funcionamento da *framework* desenvolvida. São abordados os componentes implementados bem como, a linha de pensamento por detrás das decisões tomadas no seu desenvolvimento. É também explorada a temática de integração de novos componentes na *framework*, descrevendo quais as alterações necessárias e limitações encontradas neste processo. Por fim, é detalhado o desenvolvimento de uma aplicação para a criação de lojas *online,* incluindo a sua interface gráfica e interação com a *framework* criada.

4.1. Arquitetura da Framework

A arquitetura do código gerado para a loja é baseada em componentes genéricos, sendo que a arquitetura de *back-end* da *framework* está representada na figura 6.

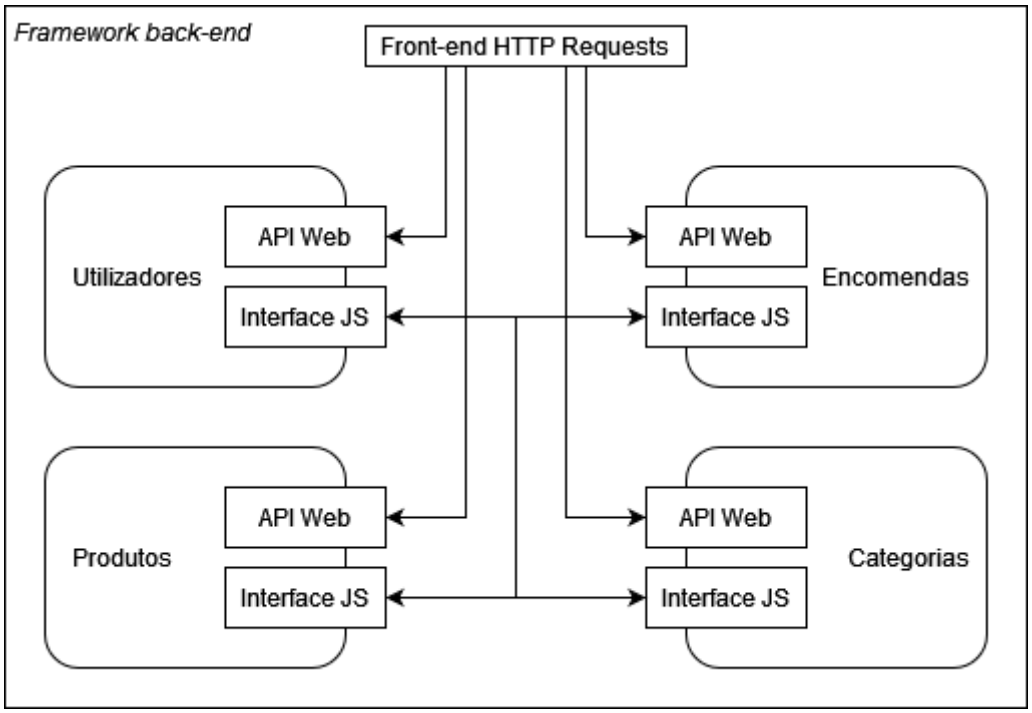


Figura 6 - Arquitetura do back-end da framework.

Cada componente (Utilizadores, Encomendas, Produtos, Categorias) comunicam entre si através de uma coleção de funções, formando assim uma interface individual para cada

componente. A interação dos componentes de *front-end* com os do *back-end,* é efetuada através de uma API, foi definida com recurso a URIs (*Uniform Resource Identifier)*.

É apresentado na Tabela 6, as funções das interfaces JS implementadas pelos componentes desenvolvidos, bem como funções que são requeridas das outras interfaces não desenvolvidas.

Tabela 6 - Interfaces JS

Componente	Interface	Uso
Utilizadores	getById(int id)	Retorna utilizador com ID fornecido.
	getByName(string name)	Retorna utilizador com o nome fornecido.
Categorias	getById(int id)	Retorna categoria com ID fornecido.
	getByName(string name)	Retorna categoria com o nome fornecido.
Encomendas	getByUser(string name)	Retorna encomendas do utilizador com o nome fornecido. (Requerida pelo componente Utilizadores)
	getByUser(int id)	Retorna encomendas do utilizador com o ID fornecido. (Requerida pelo componente Utilizadores)

A API Web desenvolvida para o componente "Utilizadores" da *framework*, tem a mesma lista de rotas apresentadas na Tabela 2, no capítulo de descrição da loja. Estão identificadas na Tabela 7, as funções definidas no controlador do componente correspondentes a cada uma dessas rotas.

Tabela 7 - Interface API Web do componente "Utilizadores"

Rota	Função JS	Argumentos
.../users/register	registerUser()	Nome, email, password. OPCIONAL: morada, contacto.
.../users/login	loginUser()	Email, password.
.../users/	getUsers()	Sem argumentos.
.../users/:id	getUserDetails()	Id do utilizador.

A arquitetura do *front-end* da *framework* tem como base os componentes de *React* utilizados. Na lista que se segue, encontram-se nomeados os componentes de topo, representantes das páginas da loja, sendo que alguns destes são construídos a partir de subcomponentes comuns a múltiplas partes da loja.

- Formulário de *login* e registo;
- Página de Perfil;
- Página de produto;
- Catálogo de produtos;
- Página de pagamento;
- Página de listagem de encomendas;
- Página de listagem de utilizadores;
- Página de listagem de produtos;
- Página de listagem de categorias;
- Página de edição/criação de utilizador;
- Página de edição/criação de produto;
- Página de edição/criação de categoria.

Na figura 7 está representado um diagrama do fluxo sequencial de tarefas para a utilização da *framework,* incluindo os passos a serem executados manualmente pelo utilizador. Os dados de configuração são provenientes do *front-end* da aplicação de criação de lojas.

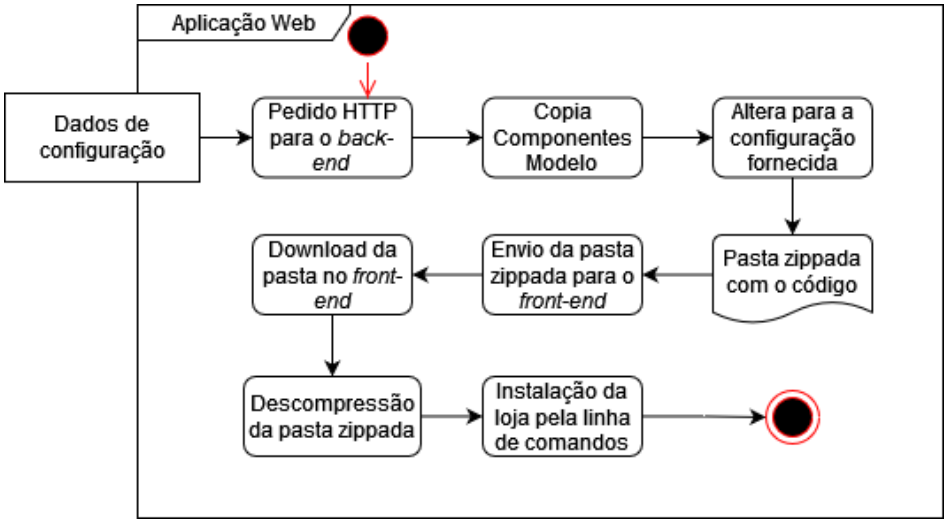


Figura 7 - Diagrama de sequência de tarefas para uso da *framework*.

4.2. Conceção do Componente “Utilizadores”

O componente “Utilizadores” é uma parte fundamental na logística de uma loja *online*, sendo que este necessita de suportar um sistema de autenticação, um sistema para gerir sessões, e funcionalidades de consulta e manutenção dos dados de cada cliente na base de dados. Com isso em mente, o componente é estruturado em três partes distintas:

a) O modelo da base de dados, onde são mapeadas as variáveis que constituem a entidade “Utilizador”.

b) O mapeamento das rotas com os diversos controladores que formam a API do servidor e que permitem executar ações sobre a informação guardada na base de dados e aplicar a lógica de negócio.

c) A interface, composta por diversos métodos disponibilizados a outros componentes, de forma a estabelecer um padrão de interação com o componente “Utilizador”, isto é, ao invés de aceder diretamente à base de dados, os componentes que requerem informação sobre um utilizador devem obter esta informação através de uma função estabelecida na interface.

Na Figura 8 é possível ver um esquema da relação entre todas as partes constituintes de um componente de *back-end*, sendo esta figura válida para todos os componentes do *back-end*.

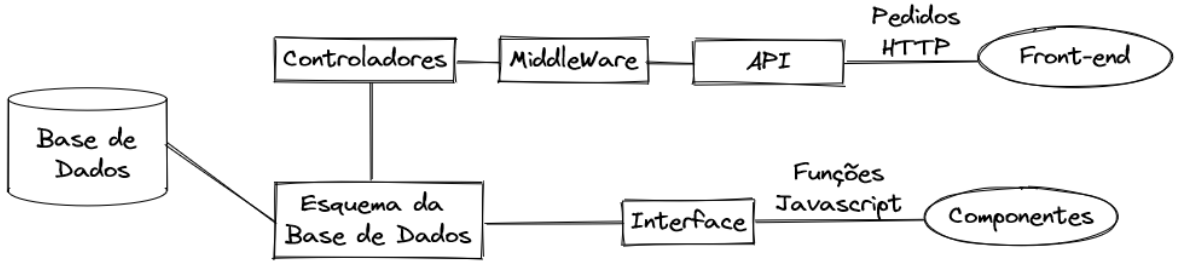


Figura 8 – Diagrama da estrutura do componente de *back-end.*

4.3. Implementação do Componente “Utilizadores”

O código do componente “Utilizadores” é constituído pelos seguintes ficheiros:

4.3.1. UserModels.js

Ficheiro com o esquema do conteúdo e do tipo de dados dos documentos que são guardados na *MongoDB* de utilizadores, incluindo regras de validação para cada campo. Este ficheiro é necessário de forma a reduzir a probabilidade de inserção de informação incorreta na base de dados.

4.3.2. UserControllers.js

Ficheiro com todas as funções de controladores que são executados quando os *endpoints* da API recebem pedidos. É nestes controladores que a lógica de negócio é definida e a manipulação da base de dados é feita. Para uso em alguns controladores, é importado o ficheiro de interface do componente "Encomendas", para poder aceder à lista de encomendas associadas a um utilizador.

4.3.3. AuthMW.js

Ficheiro com a configuração do *middleware* responsável pela autenticação dos utilizadores, isto é, verificação do tipo de utilizador e, em consequência, negar o acesso ao conteúdo caso este não possua as permissões necessárias para lhe aceder. Este *middleware* é usado em praticamente todos os outros componentes, pois estes requerem quase sempre um meio para autenticar o utilizador que envia os pedidos.

4.3.4. UserRoutes.js

Ficheiro onde os controladores são associados às diversas rotas que constituem a API, permitindo definir o *middleware* aplicado aos pedidos, conseguindo assim estabelecer quais as rotas que necessitam de permissões de acesso de administrador, ou aquelas que apenas requerem uma sessão iniciada. As rotas definidas neste ficheiro são as mesmas que as indicadas no componente de *back-end "*Utilizadores e Autenticação" da implementação da loja fictícia na secção 3.3.3.1.

4.3.5. UserInterface.js

Ficheiro com a configuração da interface deste componente, disponibilizando funções para que outros módulos do *back-end* tenham um meio de acesso aos dados dos utilizadores. Este meio de acesso pode estender-se desde simples consultas de informação sobre um utilizador, até mais complexos algoritmos para a manipulação de sessões, bem como alteração de conteúdo guardado na base de dados. Como base, são disponibilizadas funções de pesquisa de utilizadores por id, nome ou endereço de correio eletrónico.

4.4. Conceção do Componente "Categorias"

Quando se pensa na distribuição da informação numa loja *online,* uma das principais peças é, sem dúvida, a categorização. Sejam produtos, serviços ou utilizadores, um sistema de categorias permite uma melhor organização de dados, e fornece uma leve estrutura aos dados que vão sendo adicionados.

O componente "Categorias" serve o propósito de proporcionar uma forma de mapear os produtos em categorias e subcategorias, mantendo flexibilidade para adicionar e remover categorias conforme requisitado pelo utilizador. Desta forma, torna-se assim possível filtrar produtos por categorias, aumentando o nível de personalização e facilidade de identificação de produtos.

4.5. Implementação do Componente "Categorias"

A implementação do Componente "Categorias" é baseada na seguinte estrutura de código:

4.5.1. CategoryModel.js

Um ficheiro com os campos constituintes da entidade "Categoria", com regras de validação de valores inseridos e referências às categorias superiores, de forma a que o componente consiga ler os documentos da base de dados.

4.5.2. CategoryControllers.js

Um ficheiro com os controladores permitindo executar a algoritmia para listar, adicionar, remover e editar as categorias na base de dados.

4.5.3. CategoryRoutes.js

Um ficheiro com as rotas da API que dizem respeito a categorias e à sua manipulação, permitindo assim associar qual controlador é executado quando se envia um pedido para uma das rotas especificadas. As rotas definidas neste ficheiro são as mesmas que as indicadas no componente de *back-end "*Categorias" da implementação da loja fictícia na secção 3.3.3.4.

4.5.4. CategoryInterface.js

Um ficheiro onde são expostas diversas funções, com a finalidade de serem usadas por outros componentes do *back-end* para obter informação sobre categorias, formando assim uma interface. Por exemplo, se um outro componente necessitar da lista de subcategorias associada a uma categoria, ao invés de executar uma *query* à base de dados onde iria necessitar de saber o modelo de dados e os seus tipos, importa uma função desta interface necessitando apenas de fornecer o identificador da categoria da qual deseja consultar as subcategorias. Como base, são disponibilizados funções de pesquisa de categorias por *id* ou nome.

4.6. Conceção do Componente "Registo e *Login*"

A maioria das lojas *online* contêm um sistema de contas de utilizador, ou seja, cada utilizador da loja tem a ele associado um perfil. Este perfil serve para guardar dados de identificação, bem como outras informações desde históricos de encomendas, pesquisas, preferências e favoritos.

Um consumidor que consulte pela primeira vez a loja *online* não obtém automaticamente um perfil, sendo necessário por parte do utilizador se registar, escolhendo

credenciais de *login* e alguns dados identificativos. Numa futura visita, é apenas necessário fornecer as credenciais de *login* para que o utilizador tenha acesso ao seu perfil.

O componente "Registo e *Login*" pretende simplificar este processo a partir do preenchimento de formulários, sendo assim possível rapidamente gerar um perfil e registá-lo na base de dados. Em acréscimo, o componente é responsável pela gestão da sessão, isto é, o utilizador precisa apenas de inserir as suas credenciais uma vez, e por um período pré-definido de tempo. Todas as ações tomadas no site terão em conta as preferências desse utilizador. Por exemplo, ao adicionar um item como favorito ou ao carrinho, estes ficaram associados ao utilizador e disponíveis numa visita futura.

4.7. Implementação do Componente "Registo e Login"

Para a construção do componente de *front-end* "Registo e Login", foi necessário ter em conta quais os dados a pedir ao utilizador no registo de uma nova conta, a segurança na validação das credenciais, e os dados de sessão recebidos após um *login* bem-sucedido.

O componente é constituído por duas páginas, uma com o formulário de "Registo", e outra com formulário de "Login". Quando o utilizador acede ao componente, é sempre "presenteado" com o formulário de *login,* assumindo-se assim que o utilizador já criou uma conta anteriormente. Na presente implementação do componente, são pedidas como credenciais de *login,* um email e uma palavra-passe. Após a submissão das mesmas através de um pedido HTTP, é guardado na memória do *browser* o *token* recebido como resposta. É este *token* que irá servir como método de autenticação para todos os próximos pedidos HTTP feitos ao *back-end*. Na figura abaixo podemos encontrar a interface gráfica do formulário de *login* implementada.

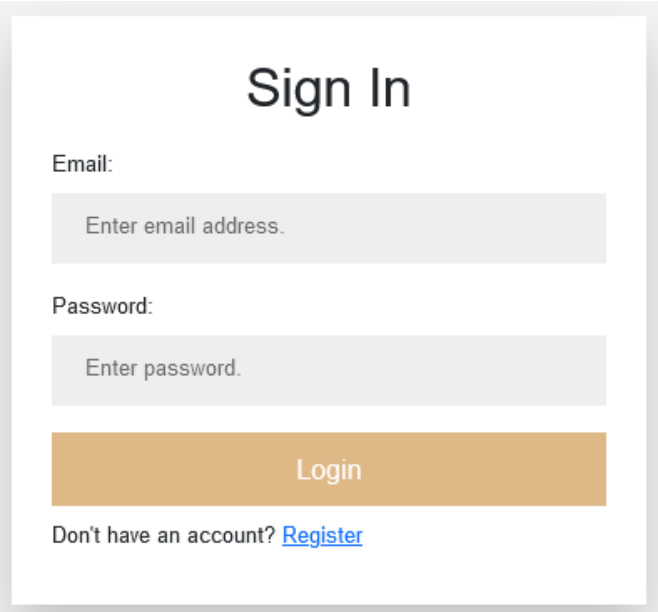


Figura 9 - Formulário de Login

Caso o utilizador nunca se tenha registado no *site*, este pode aceder rapidamente ao formulário de registo através de um *link* disponibilizado no final da página de *login*. Para que se possa proceder a um registo de utilizador, este precisa de preencher os campos obrigatórios do formulário, isto é, nome, *email* e *password*. Após a validação dos dados fornecidos pelo utilizador, estes são enviados num pedido HTTP para o servidor, onde é registado o utilizador na base de dados e criado um perfil para o mesmo. Após a confirmação do sucesso na criação do utilizador por parte do servidor, o utilizador é redirecionado para a sua página de perfil. É possível visualizar, a partir da imagem abaixo, a interface gráfica implementada para o formulário de registo.

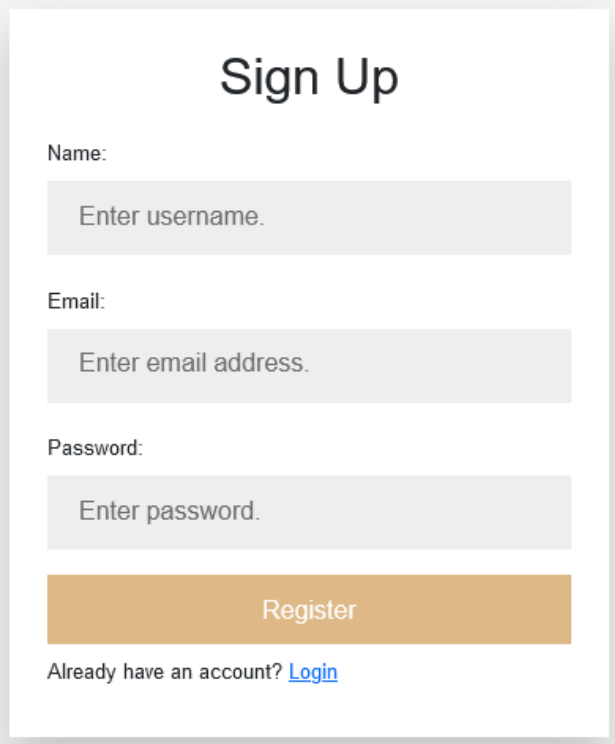


Figura 10 - Formulário de Registo

4.8. Conceção do Componente "Perfil"

Associado a uma conta existe sempre uma vasta coleção de informação, seja esta pessoal ou apenas dados estatísticos. A capacidade de visualizar tal informação de forma centralizada e bem estruturada é bastante desejável. Para tal finalidade, é usado o componente "Perfil", sendo este constituído por uma página onde são expostos dados pessoais, historial de compras e lista de itens favoritos.

4.9. Implementação do Componente "Perfil"

Na implementação do componente de *front-end* "Perfil", este é desenhado como uma única página com diversas secções para os diferentes dados a apresentar. A página consiste num conjunto de campos de texto onde é apresentada a informação pessoal associada ao cliente, como o nome, a morada, o número fiscal, número de contacto, entre outros. Também na mesma página, é possível encontrar uma tabela com uma lista das diversas encomendas executadas pelo cliente, estejam estas a decorrer ou já concluídas. Na figura 11, está representada interface da página de perfil.

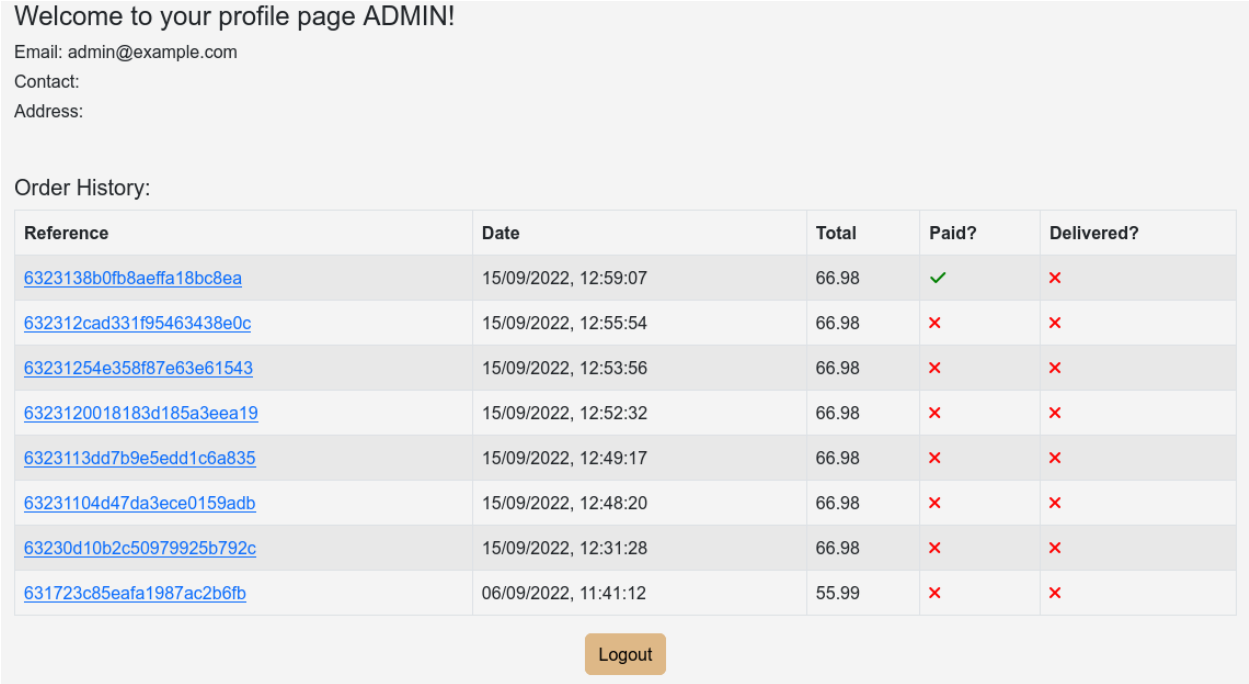


Figura 11 - Página de Perfil

4.10. Integração de Novos Componentes *Back-end* na Framework

Foi mencionado na introdução deste capítulo que o desenvolvimento da aplicação de criação de lojas tem uma abordagem modular, por forma a permitir a introdução ou remoção de componentes sem comprometer o funcionamento correto de toda a loja. Para controlar e configurar a interação entre os componentes implementados e componentes que poderão ser integrados futuramente, é necessário desenvolver o código que define quais os componentes que constituem a *framework*, onde estes estão localizados, entre outras configurações. O código correspondente a essa configuração encontra-se no ficheiro *server.js,* que é constituinte da *framework* e tem a função de ser o ponto de partida na execução do servidor da loja gerada.

4.10.1. Server.js

Um ficheiro no *back-end*, que serve de ponto de arranque do servidor, sendo este responsável por iniciar uma ligação à base de dados *MongoDB*, carregar os ficheiros de cada componente, ativar os seus *endpoints*, e ficar à escuta de pedidos HTTP vindos do *front-end*. É neste ficheiro que o programador, aquando da integração de um novo componente de *back-end*, irá manualmente configurar novos *endpoints*,

bem como, definir a localização dos ficheiros que compõem o componente que deseja adicionar. Por outras palavras, são importados os ficheiros *javascript* que contém as rotas, os controladores, o modelo da base de dados e a interface do novo componente, e é configurado através de funções da biblioteca *ExpressJS,* os URI que constituem a API do novo componente.

4.11. Integração de Novos Componentes *Front-end* na Framework

A integração de componentes no *front-end* é uma tarefa mais complexa e com algumas restrições, para que seja respeitado o método de desenvolvimento em *React, ou seja,* uso de componentes auto-contidos. Sendo considerados apenas os componentes de topo, estes têm de ser desenvolvidos a pensar na compatibilidade com a biblioteca *Redux* de gestão do estado da loja e dos dados do formulário guardados. Com esta premissa, a introdução de um componente na *framework* é conseguida através da adição do código do componente *React* onde está descrito o aspeto e comportamento do mesmo e da edição dos ficheiros de configuração da *Redux* para acomodar as diversas maneiras como o novo componente irá alterar o estado da loja. Para a total integração deste novo componente, é necessário que o nome do mesmo seja adicionado ao do componente *React* do qual este é subcomponente.

4.12. Aplicação de Criação de Lojas

A aplicação de criação de lojas gera os ficheiros de código necessários para que se possa proceder à implementação de uma loja *online* e mantê-la em serviço. A aplicação consiste num formulário *web* que recolhe, a partir do utilizador, os dados necessários para produzir corretamente os ficheiros de código. Estes dados podem variar desde simples frases como, por exemplo, o nome da loja, ou estruturas mais complexas como a lista de categorias pelos quais os produtos se estruturam. Na figura abaixo, é apresentada a interface da aplicação de criar lojas *online*, estando o formulário já preenchido com dados de exemplo.

Figura 12 - Formulário para a criação da loja

A utilização da aplicação desenvolvida começa pela recolha dos dados de configuração da loja, através do formulário de criação de loja. Após a preenchimento do referido formulário, a informação é enviada para o servidor, onde é validada e processada, sendo que os ficheiros de código são configurados em concordância com o pedido do cliente. Após a criação da pasta comprimida contendo todos os ficheiros gerados pela *framework* na implementação de uma loja *online*, é enviada para o *front-end* uma mensagem de sucesso com o URI da pasta comprimida. A partir desse momento, fica disponível para *download* a pasta com os ficheiros de código correspondentes aos componentes, quer de *back-end,* quer de *front-end*, bem como, o ficheiro *server.js,* um ficheiro com os parâmetros de configuração fornecidos no formulário inicial e um ficheiro de texto com instruções de como instalar a loja *online* através da linha de comandos.

4.13. Integração em Uma Loja Existente

Após a utilização da aplicação para gerar os ficheiros de código da loja *online*, como forma de demonstrar que os componentes criados são funcionais, estes foram integrados na loja *online* fictícia criada anteriormente, substituindo os já existentes.

Foram retiradas as duas pastas que continham o código dos componentes de *back-end,* nomeadamente "Users" e "Categories", e copiado para o seu lugar, os ficheiros criados pela *framework* correspondentes aos componentes "Utilizadores" e "Categorias", respetivamente. No ficheiro *server.js* da loja fictícia foram mudados os nomes e as localizações para corresponderem aos novos ficheiros a serem executados. Também no *front-end* foram eliminados os ficheiros correspondentes às páginas de perfil, formulário de *login* e formulário de registo, sendo colocados no seu lugar os ficheiros de código criados pela *framework.* Devido à nomenclatura dos ficheiros ser idêntica, não houve necessidade de mudar as referências aos mesmos. Após esta alteração de código na implementação da loja fictícia, a execução da mesma e a sua utilização em nada se alteraram.

4.14. Resumo

Neste capítulo foi explorado todo o desenvolvimento da *framework*, incluindo a sua arquitetura geral, o seu comportamento, os componentes implementados, bem como, a estratégia para a introdução de novos componentes na *framework* que se desenvolveu. Foi também abordada a implementação da aplicação de criação de lojas, demonstrando a sua interação com a *framework* e a interface gráfica concebida para a recolha dos parâmetros de configuração. Tendo em conta os assuntos abordados ao longo do presente capítulo, é possível concluir que os objetivos iniciais foram cumpridos.

5. CONCLUSÃO

O presente capítulo tem como intuito a apresentação da conclusão deste trabalho, bem como as limitações do mesmo e sugestões para trabalhos futuros dentro do mesmo tema desta dissertação.

5.1. Considerações Finais

Com o aumento de negócios e empresas a acolher o *e-commerce*, surgiu a necessidade de simplificar e agilizar o processo de criação de lojas *online* através de uma plataforma. Tendo em conta esta premissa, o intuito da presente dissertação consistiu na elaboração de uma plataforma que providenciasse a um programador a possibilidade de gerar, de forma rápida e simples, uma loja *online*, eliminando a necessidade do programador investir tempo a desenvolver todo um *back-end* e *front-end*. Para cumprir com esse objetivo, foi desenvolvida uma aplicação *web* que, mediante um conjunto de parâmetros inseridos num formulário, entrega ao utilizador o código gerado da loja *online*.

Para o desenvolvimento com sucesso da aplicação para criação de lojas *online*, a primeira etapa desta dissertação consistiu na construção de uma loja fictícia de forma a perceber os requisitos e opções de configuração presentes na estrutura de uma loja *online.* Após o levantamento dos requisitos, desenvolveu-se então uma aplicação *web,* onde um programador pode especificar através de um formulário a parametrização sobre a qual a ferramenta se deve basear ao criar uma loja à medida.

Ambas as etapas desta dissertação provaram ser um desafio, não só devido ao amplo leque de tecnologias e linguagens usadas, mas também devido à complexidade e extensão das soluções programadas. Foi tido em conta o cuidado de desenvolver código legível, portável e versátil, tal como discutido na secção 2.1.3, no parágrafo sobre aspetos técnicos de um *website.*

A escolha do ambiente de execução *Node* para desenvolver o *back-end* proporcionou grande flexibilidade na programação e na implementação da solução idealizada. Com um vasto catálogo de bibliotecas de terceiros, dedicadas a facilitar a integração com outras

soluções, como foi caso da biblioteca *Mongoose,* usada por proporcionar métodos para criar e manipular uma base de dados em *Mongo*. A biblioteca *Express* permitiu construir e configurar, de forma muito dinâmica, todo os aspetos relacionados com a lógica do servidor HTTP. Como, por exemplo, fornecer uma lista extensa de funções auxiliares para mapear rotas, aplicar *middleware* e construir respostas aos pedidos HTTP recebidos.

A base de dados desenvolvida com a tecnologia *MongoDB*, apresentou diversas vantagens, das quais destaco a velocidade nas *queries* feitas aos documentos guardados. A escolha de uma base de dados não-relacional, permitiu versatilidade na hora de modelar os diversos tipos de produtos num único documento.

O uso de *React* para criar o *front-end* possibilitou um desenvolvimento mais eficaz e fluído, tomando partido da estrutura em componentes e através reutilização dos mesmos em toda a interface da loja. A solução implementada para o controlo do estado da loja foi desenvolvida com ajuda da biblioteca *Redux,* beneficiando das ferramentas oferecidas pela mesma para manipular os dados apresentados em diferentes partes do *front-end.* Apesar desta mais-valia, o facto do *Redux* controlar todo o estado da loja apresentou-se como um entrave à isolação do código dos componentes a gerar pela *framework* no *front-end.* Mesmo assim, manteve-se o uso desta biblioteca pois esta reduz o número de vezes que a informação a apresentar ao utilizador é requerida ao *back-end.*

Por fim, fazendo uma análise à aplicação desenvolvida, pode-se concluir que esta representa um sistema funcional e prático que cumpre com o objetivo de fornecer uma forma rápida de criar uma loja *online.* Podemos também concluir que o levantamento de parâmetros foi realizado com sucesso, e que estes são transversais a todas as lojas. Posto isto, é possível afirmar que os objetivos propostos para esta dissertação foram executados com sucesso.

5.2. Limitações e Sugestões de Trabalho Futuro

Durante a realização desta dissertação foram identificadas algumas limitações que podem, no entanto, servir como estímulo para futuros trabalhos.

Uma das limitações da presente dissertação é a falta de uma lista mais vasta de opções para o *design* do website, criado pela aplicação. Apesar do alvo de utilizadores da aplicação serem programadores, seria uma mais-valia, aquando do preenchimento do formulário *web*,

a possibilidade de configurar de uma palete de cores a ser usada no *website*, ou escolha de um logótipo e imagem de fundo. Estes parâmetros fariam sentido serem especificados na geração da loja visto serem únicos a cada *website*. Sendo assim, uma sugestão para trabalhos futuros é a integração de mais elementos de *design* do website no processo de criação de lojas *online*.

Uma outra limitação deste trabalho está relacionada com o método de desenvolvimento em *React*. Já por si, este método de desenvolvimento é modular, mas a lógica de controlo do estado do *website*, seja usando as ferramentas nativas ao *React* ou uma biblioteca externa, é influenciada por diferentes componentes em diferentes instantes de tempo. Uma melhoria a considerar com sugestão para trabalhos futuros seria a maior segmentação do código na implementação do *front-end*, onde cada componente seria restrito a uma coleção de funções, através das quais poderia alterar estado da loja, de maneira predefinida.

Por último, uma outra condicionante deste trabalho reside no facto de que, tanto a aplicação como as lojas criadas, estão limitadas a um único idioma, sendo este o Inglês. Como recomendação para trabalhos futuros, sugere-se a implementação na *framework* de um sistema de suporte para criação lojas em múltiplos idiomas. Esta seria uma adição interessante para tornar a aplicação mais abrangente e acessível.

Referências

Abdullah, E. N., Ahmad, S., Ismail, M., & Diah, N. M. (2021, July). Evaluating E-commerce Website Content Management System in Assisting Usability Issues. In *2021 IEEE Symposium on Industrial Electronics & Applications (ISIEA)* (pp. 1-6). IEEE.

Afolabi, A. O., Oluwatobi, S., Emebo, O., Misra, S., & Garg, L. (2022). Evaluation of the Merits and Demerits Associated with a DIY Web-Based Platform for e-commerce Entrepreneurs. In *International Conference on Information Systems and Management Science* (pp. 214-227). Springer, Cham.

Banks, A., & Porcello, E. (2017). *Learning React: functional web development with React and Redux*. " O'Reilly Media, Inc.".

BuiltWith, "*eCommerce technologies Web Usage Distribution on the Entire Internet*.", 2023. [Online]. Disponível em: https://trends.builtwith.com/shop/traffic/Entire-Internet [Acedido: Jan. 06, 2023]

BuildWith, "*PrestaShop Usage Statistics*.", 2023. [Online]. Disponível em: https://trends.builtwith.com/shop/PrestaShop [Acedido: Jan. 06, 2023]

dos Santos, V. F., Sabino, L. R., Morais, G. M., & Gonçalves, C. A. (2017). E-commerce: A short history follow-up on possible trends. *International Journal of Business Administration*, *8*(7), 130-138.

Dragomirov, N. (2020). E-Commerce Platforms and Supply Chain Management–Functionalities Study. *Economic Alternatives*, *2*, 250-261.

Hoque, S. (2020). *Full-Stack React Projects: Learn MERN stack development by building modern web apps using MongoDB, Express, React, and Node. js*. Packt Publishing Ltd.

Kalia, P., Kaur, N., & Singh, T. (2017). Consumer satisfaction in e-shopping: An overview. *Indian Journal of Economics and Development*, *13*, 569-576.Khoshnampour & Nosrati (2011)

Khan, A. G. (2016). Electronic commerce: A study on benefits and challenges in an emerging economy. Global Journal of Management and Business Research.

Khurana, A. (2019). Introduction To E-Commerce. Disponível em: http://www.ddegjust.ac.in/studymaterial/mcom/mc-201.pdf

Kotian, H., & Meshram, B. B. (2017, January). A framework for quality management of e-commerce websites. In *2017 International Conference on Nascent Technologies in Engineering (ICNTE)* (pp. 1-6). IEEE.

Lin, H. F. (2007). The impact of website quality dimensions on customer satisfaction in the B2C e-commerce context. *Total Quality Management and Business Excellence*, *18*(4), 363-378.

Lixandroiu, R., & Maican, C. (2015). An Analysis On Choosing A Proper Ecommerce Platform. *Risk in Contemporary Economy*, 54-59.

Nanehkaran, Y. A. (2013). An introduction to electronic commerce. *International journal of scientific & technology research*, *2*(4), 190-193.

Niranjanamurthy, M., Kavyashree, N., Jagannath, S., & Chahar, D. (2013). Analysis of e-commerce and m-commerce: advantages, limitations and security issues. *International Journal of Advanced Research in Computer and Communication Engineering*, *2*(6), 2360-2370.

PrestaShop, "*Create and build your online business with Prestashop*.", 2023. [Online]. Available: https://www.prestashop.com/pt/sobre-nos [Acedido: Jan. 06, 2023]

PrestaShop, "*PrestaShop Addons Markplace*.", 2023. [Online]. Disponível em: https://addons.prestashop.com/pt/ [Acedido: Jan. 06, 2023]

Sant'ana, M. P., Silva, G. C., Diogo, N. G., & Nose, E. T. (2021). Comportamento do Consumidor durante a pandemia da COVID-19. *REPAE-Revista de Ensino e Pesquisa em Administração e Engenharia*, *7*(2), 54-69.

Shopify.com, "*What Is Shopify and How Does It Work*?", 2023. [Online]. Disponível em: https://www.shopify.com/blog/what-is-shopify [Acedido: Jan. 06, 2023]

Shopify.com, "*Shopify App Store*.", 2023. [Online]. Disponível em: https://trends.builtwith.com/shop/traffic/Entire-Internet [Acedido: Jan. 06, 2023]

Taher, G. (2021). E-commerce: advantages and limitations. International Journal of Academic Research in Accounting Finance and Management Sciences, 11(1), 153-165.

ThemeForest, “*Shopify Themes - Shopify Templates*.”, 2023. [Online]. Disponível em: https://themeforest.net/category/ecommerce/shopify [Acedido: Jan. 06, 2023]

Wix.com, “*Criar loja virtual | Plataforma de eCommerce | Wix.com*.”, 2023. [Online]. Disponível em: https://pt.wix.com/ecommerce/loja-virtual [Acedido: Jan. 06, 2023]

Wix.com, “*Dicas de eCommerce*.”, 2023. [Online]. Disponível em: https://pt.wix.com/blog/category/dicas-de-ecommerce [Acedido: Jan. 06, 2023]

Wix.com, “*Wix App Market | Aplicativos para o seu site | Wix.com*.”, 2023. [Online]. Disponível em: https://pt.wix.com/app-market [Acedido: Jan. 06, 2023]

WooCommerce, “*WooCommerce*.”, 2022. [Online]. Disponível em: https://pt.wix.com/app-market [Acedido: Nov. 21, 2022]