Reconstrução de Plataforma E-Commerce

Mestrado em Engenharia Informática - Computação Móvel

Cecília Silva Santos

Estágio realizado sob a orientação do Professor Doutor Marco Monteiro e sob supervisão do Engenheiro Marco Ramos.

Leiria, maio de 2020

Originalidade e Direitos de Autor

O presente relatório de estágio é original, elaborado unicamente para este fim, tendo sido devidamente citados todos os autores cujos estudos e publicações contribuíram para o elaborar.

Agradecimentos

Antes da apresentação deste projeto, não posso deixar de agradecer a algumas pessoas que direta ou indiretamente contribuíram para o sucesso do mesmo.

Em primeiro lugar gostaria de agradecer ao professor Carlos Grilo (ex coordenador do Mestrado de Engenharia Informática - Computação Móvel) pela prontidão no esclarecimento de dúvidas relativas ao relatório de estágio ao longo deste ano.

Posteriormente, gostaria de agradecer ao professor Marco Monteiro, orientador atribuído pela Escola Superior de Tecnologia e Gestão para a elaboração deste projeto, pelas dicas e melhorias sugeridas ao longo do mesmo.

Deixo também um agradecimento especial aos elementos da empresa onde decorreu o estágio, a ZENN – Websolutions, que foram incansáveis no que toca a esclarecimento de dúvidas e ajuda contínua e em especial ao Engenheiro Informático Marco Ramos que foi o orientador atribuído pela empresa para a realização deste projeto.

Gostaria de agradecer também à minha amiga Ana Filipa de Jesus, pelo tempo dedicado às orientações de português e revisões neste presente relatório, o meu bem-haja.

Por fim e não menos importante, agradeço também à família que sempre me apoiou e estive presente em todos os momentos, bons e maus, dando força para continuar este percurso.

Resumo

O presente relatório descreve as tarefas realizadas ao longo do estágio que decorreu nas instalações da ZENN – web solutions, em que os conhecimentos adquiridos ao longo do Mestrado de Engenharia Informática - Computação Móvel, nomeadamente no que diz respeito à utilização de ferramentas de gestão de projetos, gestão de código e conhecimentos adquiridos a nível de desenvolvimento de projeto, entre outros, foram cruciais para esta etapa. O estágio prolongou-se durante 9 meses e as tarefas de desenvolvimento foram todas relacionadas com a reconstrução de uma plataforma *e-commerce* já existente.

Ao longo dos anos e com a evolução da tecnologia surgiu a necessidade de atualização das tecnologias que outrora foram importantes, surgiram novas regras, legislações e tendências de lojas *online* que requerem uma constante evolução seja em termos de performance como em funcionalidades.

A antiga plataforma *e-commerce* tinha alguns problemas a nível de performance e uma grande necessidade de ser reestruturada, visto não estar a corresponder à exigência dos clientes e haver uma descontinuação da tecnologia utilizada para a gestão de base de dados, forçando assim a sua atualização. Nesta nova reestruturação salientam-se algumas novas funcionalidades: a gestão de entidades, a gestão de versões de produtos, a gestão do carrinho de compras e envio de emails.

Para construção desta nova plataforma, analisou-se em primeiro todas as funcionalidades da antiga, aproveitando as que se consideraram adequadas e implementando novas para ir de acordo às expectativas dos clientes. Ao longo do seu desenvolvimento foram feitos vários tipos de testes para garantir o bom funcionamento. Além da reestruturação da plataforma também foi implementada uma nova interface que teve como base um *template* adquirido para o efeito e adaptado para seguir os padrões de usabilidade e acompanhamento tecnológico.

Palavras-chave: *E-commerce*, NHibernate, Telerik, ASP.NET, Vue.js, Lojas *online*.

Abstract

The following report describes the tasks performed during the internship that took place at ZENN's – web solutions facilities, in which the knowledge acquired during the Master in Computer Engineering – Mobile Computing, namely about the use of project management tools, management code and knowledge acquired at the project development level, among others, were crucial for this stage. The internship lasted for 9 months and the development tasks were all related to the reconstruction of an existing e-commerce platform.

Over the years and with the evolution of technology, there was a need to update the technologies that were once important, new rules, legislation and trends for online stores have emerged that require constant evolution both in terms of performance and features.

The old e-commerce platform has some problems in terms of performance and had a major necessity of being restructured, as it is not meeting customer requirements and there was a discontinuation of the technology used for the database management, thus forcing its update. In this new restructuring, some new features are highlighted, such as entity management, product version management, shopping cart management, and email sending.

To build this new platform, all the functionalities of the previous one were first analyzed, taking advantage of those that were considered adequate and implementing new ones to go according to the customers' expectations. Throughout its development, various types of tests were carried out to ensure proper functioning. In addition to the restructuring of the platform, a new interface was also implemented, which was based on a template acquired for the purpose and adapted to follow the standards of usability and technological monitoring.

Keywords: E-commerce, NHibernate, Telerik, ASP.NET, Vue.js, Online Stores.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Siglas e Acrónimos

B2B	Business to Business
B2C	Business to Consumer
B2E	Business to Employee
B2G	Business to Government
B2M	Business to Manager
BD	Base de dados
C2B	Consumer to Business
C2C	Consumer to Consumer
CEO	Chief Executive Officer
CFO	Chief Financial Officer
CRUD	Create Read Update Delete
CSS	Cascading Style Sheet
CTO	Chief Techonology Officer
ERP	Enterprise Resource Planning
ESTG	Escola Superior de Tecnologia e Gestão
G2B	Government to Business
G2C	Government to Citizen
G2E	Government to Employess
G2G	Government to Government
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
JWT	JSON Web Tokens
MVC	Model View Controller
ORM	Object-Relational Mapping
P2P	Peer to Peer
PPC	Pay Per Click
REST	Representational State Transfer
SEO	Search Engine Optimization
SCSS	Sassy Cascading Style Sheet
SQL	Structured Query Language

SWOT	Strenghts Weaknesses Opportunities Threats
UI	User Interface
VPS	Virtual Private Server
XML	Extensible Markup Language

1. Introdução

Com a evolução da Internet surgiu também uma nova maneira de vender produtos: o *e-commerce*. Este funciona como uma plataforma que permite de forma fácil e simples a compra e venda de produtos *online*, em que o cliente só se preocupa em carregar os seus produtos no *website*, decidir quais os métodos de pagamento a adotar e fazer a sua própria gestão de encomendas. Esta é uma forma simples e rápida e com toda a logística que as lojas *online* exigem. [1]

O presente relatório descreve o trabalho desenvolvido no estágio que decorreu na empresa ZENN - web solutions entre 3 de Junho de 2019 e 31 de Março de 2020, no contexto do qual integrei na equipa de desenvolvimento responsável pela reconstrução de uma plataforma *e-commerce* não só devido à tecnologia de mapeamento de dados ter sido descontinuada, mas também à necessidade de evolução tecnológica, otimização, performance e criação de novas funcionalidades requisitadas pelos clientes. Esta reconstrução foi feita por um grupo de três pessoas incorporando a metodologia *scrumban*.

O relatório divide-se em cinco capítulos, incluindo este capítulo de introdução.

No segundo capítulo (Estágio / A ZENN – web solutions) encontra-se a descrição da empresa ZENN – web solutions no qual decorreu o estágio durante 9 meses e o plano de trabalho delineado. Além disso, também são descritas as metodologias e processos utilizados no desenvolvimento da plataforma *e-commerce,* bem como a realização de outros projetos que foram efetuados em simultâneo com o desenvolvimento da plataforma. Este capítulo finaliza-se com a análise SWOT (*Strengths Weaknesses Opportunities Threats*) e crítica relativamente à empresa.

No terceiro capítulo (Plataforma *e-commerce*), dá-se a conhecer o que é o *e-commerce* e o motivo de reestruturação da plataforma de *e-commerce* da ZENN. Comparam-se ainda outras plataformas idênticas, salientando-se também as suas diferenças.

O quarto capítulo (Projeto) aborda a arquitetura do sistema e de ambas as plataformas - a antiga e nova. Aqui está descrito todo o processo de análise de base de dados e de acesso aos dados bem como as diferenças entre as plataformas em termos de tecnologias e funcionalidades. Ainda neste capítulo encontra-se o processo de desenvolvimento da nova

interface da plataforma e comparações com a antiga, além de todos os testes que foram realizados. Por fim neste capítulo encontra-se uma análise crítica relativamente ao projeto e às propostas de melhoria do mesmo.

Por último e não menos importante, encontra-se a conclusão deste relatório com pontos chaves referentes ao desenvolvimento deste projeto da plataforma *e-commerce*.

2. Estágio / A ZENN – web solutions

Este capítulo divide-se em 5 secções: empresa, plano de estágio, metodologias e processos utilizados, outros projetos extra implementados no decurso do estágio e a análise SWOT e crítica relativamente à empresa.

Na secção 2.1 encontram-se descritas as áreas de atuação da empresa bem como a missão, visão e valores da mesma, a sua atividade e volume de negócio, entre outros.

A secção 2.2 diz respeito ao plano de estágio planeado para 9 meses com o intuito de reconstruir uma plataforma *e-commerce*.

Relativamente às metodologias e processos utilizados pela empresa na gestão de projetos, seja a nível de documentação, gestão de tarefas, gestão de código ou comunicação, podem ser encontradas na secção 2.3.

Além da reconstrução da plataforma *e-commerce*, ao longo do estágio surgiram também outros projetos envolvendo as mesmas tecnologias, estes encontram-se descritos na secção 2.4.

Na última secção deste capítulo (secção 2.5) encontra-se a análise SWOT da empresa bem como uma análise critica sobre a mesma.

2.1. A empresa

O estágio decorreu durante 9 meses, de junho de 2019 a março de 2020 nas instalações da ZENN – web solutions, empresa cujo nome de registo se intitula como Havidanainternet, Lda.

A ZENN é uma empresa situada em Leiria, com mais de 15 anos de experiência na área da transformação digital. Inicialmente o foco da sua atividade era *design*, com uma evolução ao longo dos anos para *web design*. Atualmente foca-se em desenvolvimento web e *marketing* digital. A empresa trabalha lado a lado com os seus clientes de forma a entregar a melhor solução nas seguintes áreas:

- *Websites – Designs* feitos à medida, desenvolvimento adaptado a dispositivos móveis, otimizações para motores de busca, usabilidade e *user experience* e gestão simples de conteúdos;

- *Marketing* digital – *Inbound Marketing,* com produção de conteúdos, S*earch Engine Optimization* (SEO) otimizado, gestão de redes sociais, campanhas *pay-per-click* (PPC) e email *marketing*;
- E-commerce – *Design* único e exclusivo, otimizado para motores de busca com gestão de produtos de forma simples, integrações de produtos com *Entreprise Resource Planning* (ERP), gestão de métodos de pagamento e *dashboards* de estatísticas;
- Aplicações Web – Gestão de projetos e de áreas reservadas, plataforma de *e-learning* e leilões e *software* desenvolvimento à medida;
- *Branding* digital – Apresentações digitais, vídeo-*marketing*, infográfico, fotografia digital e webvídeo.
- KOONNECT – Soluções diversas adequadas a diferentes necessidades, seja extranet ou intranet, *e-commerce business-to-business* (B2B) ou *business-to-consumer* (B2C), entre outros, recentemente reconhecidos no *Web Summit* com ênfase na plataforma de *e-learning* com uma *startup ALPHA*.

A ZENN – web solutions foca-se em soluções digitais para as empresas de modo a facilitar a gestão de clientes e a ajudar no crescimento dos seus clientes e parceiros. Nesta secção são apresentadas a visão, missão e valores da empresa bem como o seu organigrama através da Figura 1.

A Missão da ZENN é apoiar a transformação e afirmação digital de empresas e negócios através da criação de soluções informáticas únicas que facilitem o crescimento de clientes e parceiros.

Em termos de Visão, a ZENN visa ser uma referência nacional de excelência e inovação tecnológica com contributo de valor para os seus colaboradores, clientes e parceiros.

Os Valores fundamentais da empresa são: excelência, qualidade, compromisso, criatividade, inovação, respeito, trabalho de equipa, superação, foco na satisfação do cliente, proximidade, respeito pelo meio ambiente e dedicação. [2]

Organigrama

A organização da empresa é apresentada no organigrama da Figura 1:

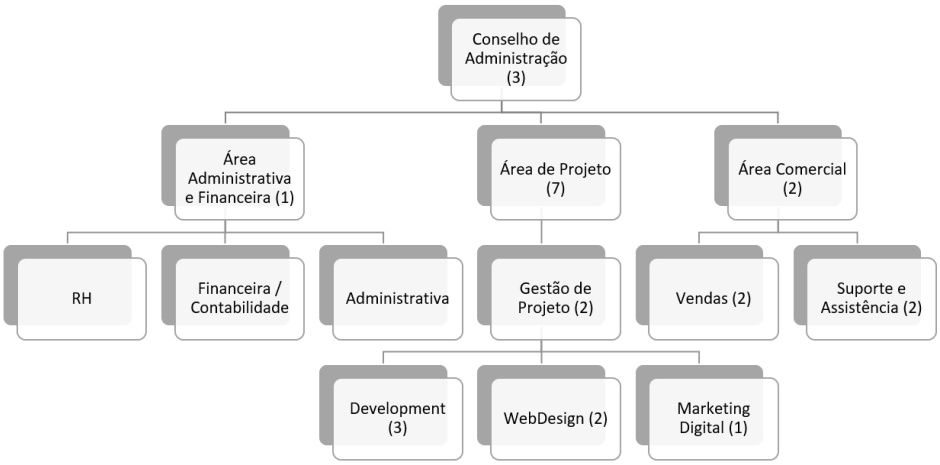


Figura 1 - Organigrama funcional da empresa [2]

Recursos humanos

A ZENN é composta por nove colaboradores formados em Engenharia Informática, *Design* e *Marketing*:

- Sérgio Cabecinhas, *Chief Executive Officer* (CEO) e fundador da ZENN – web solutions, responsável pela área de *marketing* digital e *design*;
- Rui Duarte, *Chief Financial Officer* (CFO), responsável pela área financeira e recursos humanos;
- Marco Ramos, *Chief Techonology Officer* (CTO), programador e responsável pela equipa de programação;
- Cecília Santos, *Web Developer;*
- José Oliveira, *Web Developer;*
- Joana Bento, *Web Developer;*
- Sónia Gomes, *Digital Marketeer;*
- Carolina Sousa, *Digital Designer;*
- Mariana Louro, *Digital Designer.*

Na

Figura 2 e Figura 3 encontram-se representados dois gráficos: um referente à administração

da empresa e o outro às quotas de cada um na empresa. Na Figura 3 é possível verificar-se a distribuição do capital social da empresa, sendo o CEO o sócio maioritário (com 60%) e os CTO e CFO com a restante parte repartida pelos dois (20% cada um).

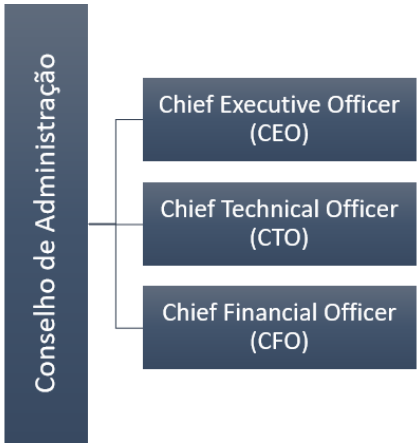


Figura 2 - Orgãos de gestão da empresa [2]

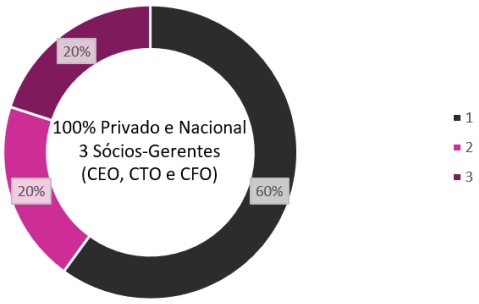


Figura 3 - Distribuição do capital social [2]

Atividade e volume de negócio

Os clientes da empresa são de diversos ramos de atividade sendo as áreas mais relevantes o comércio e retalho e as indústrias transformadoras. Entre as restantes, pode-se encontrar a sua presença no alojamento e restauração, atividades financeiras e seguros e também no setor da educação, como representado graficamente na Figura 4.

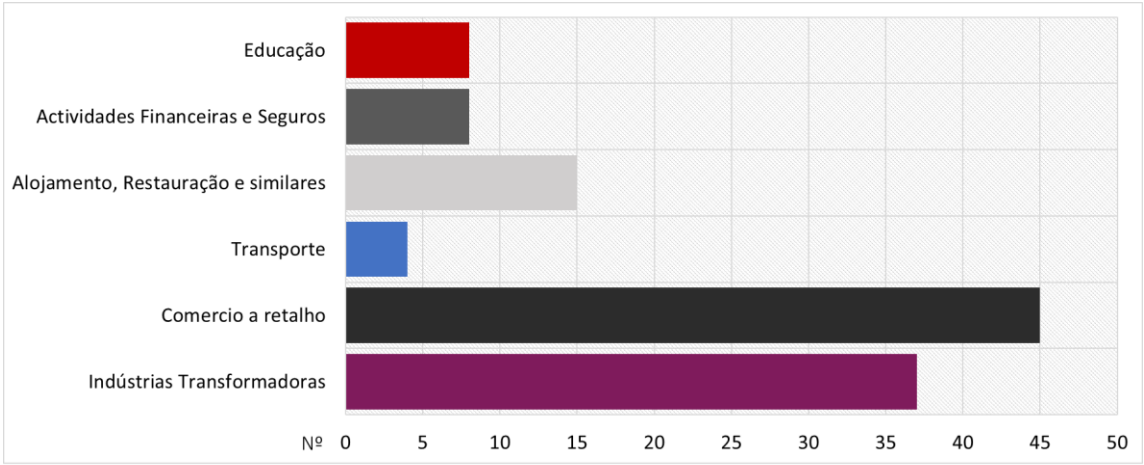


Figura 4 – Número de clientes por setor de atividade [2]

A área geográfica de clientes está distribuída entre vários países, mas assenta essencialmente em Portugal, nomeadamente na zona de Leiria, Marinha Grande e Lisboa.

Através da Figura 6 consegue-se ter uma melhor perceção do alcance geográfico da empresa no que toca aos seus clientes.

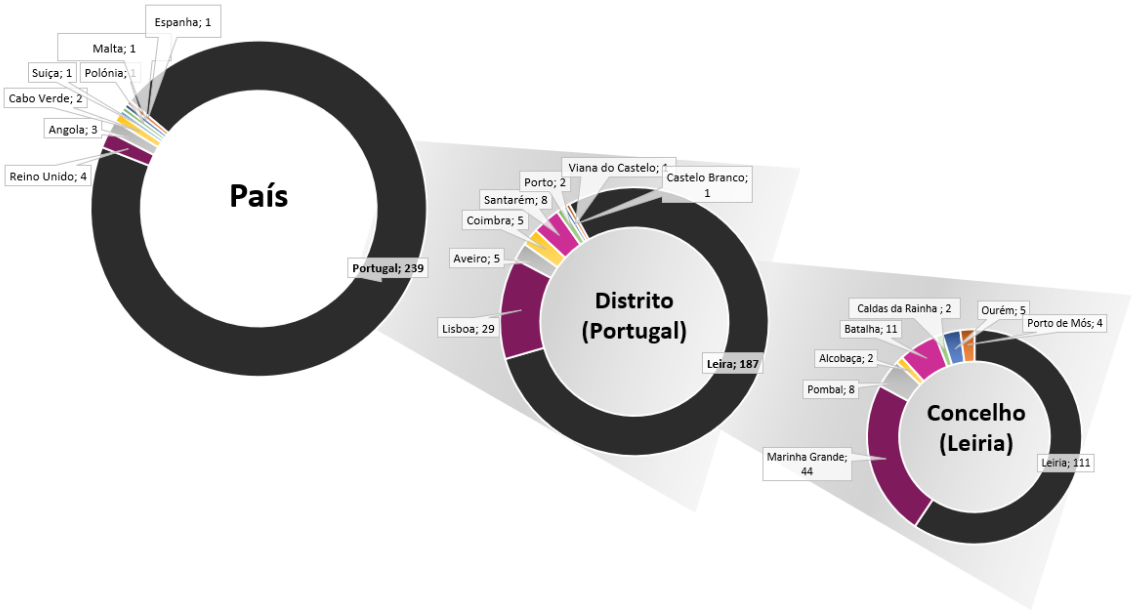


Figura 5 - Distribuição geográfica de clientes [2]

Em termos de volume de negócio a ZENN - web solutions tem evoluído ao longo dos anos essencialmente nas áreas de serviço de desenvolvimento web, *e-commerce* e *marketing* digital e *web design* (Figura 6 e Figura 7).

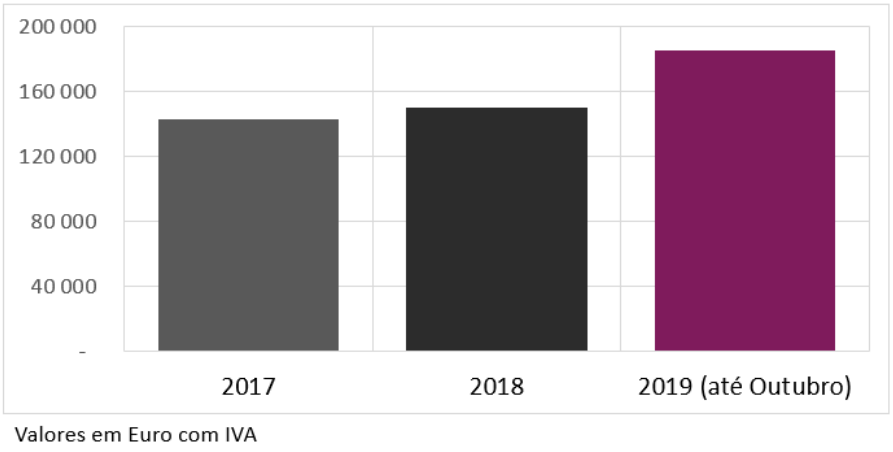


Figura 6 - Volume de negócio anual

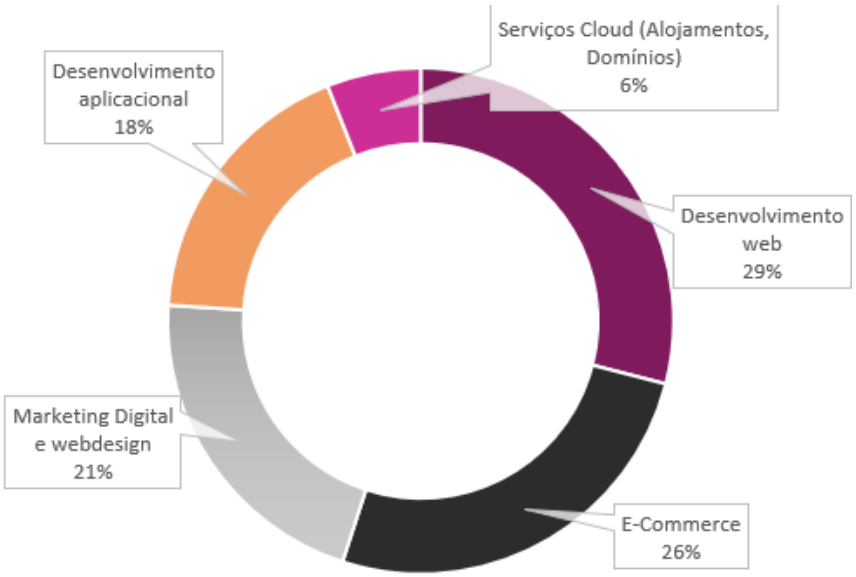


Figura 7 - Volume de negócio - áreas de serviço

2.2. **Plano de estágio**

Todas as decisões relacionadas com a arquitetura, tecnologias e metodologias de desenvolvimento da nova plataforma de *e-commerce* foram responsabilidade da empresa. A plataforma foi desenvolvida por 3 pessoas às quais foram repartidas as várias tarefas para a implementação da mesma.

Duas das tarefas descritas no plano de estágio (ver Tabela 1) foram posteriormente atribuída a outro elemento da equipa devido a serem funcionalidades urgentes e o tempo ter-se esgotado entre outras funcionalidades igualmente importantes. Como tal, as tarefas de API de autenticação e API de autorização de dados, não se encontram descritas no presente relatório.

Na Tabela 1 encontra-se representado o plano de trabalho para os nove meses de estágio, com o início em junho de 2019 e data de término em finais de março de 2020. Contudo a tabela também apresenta um campo relativo ao tempo real de execução das tarefas, pois algumas tarefas como testes unitários e API´s de autenticação e de autorização acabaram por demorar mais tempo do que o suposto devido ao surgimento de projetos paralelos que provocaram o atraso na continuação do processo de desenvolvimento. Para colmatar a situação de modo a conseguir cumprir os prazos, alocaram-se alguns recursos humanos extra ao projeto que estiveram maioritariamente focados nas API's. Todo o restante projeto correu

consoante o plano de estágio e ainda se conseguiu testar a implementação em ambientes reais.

Tabela 1 - Plano de trabalho de estágio

Plano de Trabalho (9 meses)			
Itens de ação	Data início (estimado)	Data fim (estimado)	Tempo real (conclusão)
- Análise da base de dados (BD) - Construção de um novo modelo de BD - Reconstrução de Tabelas para suporte de novas funcionalidades - Construção de script para migração de dados - Otimizações (Indexes, desnormalização)	03/06/2019	01/07/2019	01/07/2019
- Implementação em NHibernate - Acesso a dados (repositórios)	02/07/2019	02/08/2019	02/08/2019
- Testes unitários a repositórios - Testes unitários a modelo de dados	03/08/2019	01/09/2019	01/10/2019
- Server-Side - APIs: autenticação / autorização - Métodos *Create Real Update and Delete* (CRUD) - Testes funcionais / unitários	02/09/2019	01/11/2019	10/01/2020
- Implementação da parte gráfica de *User Interface* (UI) já desenhada - Reestruturação da aplicação aplicando as boas práticas - Aplicação em vue.js - Testes Unitários - Testes end-to-end	02/11/2019	31/03/2020	31/03/2020

2.3. **Metodologias e processos**

No decorrer do estágio e durante a implementação do projeto da plataforma *e-commerce* aplicou-se a metodologia *Scrumban*, que é uma metodologia ágil que resulta da aplicação de duas metodologias: *Scrum* e *Kanban*.

2.3.1. Metodologias

A metodologia *Agile* nasceu em 2001 por um pequeno grupo de pessoas que estavam cansados dos métodos tradicionais e demorosos de desenvolvimento de *software* e é conhecida por ser um processo ágil de ajuda às equipas, de forma a obter respostas rápidas e imprevisíveis sobre o projeto. Com o uso desta metodologia, a avaliação do projeto durante a fase de desenvolvimento torna-se mais rápida, visto serem organizadas reuniões regulares chamadas *sprints* ou iterações. Esta metodologia foca-se essencialmente em quatro valores:

- O foco em pessoas e suas interações em vez de nos processos e ferramentas de trabalho;
- O *software* de trabalho é mais importante que a sua própria documentação;
- A colaboração do cliente é essencial no projeto;
- Os processos devem reagir à necessidade de mudança em vez de seguir um plano à risca. [3]

Numa metodologia *Agile* existem doze princípios a ter em conta quando se desenvolve *software*:

1. Satisfazer o cliente e fornecer um *software* com continuidade e valioso;
2. Aceitar a mudança de requisitos ao longo do projeto;
3. Entregas de software funcional a curtos prazos;
4. Trabalhar em conjunto com o cliente;
5. A informação é transmitida de melhor forma quando se tem reuniões cara a cara com o cliente;
6. Motivação da equipa e bom ambiente de trabalho cria confiança e capacidade;
7. O *software* de trabalho é a melhor medida do progresso;
8. Processos ágeis promovem ambientes sustentáveis;
9. A preocupação contínua de melhoramento de *software* é excelente para relação entre técnico de desenvolvimento e *designer*;
10. A simplicidade é essencial para uma boa gestão ágil;
11. Equipas organizadas produzem melhor arquitetura, requisitos e *design*;
12. Deve haver reflexão das equipas através da inspeção e adaptação para serem mais efetivas.

Dentro da metodologia *Agile* existem várias ramificações, ou seja, várias metodologias ágeis em que cada uma tem as suas próprias práticas, terminologias e táticas. [3]

No caso da plataforma *e-commerce* desenvolvida, foram aplicadas algumas práticas e táticas das metodologias *Scrum* e *Kanban*. Estas metodologias utilizadas em conjunto deram origem a outra metodologia: o *Scrumban*.

De seguida são apresentadas as três metodologias ágeis relevantes para a plataforma *e-commerce*: *Scrum*, *Kanban* e *Scrumban*.

Scrum

A metodologia *Scrum* é uma estrutura que assenta na gestão de competências de longo alcance, de forma a controlar as iterações e incrementos nos projetos. Esta pode ser utilizada com outras metodologias ágeis dependendo na necessidade do projeto e fornece às equipas formas de estabelecer comunicação entre os membros da equipa de forma a dar opiniões sobre as funcionalidades, experiência e fazer os ajustes necessários.

Esta metodologia tem como valores:

- O compromisso, em que os membros da equipa se comprometem a atingir objetivos;
- A coragem, em que toda a equipa está focada na melhor abordagem e a trabalhar nos problemas difíceis;
- O foco, a concentração ajuda para a conclusão da tarefa e objetivos da equipa;
- A abertura, em que todos os membros da equipa e cliente estão abertos para todos os desafios que forem encontrados;
- O respeito, para que todos os membros da equipa sejam capazes e independentes.

Os principais princípios desta metodologia são:

1. Transparência - trabalhar em equipa num ambiente envolvente em que todos os membros saibam os problemas uns dos outros;
2. Inspeção - proceder a reuniões frequentes para saber os pontos de situação e refletir sobre eles de forma a perceber o funcionamento do processo;
3. Adaptação - constante investigação por parte da equipa para saber como estão a correr as coisas ou alterar processos que não fazem sentido. [4]

Relativamente ao fluxo de trabalho, nesta metodologia as tarefas são selecionadas de antemão para o próximo *sprint*. Entende-se por *sprint* um conjunto de tarefas a ser executadas por iterações (normalmente mensais). Nesta metodologia existe sempre um líder

de equipa intitulado por *Scrum Master*, que toma decisões sobre a sua equipa, distribui tarefas e responsabilidades. A equipa é composta pelos programadores, o *Scrum Master* e o *Product Owner*. Este último é responsável pela gestão de tarefas a desenvolver (*Product Backlog)*. [5]

Na Figura 8 encontra-se um esquema de *sprint* no *Scrum*, divido por três colunas onde é possível verificar-se as tarefas a realizar "todo", o trabalho a fazer na *sprint* "OnGoing" e as tarefas concluídas "Done". [6]

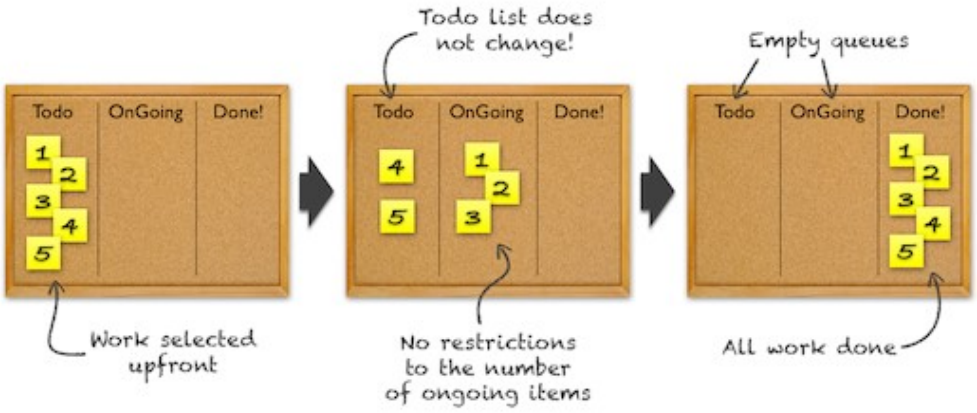


Figura 8 - Quadro de sprints *Scrumb* [6]

Kanban

A metodologia *Kanban* visa a entrega contínua de *software* sem sobrecarregar a equipa de desenvolvimento e a melhorar o seu desempenho. Neste método existem três princípios:

1. Visualizar o que se faz, ou seja, tentar perceber todos os contextos existentes;
2. Limitar a quantidade de trabalho em andamento, ou seja, equilibrar o trabalho de modo a que o fluxo de trabalho não fique comprometido;
3. Melhorar o fluxo, isto é, assim que uma tarefa ficar concluída, seguir para a próxima priorizando a máxima urgência de funcionalidade.

Esta metodologia promove também a colaboração do cliente e da equipa de forma constante, de modo a incentivar a aprendizagem e melhorar o fluxo de trabalho na equipa. [3]

Em termos de fluxo de trabalho esta metodologia limita a quantidade de trabalho em andamento e torna-se possível alterar o seguimento das tarefas em qualquer momento, ou seja, não existe um fim de *sprint*. Na Figura 9 encontra-se um esquema relativo aos quadros *Kanban*. A figura demonstra as tarefas a fazer "todo", as que estão em desenvolvimento "OnGoing" e as concluídas "Done". À medida que vão sendo concluídas, vão surgindo novas tarefas e alteração de estados de outras tarefas. [6]

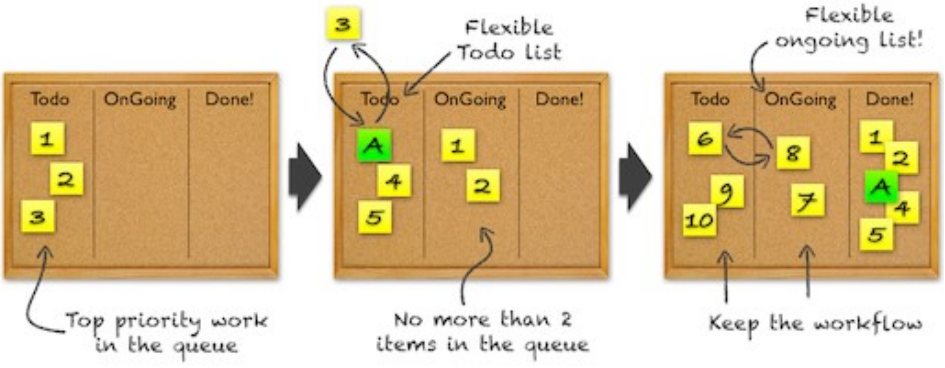


Figura 9 - Quadro *Kanban* iterações. [6]

Scrumban

A metodologia usada no desenvolvimento da plataforma foi a *Scrumban*. Esta trata-se de uma mistura de *Scrum* com o *Kanban* de forma a permitir à equipa o melhoramento contínuo do processo. Nesta metodologia o planeamento pode ser feito em intervalos regulares com revisão e retrospetiva, mas seu verdadeiro objetivo é decidir as tarefas a realizar em cada iteração, sem haver preocupação com o número de tarefas a seguir, e assim, reduzir o tempo e a sobrecarga de cada plano da iteração.

As vantagens desta metodologia são:

- A qualidade;
- As decisões no momento, ou seja, fazer decisões apenas quando é necessário;
- A continuidade de melhoramento;
- A minimização do desperdício, de forma a fazer apenas o que realmente tem valor ao cliente;
- O melhoramento do processo ao adicionar alguns valores do *Scrum* quando necessário. [6]

Relativamente ao fluxo de trabalho desta metodologia, a análise das tarefas é priorizada antes de iniciar o desenvolvimento de forma a evitar analisar ou criar muitas tarefas. Na Figura 10 encontra-se um esquema de um quadro *Scrumban* com os objetivos e priorizado por fases de desenvolvimento.

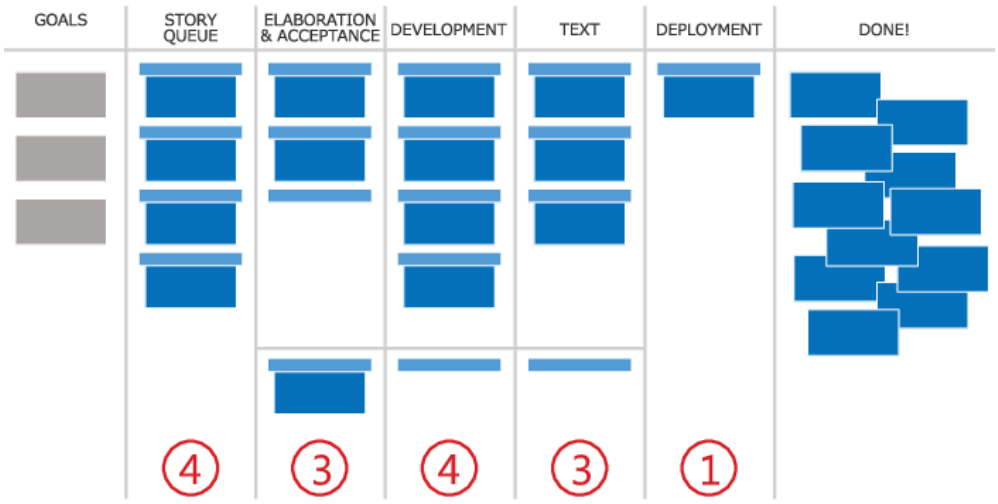


Figura 10- Quadro *Scrumban* interações. [7]

2.3.2. Ferramentas de gestão de projeto

Para a plataforma *e-commerce* foram utilizadas algumas ferramentas de gestão, seja de código, documentação ou tarefas. Todas estas ferramentas são da empresa Atlassian. A Atlassian fornece várias ferramentas para gestão de projetos ou equipas, de forma a facilitar e organizar projetos. No caso da plataforma, foi utilizado o Bitbucket para gestão de código, o Confluence para documentação e o Jira *Software* para as tarefas. Além destas ferramentas, foi utilizado também o Slack para gestão de comunicações entre a equipa, que recentemente também estabeleceu uma parceria com a Atlassian. [8]

Jira *Software*

O Jira é uma das ferramentas utilizadas para gestão das tarefas, permitindo o planeamento, criação e gestão de tarefas de forma a organizar-se mais facilmente não só o fluxo de trabalho, mas também o seu tempo. Assim, é possível calcular o tempo de duração em cada tarefa, agrupar sub-tarefas e muito mais. Esta ferramenta é normalmente utilizada em metodologias *Agile* e foi apresentada e abordada no contexto da unidade curricular de Gestão de Projeto.

Na Figura 11 apresenta-se um quadro *Kanban* relativo ao desenvolvimento da plataforma *e-commerce*, onde estão representadas as tarefas e objetivos a serem desenvolvidos, os que estão em desenvolvimento e os já concluídos. Apesar do quadro ser *kanban*, são aplicados alguns processos do *Scrum* e daí a aplicação da metodologia *Scrumban*. Cada projeto é identificado de maneira diferente de forma a atribuir a melhor metodologia de acordo os requisitos do mesmo e a interação com o cliente.

Além disso foram realizadas reuniões regulares para a avaliação das tarefas em execução e atribuição das próximas tarefas. A atribuição das tarefas foi feita pelo *Scrum Master*, após discutidas com o *Product Owner*. O fluxo de trabalho foi atribuído de forma a priorizar as tarefas mais urgentes primeiro e assim que uma tarefa é terminada, inicia-se a tarefa seguinte mais prioritária.

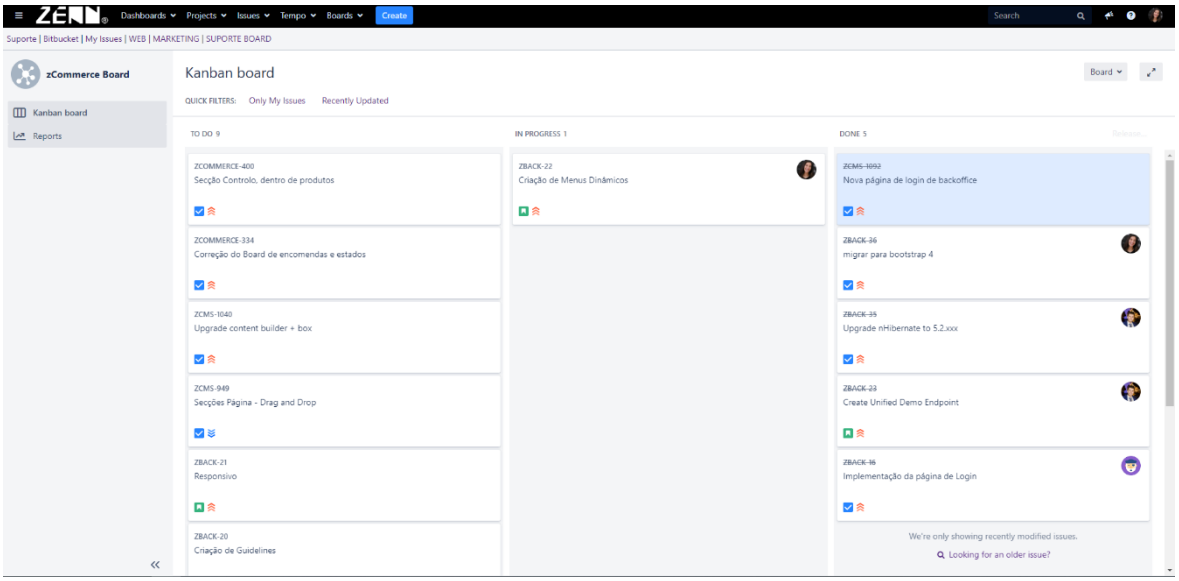


Figura 11 - Quadro *Kanban* da plataforma e-commerce

Para a gestão da plataforma *e-commerce*, foi criado no Jira *Sotfware* um projeto chamado "zCommerce", e elaborado um quadro com todas as tarefas necessárias para a sua implementação. Na Figura 12 verifica-se um exemplo de tarefas relativas a este projeto com duas *labels* em destaque: a *label* "Bug" e a *label* "Task".

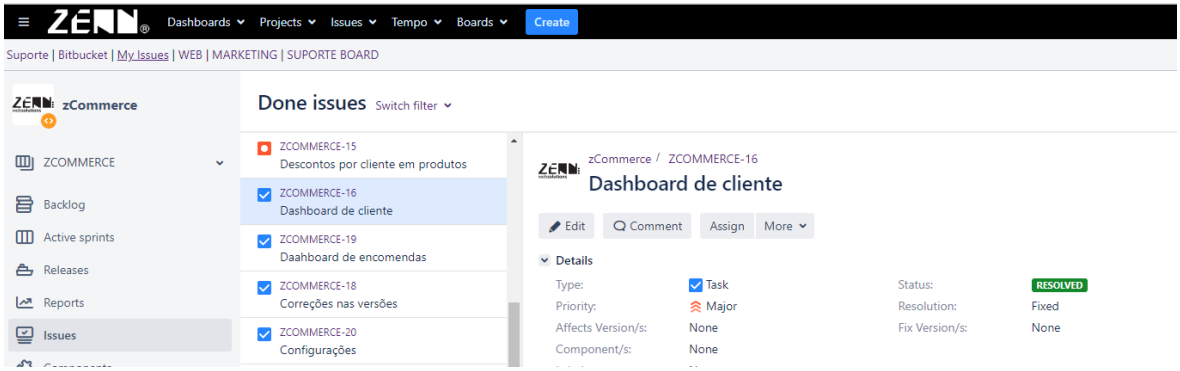


Figura 12 - Jira - tarefas "zCommerce"

As tarefas foram criadas e categorizadas em 4 tipos: Task, Story, Bug ou Epic. No quadro da plataforma *e-commerce* a maioria das tarefas encaixou na categoria de “taks”, “sub-tasks” ou “bug”. Sempre que foi criada uma nova tarefa, esta era classificada como “task”. Sempre que a funcionalidade em questão tinha ramificações, foi atribuída uma “sub-task”. Quando surgiram problemas de implementação foi atribuído a categoria “bug”. Na Figura 13 pode-se encontrar a lista de tipos de tarefas.

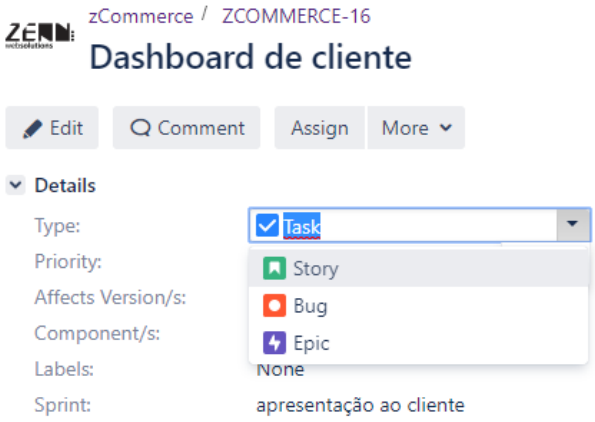


Figura 13 - Jira - Tipos de tarefas

Além da categorização por tipo, também se considerou importante categorizar a prioridade das tarefas: “Major”, “Blocker”, “Critical”, “Minor” e “Trivial”. Esta categorização permitiu a organização do fluxo de tarefas e priorização às de maior importância. As “Blocker”, como o próprio nome indica, foram consideradas bloqueantes na plataforma e, como tal, tiveram de ser imediatamente corrigidas tal como as tarefas em estado “Critical”, que se apresentava como segundo estado mais urgente. De seguida era o “Major” e depois os outros estados menos problemáticos: o “Minor” e “Trivial”. Estes foram

atribuídos a tarefas que não influenciavam o uso da plataforma. Na Figura 14 pode verificar-se a lista de prioridades de tarefas.

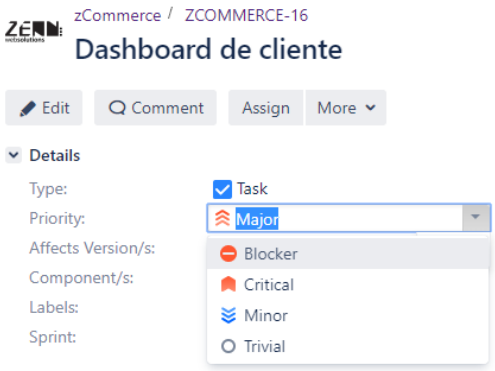


Figura 14 - Jira - Prioridades das tarefas

Confluence

O Confluence é uma plataforma que ajuda na organização e criação de documentação dos projetos tanto para o utilizador final como para o programador. Esta foi a plataforma usada para a elaboração de documentação interna no que diz respeito ao funcionamento da plataforma *e-commerce*, mas também para a ajuda do cliente através de tutoriais ilustrativos de forma a compreender melhor a utilização do *backoffice* da plataforma *e-commerce*. Na Figura 15 está representado o *dashboard* de desenvolvimento.

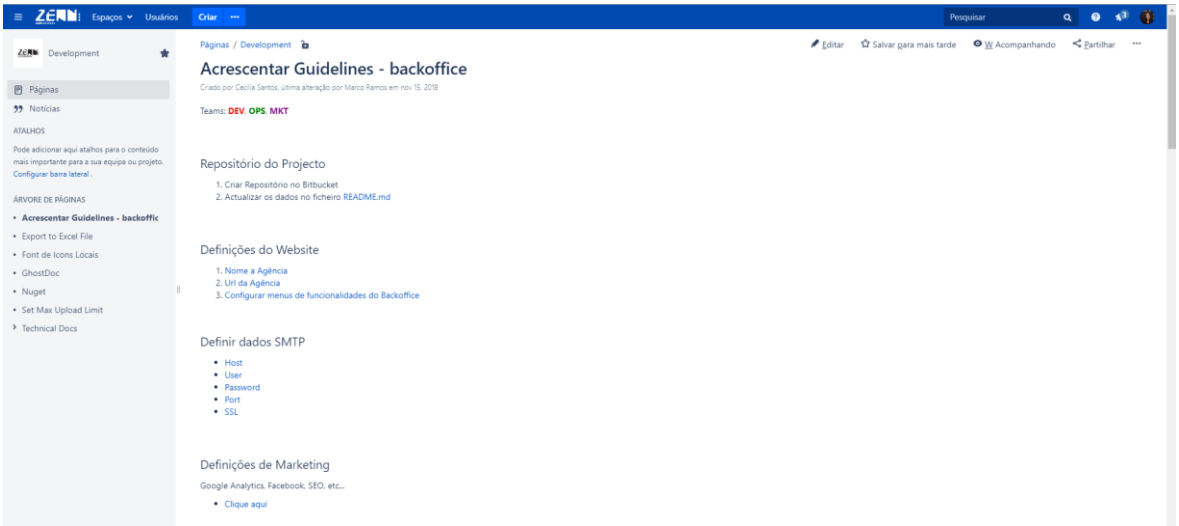


Figura 15 - Confluence - *dashboad development*

Bitbucket

Outra das ferramentas utilizadas, desta vez para gestão de código, foi o Bitbucket. Esta ferramenta também foi apresentada na unidade curricular de gestão de projeto e utilizada noutras unidades curriculares, como por exemplo, desenvolvimento de dispositivos móveis e sistemas sensíveis ao contexto.

Ela permite que o código do repositório esteja sempre disponível numa *cloud*, o que possibilita a colaboração de elaboração do código por parte de vários membros da equipa de uma forma simples e rápida. Além disso, esta ferramenta permite clonar o mesmo projeto em uma ou mais ramificações (*branchs*). Desta forma, foi possível haver uma versão estável do projeto (*branch master*) e noutras ramificações do projeto trabalhar novas funcionalidades. Depois das funcionalidades testadas e validadas era feito um *merge* delas com a *branch master* de modo a juntar todas as funcionalidades.

Na Figura 16 pode verificar-se o repositório relativo à plataforma de *e-commerce*.

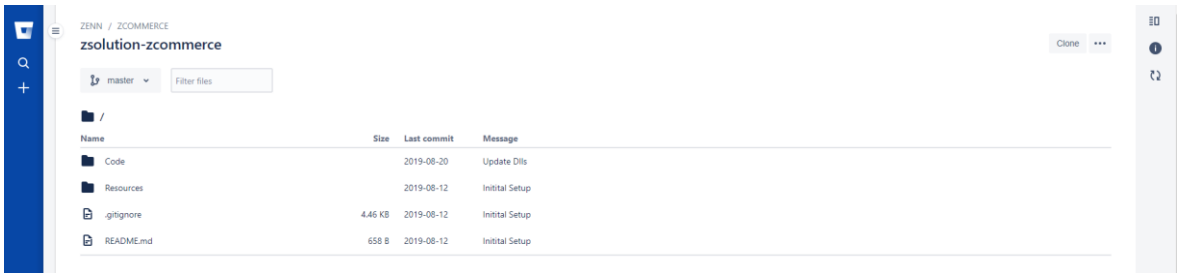


Figura 16 - Bitbucket – *zCommerce*

Slack

Para a gestão das comunicações optou-se pelo Slack pois reúne todas as ferramentas necessárias para uma boa colaboração em equipa e tem a possibilidade de criar vários canais para cada área de negócio dentro da empresa. Aqui as comunicações são instantâneas e existe a possibilidade de partilha de recursos. Na Figura 17, encontra-se uma conversa no Slack, onde é possível visualizar os vários canais e cada membro da equipa. É possível interagir por canal ou individualmente com cada membro.

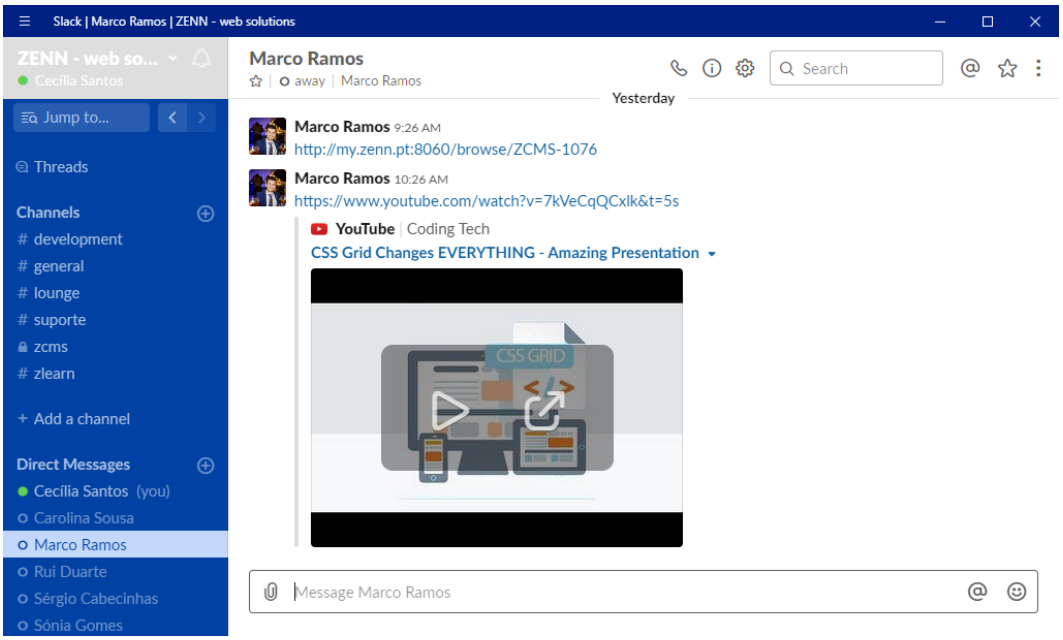


Figura 17 – Slack

2.4. Projetos Extra

Além da reconstrução da plataforma *e-commerce*, que foi o projeto principal deste estágio, por vezes foi necessário deixar o *workflow* temporariamente para conseguir resolver outras situações que surgiram, ou implementação de novos projetos. Esta secção diz respeito aos projetos desenvolvidos no decurso do estágio que não se enquadram no objetivo principal do mesmo: a reconstrução da plataforma *e-commerce*.

Projetos com plataforma e-commerce antiga

Visto que a reconstrução da plataforma ainda demorou algum tempo, algumas lojas *online* solicitadas por parte dos clientes ainda correspondiam aos critérios existentes na antiga plataforma, e como tal procedeu-se à implementação da antiga plataforma até a nova se encontrar operacional.

Estes projetos implementados utilizaram essencialmente o modelo de negócio *B2C* (ver secção 3.1), nos sectores de moldes, comunicações, têxteis e artes.

De seguida seguem alguns exemplos de projetos implementados com recurso à antiga plataforma *e-commerce*, que foram implementados com o *back-end* da plataforma, mas com um *design* de *front-end* desenhado à medida:

Mosdecor

A Mosdecor dedica-se à pintura de azulejos desde 1998, e decidiu apostar na venda *online* das suas pinturas artesanais, não só em azulejo como noutros materiais.

A Figura 18 apresenta a loja *online* desenvolvida, com possibilidade de filtrar pelo tipo de materiais utilizados.

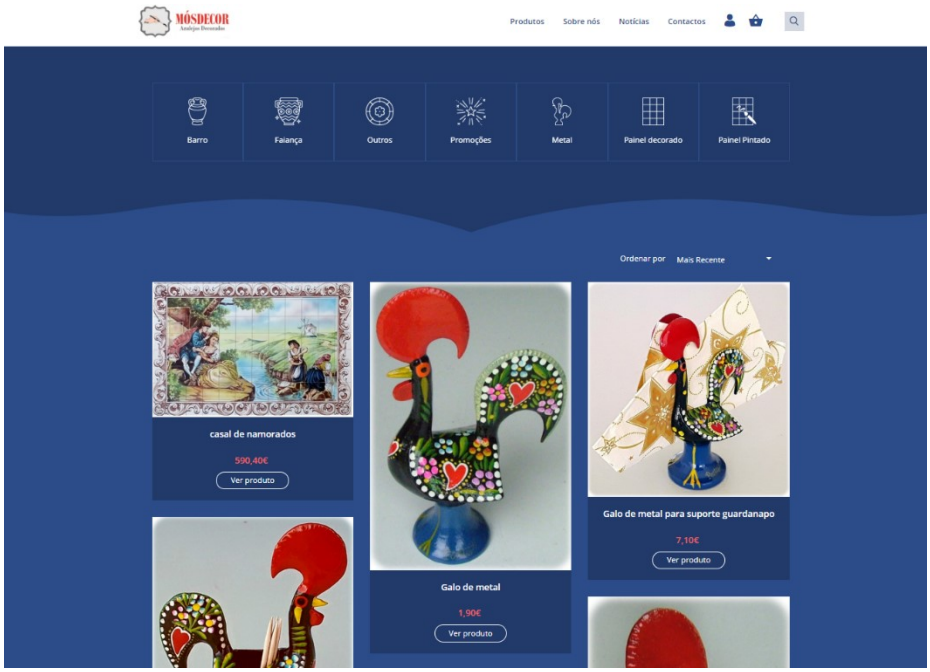


Figura 18 - Mosdecor - loja *online*

APLGadgets

Esta empresa dedica-se à instalação e comercialização de equipamentos de telecomunicações, sistemas de vigilância e distribuição de sistemas codificados de televisão, internet e telefone.

A loja *online* consistiu na venda dos equipamentos referidos em cima, com as respetivas divisões das áreas de comercialização. A Figura 19 apresenta a loja *online* desenvolvida.

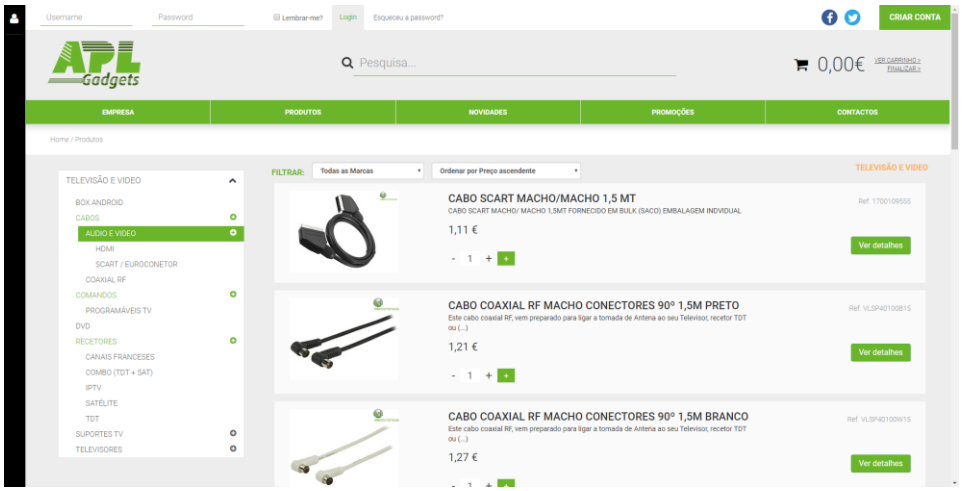


Figura 19 - APL Gadgets - loja online

Pow Wow Tribe

Esta empresa dedica-se à comercialização de roupa que deixa bronzear, tanto para homens ou mulheres. A loja *online* consiste em duas categorias de roupa: homem e mulher, como se pode verificar na Figura 20.

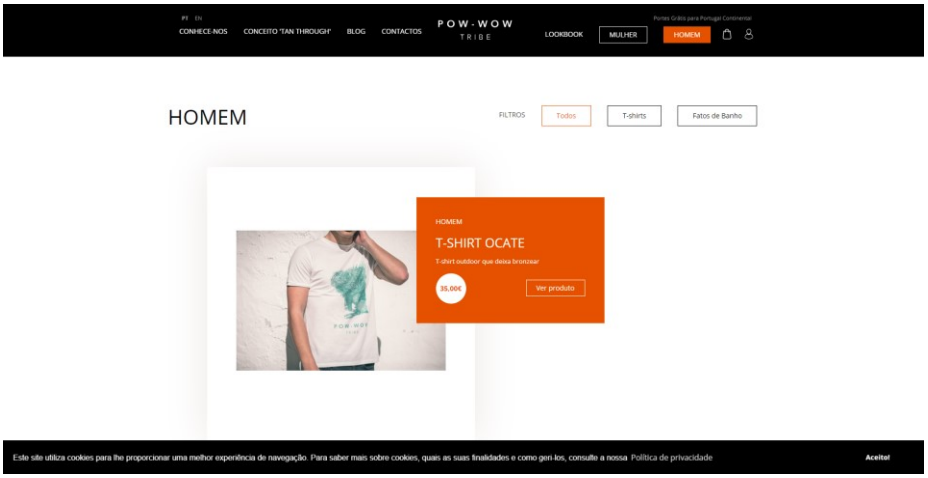


Figura 20 - Pow Wow Tribe - loja *online*

Projetos institucionais

Além das lojas *online*, também foram realizados projetos institucionais. Estes projetos foram desenvolvidos com recurso à plataforma antiga, pois além de lojas *online* também era possível criar *websites* institucionais sem qualquer tipo de relacionamento com produtos, carrinho de compras ou encomendas. Alguns exemplos são:

Grupo Moldoeste

O grupo Moldoeste está centralizado nas áreas de moldes e plásticos, e trabalha essencialmente com a indústria automóvel.

Na Figura 21 está representado o *website* institucional desta empresa.

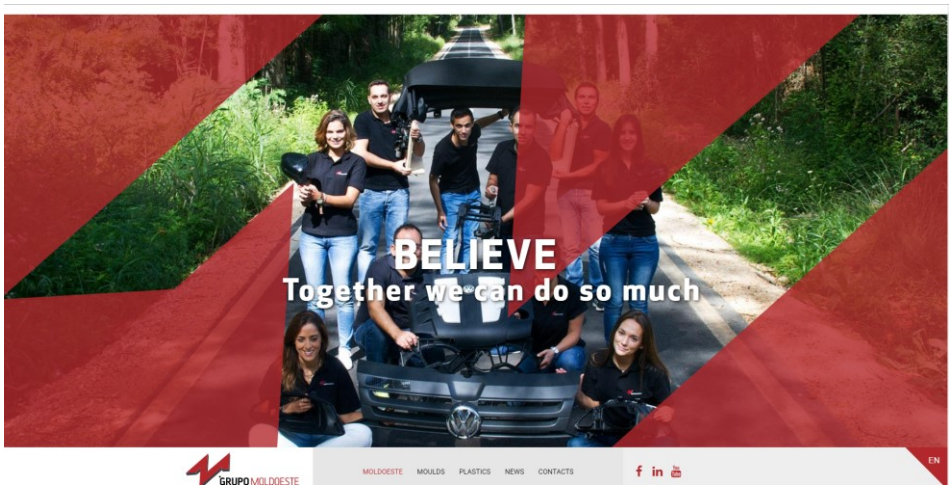


Figura 21 - Grupo Moldoeste - *website*

Lapierce

A Lapierce é uma marca de sapatos que quis apostar num *website* que tivesse a possibilidade de se converter numa loja *online*, como se pode verificar através da Figura 22.

Figura 22 - Lapierce - *website*

2.5. **Análise SWOT e crítica**

Nesta secção encontra-se a análise SWOT da empresa ZENN – web solutions, da responsabilidade da mesma, seguida de uma análise crítica relativamente ao funcionamento da mesma.

Na Figura 23 encontram-se representadas as forças, fraquezas, oportunidades e ameaças relativas à empresa. Esta é a análise SWOT que a empresa fez sobre si própria.

Figura 23 - Análise SWOT da ZENN - web solutions [2]

Análise Crítica

No final do estágio é importante fazer uma análise critica sobre o funcionamento da empresa, os seus pontos fortes e fracos, áreas a melhorar, entre outros. De seguida encontram-se descritos os pontos mais relevantes relativamente à empresa, segundo análise pessoal e análise SWOT apresentada pela empresa.

Relativamente à análise SWOT acima mencionada e de acordo o processo de adaptação que ocorreu no estágio, um dos pontos fortes da empresa é o facto da equipa ser jovem e com uma comunicação excelente entre todos os membros, isto faz com que a integração de qualquer novo elemento na equipa seja muito mais fácil.

Em termos de aprendizagem, é algo constante, pois a procura de novas tecnologias tanto como a atualização das existentes requer sempre uma pesquisa acrescida. Para além disso, a

empresa fornece aos seus colaboradores formações internas, com cursos especializados nas diversas áreas, em que são disponibilizados consoante as dificuldades ou necessidades de cada colaborador.

Em termos empresariais, as áreas de gestão dos recursos humanos e documentação são mais cruciais no que toca a melhorias a ser consideradas num futuro próximo. Situações em que a gestão dos recursos humanos não é executada eficientemente, havendo uma falha de comunicação pelo gestor de projeto no que toca aos recursos e tempo para execução das várias tarefas que constituem o projeto, o que leva ao aumento de tempo de execução face ao que foi estimado inicialmente. No entanto esta gestão tem vindo a melhorar consideravelmente e a gestão de tempo um pouco mais controlada.

Já no que diz respeito à documentação, deveria existir maior foco em elaborar documentação apropriada a cada projeto, com o máximo de detalhes possíveis, de modo a que qualquer pessoa interna ou cliente possa documentar-se sobre qualquer parte do projeto e trabalhar com as suas ferramentas, sem ter de recorrer ao programador que iniciou o mesmo. Este ponto também tem vindo a melhorar ao longo do tempo, mas ainda é um processo moroso.

3. Plataforma *e-commerce*

Neste capítulo pretende-se dar a conhecer o que é o *e-commerce* e seus modelos de negócio, bem como descrever a antiga plataforma e algumas plataformas *e-commerce* com funcionalidades relativamente parecidas a esta.

Na secção 3.1 pode encontrar-se uma pequena definição do que significa *e-commerce*. Além disso, também foram identificados os seus modelos de negócio e quais os que foram utilizados neste projeto.

A secção 3.2 diz respeito à plataforma *e-commerce* da ZENN. Nesta secção encontram-se identificados os problemas base da plataforma antiga bem como alguns requisitos identificados para a nova plataforma.

Por último, na secção 3.3, encontram-se algumas plataformas que partilham a maioria das funcionalidades com a plataforma deste projeto.

3.1. O que é *e-commerce*?

O comércio eletrónico, *e-commerce,* surgiu pela primeira vez no ano de 1991 e rapidamente se assistiu ao seu crescimento, visto que as empresas começaram a mudar o seu paradigma em termos de mercado e a expandir o seu negócio a nível digital. Esta mudança permitiu que os consumidores tomassem contacto com novas oportunidades sem ser necessário deslocamentos físicos. [9]

Entende-se por plataforma *e-commerce* um *software* que possibilita a criação de lojas virtuais de forma simples e rápida, com oferta de serviços únicos e capacidades que uma loja física não consegue atingir. [10] Temos como dois dos maiores exemplos de sucesso e inovação neste tipo de negócios, a Amazom e o Ebay, apresentando ainda serviços sustentáveis e modelos de negócio desenvolvidos internamente com integração *online* e *offline* de forma a tornar os seus negócios mais lucrativos.

Em 2015 o comércio eletrónico focou-se essencialmente em serviços *on-demand*, nomeadamente redes sociais e plataformas móveis criadas para *smartphones* e *tablets*, provocando grandes mudanças a nível social, mercado, indústrias e pequenos negócios. [9]

O comércio eletrónico é composto por vários modelos de negócio. Estes distinguem-se entre si através de um conjunto de transações consoante os agentes envolvidos no processo de troca. Os tipos de *e-commerce* existentes incidem em:

- *Business-to-business* (B2B);
- *Business-to-consumer* (B2C)*;*
- *Business-to-employee* (B2E)*;*
- *Business-to-government* (B2G)*;*
- *Business-to-manager* (B2M)*;*
- *Consumer-to-business* (C2B)*;*
- *Consumer-to-consumer* (C2C)*;*
- *Government-to-business* (G2B);
- *Government-to-citizen* (G2C)*;*
- *Government-to-employees* (G2E)*;*
- *Government-to-government* (G2G)*;*
- *Peer-to-peer* (P2P). [11]

No contexto deste trabalho considera-se mais relevante a abordagem de apenas duas das modalidades: B2B e B2C.

O modelo *business-to-business* (B2B) caracteriza-se por transações comerciais entre as organizações, como por exemplo, o fabricante e o vendedor. Esta é uma das principais modalidades de *e-commerce* com um maior volume de venda em grande escala quando comparado com o B2C, visto que ocorrem múltiplas trocas entre as organizações que envolvem a aquisição de matérias prima ou subcomponentes enquanto as transações B2C só acontecem uma vez. [11]

A modalidade de negócio *business-to-consumer* (B2C), descreve as atividades de comercialização que uma empresa tem relativamente aos seus produtos ou serviços a consumidores finais, como é o caso da Amazon. Nestes serviços ou produtos podem incluir-se atividades de *online banking*, serviços de viagens, leilões *online* e *websites* de venda de imóveis, entre outros. [11]

3.2. A restruturação da plataforma

Nesta secção pretende-se dar a conhecer a plataforma *e-commerce* antiga bem como o verdadeiro motivo da sua reconstrução e novos requisitos apresentados por parte dos clientes da empresa.

A nova plataforma foi totalmente desenvolvida pela ZENN com o intuito de dar ao cliente uma solução à sua medida, com suporte garantido, de fácil utilização, totalmente flexível e extensível. Cada cliente é único, ou seja, cada cliente tem a sua própria plataforma *e-commerce*, totalmente à sua medida, com as funcionalidades bases da plataforma e funcionalidades extra caso necessário.

A reconstrução da plataforma *e-commerce* deu-se essencialmente devido à descontinuação do *Object-Relational Mapping* (ORM) utilizado, o Telerik DataAcess Fluent. Por este motivo, todos os projetos desenvolvidos futuramente estavam postos em causa. Na secção 4.3.1 pode encontrar-se mais informações sobre o que é um ORM e na 4.3.2 sobre o ORM Telerik DataAcess Fluent.

Para além deste motivo, e de forma a melhorar a plataforma e dar suporte funcionalidades solicitadas pelos clientes da empresa, aproveitou-se para:

1. Reconstruir os modelos de negócio B2B e B2C da plataforma;
2. Otimizar performance e processos;
3. Atualizar a política de privacidade e proteção de dados;
4. Implementar novas funcionalidades.

A antiga plataforma suportava os modelos de negócio B2B e o B2C, mas a sua gestão era muito rudimentar. Assim, no caso de B2B um administrador tinha a possibilidade de adicionar determinado utilizador como revendedor, mas não conseguia gerir qual os grupos de revendedores e que regras a aplicar da sua loja *online* sobre esse mesmo grupo.

Ao identificar estes problemas, e tendo em conta a necessidade de atualizar a estrutura da base de dados, aproveitou-se para otimizar processos e performance. Deste modo tornou-se a plataforma mais intuitiva e de fácil de utilização, seja a nível de gestão de conteúdos no *front-end* como no *backoffice* com nova cara e novas funcionalidades. Das novas funcionalidades podem-se destacar por exemplo, a gestão de entidades e *emails*,

implementação de novas técnicas de venda e métodos de pagamento e alterações no carrinho para ser possível identificar "carrinhos abandonados".

A plataforma antiga de *e-commerce* era utilizada como base para cada novo projeto e no caso de ser necessário novas funcionalidades à medida do cliente, elas eram desenvolvidas sobre essa base. Esta abordagem, tornava-se numa desvantagem em caso de correções de funcionalidades, visto que se o problema fosse detetado em determinado projeto, seria provável que esse mesmo erro estivesse presente noutros projetos. Este problema tornava-se muitas vezes moroso de corrigir e acarretava custos adicionais para a empresa.

Nesta nova plataforma e com essa limitação em conta, a empresa decidiu apostar e fazer *packages* de módulos, que permite que as necessidades do cliente determinem a instalação apenas dos módulos necessários ao projeto. Caso exista alguma falha a resolver num destes módulos, a correção será feita e disponibilizada rapidamente através do *update* do *package* que ficará acessível de imediato a todos os projetos.

Além disso, tornou-se mais fácil o desenvolvimento de novas funcionalidades segundo as necessidades dos clientes, ou seja, funcionalidades que facilmente podem integrar o *core* da plataforma *e-commerce* ou podem surgir como um módulo extra por ser específico de mais. Alguns exemplos de funcionalidades pretendidas são:

- Gestão de versões de produtos (versão simples, por descritivos ou por grupo de opções);
- Gestão de preços e stocks em massa;
- Gestão de entidades;
- Gestão de emails;
- Carrinho abandonado;
- Novos métodos de pagamentos automatizados – SIBS;
- Métodos de entrega por zona – código postal (Portugal);
- Disponibilidade de Produtos;
- Personalização de emails de encomendas;
- Atualização *Google Merchant*.

3.3. Outras plataformas

Nesta secção encontram-se descritas algumas das plataformas com soluções idênticas à plataforma *e-commerce* deste projeto, com modelos de negócio semelhantes, gestão de encomendas, integrações com sistemas ERP (*Entreprise Resource Planning*) e técnicas de *marketing*. As plataformas descritas são: o Magento, Prestashop, Redicom e LV Engine. Destas plataformas, as duas primeiras são *open source*, ou seja, são plataformas de código aberto que qualquer programador pode adaptar à medida do cliente. As duas últimas são plataformas concebidas de forma a que cada cliente tenha a sua própria plataforma, ou seja, são instaladas no alojamento do cliente e configuradas unicamente e exclusivamente para o cliente e à medida do mesmo.

De seguida podem-se encontrar descritas algumas das funcionalidades mais relevantes de cada uma das plataformas e no fim um pequeno resumo geral sobre as mesmas.

Magento

O Magento é uma plataforma *e-commerce* com variadas soluções nas diversas áreas de negócio, com modelos de negócio B2B e B2C para pequenas e médias empresas a nível global.

Algumas das funcionalidades desta plataforma são:

- Fornecer recursos integrados de comércio e ferramentas para acelerar as vendas;
- Ser uma plataforma acessível, devido ao alojamento na *cloud* eliminando custos de manutenção e monitorização, ilimitada pois pode ser personalizada e evoluir conforme a evolução do negócio, e fiável graças ao seu vasto leque de comerciantes que tem crescido muito a nível mundial;
- Possibilitar o comércio omnicanal, ou seja, várias possibilidades de escolha de fluxos com diferentes modalidades de compra para que o cliente decida qual pretende escolher e comprar com diversas opções. Totalmente configuráveis pelo comprador e com uma expansão global de mercado;
- Ser adaptável a telemóveis e tablets, sendo que a maioria das vendas têm sido registadas através de dispositivos móveis;
- Possibilitar a integração com o ERP e gestão de inventários:

- Com a nova versão de *software* Magento 2, permite o acesso a funcionalidades como *drag and drop* para a criação de páginas, acelerar taxas de conversão de venda com recurso a carrinhos abandonados, entre outros. [12]

Prestashop

O Prestashop é uma plataforma que dispõe de várias soluções de *e-commerce* com a possibilidade de implementação de vários módulos consoante o modelo de negócio a aplicar. Esta plataforma está otimizada para os modelos de negócio B2B e B2C e alguns dos módulos podem ser adquiridos sem custos.

Na vasta gama de funcionalidades encontradas consoante os módulos a adquirir, distinguem-se:

- Personalização do visual da loja *online*, com temas pré-definidos;
- Técnicas de *marketing* e desenvolvimento aplicadas para uma boa indexação dos motores de busca;
- Ferramentas de *marketing* que permitem um aumento da taxa de conversão de clientes, como *chat-online*, carrinho abandonado, *pop exit retargeting* (*pop up* que aparece caso o cliente esteja a sair do site, de forma a cativá-lo de volta), entre outros;
- Métodos de pagamento automatizados com garantia de segurança;
- Visualização gráfica da evolução de encomendas;
- Integração com sistemas ERP;
- Descontos personalizados de cliente para cliente. [13]

Redicom Commerce Plataform

A Redicom é uma plataforma *e-commerce* centrada na gestão de encomendas, *marketing* e modelos de negócios B2C e B2B já descritos anteriormente na secção 3.1.

As soluções apresentadas com o modelo de negócio B2C são:

- Gestão de conteúdos, com editor *drag and drop* de forma a fornecer total autonomia na criação de conteúdos sem necessitar de programação e com uma

biblioteca de combinações associadas de forma a criar conteúdo sem qualquer tipo de esforço e adaptável a telemóveis e tablets.

- Gestão de encomendas:
    - Apresentação em tempo real com fluxo automatizado de encomendas e monitorização de pagamentos;
    - Gestão e atualização de stocks;
    - Disponibilização de códigos de *tracking* e notificações de cliente;
    - Sistemas de gestão, pagamento e transportadoras;
    - Faturação integrada com criação de faturas e personalização de documentos;
    - Trocas e devoluções com criação de notas de crédito e acompanhamento da troca ou devolução.
- *Marketing*, com criação automática de segmentos:
    - Criação de perfis completos com informações de interesses e comportamentos dos clientes;
    - Campanhas personalizadas de cliente a cliente com produtos do seu interesse: vouchers, descontos de aniversário, carrinho abandonado, entre outros.
- Comércio Omnicanal, que fornece várias possibilidades de escolha de fluxos com diferentes modalidades de compra para que o cliente decida qual pretende escolher e comprar. [14]

Já as soluções apresentadas com o modelo de negócio B2B são:

- Gestão de Encomendas:
    - Importações e exportações via CSV/Excel para uma gestão mais eficiente das ordens de encomenda;
    - Soluções de gestão de armazém integradas com o ERP;
    - Sistemas de envio e pagamento;
    - Gráficos evolutivos das vendas e monitorização de campanhas.
- Integrações e possibilidade de interligação das regras de negócio do ERP com as regras da plataforma de forma a obter as informações do produto mais atualizadas em ambos os lados;

- Comércio Global, com flexibilidade de expansão do negócio a nível mundial com multi-moeda, multi-idioma e multi-operações, com métodos de pagamento diferentes e com stocks, preços e campanhas personalizadas conforme o mercado;
- *Marketing* e ferramentas capazes de adaptar campanhas conforme o negócio como por exemplo descontos por volume, preços personalizados por cliente, vouchers, entre outros, tal como já apresentado na solução B2C. [14].

LVEngine

Esta plataforma *e-commerce* disponibiliza vários serviços e soluções aos seus clientes de forma a melhorar a estratégia do seu negócio *online*, com técnicas de venda através de ferramentas de *marketing* e possíveis integrações com ERP. Os modelos de negócio adotados nesta plataforma foram o B2B e B2C e é considerada uma plataforma escalável consoante o crescimento do negócio.

Dentro das soluções disponíveis na plataforma *e-commerce*, destacam-se:

- *Marketing* e vendas, com campanhas de *marketing automation* multi-canal (email, sms, *website* e redes sociais) e automatização de emails de aviso consoante o tipo de cliente;
- Gestão de pagamentos, com várias formas de pagamento *online* ou por cartão consideradas 100% seguras;
- Gestão de entregas, com vários fornecedores para a expedição e entregas de encomendas;
- Gráficos evolutivos, com informação em tempo real dos valores e indicadores sobre as vendas, produtos mais vendidos, visitantes, clientes da loja, entre outros;
- Gestão de stocks e preços, com possível integração com sistemas ERP de modo a manter sempre os produtos atualizados;
- Gestão de encomendas, com regras específicas para o processamento das mesmas;
- Promoções com códigos de desconto, para campanhas específicas ou a determinados clientes e produtos;
- Otimização de indexação dos motores de busca através de técnicas de *marketing*. [15]

Resumo

Em suma, todas estas plataformas se focam nos modelos de negócio B2B e B2C, com várias soluções desde *marketing* de venda a possíveis integrações com sistemas ERP, gestão de encomendas, pagamentos automatizados, entre outros, mediante a necessidade dos clientes.

A Redicom e o Magento (versão 2) proporcionam aos seus clientes um editor simples de conteúdo *drag and drop* de forma a facilitar a criação e edição de páginas. O Prestashop também tem um módulo extra de instalação desta funcionalidade e muitos outros módulos que podem ser integrados na plataforma consoante a necessidade. Esta plataforma tem ainda a vantagem de poder ser obtida de forma gratuita. Contudo a maioria dos módulos implica custos extra, tal como o editor de conteúdo *drag and drop*, técnicas de *marketing*, carrinho abandonado, entre outros.

O Magento e o Prestashop apesar de garantirem diversas soluções para lojas *online* e serem *open source*, são plataformas genéricas, isto é, a customização das lojas *online* de cliente para cliente são limitadas ou são utilizados temas pré-definidos. Já o Redicom e a LVEngine têm mais facilidade de adaptação a cada tipo de cliente, pois o controlo sobre estas plataformas é tratado como se cada cliente fosse único de modo a fornecer as soluções mais adequadas a eles e com outro tipo de personalização.

Pode-se concluir que todas estas plataformas partilham funcionalidades similares entre si e com a plataforma descrita neste projeto. O Magento e o Prestashop são plataformas *open source* que podem ser desenvolvidas à medida do cliente, mas apresentam implementações mais genéricas enquanto as outras duas conseguem ser mais específicas de cliente para cliente. As plataformas *e-commerce* Redicom e LVEngine inserem-se melhor no conceito da plataforma reconstruída neste projeto, devido a serem customizadas para o cliente e não pelo cliente. Além disso, ambas foram construídas de raiz por parte das próprias empresas, o que se torna numa vantagem em termos de suporte contínuo e com facilidade em desenvolver funcionalidades à medida do cliente.

4. Projeto

Como referido anteriormente, o estágio consistiu na reconstrução da plataforma *e-commerce* desenvolvida pela empresa, devido essencialmente à descontinuidade tecnológica do ORM utilizado para mapeamento de dados: o Telerik DataAcess Fluent (descrito na secção 4.3.2). Para além disso, surgiram novos requisitos por parte dos clientes e nas suas variadas áreas como descrito na secção 3.2)

Este capítulo diz respeito a todo o processo que envolveu a restruturação da plataforma, desde a sua arquitetura e tecnologias utilizadas, à análise dos dados da antiga plataforma e desenvolvimento de novas funcionalidades passando pela alteração do ORM, que era o problema base da plataforma.

A secção 4.1 debruça-se sobre as análises das arquiteturas das plataformas: a do sistema, a antiga e a nova. Desta forma consegue-se ter uma melhor perceção do funcionamento das plataformas bem como a evolução tecnologia e funcional.

A secção 4.2. diz respeito a uma primeira análise à base de dados antiga, de forma a servir de base para a construção da estrutura para a nova plataforma. Para atingir esse objetivo, foi fornecida por parte da empresa a BD juntamente com os novos requisitos para a construção da nova BD, de modo a tornar possível o relacionamento entre elas. Após a análise foi pedida a implementação do novo modelo, bem como a *Structured Query Language* (*SQL*) para a migração de dados com as respetivas otimizações.

Depois de toda a análise tratou-se do mapeamento dos dados, com a ajuda do NHibernate descrito em pormenor na secção 4.3.3 de forma a garantir o acesso aos dados na plataforma, como se pode verificar na secção 4.3. Nesta secção também se pode encontrar uma descrição mais aprofundada do que é um ORM, bem como o ORM utilizado previamente e o que foi agora implementado.

Na secção 4.4 encontram-se algumas mudanças importantes entre a antiga e a nova plataforma *e-commerce* em termos de UI (*User Interface*), usabilidade e disposição de funcionalidades.

A secção 4.5 deste capítulo discrimina os testes feitos na plataforma, sejam eles unitários, integração, manuais ou de carga.

A última secção deste capítulo (secção 4.6), diz respeito à análise crítica sobre o projeto bem como propostas de melhoria.

4.1. Arquitetura e tecnologias

Nesta secção são apresentadas as diversas arquiteturas: a do sistema, a da antiga e da nova plataforma, bem como identificar as principais tecnologias utilizadas no desenvolvimento da nova plataforma.

4.1.1. Arquitetura do sistema

A plataforma *e-commerce* "zCommerce" é uma aplicação *web*, ou seja, projetada para ser utilizada por *browsers* através de um servidor *Hypertext Transfer Protocol* (HTTP) ou *Hypertext Transfer Protocol Secure* (HTTPS) e com ligação à Internet.

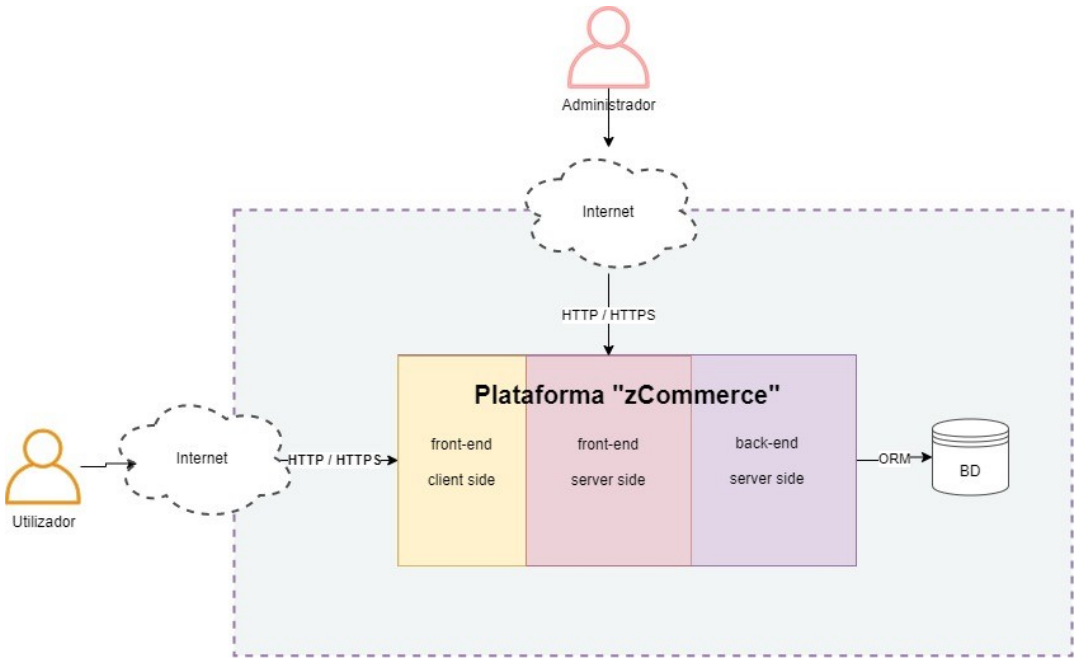


Figura 24 - Arquitetura do Sistema

Na Figura 24 encontra-se a arquitetura do sistema da plataforma, em que o utilizador com internet acede através do browser à plataforma, e esta, por sua vez, comunica com a base de dados através do ORM. Na figura encontram-se apresentados dois atores: o utilizador e o administrador. O primeiro é o cliente final que tem acesso apenas ao *front-end* da aplicação, ou seja, é lhe permitido comprar artigos na plataforma. O segundo é a pessoa que trata de toda a gestão de utilizadores, encomendas, devoluções, entre outros.

4.1.2. Arquitetura da antiga plataforma

A arquitetura da antiga plataforma encontra-se representada na Figura 25.

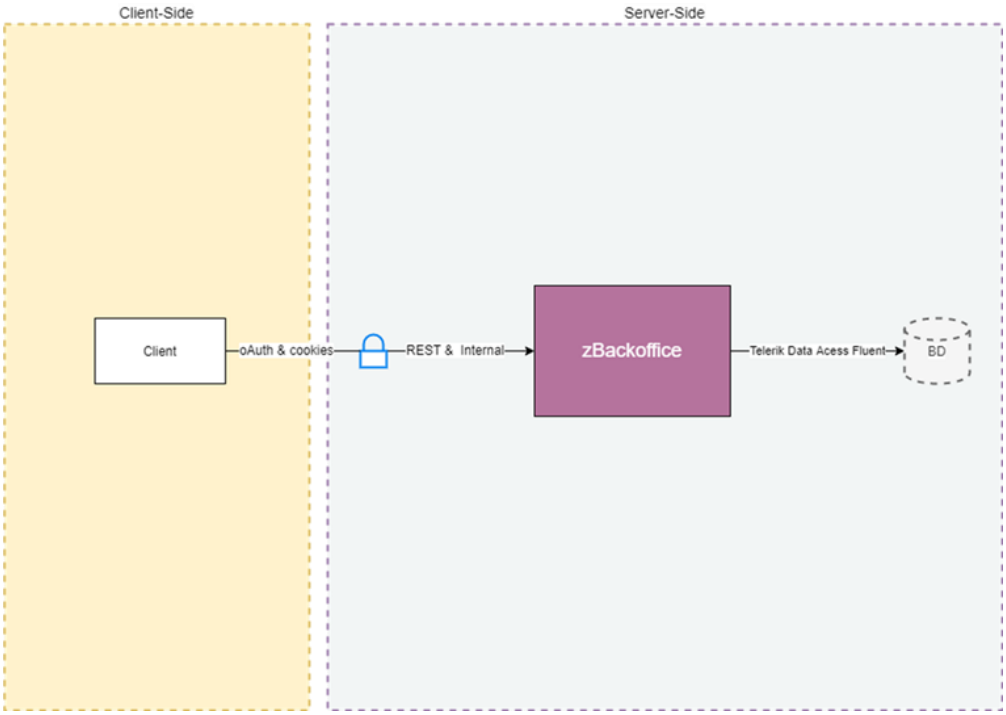


Figura 25 - Arquitetura da solução

Esta foi desenvolvida apenas com um único módulo, o "zBackoffice", que conta com uma ligação à base de dados e ao "Client". Neste módulo as comunicações com a base de dados eram estabelecidas através do ORM (Telerik Data Acess Fluent), que por sua vez, permitia o mapeamento dos dados para a plataforma (ver secção 4.3.2).

A comunicação entre o "zBackoffice" e o "Client" é feita através do protocolo HTTP ou HTTPS com base no padrão arquitetural *Representational State Transfer* (REST) e também através procedimentos internos como funções desenvolvidas internamente que permitiram a comunicação com o "Client".

O "Client" designa-se como o *front-end* da aplicação, a parte visível ao utilizador que comunica com o "zBackoffice". Este, por sua vez era responsável pela ligação à base de dados de forma a estabelecer a comunicação entre eles.

A comunicação entre o "Client" e o "zBackoffice" é estabelecida em HTTP ou HTTPS com recurso a mecanismos de autenticação como o armazenamento de *cookies*, no *browser*, de forma a guardar os dados de acesso do utilizador e manter a sua sessão. Esta solução foi implementada de forma a conseguir estabelecer autenticação de utilizadores à plataforma sem sobrecarregar o servidor com armazenamento de estados de sessão ou requisitos de acesso. O resultado disso seria sempre a criação de uma cookie com todas as informações necessárias e a cada nova solicitação de *cookie*, a sessão era des-serializada. [16] No caso de autenticação com entidades externas, como é o caso Facebook, é necessário a autorização, e como tal implementou-se o padrão oAuth. Este é um padrão aberto que permite a autenticação de utilizadores com contas como Facebook, Twitter, Google, entre outros, em *websites* de terceiros sem necessidade de expor as suas *passwords*. [17]

4.1.3. Arquitetura da nova plataforma

Na arquitetura do novo sistema, o módulo do "zBackoffice" mencionado anteriormente foi repartido em vários módulos, alguns interligados entre si para uma melhor comunicação e diferenciação entre eles.

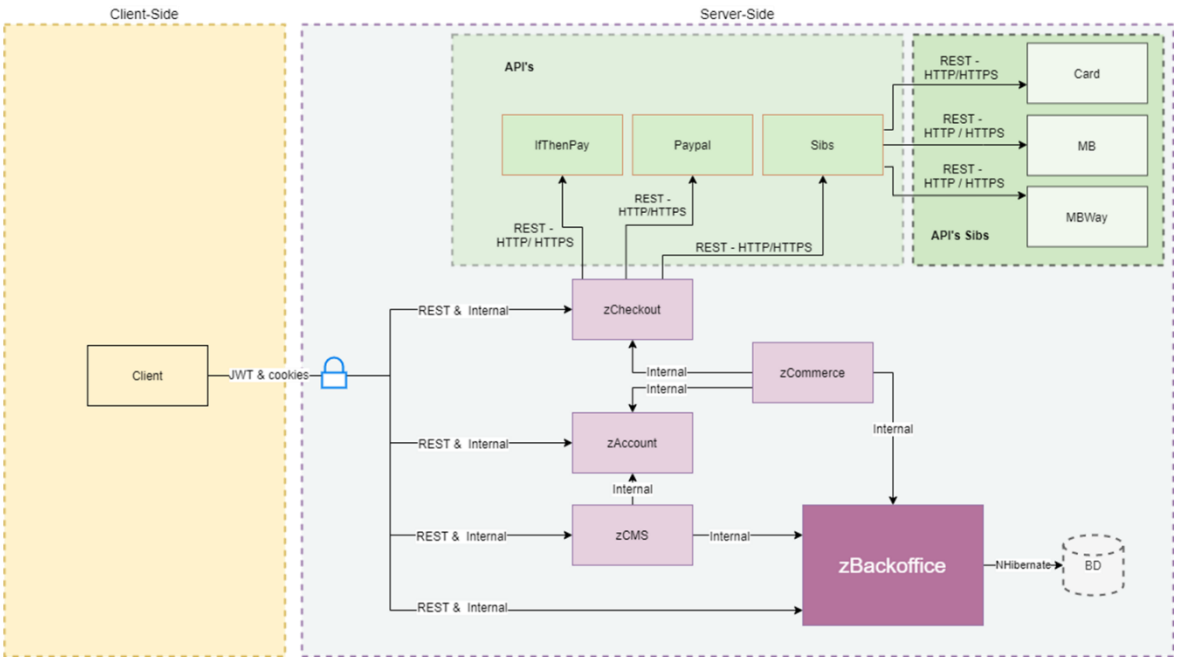


Figura 26 - Arquitetura da nova solução

A comunicação entre o "Client" e os módulos "zCheckout", "zAccount", "zCms" e "zBackoffice" é feita em HTTP ou HTTPS e utiliza autenticação com base em *cookies* e *JSON Web Tokens* (JWT). O JWT é uma forma compacta e totalmente independente de

transmitir informações com segurança entre ambas as partes como um objeto JSON, tornando-se ideal para autenticação e autorização de dados e uma mais valia nos carrinhos de compras visto permitir guardar todos os dados necessários daquele utilizador em objeto JSON de forma a não perder qualquer informação e sem ter de ser acedido por um utilizador não autorizado. [18]

Depois da autenticação, a comunicação é estabelecida através de REST, com operações HTTP ou HTTPS e procedimentos internos como funções que permitiam a comunicação com o "Client".

O "zChechout" é o módulo do carrinho de compras, que comunica entre o "zCommerce" e os módulos relativos aos métodos de pagamento por REST via HTTP ou HTTPS, como o IfTheyPay, Paypal e SIBS. No caso da SIBS, irá comunicar posteriormente com o mesmo protocolo de comunicação com as suas API's: o Card, MB e MBWay.

O módulo "zBackoffice" contém todas as funcionalidades necessárias para funcionar como um *core* da aplicação, o que torna, na prática, no modulo principal onde são instalados os outros módulos adicionais em caso de necessidade, como por exemplo: o "zAccount", "zCMS" e o "zCommerce".

A instalação do "zCommerce" depende da instalação do "zCheckout" e do "zAccount" pois são estes módulos que tornam possíveis as vendas *online*. O "zCommerce" é responsável pelos produtos da loja *online* e todas as funcionalidades ligadas a eles. Já o "zCheckout" diz respeito apenas ao processo de finalização de compra. Por fim, o "zAccount" é o módulo responsável pela criação de contas.

A instalação do módulo "zCMS" não depende da instalação do "zAccount", "zCheckout" e "zCommerce" uma vez que funciona como um *website* institucional sem necessidade de uma plataforma *e-commerce*. Apesar da independência deste módulo, o "zAccount" pode ser adicionado a ele no caso de, por exemplo, ser necessário criar uma área reservada.

Além de ser o core da aplicação e funcionar como base da mesma, o módulo "zBackoffice" estabelece a ligação com a base de dados através do ORM NHibernate. Foi a solução apresentada para substituir o Telerik Data Access Fluent. Os dois ORM's podem ser encontrados com mais detalhe nas secções 4.3.2 e 4.3.3.

4.1.4. Tecnologias

Nesta secção pretende-se dar a conhecer as tecnologias utilizadas na plataforma, quer seja do lado do servidor (*back-end*) como do lado do cliente (*front-end*) e uma breve descrição das mesmas.

Tecnologias do lado do servidor

As tecnologias do lado do servidor dizem respeito ao *backend* da plataforma. Aquando o *refactoring* da plataforma antiga, a estrutura manteve-se (*backend*/*front-end*), sofrendo apenas ajustes de código para melhoramento de performance e otimização.

Em ambas as plataformas, a antiga e a nova, o desenvolvimento foi em ASP.NET. Esta é uma tecnologia de desenvolvimento dividida por 2 tipos: *webforms* e *model view controller* (MVC). Neste projeto em concreto, optou-se por escolher o padrão MVC para o desenvolvimento. O padrão é responsável pela apresentação da aplicação, criando uma base de desenvolvimento de código que facilita a manutenção e adição de funcionalidades ao código e foca-se em 3 elementos principais: *Model* (responsável pela parte lógica aplicacional), *View* (responsável pela apresentação da interface ao utilizador) e *Controller* (responsável por controlar os outros elementos de forma a estabelecer a ligação entre eles). [19]

Tecnologias do lado do cliente

Quando se menciona tecnologias do lado do cliente, refere-se à implementação da plataforma em termos de *front-end* acessível ao utilizador final da plataforma.

Com a utilização do padrão MVC do ASP.NET no lado do servidor, torna-se possível recorrer à sintaxe *razor.* Esta é uma linguagem de marcação que permite renderizar os dados do lado do servidor para as *views* na extensão *.cshtml*, e também a renderização de código HyperText Markup Language (HTML).

Além da utilização do *razor*, continuou-se a utilizar o *HTML, e cascading style sheets* (CSS). Este último foi substituído por *Sassy Cascading Style Sheet* (SCSS*)* na nova plataforma*,* o que possibilitou o uso de variáveis e funções de forma a otimizar código.

Por correr do lado do cliente, o *vue.js* também foi utilizado em algumas partes do código, como é o caso das grelhas de listagem de dados de forma a tornar a interação do utilizador com a plataforma mais otimizada, flexível e intuitiva. O *Vue.js* apresenta-se como uma *framework* de *front-end* em *javascript* para construção de *interfaces* de utilizador. Foca-se nas camadas visuais da aplicação, as *views* e permite a integração com qualquer biblioteca ou projeto. [20] Apenas foi utilizado no projeto em algumas partes de código, porque ainda requerer alguma curva de aprendizagem que nem toda a equipa tinha na totalidade. Esta mudança para *vue.js* aconteceu de forma a melhorar a performance em termos de navegação e evitar uma total atualização das páginas. Desta forma foi possível minimizar os pedidos HTTP ou HTTPS ao servidor, o que, na plataforma antiga se tornava um problema.

4.2. Análise à base de dados

Esta seção descreve a análise efetuada à base de dados antiga e à construção da nova, segundo os requisitos já existentes e os novos requisitos.

A análise foi iniciada através de um ficheiro *SQL* da base de dados antiga que foi importado no *software* de desenvolvimento de base de dados *MySQL Workbench.* Este *software* é uma ferramenta útil no que diz respeito às representações gráficas para quem deseja desenvolver base de dados uma vez que permite não só visualizar graficamente o ficheiro *SQL* importado, como desenhar novas tabelas e fazer o respetivo relacionamento entre elas, migrações e comparações de BD, entre outros. [21]

De acordo com a análise da base dados da antiga plataforma, foi possível identificar vários problemas, nomeadamente relacionados com funcionalidades que não correspondiam na totalidade aos requisitos dos clientes, a descontinuação de funcionalidades e a necessidade de implementação de novas. Os diagramas das bases de dados das duas plataformas encontram-se em anexo, em que o Anexo A corresponde à base de dados da antiga plataforma e o Anexo B à base de dados da nova plataforma. Devido à sua complexidade e extensão, torna-se impraticável a representação de ambos os diagramas numa página única, neste sentido foram repartidos em várias páginas, o que originou algumas dificuldades na visualização dos relacionamentos entre tabelas.

4.2.1. Tabelas descontinuadas

Ao analisar a antiga base de dados e segundo os novos requisitos apresentados por parte dos clientes, algumas tabelas da base de dados tornaram-se obsoletas devido às

funcionalidades terem sido descontinuadas ou renovadas com diferentes abordagens. Como tal, pode-se apontar como exemplo de algumas das tabelas descontinuadas a "newsletter" e "reviews".

Relativamente a regras e políticas de proteção de dados, estas foram alteradas em maio de 2018 de forma a diminuir o risco de troca de dados pessoais dos utilizadores de serviços. Devido a isso, a encriptação de dados na base de dados de lojas *online* passou a ser obrigatória, pois só assim se consegue garantir a proteção e confidencialidade dos dados e utilizadores. Com base nesta nova obrigação decidiu-se extinguir a tabela "newsletter" pois além de não seguir as novas normas relativas à proteção de dados, também já existia outra maneira de gerir os dados de utilizadores para a subscrição de *newsletter*: os formulários dinâmicos. São formulários totalmente construídos na plataforma pelo administrador, sem qualquer intervenção por parte do programador. Na antiga plataforma eles já existiam, mas era necessário a implementação da encriptação dos dados para salvaguardar a privacidade dos mesmos.

Como já dito anteriormente, uma das funcionalidades descontinuadas foi a "reviews*"*. Esta funcionalidade permitia adicionar comentários aos produtos de forma a tornar esse produto mais popular e com opiniões por parte dos clientes. As *reviews* hoje em dia são feitas através das redes sociais, onde são colocados comentários e atribuídos *rankings* aos produtos, sendo assim não só desnecessária esta funcionalidade na plataforma, como uma fonte de trabalho adicional para o cliente, visto ter de gerir os comentários por cada produto. Apesar da descontinuação, manteve-se um sistema de *ranking* simples sem necessidade de comentários.

4.2.2. Encomendas

O processo de fazer uma encomenda foi estruturado segundo os requisitos solicitados por parte dos clientes, mas também de forma a melhorar algumas funcionalidades como o processo de compra do produto e a gestão da encomenda.

Era já possível fazer encomendas na plataforma com base no país e distrito, mas não com base no código postal, ou seja, com recurso à localização exata. Para possibilitar esse novo requisito, a tabela associada à localização foi reestruturada e renomeada como se pode verificar nas Figura 27 e Figura 28, que representam, respetivamente, o antes e o depois da reestruturação.

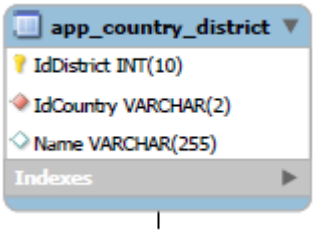


Figura 27 - Tabela "app_country_district" da base de dados

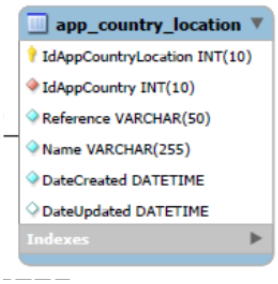


Figura 28 - Tabela "app_country_location"

Cupões

Os cupões de desconto na antiga plataforma apenas podiam ser atribuídos na totalidade da encomenda e não a artigos em específicos, por percentagens de desconto ou a categorias específicas. Como tal, as tabelas apresentadas na Figura 29 e Figura 30 foram renovadas de forma a suportar essa funcionalidade.

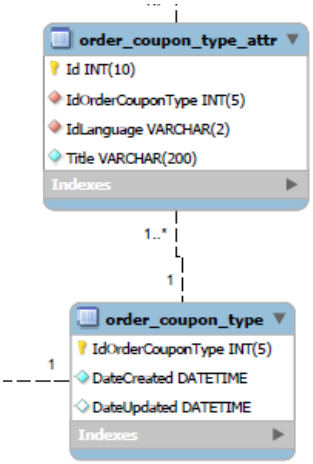


Figura 29 - Tabelas "order_coupon_type_attr" e "order_coupon_type"

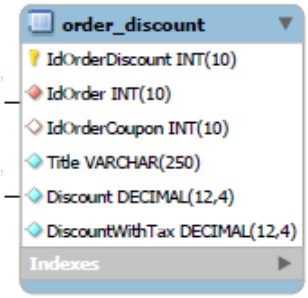


Figura 30 - Tabela "order_discount" da base de dados

Devoluções

Ao fazer uma devolução na antiga plataforma, apenas algumas informações sobre esse produto é que eram guardadas na base de dados o que, por vezes, se revelava insuficiente, nomeadamente em situações de compra de um produto com uso de um cupão de desconto, em que essa informação não era recolhida. De forma a contornar esta situação e garantir todas as informações necessárias da encomenda, como por exemplo, o tipo de pagamento, a identificação de entrega de encomenda, cupões, entre outros, foram alteradas as tabelas e dado um nome mais sugestivo “line”. Na Figura 31 pode verificar-se o antes e na Figura 32 o depois das tabelas de encomendas e devoluções.

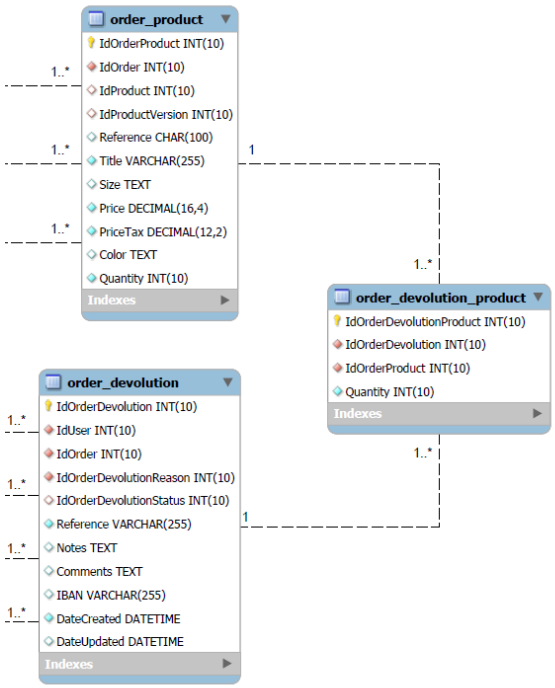


Figura 31 - Esquema das tabelas "order_product", "order_devolution_product" e "order_devolution" da base de dados

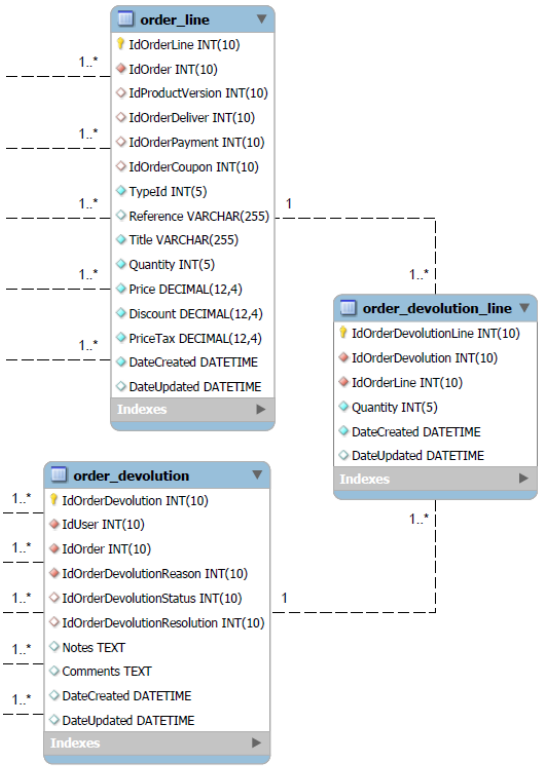


Figura 32 - Esquema das tabelas "order_line", "order_devolution_line" e "order_devolution" da nova base de dados

4.2.3. Produtos

Foi nos produtos que a maioria das alterações surgiram, pois ao longo do tempo os clientes tornaram-se mais exigentes nos seus requisitos e a plataforma antiga não suportava na totalidade esses requisitos.

Versões de produtos

A plataforma antiga foi concebida para suportar versões de produtos, ou seja, suportar a possibilidade de existir várias combinações do mesmo produto só que apenas de cores e tamanhos. Isto permitia reutilizar o mesmo produto com as várias combinações possíveis em vez de criar o mesmo produto várias vezes para corresponder às mesmas. Esta situação não era considerada uma solução que satisfizesse todos os clientes, pois nem todos os negócios funcionam apenas com cores ou tamanhos. Entre outras características que não eram garantidas por este modelo pode-se apontar como exemplo: combinações de cores com dimensões e materiais. Na Figura 33 e Figura 34 pode visualizar-se as tabelas com as combinações disponíveis na base de dados antiga.

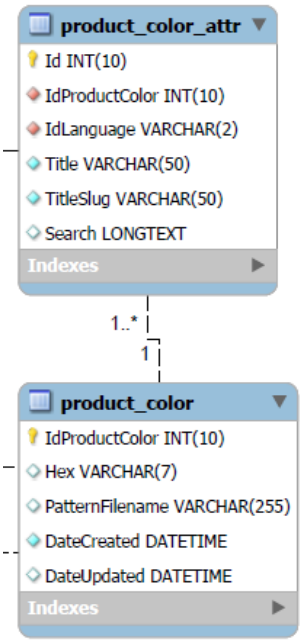


Figura 33 - Tabelas "product_color_attr" e "product_color" da base de dados

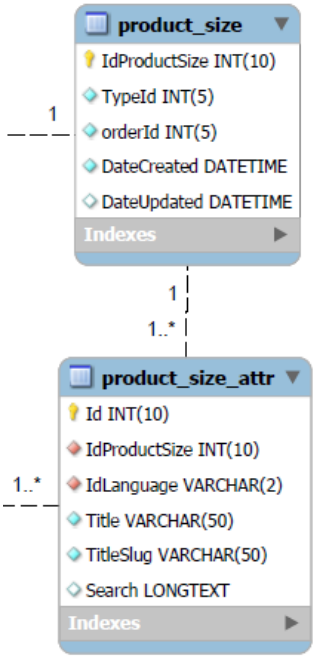


Figura 34 - Tabelas "product_size" e "product_size_attr" da base de dados

As Figura 35 e Figura 36 representam a solução deste problema de versionamento, solução que passou por se criar novas tabelas para suportar esses requisitos avaliando o produto em 3 tipos de versão diferentes: versão simples, versão por descritivos e versão por grupo de opções.

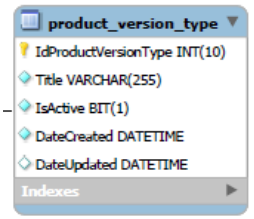


Figura 35 - Tabela - "product_version_type" da nova base de dados

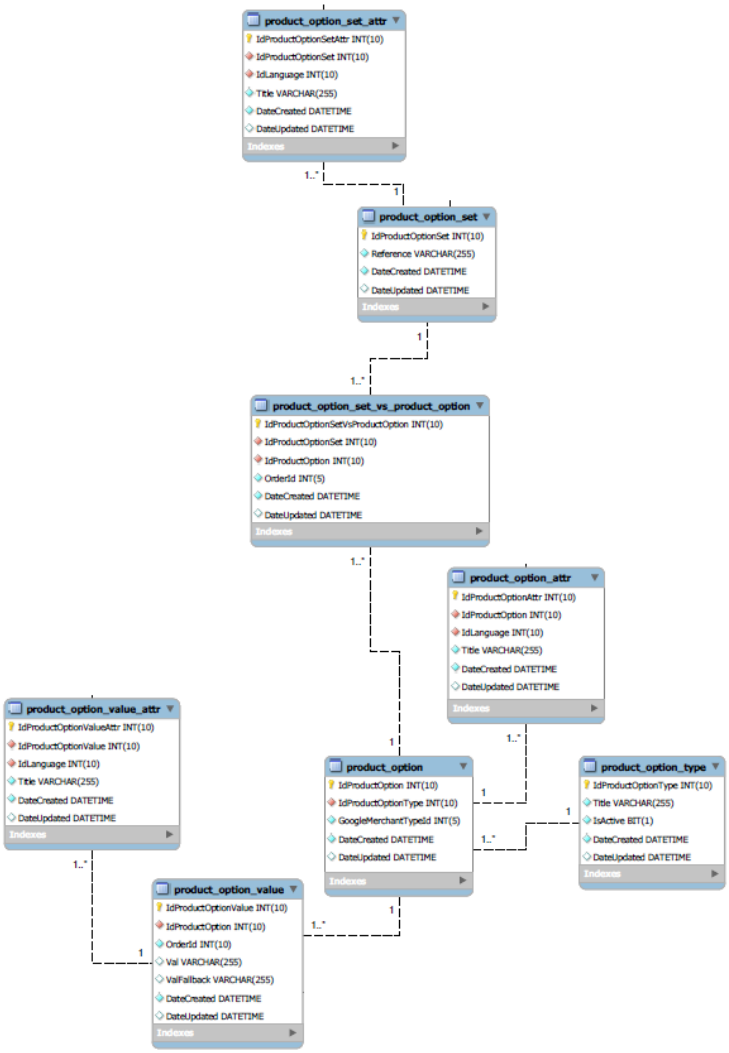


Figura 36 - Tabelas relativas às opções de produto danova base de dados

Wishlists

As *wishlists* nas lojas *online* é uma forma de guardar produtos que determinado utilizador gostou. Na plataforma antiga apenas se podia adicionar à *wishlist* a primeira versão do produto encontrada, ou seja, nem sempre era possível obter na *wishlist* o tamanho e cor desejado daquele produto. Para colmatar este tipo de situações, as tabelas foram reescritas de forma a conseguir adicionar qualquer versão do produto aos favoritos. Na Figura 37 estão representadas as estruturas das tabelas *wishlists* originais, e na Figura 38 as tabelas reconstruídas.

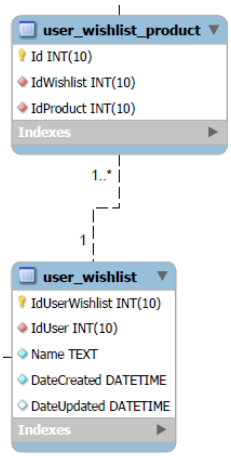


Figura 37 - Tabelas "user_wishlist_product " e "user_wishlist" da base de dados

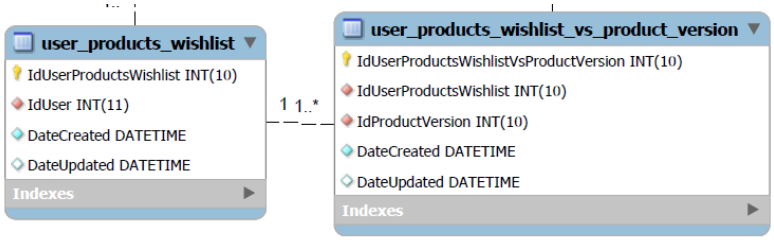


Figura 38 - Tabelas "user_products_wishlist" e "user_products_wishlist_vs_product_version" da nova base de dados

4.2.4. Utilizador

A estrutura da tabela do utilizador também foi alterada, pois nesta nova base de dados um utilizador está associado a uma entidade e é gerido por ela, como podemos perceber na secção 4.2.5. Além disso, como já foi dito na secção 4.2.1 a lei foi alterada recentemente relativamente às políticas de privacidade e proteção de dados, e a obrigatoriedade da plataforma dar seguimento a essas novas regras faz com que o utilizador seja sempre obrigado a aceitar os termos de uso e condições da plataforma bem como a recusa ou autorização de tratamento dos seus dados. Surgem logo no início de criação de conta, *login* ou formulários no qual é necessário o preenchimento de dados pessoais. Na Figura 39 pode verificar-se a tabela acrescentada relativa à aceitação dos termos e privacidade dos utilizadores.

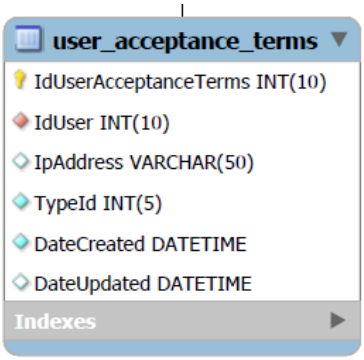


Figura 39 - Tabela "user_acceptance_terms" da nova base de dados

Carrinho

A estrutura do carrinho também mudou um pouco, pois na base de dados antiga guardava-se apenas uma cookie do carrinho com a versão base do produto, isto é, a primeira versão encontrada e o *Internet Protocol* (IP) relativo aquele utilizador. Por falta de informações, decidiu-se acrescentar uma nova tabela que se encontra representada na Figura 40 que guarda além dessas informações, outras relativas à versão do produto, taxas, promoções, cupões aplicados, entre outros. Isto permitiu não só reter todas as informações necessárias como também serem aplicadas técnicas de *marketing* sobre o carrinho.

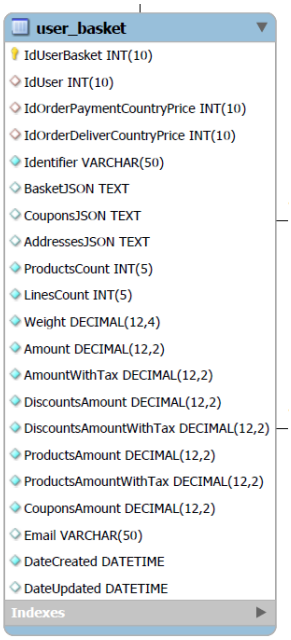


Figura 40 - Tabela "user_basket" da nova base de dados

4.2.5. Novas Funcionalidades

Depois desta análise à antiga base de dados e da descrição das tabelas que foram descontinuadas ou substituídas por outras, segue-se a explicação relativa a novas funcionalidades da plataforma, como os *templates* de email, gestão de entidades, gestão de produtos e métodos de entrega por zona.

Emails

Uma das necessidades e requisitos considerados pelos clientes foram os emails enviados pelo sistema referentes às encomendas, alterações de encomendas, devoluções, entre outros, não poderem ser customizados por eles próprios, visto que na antiga plataforma estes emails seguiam um *template* não personalizável. Nas tabelas representadas na Figura 41 pode verificar-se esta nova funcionalidade que permite ao administrador da plataforma criar o seu próprio *template* nas várias línguas para cada tipo de email.

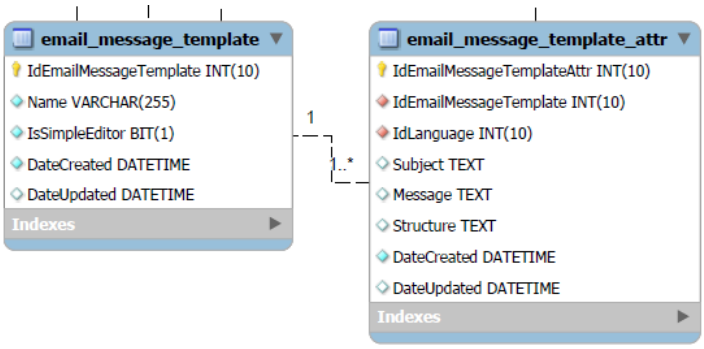


Figura 41 - Tabelas "email_messagem_template" e "email_mesage_template_attr" da nova base de dados

Entidades

Os utilizadores na antiga base de dados eram separados por grupos de forma a conseguir distinguir entre um utilizador normal e outro tipo de utilizador, como por exemplo, um fornecedor. Nesta nova base de dados, a gestão de utilizadores foi alterada de forma a poderem ser geridos por entidades e grupo de entidades, conseguindo definir características diferentes para cada grupo, como as suas próprias moradas, descontos e contactos. Na Figura 42 estão representadas as tabelas das entidades e os seus relacionamentos na nova base de dados.

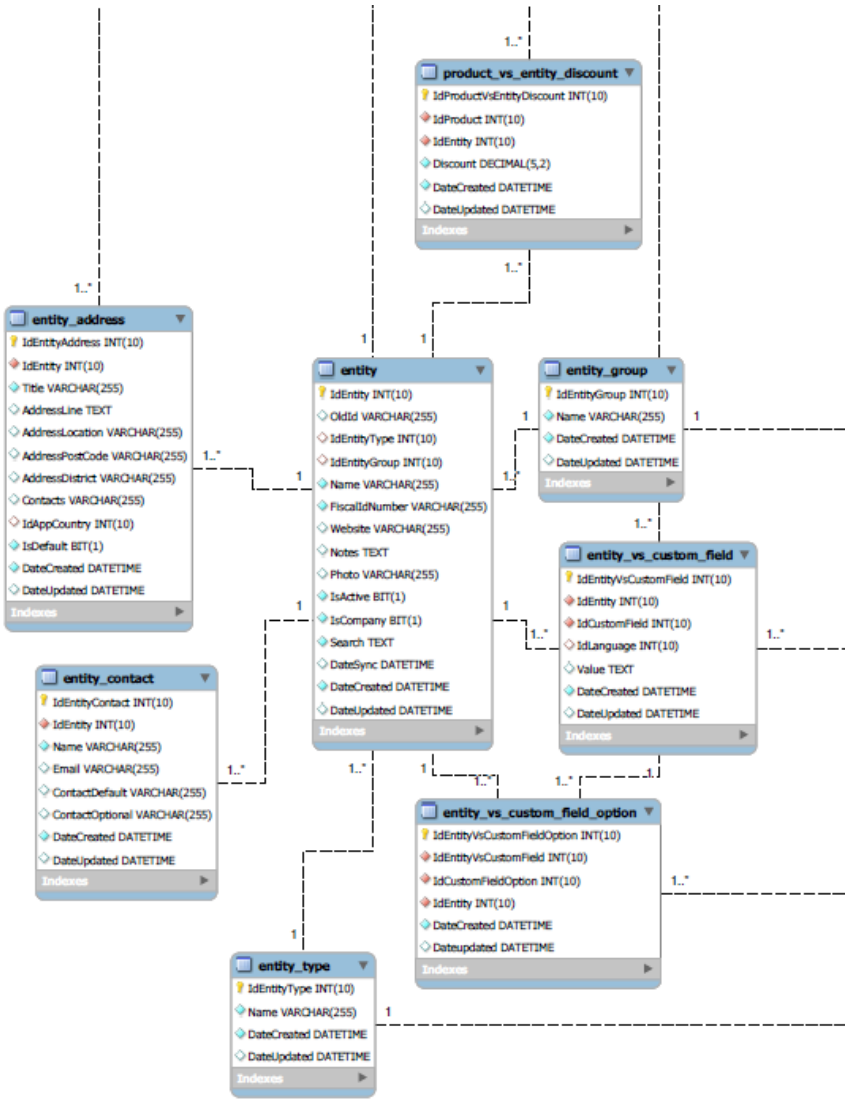


Figura 42 - Tabelas relacionadas com as entidades da nova base de dados

Produtos

Como descrito na secção 4.2.3, foi nos produtos onde existiram mais mudanças. Assim, foram acrescentadas novas funcionalidades aos produtos como a disponibilidade, as opções de versão, métodos de entrega por zona, entre outras. Estas alterações permitiram o alcance a outro tipo de clientes e suas áreas de negócio permitindo assim uma plataforma *e-commerce* versátil a qualquer tipo de negócio.

Disponibilidade de produtos

A disponibilidade do produto na antiga plataforma apenas tinha dois estados: disponível ou indisponível e este era gerido através do stock do produto. Nesta nova plataforma foi acrescentado um novo estado, o sob consulta, e a gestão passou a ser feita de forma independe do stock, o que prevê que o mesmo produto pode ter stock e estar sob consulta. Na Figura 43 pode verificar-se as novas tabelas relativas à implementação da disponibilidade do produto, bem como a possibilidade desses estados serem *multilingue*.

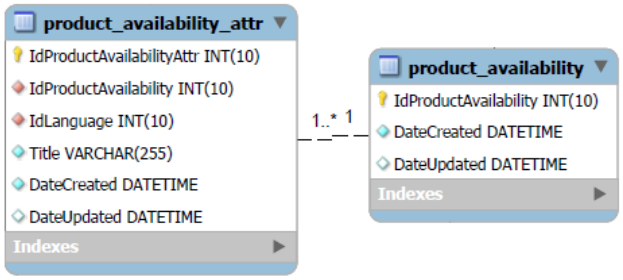


Figura 43 - Tabelas "product_availability_attr" e "product_availability" da nova base de dados

Opções de versão de produto

A grande mudança nos produtos surge com os tipos de produto, opções e preços, pois um produto pode ter três tipos de versão:

- Versão simples, em que o produto tem apenas tem um preço;
- Versão por descritivos, em que o produto é composto, por exemplo, por vários tamanhos e o preço difere consoante o tamanho e quantidades;
- Grupo de opções, em que o produto pode ter vários grupos de opções (por exemplo, cor, tamanho e dimensão), e pode obter todas as combinações possíveis e vários preços;

A Figura 44 apresenta a tabela relativa aos tipos de versões.

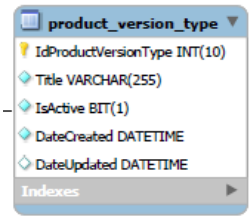


Figura 44 - Tabela - "product_version_type" da nova base de dados

Ao ter em conta estes novos dados, surgiu a necessidade de colocar preços e fazer conjuntos por versões e permitir novas combinações de forma a que possibilitassem a adição de versões em massa e colocar respetivos preços por versão. Na Figura 45 encontram-se representadas as tabelas que dizem respeito às opções de produto e aos seus relacionamentos, já na Figura 46 as tabelas representadas são as versões do produto por preço.

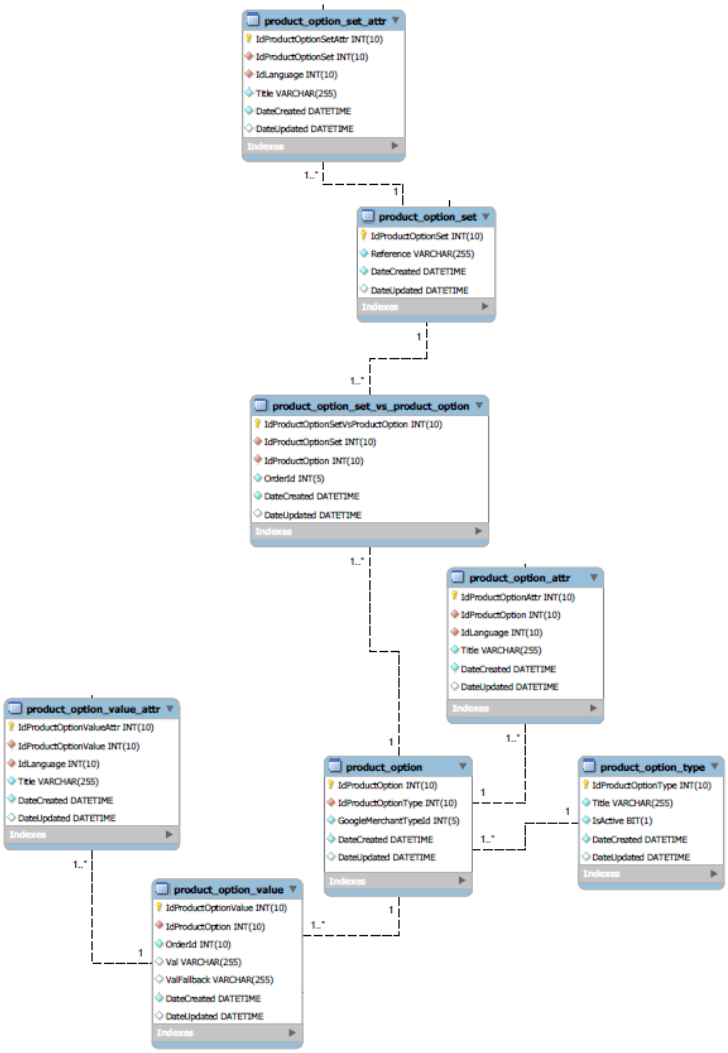


Figura 45 - Tabelas relativas as opções de produto da nova base de dados

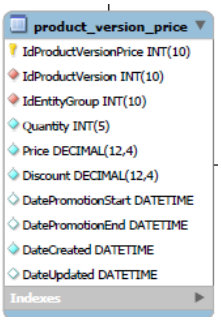


Figura 46 - Tabela "product_version_price" da nova base de dados

A criação das opções permitiu o relacionamento com os próprios produtos que se encontram representados na Figura 47 e Figura 48. Nas figuras pode verificar-se também o relacionamento com as imagens que criou a opção de que quando selecionada uma cor, esta pudesse de forma automática identificar a imagem associada.

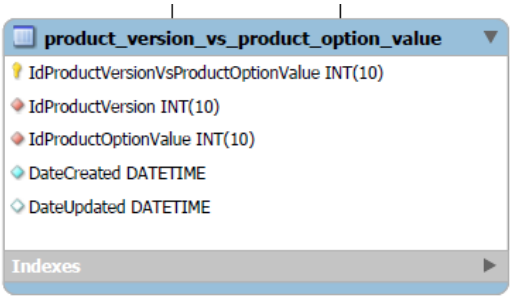


Figura 47 - Tabela "product_version_vs_product_option_value" da nova base de dados

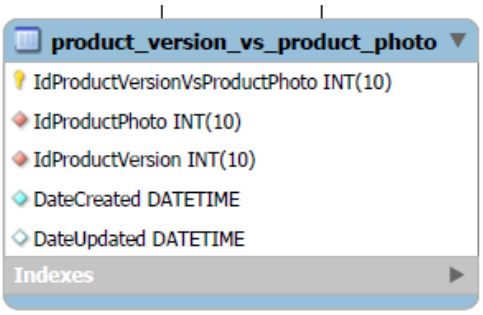


Figura 48 - Tabela "product_version_vs_product_photo" da nova base de dados

Métodos de entrega por zona

No caso das encomendas o esquema também sofreu algumas alterações: as entregas passaram a ser por zona em vez de ser só por país. Deste modo, tornou-se possível a deteção de moradas através do código postal de determinada zona do país. Na Figura 49 é possível verificar-se o relacionamento entre as tabelas.

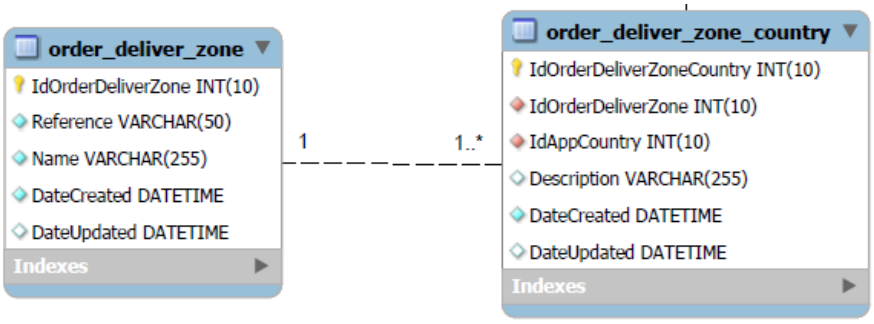


Figura 49 - Tabelas "order_deliver_zone" e "order_deliver_zone_country" da nova base de dados

A tabela identificada através da Figura 50 (order source) diz respeito à fonte da encomenda, isto é, se foi efetuada por multibanco, paypal, entre outros.

Já no caso da tabela da Figura 51, guarda o estado relativo à encomenda, ou seja, se está paga, pendente, em despacho, anulada ou entregue.

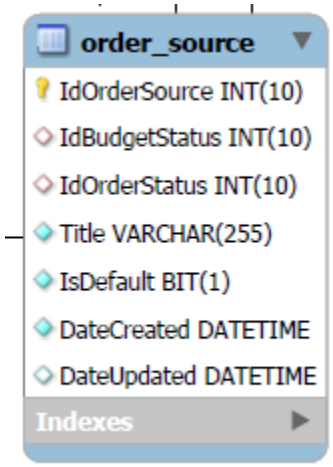


Figura 50 - Tabela "order_source" da nova base de dados

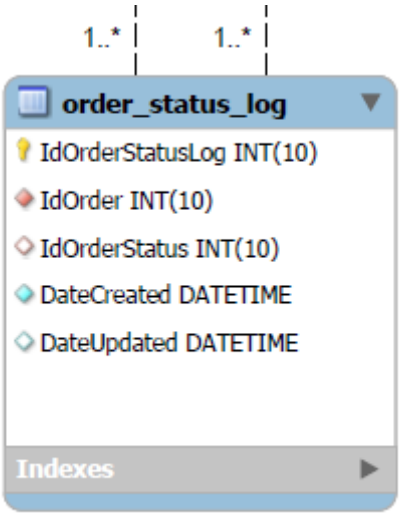


Figura 51 - Tabela "order_status_log" da nova base de dados

Nas devoluções, é questionado o motivo de devolução do item e com isso inicia-se um processo de devolução, representado na Figura 52. Cada processo é composto por várias fases: em análise, em tratamento, resolvida ou anulada.

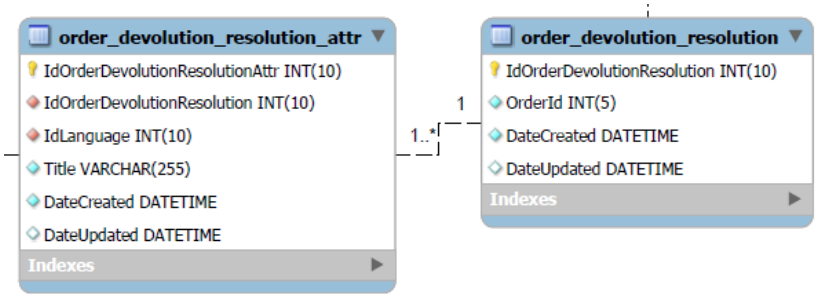


Figura 52 - Tabelas "order_devolution_resolution_attr" e "order_devolution_resolution" da nova base de dados"

4.3. Acesso a dados

Depois da análise e elaboração do script *SQL* com a estrutura da nova base de dados tratou-se dos acessos aos dados através de repositórios de forma a conseguir aceder aos dados de forma fácil, rápida e reutilizável para todas operações necessárias.

A implementação da estrutura e mapeamento dos dados da nova plataforma foi feita em NHibernate (ver seção 4.3.3) que veio substituir o Telerik Data Access Fluent descontinuado em 2014 como já referido (ver secção 4.3.2).

A secção 4.3.4 diz respeito ao processo de mapeamento utilizado para tratamento dos dados para a nova plataforma *e-commerce* através do ORM escolhido.

4.3.1. O que é um ORM?

Antes de se explicar cada um dos ORM's utilizados, convém perceber o que realmente é um ORM.

Um ORM ou *Object-Relational Mapping* é uma técnica de mapeamento que permite fazer uma associação dos objetos com os dados que estes representam. Existem vários *softwares* de mapeamento de dados que apresentam diferentes implementações, mas todas elas focadas no mesmo objetivo: fornecer o mapeamento dos dados presentes nas bases de dados relacionais em instâncias (objetos) de classes. As diferenças entre a forma como os dados são mantidos e geridos entre uma aplicação desenvolvida com base numa linguagem de programação orientada a objetos e uma base de dados relacional são:

- Os objetos são temporários e instanciados em tempo de execução, enquanto os dados no modelo relacional são persistentes;
- Os objetos não têm uma estrutura fixa enquanto os dados da BD têm;
- A linguagem *SQL* utilizada para interagir com as bases de dados relacionais é uma linguagem *standard*, enquanto nas linguagens orientadas a objetos, cada uma tem a sua própria sintaxe para chamada de métodos. [22]

Os *softwares* de ORM são responsáveis pelo cruzamento entre o objeto e a relação de forma a fornecer uma interface no caso das linguagens de programação orientadas a objetos. Desta forma, apenas é necessário fazer chamadas aos métodos e estes interagem diretamente com a BD na criação e execução de *SQL* sem necessidade de manipulação direta de *SQL*.

Um ORM tem a capacidade de criar modelos de dados internos que podem variar conforme a implementação e as chamadas à BD são convertidas através desse modelo interno antes de serem executadas. Esta funcionalidade permite que não seja necessário código adicional para conexão à BD ou fazer *querys SQL* para acesso aos dados porque o ORM já trata dessa questão. [22]

Uma das principais vantagens em usar um ORM reside na necessidade de escrever menos código e assim conseguir ter uma maior produtividade sobre a aplicação, tomando o código mais elegante e de fácil manutenção. [23]

Neste projeto a plataforma antiga utilizou o ORM Telerik Data Acess Fluent, e na nova plataforma foi implementado o ORM NHibernate. Ambos se encontram descritos nas secções 4.3.2 e 4.3.3.

4.3.2. Telerik Data Access Fluent

O Telerik Data Access é um *software* que permite o desenvolvimento de aplicações de *software* orientadas a dados, com o principal objetivo de resolver os problemas de incompatibilidades de impedância objeto-relacional. O uso deste *software* permite aos programadores de aplicações orientadas a dados poderem concentrarem-se apenas na lógica dos seus problemas e não na resolução de problemas provocados pelos mecanismos de armazenamento ou recuperação de dados, pois os dados são trabalhados na forma de objetos sem necessidade de interagir diretamente com as tabelas e colunas da BD. [24]

Um dos principais módulos utilizados neste projeto foi o Telerik Data Access Fluent, desenvolvido com o objetivo de mapear os dados por código. O módulo ajuda a definir o tipo de mapeamento a fazer e a executar operações de criação, recuperação, atualização e eliminação de classes relativas ao mapeamento dados da BD. O *software* foi utilizado para o mapeamento das tabelas da BD através do diagrama do modelo relacional, de forma simples e sem necessidade de fazer código adicional, pois no diagrama era possível fazer toda a atualização da informação vinda da BD. [25]

Contudo, o Telerik Data Acess Fluent foi descontinuado a partir de 2014, o que levou à procura de outro ORM para a plataforma.

4.3.3. NHibernate

O NHibernate é uma das soluções de mapeamento de dados para plataformas cujo desenvolvimento é em .NET e é uma solução de ORM *open source*. [26] Esta solução utiliza descrições XML (*Extensible Markup Language*) das entidades e seus relacionamentos, de forma a gerar automaticamente código *SQL* para carregar e guardar os objetos. Paralelamente também suporta a funcionalidade de ser o próprio programador a fazer o mapeamento dos dados e seus atributos através do seu código fonte.

Este ORM contém classes persistentes, ou seja, não é necessário implementar nenhuma interface ou heranças de outras classes e o objeto também não tem de seguir um modelo de programação restrito. Isto torna possível transportar a lógica de negócio utilizando objetos .NET simples ou objetos que diferem de idioma.

O NHibernate é conhecido não só pela sua facilidade de mapeamentos e acesso a dados como também pelas opções de mapeamento rápidas, os atributos. São interfaces utilizadas em .NET que facilitam o mapeamento dos dados que substituem o mapeamento com base em XML. [27]

Para escolha da tecnologia ORM para a plataforma *e-commerce* foram tidos em consideração duas hipóteses: o Entity Framework (ORM da Microsoft) e NHibernate. Na altura da decisão sobre que ORM utilizar, o NHibernate era o mais recomendado pela comunidade devido à fácil utilização e rapidez no acesso e mapeamento dos dados, e como tal, a empresa decidiu optar por este.

4.3.4. Implementação NHibernate

Nesta secção encontra-se a estrutura aplicacional do projeto bem como o processo de desenvolvimento relativo ao mapeamento dos dados da BD para a plataforma de forma a ficarem acessíveis no lado aplicacional através do ORM NHibernate.

A Estrutura da plataforma

A estrutura do projeto utilizada para o mapeamento dos dados foi feita com recurso à base estrutural já anteriormente desenvolvida, ou seja, o desenvolvimento de uma aplicação dentro da solução para a gestão de repositórios e entidades. Nesta nova plataforma desenvolveu-se uma aplicação para cada módulo dentro da mesma solução, de forma a gerir os dados respetivos a cada módulo como demonstra a Figura 53.

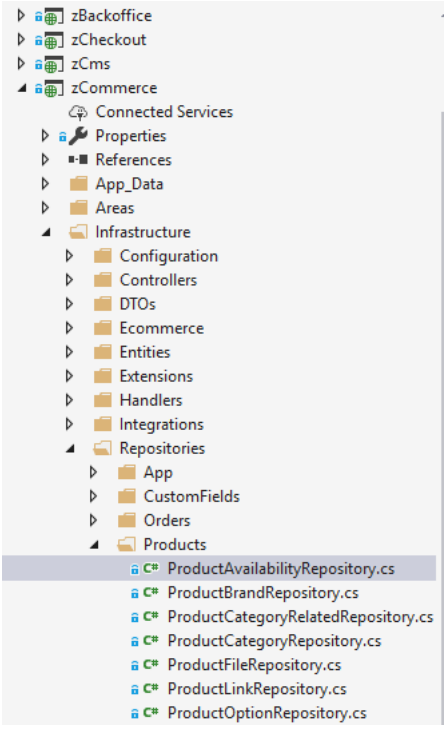


Figura 53 - Plataforma – repositórios

Assim, conseguiu-se fazer mais facilmente a distinção entre o que faz parte da base do projeto (zBackoffice) e de cada módulo: *website* institucional (zCms), carrinho de compras (zCheckout), minha conta (zAccount) e *e-commerce* (zCommerce).

A separação das aplicações foi feita também com o intuito de fazer, posteriormente, atualizações da plataforma *e-commerce* de forma mais simples e rápida através de *packages*, característica que a plataforma antiga não possuía e, como tal, as alterações ou correções eram, quase sempre, um processo moroso.

Ao agrupamento de funcionalidades em cada uma das aplicações deu-se o nome de "Infrastructure" e cada aplicação tem os seus próprios repositórios (Repositories) e entidades (Entities). Nos repositórios foram feitas as operações necessárias para obter os dados da BD, como por exemplo: as entidades, os produtos, os formulários, os utilizadores, entre outros (ver Figura 53).

As "Entities" são responsáveis por fazer o mapeamento dos dados da base de dados. Depois desse mapeamento são feitas as operações nos repositórios, como métodos e funções de forma a obter ou guardar informações na base de dados.

NHibernate e Mapeamento

A utilização do NHibernate para o mapeamento dos dados requer que os campos sejam criados diretamente na BD e posteriormente no lado aplicacional que seja criada uma classe específica com todas as especificações de cada campo. Na Figura 54 encontra-se o mapeamento da tabela "languages" da BD através no NHibernate.

```
using NHibernate.Mapping.ByCode.Conformist;

namespace ApplicationLib.Entities
{
    public class Language : BaseEntity
    {
        public Language()
        {
        }

        public virtual int IdLanguage { get; set; }

        public virtual string Name { get; set; }
        public virtual string Reference { get; set; }
        public virtual string Code { get; set; }
        public virtual string Direction { get; set; }
        public virtual bool IsActive { get; set; }
        public virtual bool IsDefault { get; set; }

        public new virtual bool IsNew => IdLanguage == 0;

        public override IList<string> Validate()
        {
            var errors = new List<string>();

            if (!Utilities.Validations.IsValidString(this.Name, false, 1, 150))
                errors.Add(LanguageService.Get("Error_NameInvalid", ApplicationLib.Memory.Session.IdLanguage, "Nome inválido"));

            if (!Utilities.Validations.IsValidString(this.Code, false, 1, 10))
                errors.Add("Codigo inválido");

            if (!Utilities.Validations.IsValidString(this.Direction, false, 1, 10))
                errors.Add("Direção inválido");

            if (!Utilities.Validations.IsValidString(this.Reference, false, 1, 2))
                errors.Add("Referência inválido");

            return errors;
        }
    }
}
```

Figura 54 - Mapeamento da tabela "Language" da Base de dados

Ao nome da classe do mapeamento chamou-se "Language", tal como a sua referência na BD. Nesta classe foram criadas variáveis do tipo "virtual" com os nomes e tipos exatamente iguais aos que foram criados na BD de forma a ser facilitado o mapeamento, como demonstra a Figura 55.

```
46  namespace ApplicationLib.Entities.Mappings
47  {
48      public class LanguageMapping : ClassMapping<Language>
49      {
50          public LanguageMapping()
51          {
52              Table("language");
53              Lazy(true);
54
55              Id( x => x.IdLanguage, x => { x.Column("IdLanguage"); x.Generator(Generators.Identity); });
56
57              Property( x => x.Name, x => { x.Column("Name"); x.NotNullable(true); x.Length(150);});
58              Property( x => x.Reference, x => { x.Column("Reference"); x.NotNullable(true); x.Length(2); });
59              Property( x => x.Code, x => { x.Column("Code"); x.NotNullable(true); x.Length(10);});
60              Property( x => x.Direction, x => { x.Column("Direction"); x.NotNullable(true); x.Length(10);});
61              Property( x => x.IsActive, x => { x.Column("IsActive"); x.NotNullable(true); });
62              Property( x => x.IsDefault, x => { x.Column("IsDefault"); x.NotNullable(true); });
63              Property( x => x.DateCreated, x => { x.Column("DateCreated"); x.NotNullable(true); });
64              Property( x => x.DateUpdated, x => { x.Column("DateUpdated"); x.NotNullable(false); });
65          }
66      }
67  }
68
```

Figura 55 - Mapeamento da tabela "Language" da Base de dados

O mapeamento dos dados consiste em identificar qual a tabela a ser mapeada e com recurso às variáveis virtuais, conseguir fazer a correspondência entre eles e a BD.

Todo o processo é feito através de *lambda expressions* de modo a simplificar todo o processo sem ser necessário recorrer ao uso de *SQL*. As *lambda expressions,* ou funções anónimas são expressões que especificam um objeto de uma função anónima. Quando utilizada uma expressão *lambda,* é criada uma função automaticamente pelo compilador com recurso a variáveis locais que podem ser passadas como argumento ou devolver valores. [28] Um exemplo de *lambda expression* encontra-se representado na Figura 56 referente ao mapeamento da coluna “IdLanguage” da tabela “Language” da base de dados. O exemplo que se segue é composto por uma variável que identifica o objeto que é utilizado para fazer a ligação entre a varável virtual “IdLanguage” e a coluna da base de dados “IdLanguage”:

```
Id(x => x.IdLanguage, x => { x.Column("IdLanguage"); x.Generator(Generators.Identity); });
```

Figura 56 - *Lambda expression* do mapeamento da coluna "IdLanguage" da tabela “Language” da base de dados.

Todos os campos foram mapeados de acordo com os dados inseridos na BD para que o mapeamento seja considerado correto. Assim em situações em que um campo da BD tenha o valor a *null*, este valor tem de ser na mesma representado na classe do mapeamento para que a correspondência seja feita corretamente.

4.4. UI da Plataforma

Nesta secção encontram-se descritas algumas diferenças entre a antiga plataforma e a nova em termos de UI. Além disso, também estão descritas as tecnologias utilizadas na nova plataforma e o processo relativo ao desenvolvimento da interface (*front-end*).

Em ambas as plataformas - a antiga e a nova - a UI foi desenvolvida de forma a ter em conta a experiência de utilizador, ou seja, simplificar a sua utilização em termos de usabilidade e criação de conteúdos. Na antiga plataforma, cada item no *front-end* aplicacional (visualização de utilizador final) era composto por um botão de acesso direto aos detalhes do mesmo, o que facilitava a sua edição como demonstra a Figura 57.

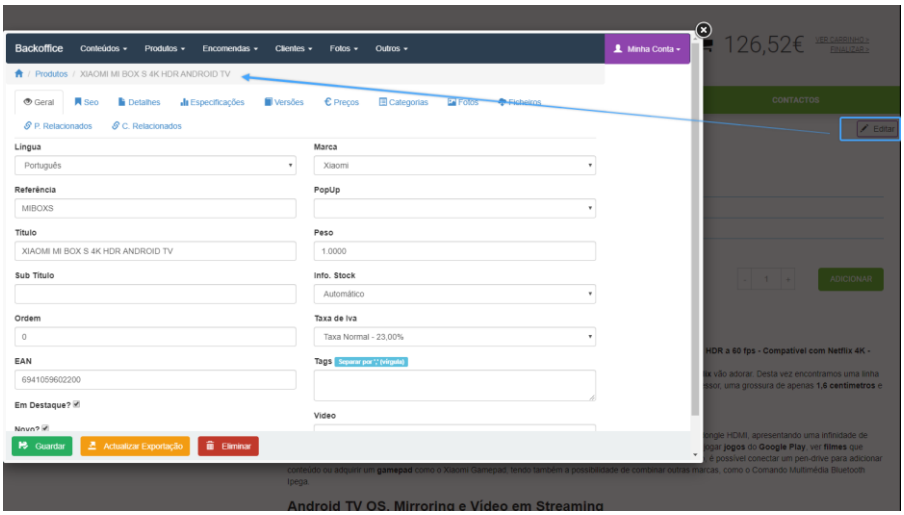


Figura 57 - Backoffice da Antiga Plataforma

No caso da nova plataforma o objetivo foi manter a facilidade de edição apesar da interface ter mudado. Para isso, criou-se um *widget* em *JavaScript* com a possibilidade de editar alguns conteúdos simples como por exemplo título, introdução e visibilidade do produto, mas também criar atalhos para as funcionalidades de *backoffice* com edições mais complexas, como é o caso dos preços e versões. Na Figura 58 encontra-se o *widget* criado para facilitar a criação e edição de conteúdos da nova plataforma.

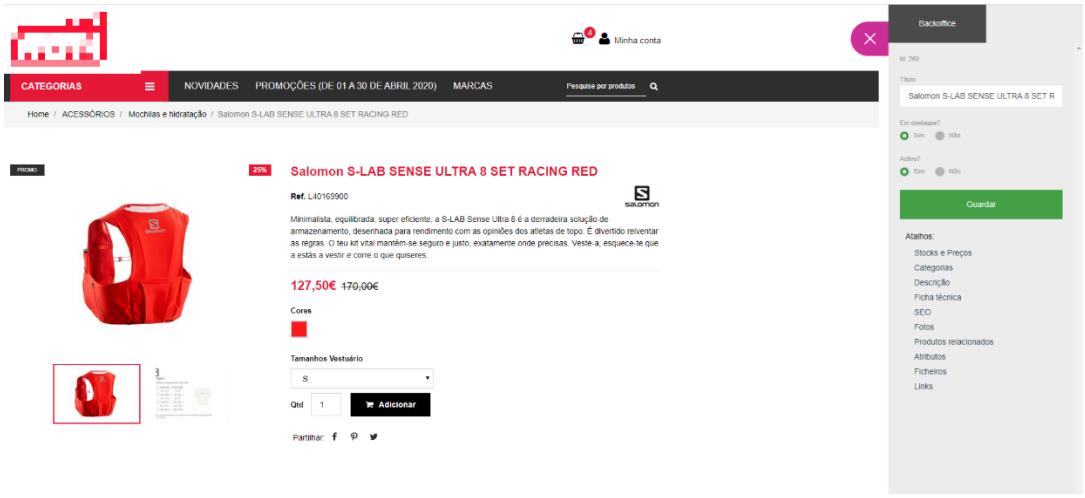


Figura 58 - Widget da Nova Plataforma que dá acesso ao *backoffice* diretamente ou através de atalhos.

Na nova plataforma, como já referido anteriormente na secção 4.1.4, a tecnologia utilizada foi ASP.NET com padrão MVC permitindo que no lado aplicacional fosse possível a renderização dos dados vindos do servidor através da sintaxe *razor* que, por sua vez, irá geral o conteúdo em HTML para acesso ao mesmo através do cliente (*browser*).

Para o desenvolvimento da sua interface, foi adaptado um *template* base desenvolvido por outra equipa de forma a garantir todas as funcionalidades e regras de usabilidade. Na Figura 59 encontra-se o *template* base utilizado para a plataforma.

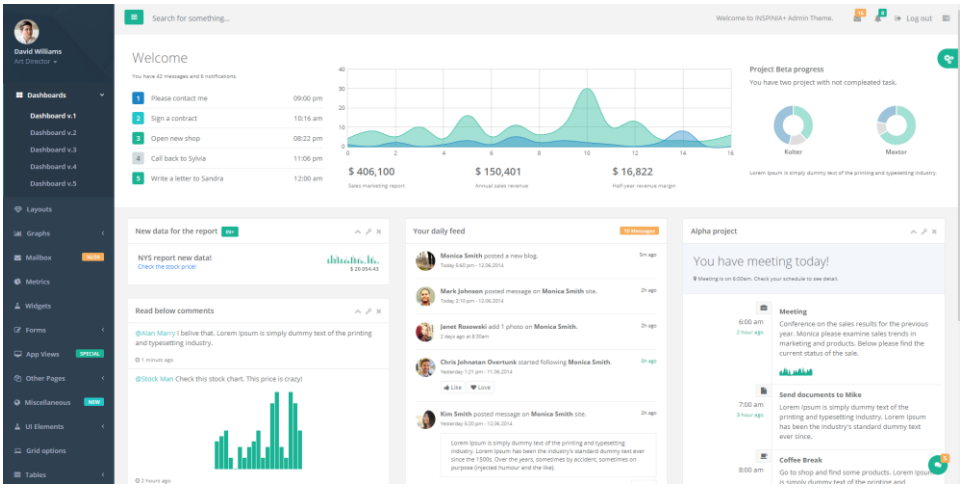


Figura 59 - Template UI da plataforma [29]

No lado aplicacional, as linguagens utilizadas foram: *HTML5, CSS3, SCSS, JavaScript, Vue.js,* além de alguns plugins desenvolvidos em *jQuery*.

O *template* base adquirido para a plataforma foi desenvolvido em HTML5, CSS3 e *JavaScript* com a utilização da *framework bootstrap* 3.4. Aproveitaram-se algumas funcionalidades em *JavaScript* e atualizou-se a *framework* de *bootstrap*. Na Figura 60 encontra-se a imagem referente à interface da nova plataforma.

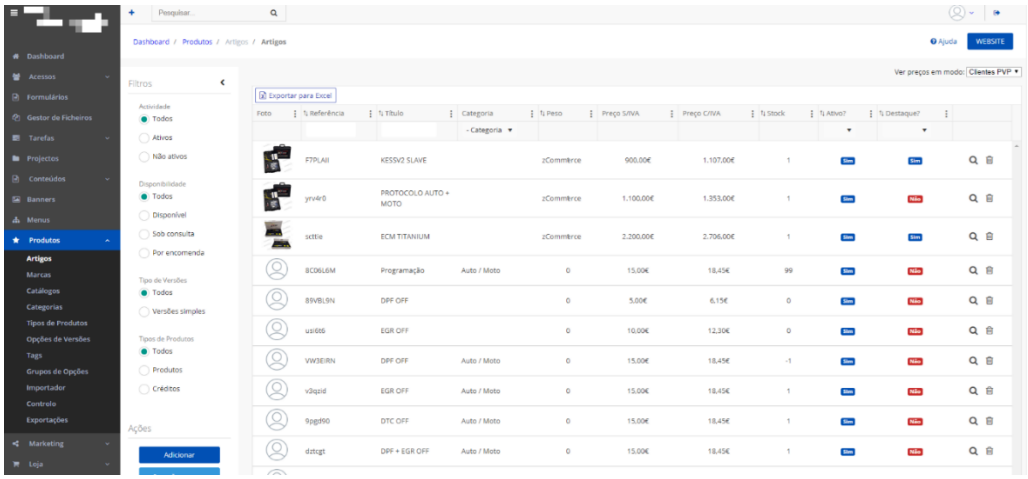


Figura 60 - UI da nova plataforma

Bootstrap

O *bootstrap* apresenta-se como *framework open source* para a *web* que permite facilitar o desenvolvimento de *websites* e adaptação para dispositivos móveis de forma simples e rápida. [30]

Ao fazer a adaptação do *template* para a plataforma, decidiu-se atualizar a *framework bootstrap* de 3.4 para 4.4 e também o CSS3 para SCSS. O SCSS é uma extensão do CSS que permite simplificar o modo de desenvolvimento das folhas de estilo, pois torna-se possível a criação de variáveis, funções, *mixins*, entre outros. [31]

O SCSS do *bootstrap* permitiu obter várias funcionalidades já desenvolvidas como por exemplo: funções que permitem a alteração do tamanho de letra dependendo da resolução, *mixins* para as transições que permitem agrupar as propriedades que diferem nos diversos browsers, possibilidade de personalizar rapidamente os estilos globais da plataforma através de variáveis, como formulários, cores, tamanhos, tipos de letra, entre outros. Na Figura 61 pode encontrar-se um excerto de código que demonstra a alteração de uma variável de cor que afeta de imediato toda a plataforma nos vários sítios onde esta foi utilizada.

```scss
// Body
$body-bg: $white;
$body-color: $black;


// Paragraphs
$paragraph-margin-bottom: 0;


// Grid breakpoints
$grid-breakpoints: ( xs: $breakpoint-xs, sm: $breakpoint-sm, md: $breakpoint-md, lg: $breakpoint-lg, xl: $breakpoint-xl );


// Grid containers
$container-max-widths: ( sm: $container-sm, md: $container-md, lg: $container-lg, xl: $container-xl );


// Fonts
$font-family-base: 'Open Sans', sans-serif;
$font-family-title: $font-family-base;
$font-family-icon: 'icomoon';
$font-size-base: rem(14px);
$line-height-base: 1.5;

$font-weight-light: 300;
$font-weight-normal: 400;
$font-weight-bold: 700;

$h1-font-size: rem(40px);
$h2-font-size: rem(36px);
$h3-font-size: rem(24px);
$h4-font-size: rem(20px);
$h5-font-size: rem(18px);
$h6-font-size: rem(14px);
```

Figura 61 - Variáveis em SCSS

A *framework bootstrap* também trouxe consigo um componente de *JavaScript* que permitiu a utilização de *popUps, accordions* entre outras funcionalidades. Na Figura 62 encontra-se um exemplo de *popUp* do *bootstrap* com recurso a *JavaScript.*

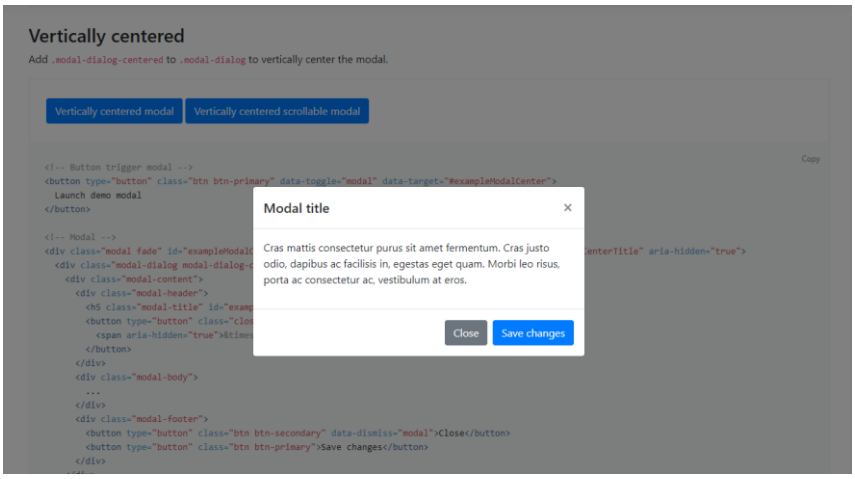


Figura 62 - Modal *Bootstrap* [32]

Ainda no mundo do *JavaScript,* foi utilizado *vue.js (*descrito na secção 4.1.4) para algumas funcionalidades nomeadamente grelhas *(grids)* de dados vindos do servidor. O *vue.js*, como foi dito anteriormente, não foi utilizado como um "todo" na plataforma visto que a sua curva inicial de aprendizagem ainda se torna extensa e nem todos os membros da equipa tinham os conhecimentos necessários sobre esta tecnologia. Desta forma foi utilizado apenas nas grelhas de dados através de um componente para *vue.js*: o Kendo UI da Telerik.

Kendo UI

O Kendo UI é uma *framework* de *front-end* do Telerik que fornece várias ferramentas de forma a enriquecer a programação web e com uma sintaxe acessível. Com esta ferramenta não é necessário especificar atributos em elementos HTML para que o *JavaScript* do Kendo UI consiga interpretá-los pois todo o processo pode ser feito do lado do servidor e simplesmente ser mostrado no *front-end*. [33] Esta *framework* fornece vários componentes, consoante as linguagens a utilizar são elas: *jQuery, Angular, React* e *Vue.js*.

Neste projeto foi utilizado o componente de *vue.js* do Kendo UI para a renderização de dados nas tabelas. Esta implementação simplificou o processo de desenvolvimento e tornou possível vários tipos de filtros.

Foto	Referência	Título	Categoria	Marca	Peso	Preço S/IVA
	FT-250-24410-TR (Versões simples)	Frasco PET tubo de 250 ml transparente	Frascos		0.0264	0,29€
	VPS-67-8-BR (Versões simples)	Disco Vedante Adesivo em PS Ø67,8 x 0,58mm Sealed for your protection	Tampas / Vedantes		0.0008	0,13€
	VCG-18-TR (Versões simples)	Vedante Conta gotas para frascos de vidro	Tampas / Vedantes		0.001	0,10€

Figura 63 - Kendo UI Grid Telerik

Na Figura 63 encontra-se representada uma implementação de uma *grid* do Kendo UI na nova plataforma. Trata-se de uma tabela em que cada cabeçalho é responsável por vários tipos de filtros e ordenações para cada coluna da mesma.

Para facilitar a navegação entre páginas, carregamento de dados das páginas, tratamento de imagens e renderização de conteúdo, contou-se com ajuda de alguns elementos externos, tais como o *barba.js*, o *Pixie*, o *ContentBuilder* e o *ContentBox*.

Barba.js

O *Barba.js* trata-se de uma *framework* que ajuda na criação de movimentos e transações de páginas de forma simples e suave através de data-atributos nas *tags* HTML para conseguir fazer a correspondência ao código *JavaScript*. Desta forma, torna-se possível fazer

transições de página com movimentos suaves que normalmente são complicadas. Para além disso, o *barba.js* proporciona também um sistema de cache da página, isto é, apenas é necessário carregá-la a primeira vez que se acede e assim torna-se a navegação mais rápida e eficaz. [34]

Esta *framework* foi utilizada de forma a fazer as transições suaves consoante a mudança de separadores ou páginas e para aplicar cache nas páginas de forma a tornar uma navegação mais fluída visto que o *vue.js* não foi utilizado na sua íntegra pelos motivos referidos anteriormente.

Pixie – Image Editor

O *Pixie* é de um *plugin jQuery* para tratamento de imagens que veio substituir o *Creative SDK* existente na antiga plataforma.

O *Creative SDK* é uma *framework* da Adobe que fornece um conjunto de ferramentas que podem ser adicionadas aos dispositivos móveis. Aquando a criação da antiga plataforma, esta *framework* fornecia uma funcionalidade exclusivamente dedicada à web que permitia que a partir do browser se conseguisse editar imagens em tempo real e sem ser obrigatório ter um login da Adobe, funcionalidade que foi descontinuada. [35]

O *Pixie* contém várias funcionalidades para tratamento de imagem como recorte, redimensionamento, filtros de cores, entre outros. O editor é totalmente adaptado para dispositivos móveis e com possível personalização de temas do próprio editor. É também de fácil integração com projetos ou aplicações. [36] Na nova plataforma, este editor foi utilizado para edição de fotos de produtos e de páginas. Na Figura 64 encontra-se um exemplo da utilização do *Pixie* para edição de fotos de um produto.

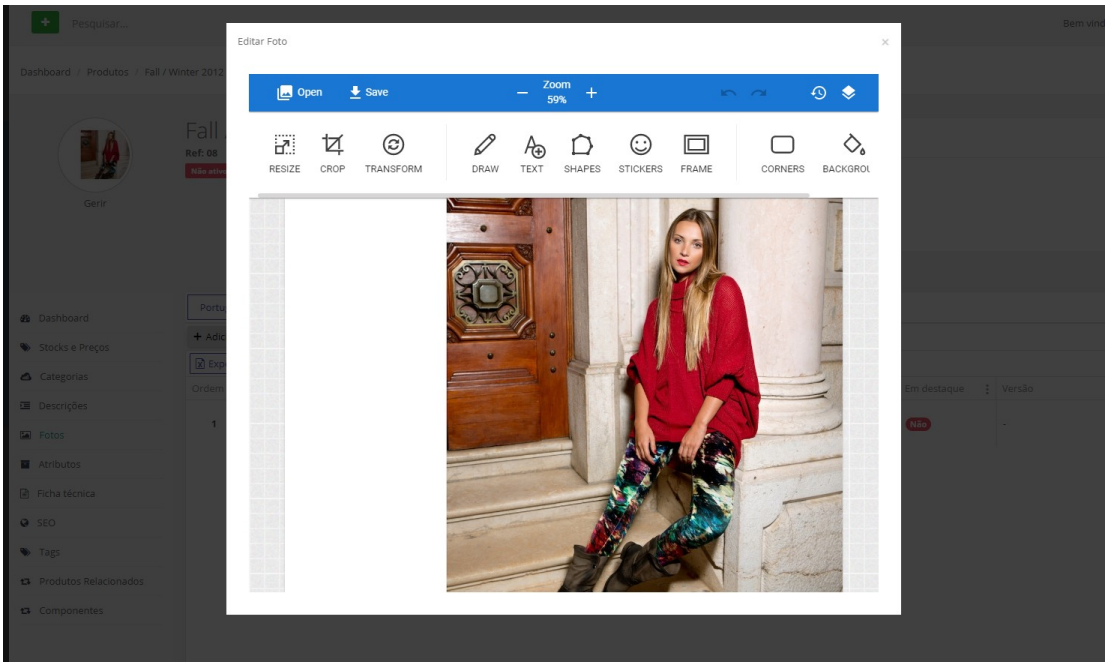


Figura 64 – Edição de foto através do *plugin Pixie* na nova plataforma

ContentBuilder e ContentBox

O *Contentbuilder e ContentBox* são *frameworks* em *jQuery* que permitem a edição, criação e alteração de conteúdos de forma simples através de blocos de conteúdo. São utilizados de forma a que o conteúdo possa ser alterado diretamente no *front-end*, pelo utilizador final, sem que este tenha a necessidade de ter um bom conhecimento de programação de modo a conseguir fazer conteúdo e torná-lo apelativo.

O *Contentbuilder* é composto por vários blocos de conteúdos divididos em diversas categorias: preços, serviços, portfólios, parágrafos, títulos, entre outros. Foram blocos personalizados com estilos diferentes para que o utilizador possa usar com um simples arrastar para a área pretendida. Além disso, também é possível criar novos blocos de conteúdo consoante o *design* de *layout* do cliente e dispõe de uma barra de ferramentas que possibilita a alteração de formação de texto como cor e tamanho, colocação de imagens e links, entre outros. Na Figura 65 encontra-se um exemplo de utilização do *ContentBuilder*

na construção de conteúdo seja de produtos ou páginas, como por exemplo: “Guia de Compras”.

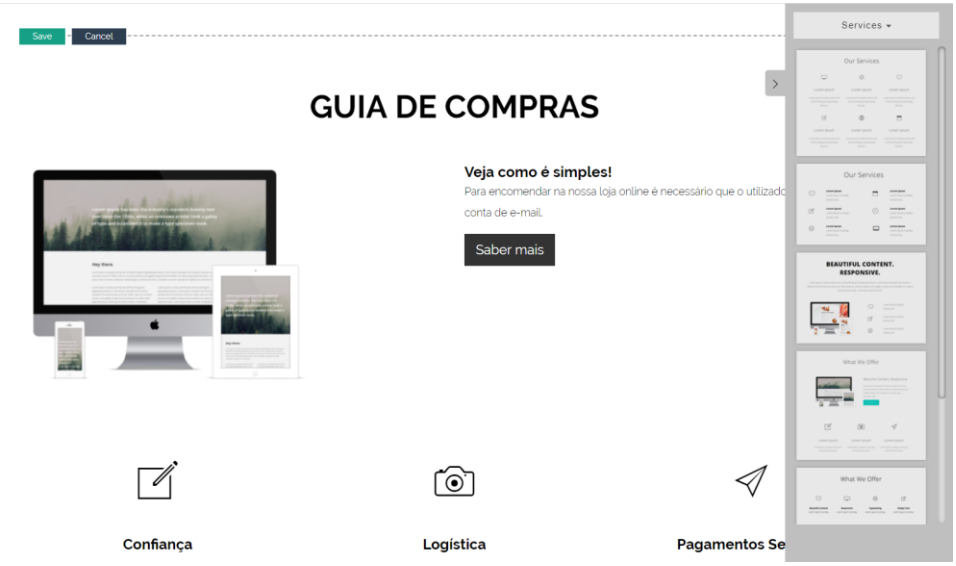


Figura 65 - Utilização do Contenbuilder na construção de conteúdo.

O *ContentBox* é uma ferramenta que funciona aliada ao *ContentBuilder* pois permite a criação de secções para que possam ser arrastados os blocos de conteúdo. [37] Na Figura 66 encontra-se um exemplo relativo à criação de secções no *ContentBox* que pode ser utilizado de igual modo na produção de conteúdo seja em páginas ou produtos.

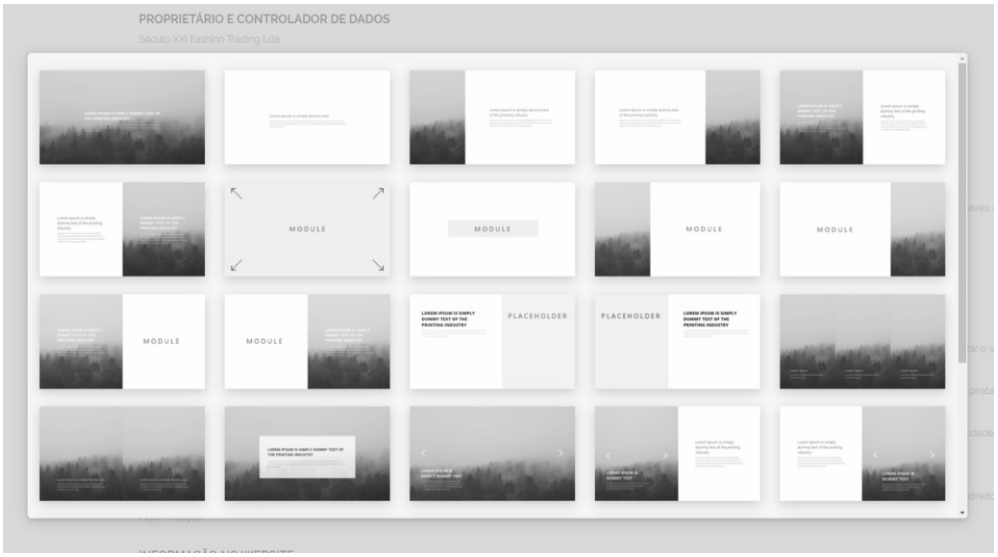


Figura 66 - Secções do *ContentBox*

4.5. Testes

Nesta secção encontram-se todos os testes realizados na plataforma, desde os testes unitários até aos testes carga.

No decorrer do desenvolvimento foram implementados testes unitários e de integração de forma a garantir todo o funcionamento da plataforma (ver secção 4.5.1 e secção 4.5.2).

Depois destes testes foram realizados testes manuais, ou seja, testes com o utilizador final para perceber se a plataforma estava intuitiva e funcional (ver secção 4.5.3).

Por fim, na secção 4.5.4 encontram-se os testes de carga feitos à plataforma *e-commerce* a dois casos reais com afluências de utilizadores e ambientes diferentes.

4.5.1. Testes Unitários

Depois da implementação do ORM para o mapeamento dos dados e acesso aos mesmos estar concluído, como referido na secção 4.3.4, tratou-se dos testes unitários aos repositórios de forma a perceber se as validações estariam corretas.

Os testes unitários são testes menores que servem para testar qualquer *software.* Os testes permitem verificar se uma função específica está a funcionar corretamente e garantir que a aplicação retorne o valor correspondente, mesmo que o código base seja alterado. Nesta fase de testes aplicaram-se testes em estruturas internas, testes de funcionalidades e testes de segurança. [38]

Foram feitos vários testes às funcionalidades da plataforma, no entanto os testes seguintes referem-se apenas à gestão por entidade. As restrições da entidade são:

- O número fiscal tem de ser obrigatório;
- O nome da entidade não pode ser maior que 255 caracteres;
- O nome da entidade tem de ser preenchido.

Para validar estas restrições foram feitos os testes identificados na Tabela 2:

Tabela 2 - Restrições de entidade

Condição	**Resultado** (verdadeiro/falso)
Se a entidade não tiver nem nome nem número fiscal.	Retornar falso

Se a entidade tiver número fiscal	Retornar verdadeiro
Se a entidade tiver nome	Retornar verdadeiro
Se a entidade tiver nome, mas sem número fiscal	Retornar falso
Se o nome da entidade tiver mais do que 255 caracteres.	Retornar falso

O código da plataforma e os testes foram feitos através do Visual Studio 2019. Foi criada uma aplicação dentro da solução do projeto dedicada exclusivamente para os testes, e dividido por pastas consoante a área a tratar. Na Figura 67 encontra-se a estrutura de pastas dos testes bem como as classes abstratas que ajudaram na implementação dos mesmos. As classes abstratas foram utilizadas para simular a criação de dados, como por exemplo uma entidade.

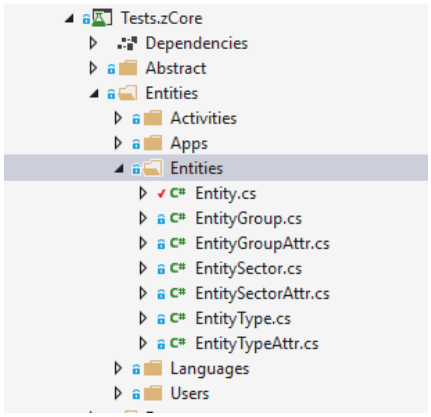


Figura 67 - Estrutura de pastas da Aplicação de Testes

Para a criação dos testes foram importadas duas *frameworks*: o *XUnit* e o *Shouldly*. O XUnit é uma ferramenta *open source* para criar testes unitários em várias plataformas e o *Shouldly* é uma *framework* adicional que permite substituir o tradicional "Assert That" por uma sintaxe mais legível para humanos: o "Should Be". [39]

Em cada classe foram criados testes para validação das regras e condições aplicadas. Para isso, cada método corresponde a um teste e tem de ter a propriedade "Fact" ou "Theory". O "Fact" indica um conjunto estático e único de condições de testes em que os testes são executados apenas uma vez porque é um conjunto de valores que é passado para o sistema de testes. Já o "Theory" indica que se vai testar mais do que um conjunto de

condições. Neste último caso, o sistema de testes executa-os quantas vezes houver conjuntos de condições e sempre pela ordem dos parâmetros colocados no método de testes. [40]

Na Figura 68 encontra-se um excerto de código referente à importação das duas *frameworks XUnit* e *Shouldly* bem como a construção de um teste com a utilização do "Fact".

```
Tests.zCore.Entities.EntityValidatorTest                    Entity_Invalid()
using Shouldly;
using System.Linq;
using Tests.zCore.Abstract;
using Xunit;
using zModule.zCore.Entities;

namespace Tests.zCore.Entities
{
    0 references
    public class EntityValidatorTest
    {
        [Fact]
        0 references
        public void Entity_Invalid()
        {
            /*
             * Props to validate:
             * {Name}
             * {FiscalIdNumber}
             */
            var entity = new Entity();
            var validator = new EntityValidator();

            // Name - Empty
            // FiscalIdNumber - Empty
            validator.Validate(entity).IsValid.ShouldBeFalse();
            validator.Validate(entity).Errors.Select(s => s.PropertyName).ShouldContain(nameof(Entity.Name));
            validator.Validate(entity).Errors.Select(s => s.PropertyName).ShouldContain(nameof(Entity.FiscalIdNumber));
```

Figura 68 - Testes à Entidade

Neste teste da Figura 68 foi aplicada a primeira condição da Tabela 2. O objetivo do teste é testar se a entidade tem nome e número fiscal e retornar falso no caso de não ter. Na criação do método foram colocadas as propriedades a testar em comentário - a entidade e número fiscal - para se mais fácil de identificar o que se está a testar e quais as condições a testar. A partir daí foram feitos os testes através de *lambda expressions* e com recurso à *framework Shouldly*, de forma a retornar as condições pretendidas. Na Figura 69 encontram-se os restantes testes feitos à Entidade através de *lambda expressions*.

```
// Name - Okay
// FiscalIdNumber - Empty
// Website - Empty
entity.Name = "Zenn";
validator.Validate(entity).IsValid.ShouldBeFalse();
validator.Validate(entity).Errors.Select(s => s.PropertyName).ShouldContain(nameof(Entity.FiscalIdNumber));

// Name - Okay
// FiscalIdNumber - Empty
validator.Validate(entity).IsValid.ShouldBeFalse();

// Name - Okay
// FiscalIdNumber - Empty
validator.Validate(entity).IsValid.ShouldBeFalse();
validator.Validate(entity).Errors.Select(s => s.PropertyName).ShouldContain(nameof(Entity.FiscalIdNumber));

// Name - Greater than 255
// FiscalIdNumber - Empty
entity.Name = EntityExtensions.GenerateString('a', 500);
validator.Validate(entity).IsValid.ShouldBeFalse();
validator.Validate(entity).Errors.Select(s => s.PropertyName).ShouldContain(nameof(Entity.Name));
validator.Validate(entity).Errors.Select(s => s.PropertyName).ShouldContain(nameof(Entity.FiscalIdNumber));
```

Figura 69 - Testes à Entidade (parte 2)

Os testes podem ser corridos de forma global ou um teste em particular. Depois de concluir cada teste este foi analisado se falhou ou passou conforme cada condição esperada, e caso o resultado não fosse o esperado o erro era corrigido de imediato.

No caso dos testes à funcionalidade Entidade, foram corridos 50 testes na totalidade e passaram todos. Estes testes demoraram na totalidade 2,3 segundos como se pode verificar através da Figura 70.

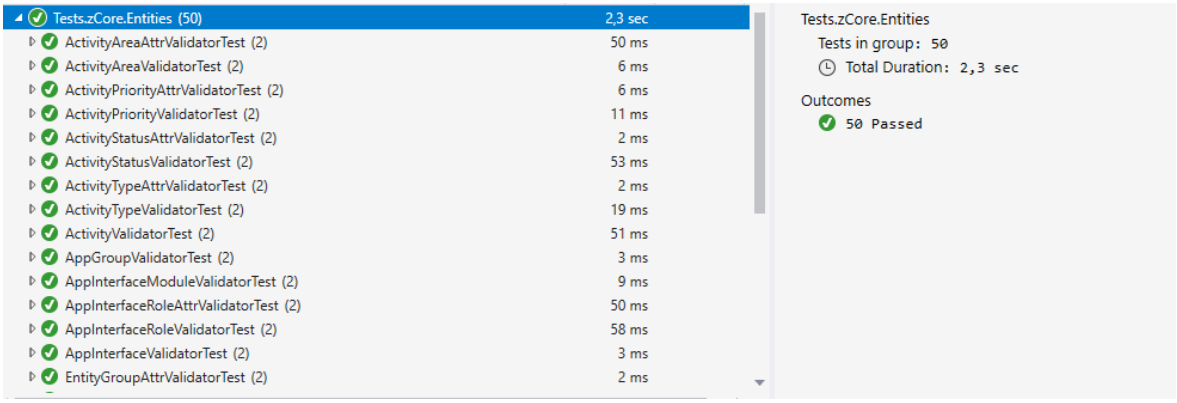


Figura 70 - Testes Unitários à funcionalidade Entidade.

4.5.2. Testes de Integração

Depois dos testes unitários, seguem-se os testes de integração. Estes têm como objetivo encontrar falhas na integração do sistema e não falhas de funcionalidades e aplicam-se a interfaces e dependências entre componentes. [41]

Os testes de integração seguintes dizem respeito à funcionalidade "Entidade", no entanto foram feitos vários testes a todas as funcionalidades da plataforma.

No caso da nova plataforma, uma Entidade tem a obrigatoriedade de estar associada a um grupo de entidade e este grupo é obrigado a ter pelo menos uma linguagem. Para testar a criação de grupos de entidade e validação da linguagem seguem-se as seguintes restrições:

- Ter pelo menos uma linguagem;
- Ter pelo menos um registo de grupo de entidade nessa língua.

Para validar estas restrições foram feitos os seguintes testes representados na Tabela 3:

Tabela 3 - Restrições de apagar uma entidade

Condição	**Resultado**
Criar uma única instância de grupo entidade.	- Retornar verdadeiro - Retornar resultado maior ou igual a 1
Verificar se o grupo de entidade foi criado na base de dados com a língua associada	- Retornar *not null* - Retornar título do grupo na língua correspondente

Na Figura 71 encontra-se o teste realizado com os requisitos identificados. Depois da identificação do teste, foi criado o grupo de entidade e a linguagem na base de dados bem como o relacionamento entre elas de forma a fazer o teste com os dados reais.

```
[Fact]
0 references
public async Task Handle_Success()
{
    /*
    * Requirements:
    * At least 1 Language
    * At least 1 EntityGroupAttrs
    */

    if (!LocalContext.Languages.Any())
    {
        var pt = Languages.GetPortuguese();
        if (!LocalContext.Languages.Any(s => s.Reference == pt.Reference))
            LocalContext.Languages.Add(pt);
        LocalContext.SaveChanges();
    }

    /*
   * Test:
   * Create a single instance of Entity Group.
    */

    var command = new CreateEntityGroupCommand()
    {
        Title = "Grupo",
        IdLanguage = LocalContext.Languages.FirstOrDefault().IdLanguage,
    };

    var handler = new CreateEntityGroupCommand.CreateEntityGroupCommandHandler(new EntityGroupRepository(LocalContext,
    var result = await handler.Handle(command, CancellationToken.None);

    // Command should return OK!
    result.IsValid.ShouldBeTrue();
    result.Payload.ShouldBeGreaterThanOrEqualTo((uint)1);

    // Check if EntityGroup has created in database
    var entity = await LocalContext.EntityGroups.FirstOrDefaultAsync(s => s.IdEntityGroup == result.Payload);
    entity.ShouldNotBeNull();
    entity.GetAttr(command.IdLanguage).Title.ShouldBe(command.Title);
```

Figura 71 - Teste ao Eliminar um tipo de entidade numa entidade associada

Foram corridos 29 testes de integração em várias funcionalidades e passaram todos. Estes testes demoraram na totalidade 16.3 segundos como se pode verificar através da Figura 72.

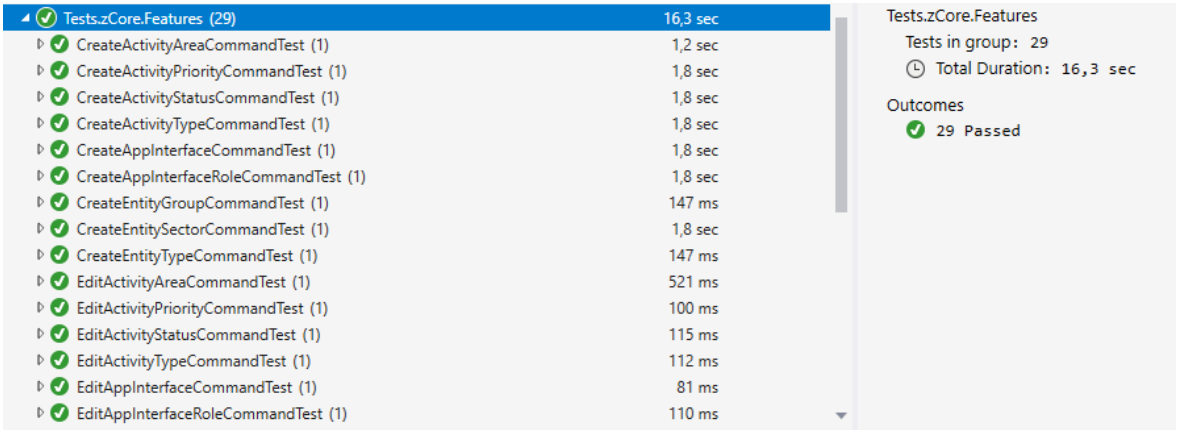


Figura 72 - Testes de Integração da nova plataforma

4.5.3. Testes Manuais

Nesta secção encontram-se descritos alguns dos testes manuais que foram realizados depois da implementação da nova plataforma.

Os testes são realizados por seres humanos com recurso a uma lista de tarefas a executar, passo-a-passo, de forma a perceber se os cenários se encontram coerentes com a expectativa e resultado final da execução. [42]

Foram realizados vários testes com utilizadores finais que testaram a plataforma nas suas diversas funcionalidades. Na Tabela 4 encontram-se alguns dos casos de uso que foram utilizados com 10 utilizadores finais que testaram a plataforma pela primeira vez. Todos estes utilizadores tinham conhecimentos de outras plataformas de *e-commerce*.

Tabela 4 - Casos de uso para os testes Manuais

Nº de casos de uso	**Casos de uso**
1	Criar um produto
2	Criar uma categoria
3	Adicionar uma categoria a um produto
4	Adicionar preço ao produto
5	Adicionar descrição ao produto na língua inglesa
6	Adicionar preço por quantidade
7	Tornar obrigatório um campo de formulário

Depois de apresentados os casos de uso, seguiram-se os testes relativos ao grupo de utilizadores. Os testes foram verificados manualmente por cada utilizador relativamente ao número de cliques em cada caso de uso. A Tabela 5 apresenta a quantidade mínima e máxima de cliques que os utilizadores realizaram em cada caso de uso.

Tabela 5 - Tabela de expectativas dos Caso de Uso

Nº Casos de uso	**Expectativa (número de passos)**	**Média de execução (número de passos)**	**Observações gerais de utilização**
1	4 cliques	4 a 6 cliques	No geral os utilizadores não tiveram problemas em criar produtos.

2	4 cliques	4 a 5 cliques	No geral os utilizadores não tiveram problemas em criar categorias.
3	6 cliques	8 a 12 cliques	Vários utilizadores tiveram dificuldade em entrar no detalhe de produto, clicaram no título do produto para aceder ao detalhe.
4	6 cliques	9 a 12 cliques	Alguma dificuldade em encontrar o botão de gravar preço.
5	6 cliques	16 a 20 cliques	Existiu alguma dificuldade em saber onde estava a opção de mudar de língua.
6	8 cliques	10 a 14 cliques	Existiu dificuldades em encontrar a opção para criar um preço por quantidade.
7	6 cliques	16 a 25 cliques	Existiu grandes dificuldades em perceber onde alterar a opção. Alguns dos utilizadores não terminaram a tarefa com sucesso.

De acordo com a Tabela 5 pode-se dizer que não houve dificuldades a criar produtos e categorias, mas os restantes casos de uso apresentam alguns problemas de usabilidade a resolver na plataforma devido às dificuldades de acesso. Na Figura 73 encontra-se um gráfico referente à média de cliques dados e a expectativa de forma a perceber melhor os resultados.

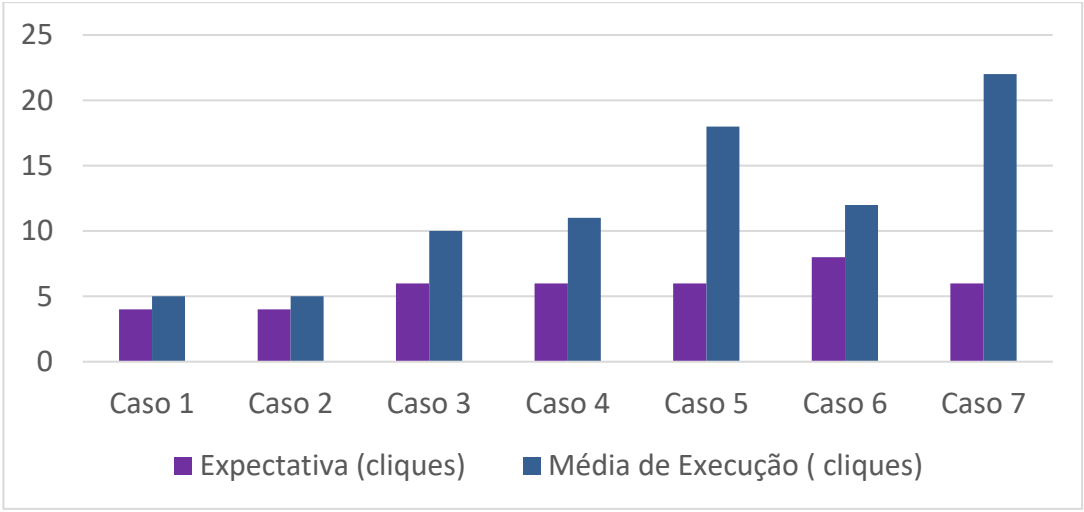


Figura 73 - Gráfico de Média de cliques por caso de uso

Depois da análise do gráfico representado na Figura 73 pode-se concluir que os maiores problemas de usabilidade requerem:

- A uniformização das grelhas de dados para se aceder aos detalhes em toda a plataforma;
- A melhoria da interface de forma mais intuitiva para criar preços num produto;
- A melhoria da interface para se tornar mais intuitiva a mudança de língua;
- A melhoria da interface para criação e edição de campos de formulário.

Alguns dos problemas de usabilidade detetados após esta análise foram resolvidos e outros menos problemáticos ficaram para uma próxima atualização da plataforma.

4.5.4. Testes de Carga

Os testes de carga são utilizados de forma a garantir o funcionamento dos sistemas quando existem picos de acessos. São executados para saber se o sistema está preparado para aguentar múltiplos acessos sem gerar graves problemas de lentidão ou mesmo quebra do serviço. No caso de isto acontecer, poderá prejudicar não só a experiência do utilizador como a do serviço.

Nestes testes é realizado um estudo junto do cliente para saber quais as principais operações realizadas no seu sistema de modo a identificar as que possam exigir mais do mesmo. Posteriormente são criados testes automatizados que simulam as operações em simultâneo de vários utilizadores de forma a perceber a resistência do sistema. [43]

Para testar a plataforma com testes de carga foram utilizados dois ambientes distintos: um caso num servidor partilhado e outro caso numa *Virtual Private Server* (VPS). Além disso, foi analisada a média de utilizadores em simultâneo e pedidos ao servidor bem como quais páginas com maior afluência em cada plataforma. Os testes foram feitos numa plataforma específica para testes de carga o *loader.io*, que é um serviço gratuito de testes de carga que permite a realização de testes de stress nas aplicações *web* com várias conexões em simultâneo. [44]

O primeiro caso diz respeito à implementação da plataforma numa loja de artigos desportivos: o RUN.PT. Neste caso o número de utilizadores em simultâneo era aproximadamente 10, a página "Produtos" foi identificada com maior afluência e o ambiente encontrava-se num servidor partilhado. Na Figura 74, encontra-se um gráfico relativo à afluência do *website* num teste de carga que poderá conter até 75 utilizadores por teste e os pedidos por utilizador na página de "Produtos" podem exceder 68 pedidos.

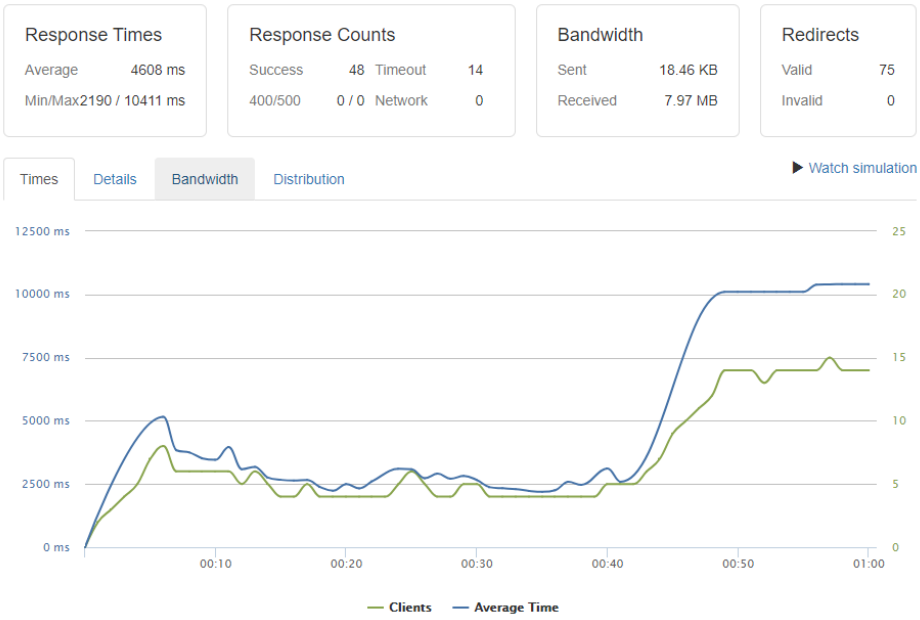


Figura 74 - Teste de carga ao RUN.PT [45]

De acordo com o que se pode verificar na Figura 74 neste teste o servidor teve até 15 utilizadores em simultâneo e verificou-se alguma taxa de rejeição. As métricas encontram-se um pouco abaixo das espectativas ao longo do teste bem como o tempo de resposta do servidor perante a quantidade de pedidos e acessos. Com o aumento do número de utilizadores em simultâneo no *website* e considerando que estamos num servidor partilhado, o tempo de resposta do servidor foi aumentando consideravelmente, o que não é, de todo, o ideal.

O segundo caso trata-se de uma loja de roupa feminina: a SAHOCO. Neste caso o número de utilizadores em simultâneo rondava aproximadamente entre 50 a 80 utilizadores, a página com maior afluência era a "Nova Coleção" e o ambiente encontrava-se numa VPS. Na Figura 75 encontra-se um gráfico relativo à afluência do *website* num teste de carga em que poderá ter até 500 utilizadores em simultâneo durante 1 segundo e os pedidos por utilizador na página de "Nova Coleção" pode exceder 94.

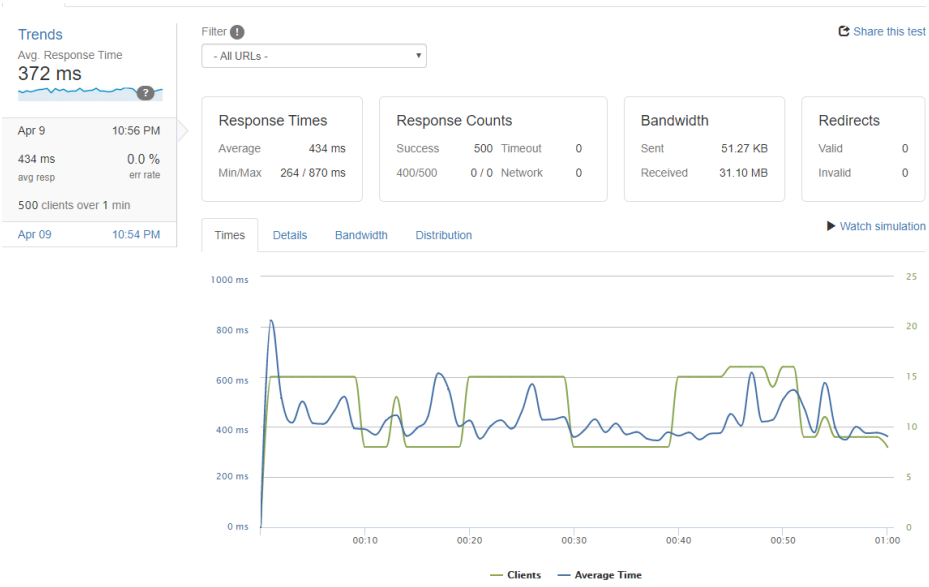


Figura 75 - Testes de Carga ao SAHOCO [45]

Segundo o gráfico da Figura 75, o servidor teve para cima de 20 utilizadores em simultâneo durante um segundo no qual não se verificou qualquer taxa de rejeição. O acompanhamento oscilou devido à quantidade de utilizadores, mas manteve sempre uma linha contínua perante o tempo de resposta do servidor e da quantidade de pedidos.

4.6. **Análise crítica e proposta de melhorias**

Nesta secção encontram-se descritas algumas considerações relativas à análise e implementação de tecnologias e propostas de melhorias.

Durante a análise e implementação da plataforma todas as decisões a nível tecnológico e funcional foram feitas pela empresa, como já referido anteriormente na secção 2.2. Algumas das tecnologias não foram usadas na totalidade e outras deveriam ser repensadas futuramente.

Uma das tecnologias que podia ter sido aproveitada na totalidade era o *vue.js*. Este foi aplicado parcialmente devido à curva de aprendizagem exigida e escassez de recursos humanos da empresa. Ao aplicar esta tecnologia na totalidade, a estrutura da aplicação teria de ser totalmente reconstruída (pelo menos do lado do cliente) e tanto o *razor* como o *barba.js (plugin* utilizado para as transições suaves e cache de páginas), deixariam de ser necessários e a aplicação passava a utilizar apenas API's para obter os dados do servidor e

serem geridos através de *JavaScript* e renderizados com a ajuda do *vue.js*. Desta forma a UI tornar-se-ia mais fluída.

Outra das situações que poderia ser melhorada era a uniformização das grelhas de dados da UI, uma vez que os testes manuais identificaram alguns problemas de usabilidade destas grelhas no acesso ao detalhe do item. Este acesso deveria ser conseguido através do título e da imagem da lupa mas nem todas as grelhas respeitavam esse processo e isso provocou confusão no utilizador.

Na nova plataforma uma das funcionalidades mapeada da antiga foi gestão de menus. Esta consistia numa implementação baseada em 2 tipos de menu: cabeçalho e rodapé, que a empresa considerou ser o suficiente. No entanto, o ideal seria a possibilidade de criar vários menus e atribuir links internos, externos ou mesmo documentos de forma a expandir as opções de menu.

Já no que diz respeito a lojas *online*, surgiram novos clientes que levantaram algumas questões sobre as funcionalidades da plataforma *e-commerce* nomeadamente a possibilidade de gestão de projetos, orçamentos e créditos por entidade ou cliente. Seriam boas apostas de implementações futuras para enriquecimento da plataforma.

5. Conclusão

O principal objetivo deste projeto foi a reconstrução de uma plataforma *e-commerce* devido, essencialmente, à descontinuação do ORM utilizado. Conclui-se que este objetivo proposto foi atingido, bem como a implementação de novas funcionalidades, principalmente ao nível da gestão de produtos e de entidades.

De acordo com o plano de estágio inicialmente previsto, houve uma discrepância temporal nos testes unitários, que acabaram por demorar mais tempo do que o suposto. Isto deveu-se ao surgimento de projetos paralelos que provocou o atraso na continuação do processo de desenvolvimento, principalmente no que toca ao tempo alocado às API's de autenticação e de autorização. De forma a conseguir cumprir com os prazos, alocaram-se alguns recursos humanos extra ao projeto, recursos que estiveram maioritariamente focados nas API's. Todo o restante projeto correu consoante o plano de estágio e ainda se conseguiu testar a implementação em ambientes reais.

Ao longo do estágio foram aplicados vários conhecimentos adquiridos na Licenciatura de Tecnologias de Informação e Multimédia, como usabilidade, interface pessoa-máquina, análise de base de dados e engenharia de *software*, além dos conhecimentos adquiridos no Mestrado de Engenharia Informática - Computação Móvel no que diz respeito à gestão de projetos e testes unitários. Com este projeto foi possível ainda obter novos conhecimentos nomeadamente no que toca ao ORM NHibernate, JWT, barba.js e SCSS.

Como já mencionado, todas as decisões foram tomadas por parte da empresa apesar de nem sempre haver concordância da minha parte com algumas abordagens utilizadas. Essa discrepância foi mais acentuada no que toca à utilização parcial do *vue.js* e à aplicação do *barba.js* para a colmatar. No meu ponto de vista, a utilização total do *vue.js* traria mais vantagens aplicacionais para o projeto em termos de fluidez e a possibilidade de remoção de outros *plugins*.

No que toca à metodologia utilizada na plataforma revelou-se ser uma boa abordagem, pois o projeto correu dentro do planeado bem como a execução das tarefas. Já no processo de desenvolvimento da plataforma, no meu ponto de vista acho que devia ter sido dedicado mais tempo à elaboração de documentação das funcionalidades, seja documentação interna ou de utilização da plataforma de forma a servir de tutoriais para os clientes.

Este projeto revelou-se vantajoso no que toca ao enriquecimento profissional tanto em aprendizagem como em experiência, principalmente no que diz respeito às novas tecnologias existentes no mercado.

Bibliografia

[1] K. C. T. C. G. Laudon, E-commerce 2016: business, technology, society, Twelfth Edition, 2016.

[2] ZENN - websolutions, "Apresentação ZENN," Leiria, 2018.

[3] L. Gonçalves, "Organisational Mastery," 3 1 2020. [Online]. Available: https://luis-goncalves.com/pt-pt/o-que-e-metodologia-agile/. [Acedido em 3 12 2019].

[4] Agile Alliance, "Scrum," [Online]. Available: https://www.agilealliance.org/glossary/scrum/. [Acedido em 13 1 2019].

[5] L. Gonçalves, "AGILE PRODUCT OWNER, O GUIA COMPLETO DE TUDO O QUE PRECISA DE SABER," 1 9 2019. [Online]. Available: https://luis-goncalves.com/pt-pt/agile-product-owner/. [Acedido em 14 11 2019].

[6] S. Pahuja, "Agile alliance," 2017. [Online]. Available: https://www.agilealliance.org/what-is-scrumban/. [Acedido em 13 12 2019].

[7] Tehnicki Vjesnik, "A Scrumban Integrated Gamification Approach To Guide Software Process Improvement: A Turkish Case Study," *A Scrumban Integrated Gamification Approach To Guide Software Process Improvement: A Turkish Case Study,* p. 245, 23 02 2016.

[8] Atlassian, "Atlassian," [Online]. Available: https://www.atlassian.com. [Acedido em 30 11 2019].

[9] E. Ramos, E-commerce, FGV Editora, 2016.

[10] ComSchool, "Performace Digital," 5 2019. [Online]. Available: https://news.comschool.com.br/o-que-e-uma-plataforma-de-e-commerce/. [Acedido em 22 10 2019].

[11] World Applied Programming, "Taking a look at different types of e-commerce," *World Applied Programming,* p. 5, 2 6 2011.

[12] Magento, "Magento," [Online]. Available: https://magento.com/. [Acedido em 03 01 2020].

[13] PrestaShop SA., "Prestashop," 2007-2020. [Online]. Available: https://addons.prestashop.com/en/. [Acedido em 23 12 2019].

[14] Redicom, "Redicom," 2002-2019. [Online]. Available: https://www.redicom.pt/pt/. [Acedido em 23 12 2019].

[15] LVEngine, "LVEngine," 2019. [Online]. Available: LVEngine. [Acedido em 23 11 2019].

[16] Auth0, "Web Apps vs Web APIs / Cookies vs Tokens," 2013-2018. [Online]. Available: https://auth0.com/docs/design/web-apps-vs-web-apis-cookies-vs-tokens. [Acedido em 11 12 2019].

[17] R. SOBERS, "Rob Sobers," 30 8 2018. [Online]. Available: https://www.varonis.com/blog/what-is-oauth/. [Acedido em 15 1 2020].

[18] S. Peyrott, The JWT Handbook, 2016-2018.

[19] Henrique, "DEVMEDIA," 2014. [Online]. Available: https://www.devmedia.com.br/introducao-ao-asp-net-mvc/31878. [Acedido em 13 2 2020].

[20] Vue.js, "Vue.js," 2014-2020. [Online]. Available: https://vuejs.org/v2/guide/index.html. [Acedido em 12 02 2020].

[21] Oracle Corporation and/or its affiliates, "MySQL Workbench," Oracle, [Online]. Available: https://www.mysql.com/products/workbench/. [Acedido em 14 9 2019].

[22] C. S. M. A. Derek Colley, "The Impact of Object-Relational Mapping Frameworks on Relational Query Performance," em *ICCECE 2018*, Southend, Essex, UK, 2018.

[23] DEVMEDIA, "Análise dos melhores ORM (Object-Relational Mapping) para plataforma .NET," 2007. [Online]. Available: https://www.devmedia.com.br/analise-dos-melhores-orm-object-relational-mapping-para-plataforma-net/5548. [Acedido em 23 12 2019].

[24] Telerik, "Introducing Telerik® Data Access," Progress Software Corporation and/or its subsidiaries or affiliates, 2017. [Online]. Available: https://docs.telerik.com/data-access/data-access-introduction. [Acedido em 8 11 2019].

[25] Telerik, "Telerik.DataAccess.Fluent," Telerik, 2014. [Online]. Available: https://www.nuget.org/packages/Telerik.DataAccess.Fluent/2014.2.918.1. [Acedido em 17 11 2019].

[26] NHibernate, "NHibernate," [Online]. Available: https://nhibernate.info. [Acedido em 8 10 2019].

[27] DEVMedia, "Introdução ao NHibernate – Framework para Mapeamento Objeto-Relacional," 2013. [Online]. Available: https://www.devmedia.com.br/introducao-ao-nhibernate-framework-para-mapeamento-objeto-relacional/28671. [Acedido em 2 12 2019].

[28] J. J. D. G. B. S. Jeremiah Willcock, "Lambda expressions and closures for C++," *Lambda expressions and closures for C++,* p. 14, 26 2 2006.

[29] INSPINIA - Responsive Admin Theme, "INSPINIA - Responsive Admin Theme," 2015. [Online]. Available: https://wrapbootstrap.com/theme/inspinia-responsive-admin-theme-WB0R5L90S. [Acedido em 11 4 2020].

[30] J. S. F. b. D. Winer, Bootstrap: Responsive Web Development, New York: O'Reilly Media, 2013.

[31] N. W. C. E. J. A. Hampton Catlin, "SASS," 2006-2020. [Online]. Available: https://sass-lang.com/documentation/syntax. [Acedido em 27 4 2020].

[32] Bootstrap Team, "Modal," 2011-2020. [Online]. Available: https://getbootstrap.com/docs/4.4/components/modal/#vertically-centered. [Acedido em 21 3 2020].

[33] J. Adams, Learning Kendo UI Web Development, Birmingham - Mumbai: Packt Publishing Ltd, 2013.

[34] xavierfoucrier, "Barba.js," 2020. [Online]. Available: https://barba.js.org/docs/plugins/preset/. [Acedido em 25 3 2020].

[35] Telerik, "Advanced Image Editing in the Browser," 11 2015. [Online]. Available: https://www.telerik.com/blogs/advanced-image-editing-in-the-browser. [Acedido em 8 4 2020].

[36] Envato market, "Pixie - Image Editor," 2020. [Online]. Available: https://codecanyon.net/item/pixie-image-editor/10721475. [Acedido em 3 3 2020].

[37] Innovastudio, "ContentBuilder.js," [Online]. Available: https://innovastudio.com/content-builder.aspx. [Acedido em 10 4 2020].

[38] D. Lima, 17 8 2017. [Online]. Available: https://medium.com/@dayvsonlima/entenda-de-uma-vez-por-todas-o-que-s%C3%A3o-testes-unit%C3%A1rios-para-que-servem-e-como-faz%C3%AA-los-2a6f645bab3. [Acedido em 4 4 2020].

[39] Media Inc, "Improve Test Asserts with Shouldly," 19 8 2015. [Online]. Available: https://visualstudiomagazine.com/articles/2015/08/01/improve-test-asserts-with-shouldly.aspx. [Acedido em 04 04 2020].

[40] A. Carlos, 7 1 2019. [Online]. Available: https://medium.com/beelabsolutions/aprendendo-testes-unit%C3%A1rios-com-xunit-e-shouldly-2c86bc84bd4b. [Acedido em 31 Março 2020].

[41] DEVMedia, "Teste de integração na prática," 2014. [Online]. Available: https://www.devmedia.com.br/teste-de-integracao-na-pratica/31877. [Acedido em 10 4 2020].

[42] A. Sales, "Qual a diferença entre testes manuais, testes automatizados e crowdtesting?," 27 04 2018. [Online]. Available: https://medium.com/@andersontestr/qual-a-diferen%C3%A7a-entre-testes-manuais-testes-automatizados-e-crowdtesting-b8f7caa13875. [Acedido em 10 4 2020].

[43] Prime Control, "Automação de Testes - o guia do gestor," em *Automação de Testes - o guia do gestor*, Curitiba, DRB Marketing, 2018, p. 29.

[44] Loader, "Loader," SendGrid Labs, 2016. [Online]. Available: https://loader.io. [Acedido em 28 4 2020].

[45] SendGrid, "Loader," 2020. [Online]. Available: https://loader.io/. [Acedido em 8 4 2020].

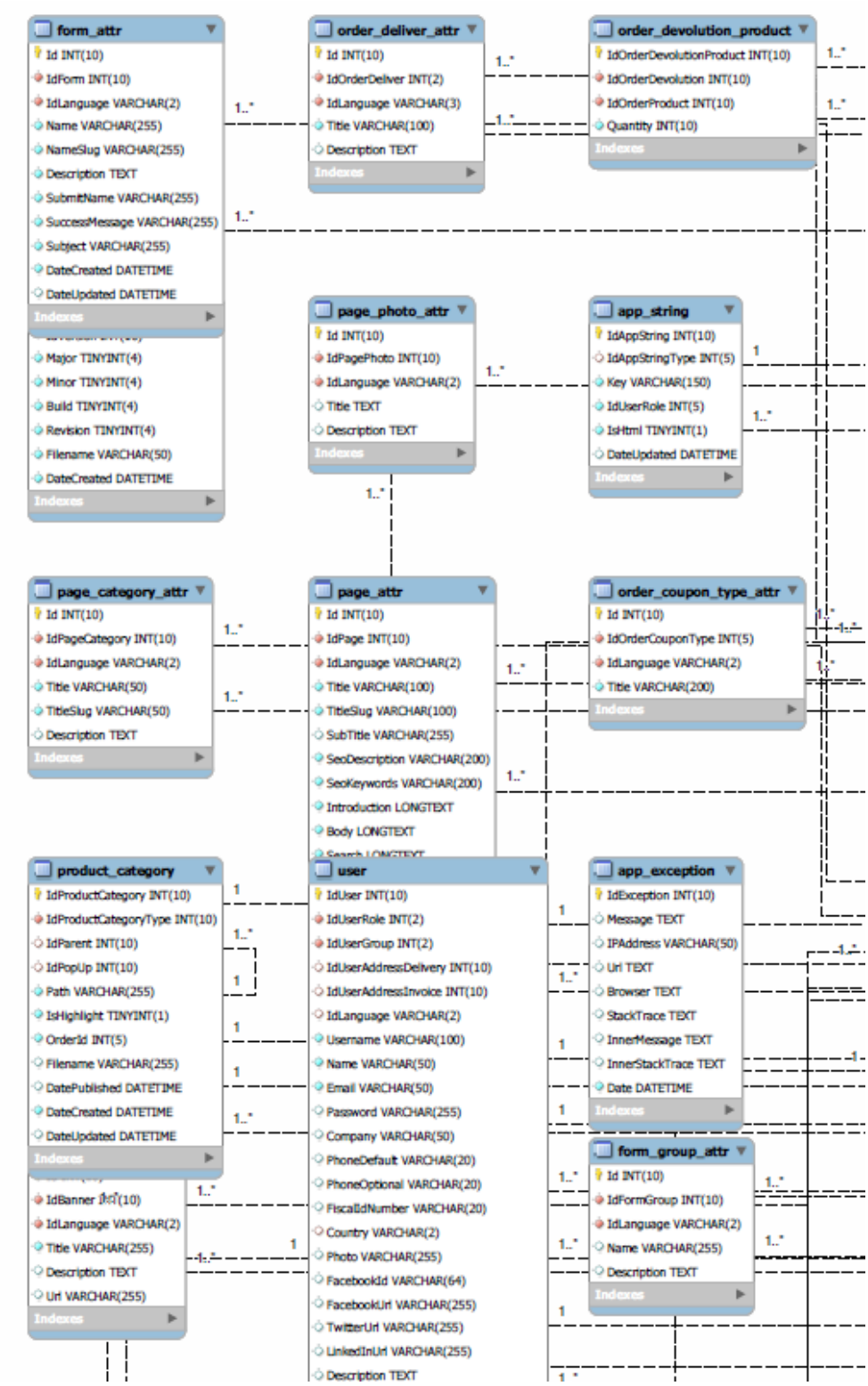
[46] L. Z. R. Mendes, "E-commerce : origem, desenvolvimento e perspectivas," Porto Alegre, 2013.

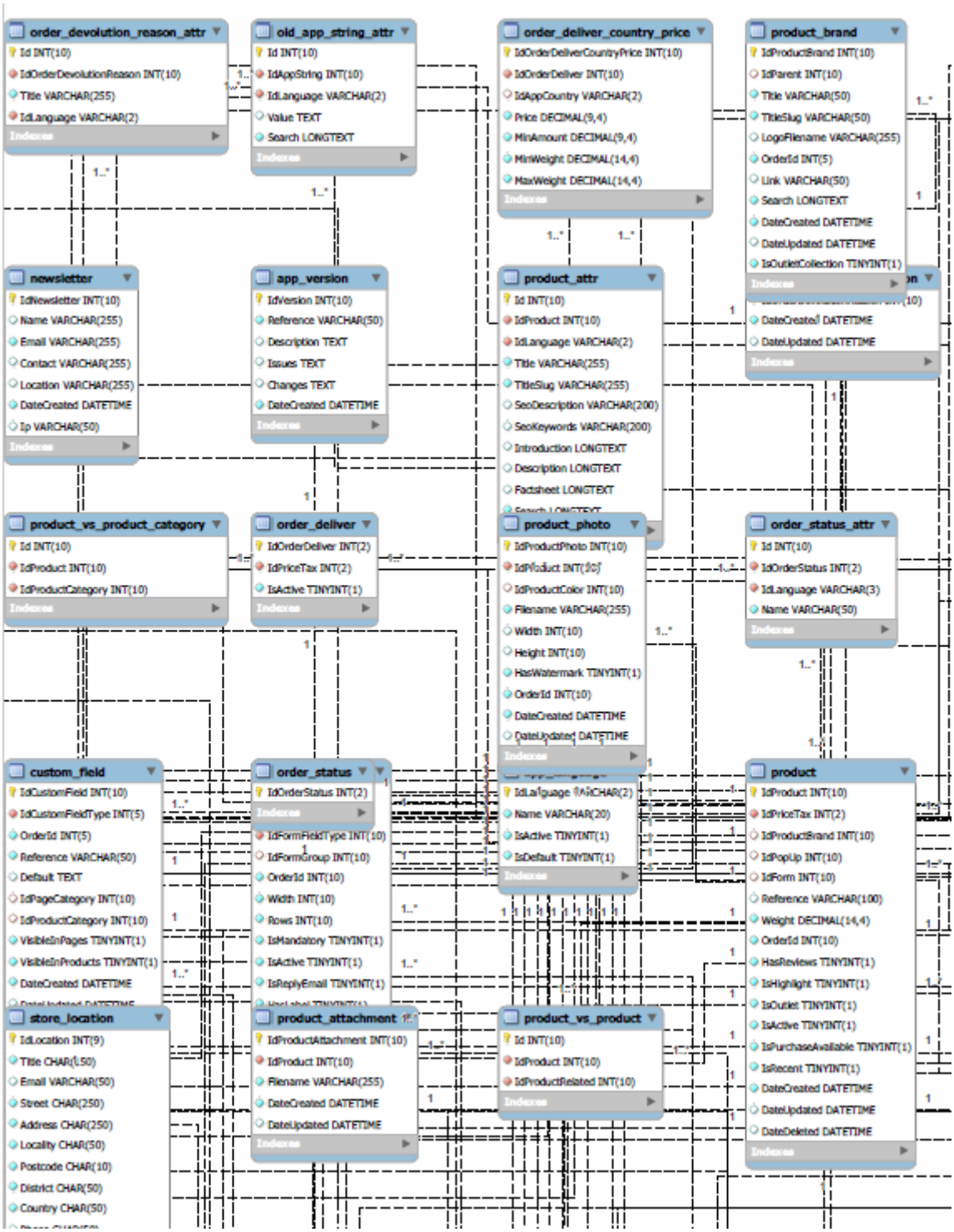
[47] Microsoft, "ASP.NET," [Online]. Available: https://dotnet.microsoft.com/apps/aspnet. [Acedido em 17 1 2020].

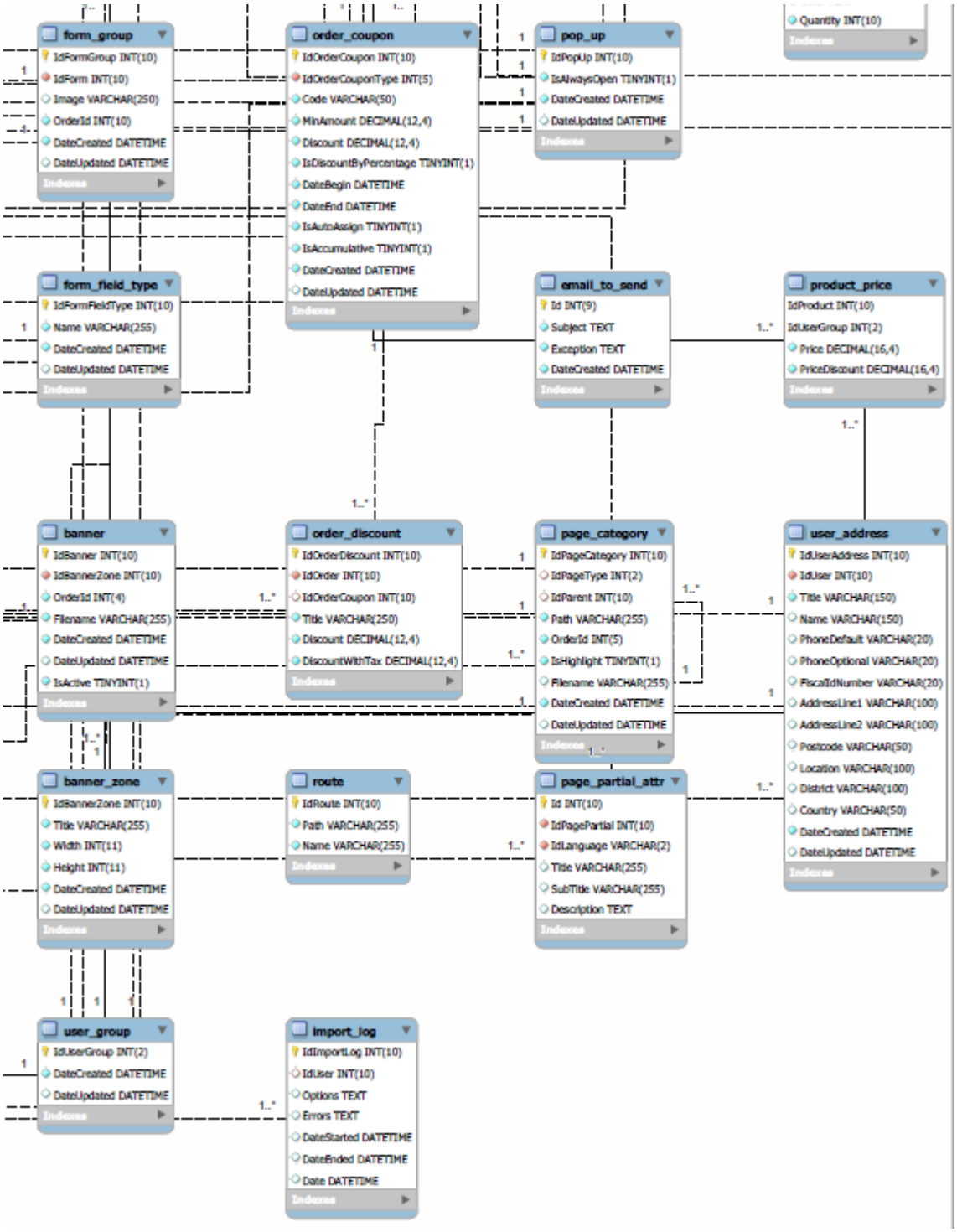
[48] Sebastián Peyrott, "JWT Handbook," em *JWT Handbook*, 2016-2018.

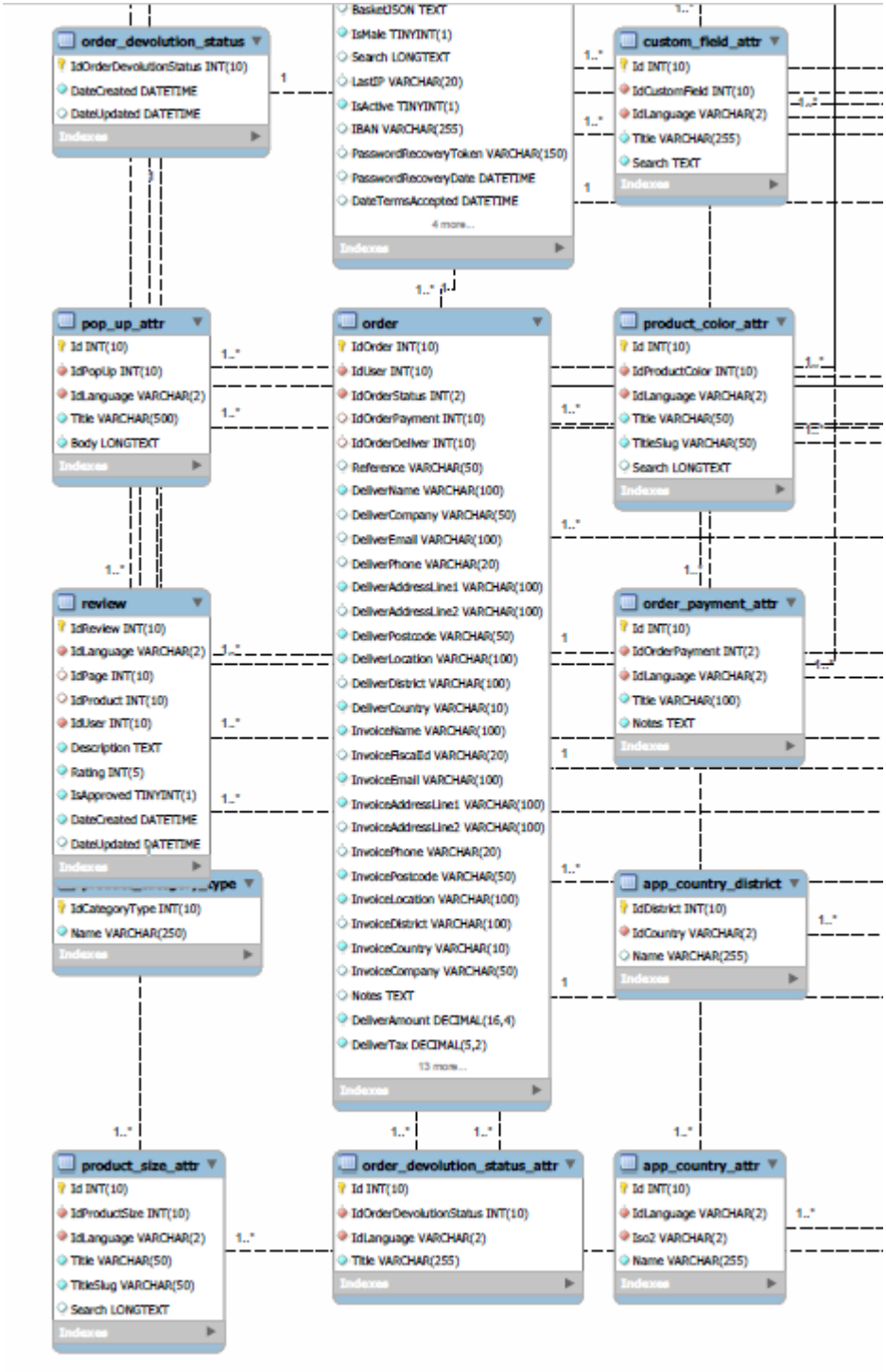
Anexos

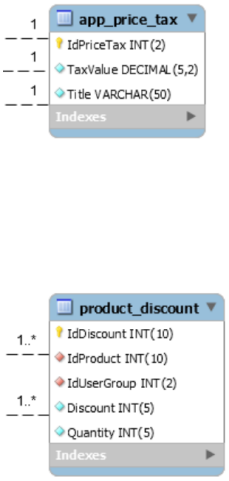
Anexo A




order_devolution_reason_attr
Id INT(10)
IdOrderDevolutionReason INT(10)
Title VARCHAR(255)
IdLanguage VARCHAR(2)
old_app_string_attr
Id INT(10)
IdAppString INT(10)
IdLanguage VARCHAR(2)
Value TEXT
Search LONGTEXT
order_deliver_country_price
IdOrderDeliverCountryPrice INT(10)
IdOrderDeliver INT(10)
IdAppCountry VARCHAR(2)
Price DECIMAL(9,4)
MinAmount DECIMAL(9,4)
MinWeight DECIMAL(14,4)
MaxWeight DECIMAL(14,4)
product_brand
IdProductBrand INT(10)
IdParent INT(10)
Title VARCHAR(50)
TitleSlug VARCHAR(50)
LogoFilename VARCHAR(255)
OrderId INT(5)
Link VARCHAR(50)
Search LONGTEXT
DateCreated DATETIME
DateUpdated DATETIME
IsOutletCollection TINYINT(1)
newsletter
IdNewsletter INT(10)
Name VARCHAR(255)
Email VARCHAR(255)
Contact VARCHAR(255)
Location VARCHAR(255)
DateCreated DATETIME
Ip VARCHAR(50)
app_version
IdVersion INT(10)
Reference VARCHAR(50)
Description TEXT
Issues TEXT
Changes TEXT
DateCreated DATETIME
product_attr
Id INT(10)
IdProduct INT(10)
IdLanguage VARCHAR(2)
Title VARCHAR(255)
TitleSlug VARCHAR(255)
SeoDescription VARCHAR(200)
SeoKeywords VARCHAR(200)
Introduction LONGTEXT
Description LONGTEXT
Factsheet LONGTEXT
DateCreated DATETIME
DateUpdated DATETIME
product_vs_product_category
Id INT(10)
IdProduct INT(10)
IdProductCategory INT(10)
order_deliver
IdOrderDeliver INT(2)
IdPriceTax INT(2)
IsActive TINYINT(1)
product_photo
IdProductPhoto INT(10)
IdProduct INT(10)
IdProductColor INT(10)
Filename VARCHAR(255)
Width INT(10)
Height INT(10)
HasWatermark TINYINT(1)
OrderId INT(10)
DateCreated DATETIME
DateUpdated DATETIME
order_status_attr
Id INT(10)
IdOrderStatus INT(2)
IdLanguage VARCHAR(3)
Name VARCHAR(50)
custom_field
IdCustomField INT(10)
IdCustomFieldType INT(5)
OrderId INT(5)
Reference VARCHAR(50)
Default TEXT
IdPageCategory INT(10)
IdProductCategory INT(10)
VisibleInPages TINYINT(1)
VisibleInProducts TINYINT(1)
DateCreated DATETIME
order_status
IdOrderStatus INT(2)
IdFormFieldType INT(10)
IdFormGroup INT(10)
OrderId INT(10)
Width INT(10)
Rows INT(10)
IsMandatory TINYINT(1)
IsActive TINYINT(1)
IsReplyEmail TINYINT(1)
IdLanguage VARCHAR(2)
Name VARCHAR(20)
IsActive TINYINT(1)
IsDefault TINYINT(1)
product
IdProduct INT(10)
IdPriceTax INT(2)
IdProductBrand INT(10)
IdPopUp INT(10)
IdForm INT(10)
Reference VARCHAR(100)
Weight DECIMAL(14,4)
OrderId INT(10)
HasReviews TINYINT(1)
IsHighlight TINYINT(1)
IsOutlet TINYINT(1)
IsActive TINYINT(1)
IsPurchaseAvailable TINYINT(1)
IsRecent TINYINT(1)
DateCreated DATETIME
DateUpdated DATETIME
DateDeleted DATETIME
store_location
IdLocation INT(9)
Title CHAR(150)
Email VARCHAR(50)
Street CHAR(250)
Address CHAR(250)
Locality CHAR(50)
Postcode CHAR(10)
District CHAR(50)
Country CHAR(50)
product_attachment
IdProductAttachment INT(10)
IdProduct INT(10)
Filename VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
product_vs_product
Id INT(10)
IdProduct INT(10)
IdProductRelated INT(10)

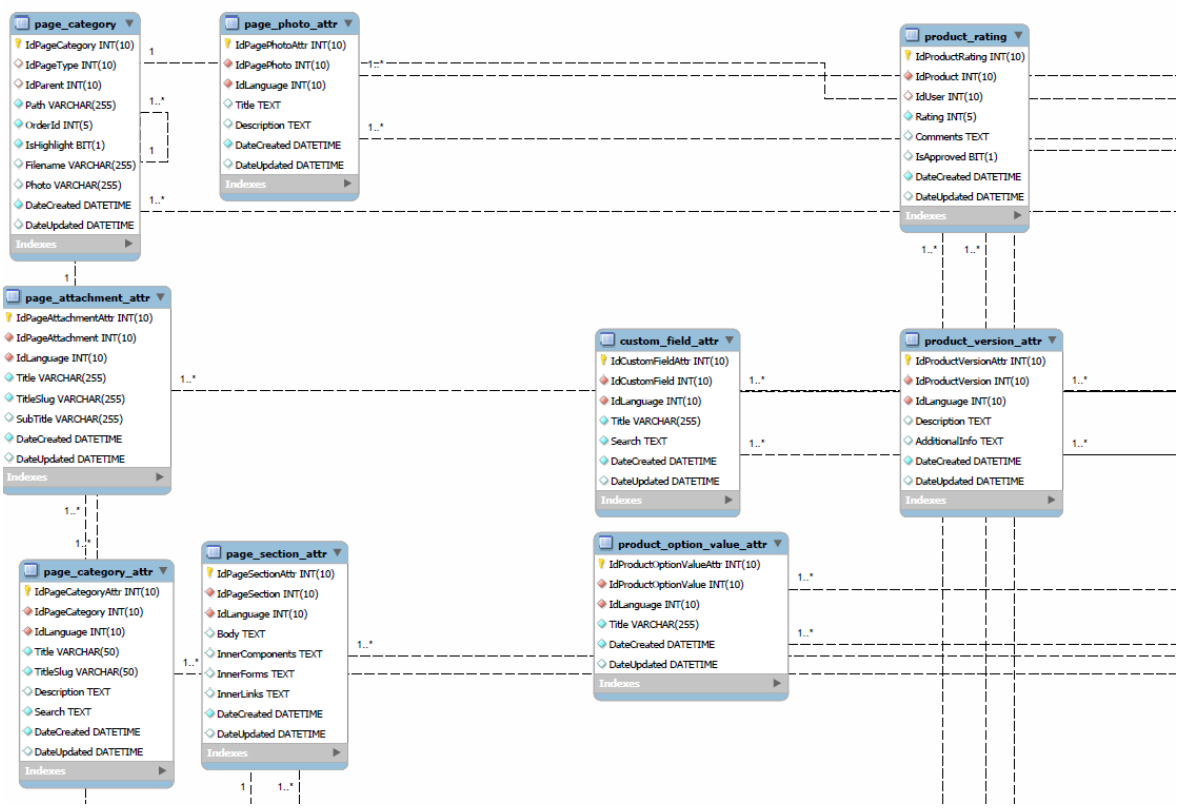

form_group
IdFormGroup INT(10)
IdForm INT(10)
Image VARCHAR(250)
OrderId INT(10)
DateCreated DATETIME
DateUpdated DATETIME
order_coupon
IdOrderCoupon INT(10)
IdOrderCouponType INT(5)
Code VARCHAR(50)
MinAmount DECIMAL(12,4)
Discount DECIMAL(12,4)
IsDiscountByPercentage TINYINT(1)
DateBegin DATETIME
DateEnd DATETIME
IsAutoAssign TINYINT(1)
IsAccumulative TINYINT(1)
DateCreated DATETIME
DateUpdated DATETIME
pop_up
IdPopUp INT(10)
IsAlwaysOpen TINYINT(1)
DateCreated DATETIME
DateUpdated DATETIME
Quantity INT(10)
form_field_type
IdFormFieldType INT(10)
Name VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
email_to_send
Id INT(9)
Subject TEXT
Exception TEXT
DateCreated DATETIME
product_price
IdProduct INT(10)
IdUserGroup INT(2)
Price DECIMAL(16,4)
PriceDiscount DECIMAL(16,4)
banner
IdBanner INT(10)
IdBannerZone INT(10)
OrderId INT(4)
Filename VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
IsActive TINYINT(1)
order_discount
IdOrderDiscount INT(10)
IdOrder INT(10)
IdOrderCoupon INT(10)
Title VARCHAR(250)
Discount DECIMAL(12,4)
DiscountWithTax DECIMAL(12,4)
page_category
IdPageCategory INT(10)
IdPageType INT(2)
IdParent INT(10)
Path VARCHAR(255)
OrderId INT(5)
IsHighlight TINYINT(1)
Filename VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
user_address
IdUserAddress INT(10)
IdUser INT(10)
Title VARCHAR(150)
Name VARCHAR(150)
PhoneDefault VARCHAR(20)
PhoneOptional VARCHAR(20)
FiscalIdNumber VARCHAR(20)
AddressLine1 VARCHAR(100)
AddressLine2 VARCHAR(100)
Postcode VARCHAR(50)
Location VARCHAR(100)
District VARCHAR(100)
Country VARCHAR(50)
DateCreated DATETIME
DateUpdated DATETIME
banner_zone
IdBannerZone INT(10)
Title VARCHAR(255)
Width INT(11)
Height INT(11)
DateCreated DATETIME
DateUpdated DATETIME
route
IdRoute INT(10)
Path VARCHAR(255)
Name VARCHAR(255)
page_partial_attr
Id INT(10)
IdPagePartial INT(10)
IdLanguage VARCHAR(2)
Title VARCHAR(255)
SubTitle VARCHAR(255)
Description TEXT
user_group
IdUserGroup INT(2)
DateCreated DATETIME
DateUpdated DATETIME
import_log
IdImportLog INT(10)
IdUser INT(10)
Options TEXT
Errors TEXT
DateStarted DATETIME
DateEnded DATETIME
Date DATETIME

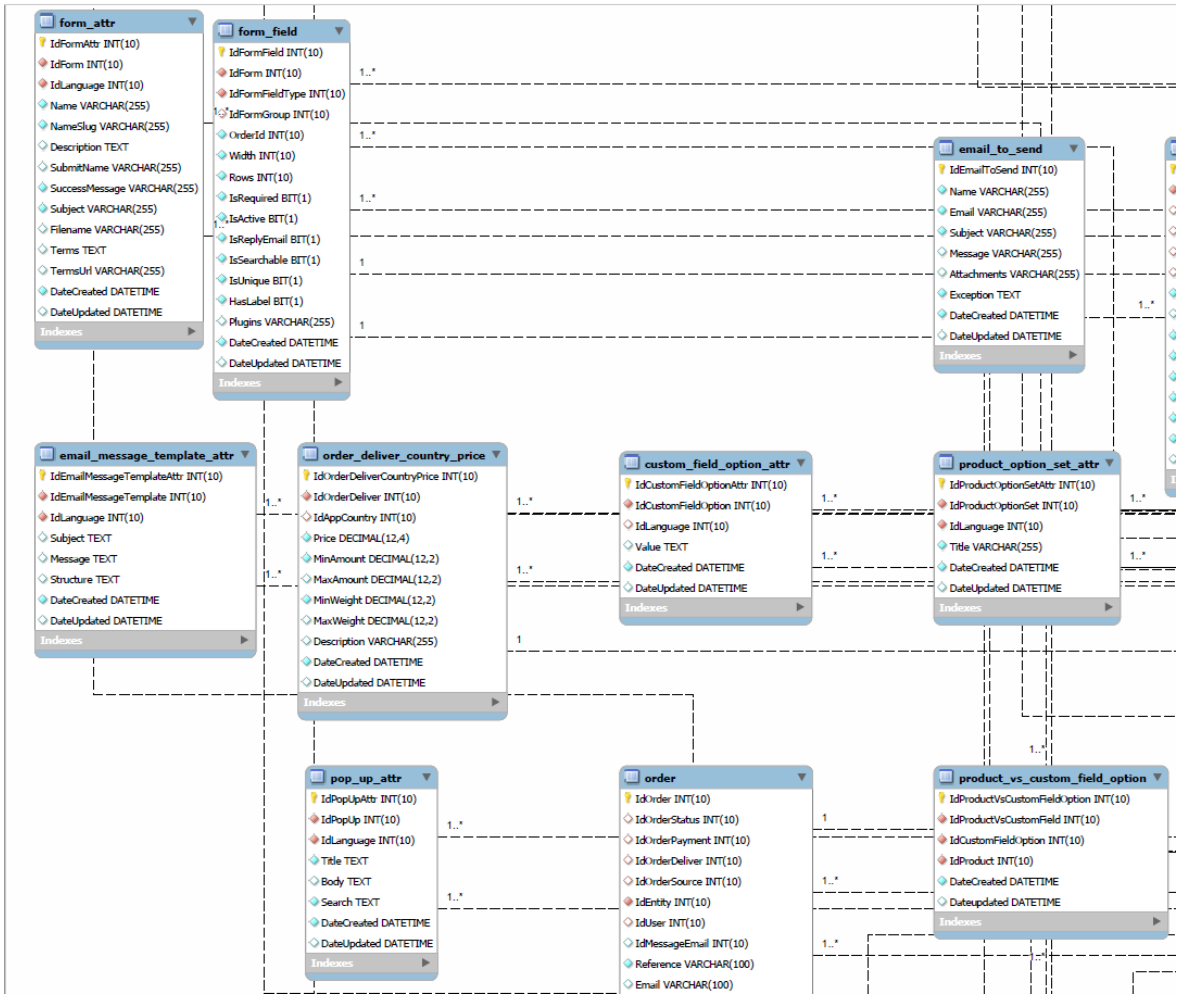

order_devolution_status
IdOrderDevolutionStatus INT(10)
DateCreated DATETIME
DateUpdated DATETIME
BasketJSON TEXT
IsMale TINYINT(1)
Search LONGTEXT
LastIP VARCHAR(20)
IsActive TINYINT(1)
IBAN VARCHAR(255)
PasswordRecoveryToken VARCHAR(150)
PasswordRecoveryDate DATETIME
DateTermsAccepted DATETIME
custom_field_attr
Id INT(10)
IdCustomField INT(10)
IdLanguage VARCHAR(2)
Title VARCHAR(255)
Search TEXT
pop_up_attr
Id INT(10)
IdPopUp INT(10)
IdLanguage VARCHAR(2)
Title VARCHAR(500)
Body LONGTEXT
order
IdOrder INT(10)
IdUser INT(10)
IdOrderStatus INT(2)
IdOrderPayment INT(10)
IdOrderDeliver INT(10)
Reference VARCHAR(50)
DeliverName VARCHAR(100)
DeliverCompany VARCHAR(50)
DeliverEmail VARCHAR(100)
DeliverPhone VARCHAR(20)
DeliverAddressLine1 VARCHAR(100)
DeliverAddressLine2 VARCHAR(100)
DeliverPostcode VARCHAR(50)
DeliverLocation VARCHAR(100)
DeliverDistrict VARCHAR(100)
DeliverCountry VARCHAR(10)
InvoiceName VARCHAR(100)
InvoiceFiscalId VARCHAR(20)
InvoiceEmail VARCHAR(100)
InvoiceAddressLine1 VARCHAR(100)
InvoiceAddressLine2 VARCHAR(100)
InvoicePhone VARCHAR(20)
InvoicePostcode VARCHAR(50)
InvoiceLocation VARCHAR(100)
InvoiceDistrict VARCHAR(100)
InvoiceCountry VARCHAR(10)
InvoiceCompany VARCHAR(50)
Notes TEXT
DeliverAmount DECIMAL(16,4)
DeliverTax DECIMAL(5,2)
product_color_attr
Id INT(10)
IdProductColor INT(10)
IdLanguage VARCHAR(2)
Title VARCHAR(50)
TitleSlug VARCHAR(50)
Search LONGTEXT
review
IdReview INT(10)
IdLanguage VARCHAR(2)
IdPage INT(10)
IdProduct INT(10)
IdUser INT(10)
Description TEXT
Rating INT(5)
IsApproved TINYINT(1)
DateCreated DATETIME
DateUpdated DATETIME
order_payment_attr
Id INT(10)
IdOrderPayment INT(2)
IdLanguage VARCHAR(2)
Title VARCHAR(100)
Notes TEXT
ype
IdCategoryType INT(10)
Name VARCHAR(250)
app_country_district
IdDistrict INT(10)
IdCountry VARCHAR(2)
Name VARCHAR(255)
product_size_attr
Id INT(10)
IdProductSize INT(10)
IdLanguage VARCHAR(2)
Title VARCHAR(50)
TitleSlug VARCHAR(50)
Search LONGTEXT
order_devolution_status_attr
Id INT(10)
IdOrderDevolutionStatus INT(10)
IdLanguage VARCHAR(2)
Title VARCHAR(255)
app_country_attr
Id INT(10)
IdLanguage VARCHAR(2)
Iso2 VARCHAR(2)
Name VARCHAR(255)

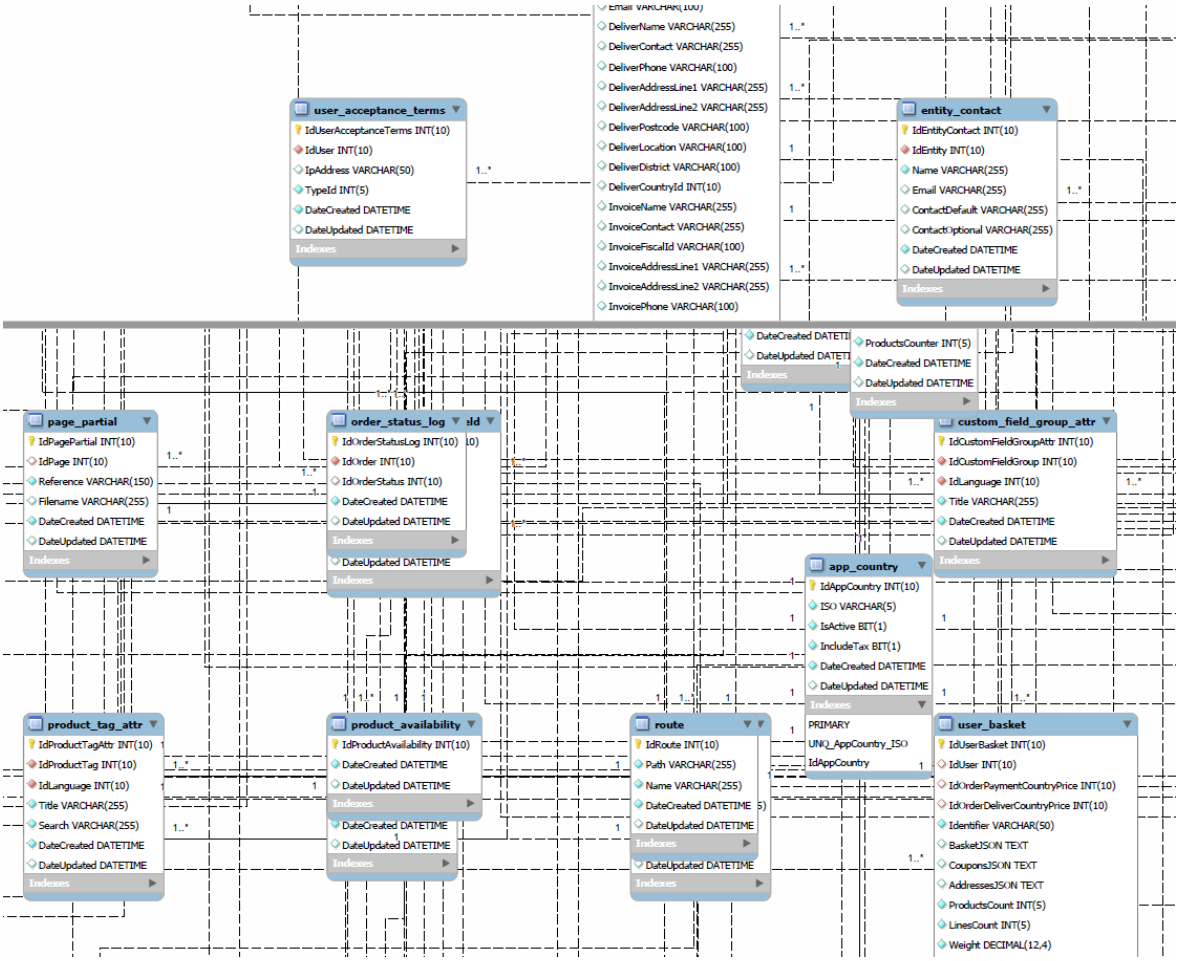


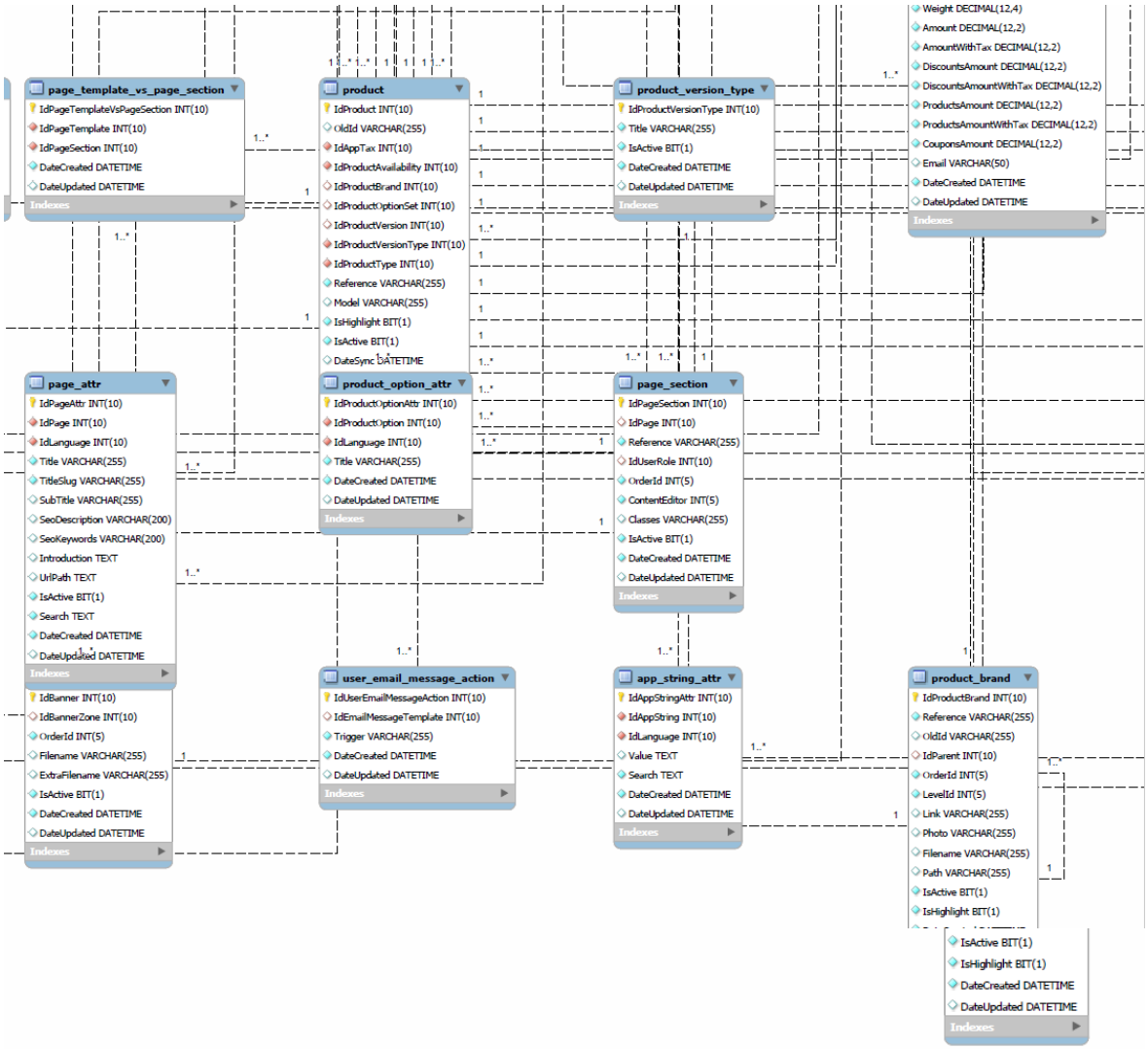
Anexo A - Base de dados da antiga plataforma

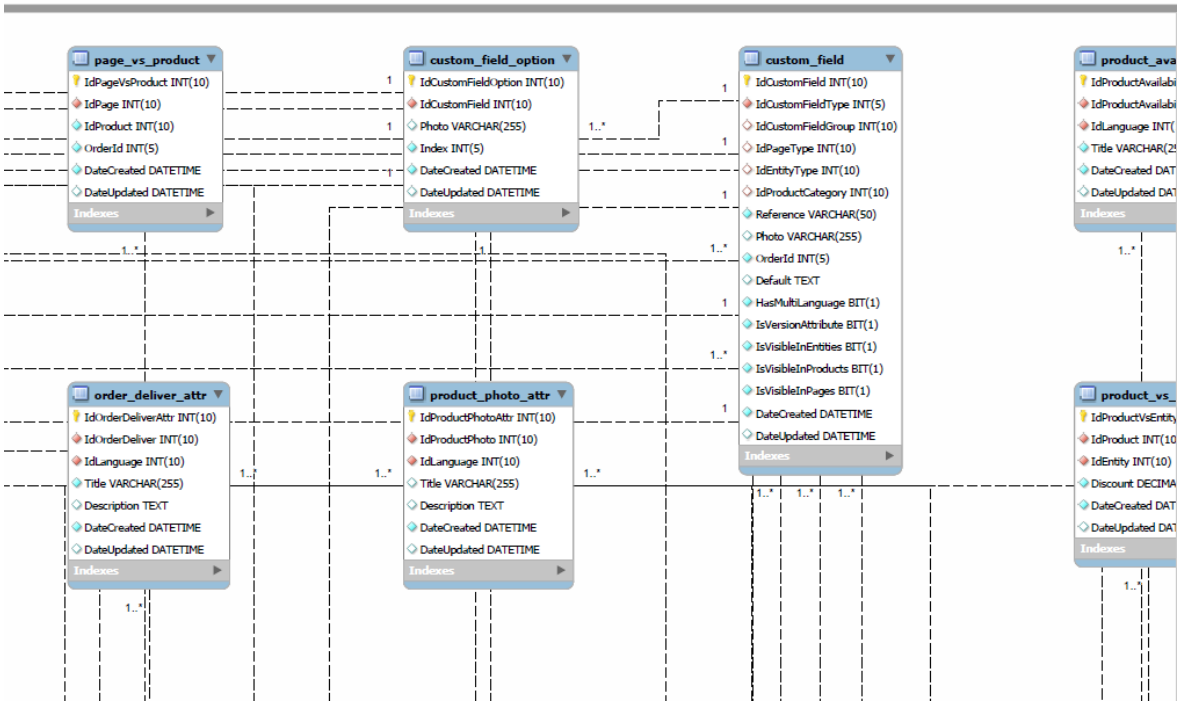
Anexo B

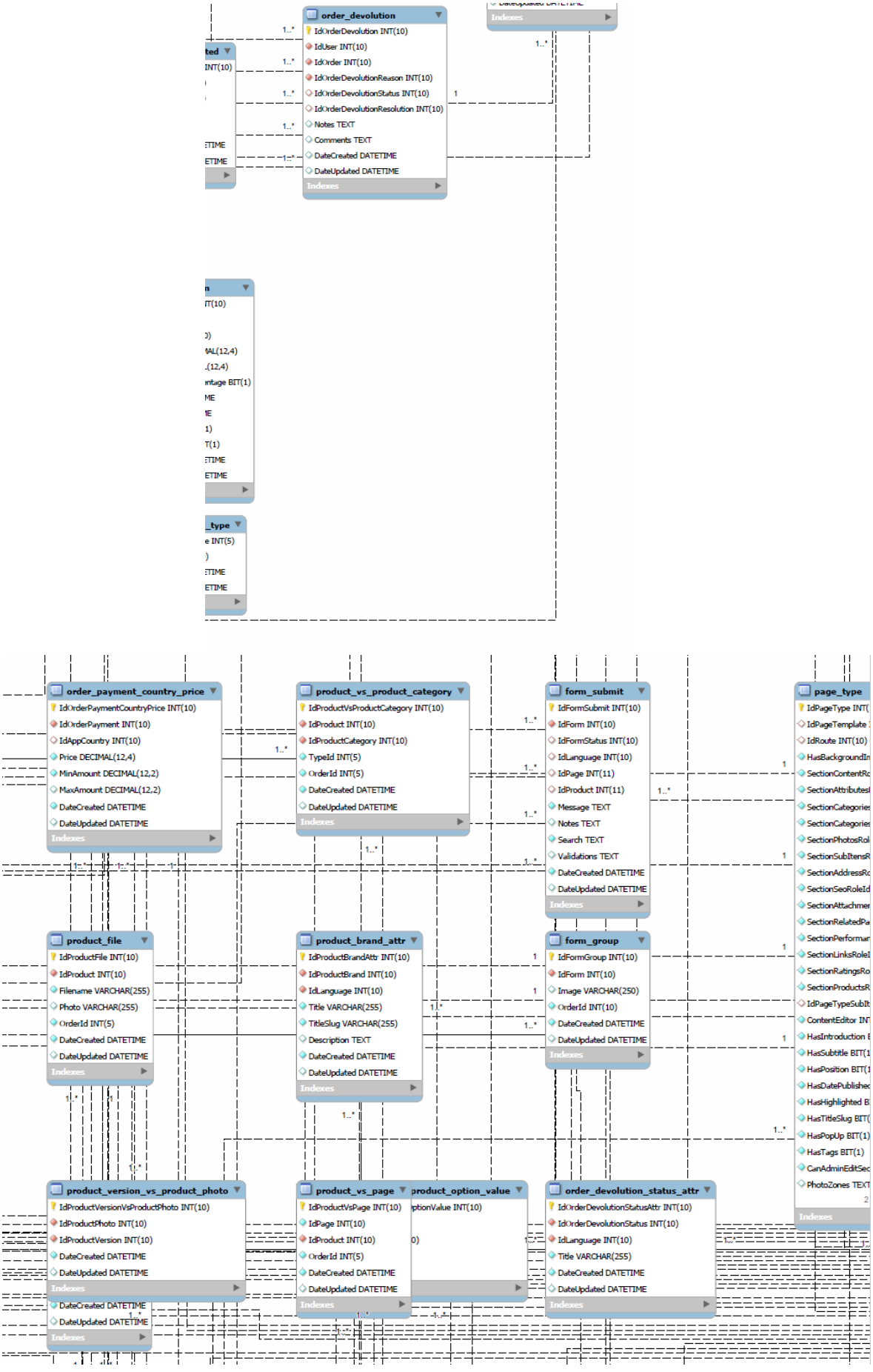

page_category
IdPageCategory INT(10)
IdPageType INT(10)
IdParent INT(10)
Path VARCHAR(255)
OrderId INT(5)
IsHighlight BIT(1)
Filename VARCHAR(255)
Photo VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_photo_attr
IdPagePhotoAttr INT(10)
IdPagePhoto INT(10)
IdLanguage INT(10)
Title TEXT
Description TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_rating
IdProductRating INT(10)
IdProduct INT(10)
IdUser INT(10)
Rating INT(5)
Comments TEXT
IsApproved BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_attachment_attr
IdPageAttachmentAttr INT(10)
IdPageAttachment INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
TitleSlug VARCHAR(255)
SubTitle VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
custom_field_attr
IdCustomFieldAttr INT(10)
IdCustomField INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Search TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_version_attr
IdProductVersionAttr INT(10)
IdProductVersion INT(10)
IdLanguage INT(10)
Description TEXT
AdditionalInfo TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_category_attr
IdPageCategoryAttr INT(10)
IdPageCategory INT(10)
IdLanguage INT(10)
Title VARCHAR(50)
TitleSlug VARCHAR(50)
Description TEXT
Search TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_section_attr
IdPageSectionAttr INT(10)
IdPageSection INT(10)
IdLanguage INT(10)
Body TEXT
InnerComponents TEXT
InnerForms TEXT
InnerLinks TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_option_value_attr
IdProductOptionValueAttr INT(10)
IdProductOptionValue INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes

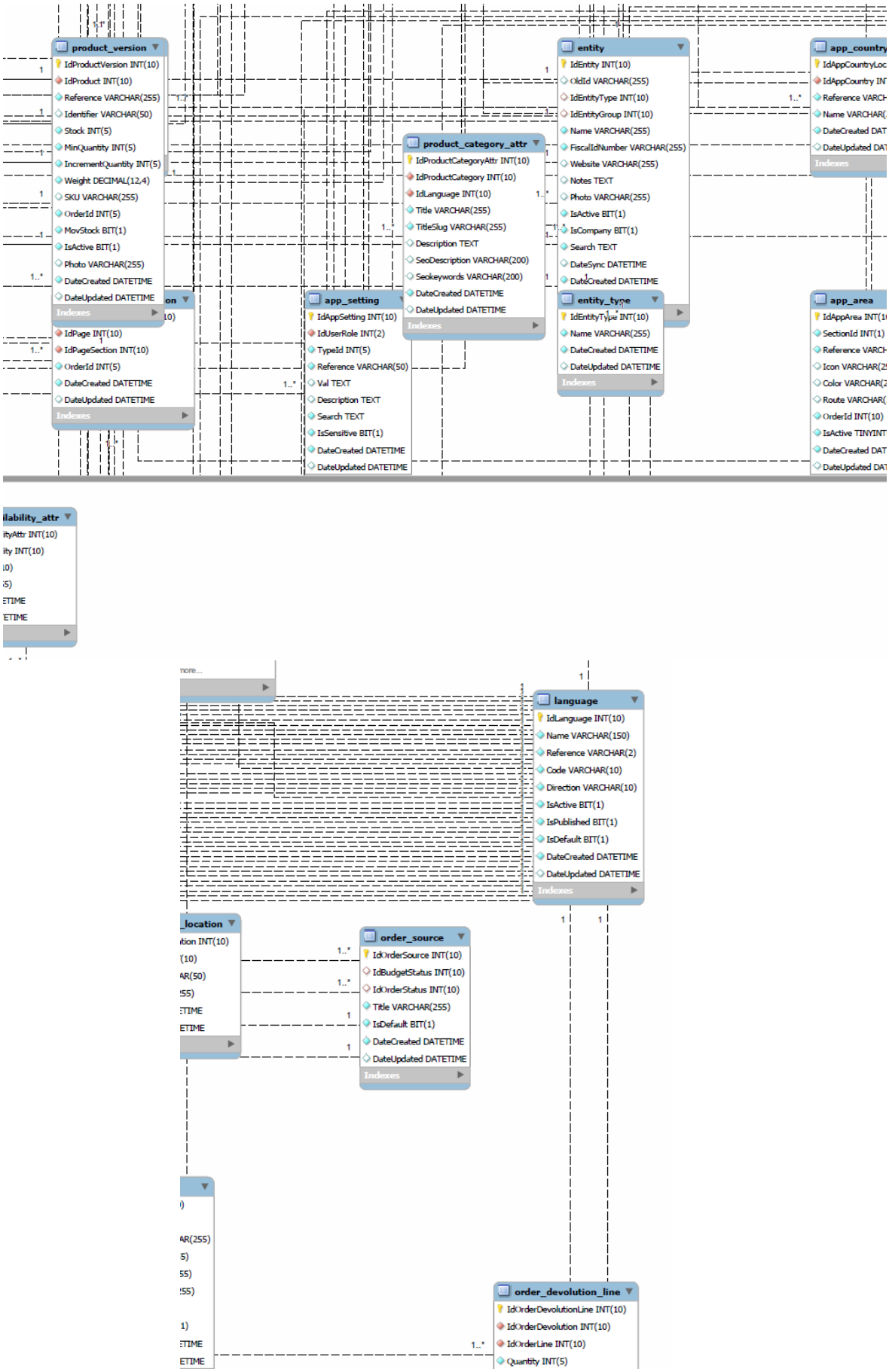

form_attr
IdFormAttr INT(10)
IdForm INT(10)
IdLanguage INT(10)
Name VARCHAR(255)
NameSlug VARCHAR(255)
Description TEXT
SubmitName VARCHAR(255)
SuccessMessage VARCHAR(255)
Subject VARCHAR(255)
Filename VARCHAR(255)
Terms TEXT
TermsUrl VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
form_field
IdFormField INT(10)
IdForm INT(10)
IdFormFieldType INT(10)
IdFormGroup INT(10)
OrderId INT(10)
Width INT(10)
Rows INT(10)
IsRequired BIT(1)
IsActive BIT(1)
IsReplyEmail BIT(1)
IsSearchable BIT(1)
IsUnique BIT(1)
HasLabel BIT(1)
Plugins VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
email_to_send
IdEmailToSend INT(10)
Name VARCHAR(255)
Email VARCHAR(255)
Subject VARCHAR(255)
Message VARCHAR(255)
Attachments VARCHAR(255)
Exception TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
email_message_template_attr
IdEmailMessageTemplateAttr INT(10)
IdEmailMessageTemplate INT(10)
IdLanguage INT(10)
Subject TEXT
Message TEXT
Structure TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_deliver_country_price
IdOrderDeliverCountryPrice INT(10)
IdOrderDeliver INT(10)
IdAppCountry INT(10)
Price DECIMAL(12,4)
MinAmount DECIMAL(12,2)
MaxAmount DECIMAL(12,2)
MinWeight DECIMAL(12,2)
MaxWeight DECIMAL(12,2)
Description VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
custom_field_option_attr
IdCustomFieldOptionAttr INT(10)
IdCustomFieldOption INT(10)
IdLanguage INT(10)
Value TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_option_set_attr
IdProductOptionSetAttr INT(10)
IdProductOptionSet INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
pop_up_attr
IdPopUpAttr INT(10)
IdPopUp INT(10)
IdLanguage INT(10)
Title TEXT
Body TEXT
Search TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order
IdOrder INT(10)
IdOrderStatus INT(10)
IdOrderPayment INT(10)
IdOrderDeliver INT(10)
IdOrderSource INT(10)
IdEntity INT(10)
IdUser INT(10)
IdMessageEmail INT(10)
Reference VARCHAR(100)
Email VARCHAR(100)
product_vs_custom_field_option
IdProductVsCustomFieldOption INT(10)
IdProductVsCustomField INT(10)
IdCustomFieldOption INT(10)
IdProduct INT(10)
DateCreated DATETIME
Dateupdated DATETIME
Indexes

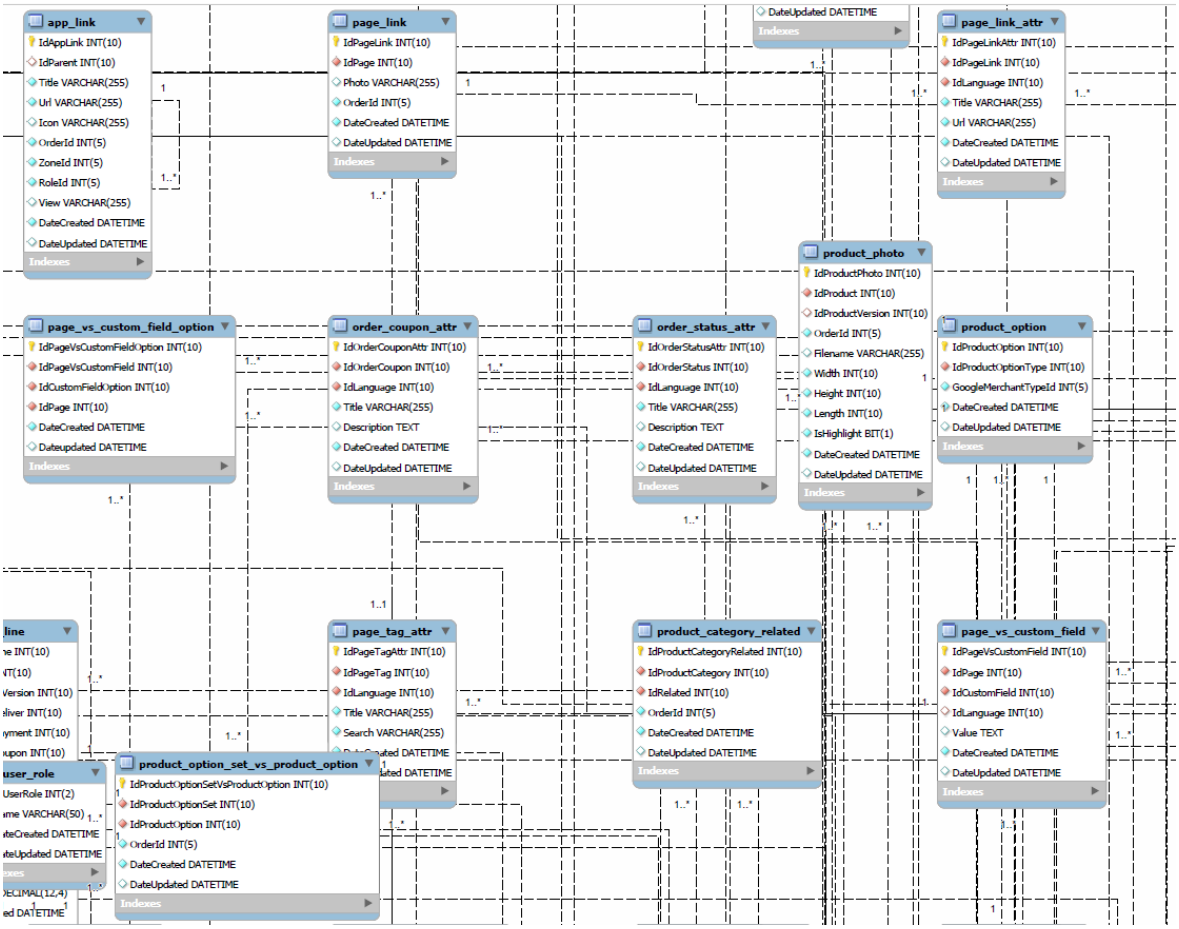

user_acceptance_terms
IdUserAcceptanceTerms INT(10)
IdUser INT(10)
IpAddress VARCHAR(50)
TypeId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
DeliverName VARCHAR(255)
DeliverContact VARCHAR(255)
DeliverPhone VARCHAR(100)
DeliverAddressLine1 VARCHAR(255)
DeliverAddressLine2 VARCHAR(255)
DeliverPostcode VARCHAR(100)
DeliverLocation VARCHAR(100)
DeliverDistrict VARCHAR(100)
DeliverCountryId INT(10)
InvoiceName VARCHAR(255)
InvoiceContact VARCHAR(255)
InvoiceFiscalId VARCHAR(100)
InvoiceAddressLine1 VARCHAR(255)
InvoiceAddressLine2 VARCHAR(255)
InvoicePhone VARCHAR(100)
entity_contact
IdEntityContact INT(10)
IdEntity INT(10)
Name VARCHAR(255)
Email VARCHAR(255)
ContactDefault VARCHAR(255)
ContactOptional VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
ProductsCounter INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_partial
IdPagePartial INT(10)
IdPage INT(10)
Reference VARCHAR(150)
Filename VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_status_log
IdOrderStatusLog INT(10)
IdOrder INT(10)
IdOrderStatus INT(10)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
DateUpdated DATETIME
Indexes
custom_field_group_attr
IdCustomFieldGroupAttr INT(10)
IdCustomFieldGroup INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
app_country
IdAppCountry INT(10)
ISO VARCHAR(5)
IsActive BIT(1)
IncludeTax BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
PRIMARY
UNQ_AppCountry_ISO
IdAppCountry
product_tag_attr
IdProductTagAttr INT(10)
IdProductTag INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Search VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_availability
IdProductAvailability INT(10)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
DateCreated DATETIME
DateUpdated DATETIME
Indexes
route
IdRoute INT(10)
Path VARCHAR(255)
Name VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
DateUpdated DATETIME
Indexes
user_basket
IdUserBasket INT(10)
IdUser INT(10)
IdOrderPaymentCountryPrice INT(10)
IdOrderDeliverCountryPrice INT(10)
Identifier VARCHAR(50)
BasketJSON TEXT
CouponsJSON TEXT
AddressesJSON TEXT
ProductsCount INT(5)
LinesCount INT(5)
Weight DECIMAL(12,4)

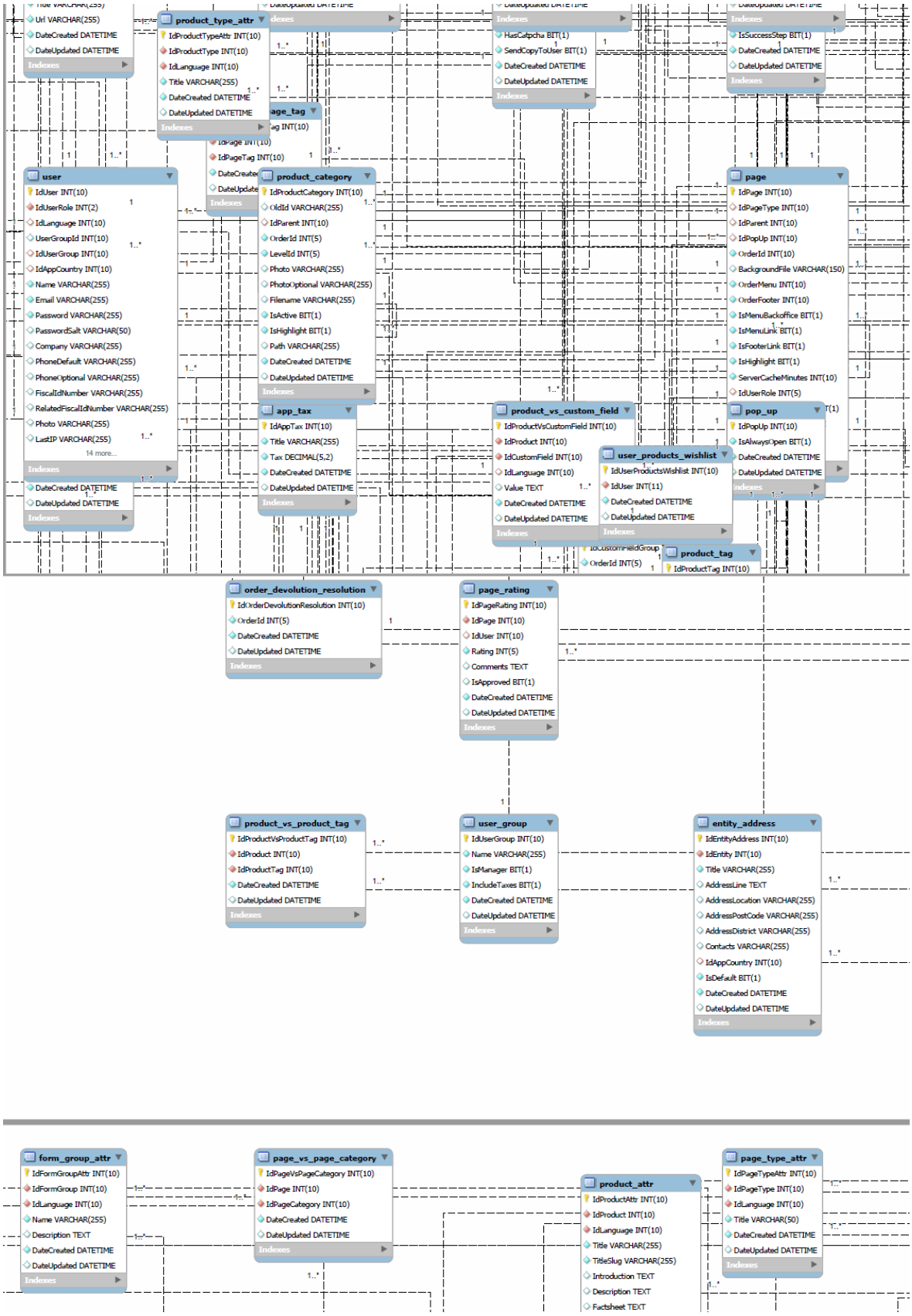

page_template_vs_page_section
IdPageTemplateVsPageSection INT(10)
IdPageTemplate INT(10)
IdPageSection INT(10)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product
IdProduct INT(10)
OldId VARCHAR(255)
IdAppTax INT(10)
IdProductAvailability INT(10)
IdProductBrand INT(10)
IdProductOptionSet INT(10)
IdProductVersion INT(10)
IdProductVersionType INT(10)
IdProductType INT(10)
Reference VARCHAR(255)
Model VARCHAR(255)
IsHighlight BIT(1)
IsActive BIT(1)
DateSync DATETIME
product_version_type
IdProductVersionType INT(10)
Title VARCHAR(255)
IsActive BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
Weight DECIMAL(12,4)
Amount DECIMAL(12,2)
AmountWithTax DECIMAL(12,2)
DiscountsAmount DECIMAL(12,2)
DiscountsAmountWithTax DECIMAL(12,2)
ProductsAmount DECIMAL(12,2)
ProductsAmountWithTax DECIMAL(12,2)
CouponsAmount DECIMAL(12,2)
Email VARCHAR(50)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_attr
IdPageAttr INT(10)
IdPage INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
TitleSlug VARCHAR(255)
SubTitle VARCHAR(255)
SeoDescription VARCHAR(200)
SeoKeywords VARCHAR(200)
Introduction TEXT
UrlPath TEXT
IsActive BIT(1)
Search TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_option_attr
IdProductOptionAttr INT(10)
IdProductOption INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_section
IdPageSection INT(10)
IdPage INT(10)
Reference VARCHAR(255)
IdUserRole INT(10)
OrderId INT(5)
ContentEditor INT(5)
Classes VARCHAR(255)
IsActive BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
IdBanner INT(10)
IdBannerZone INT(10)
OrderId INT(5)
Filename VARCHAR(255)
ExtraFilename VARCHAR(255)
IsActive BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
user_email_message_action
IdUserEmailMessageAction INT(10)
IdEmailMessageTemplate INT(10)
Trigger VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
app_string_attr
IdAppStringAttr INT(10)
IdAppString INT(10)
IdLanguage INT(10)
Value TEXT
Search TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_brand
IdProductBrand INT(10)
Reference VARCHAR(255)
OldId VARCHAR(255)
IdParent INT(10)
OrderId INT(5)
LevelId INT(5)
Link VARCHAR(255)
Photo VARCHAR(255)
Filename VARCHAR(255)
Path VARCHAR(255)
IsActive BIT(1)
IsHighlight BIT(1)
IsActive BIT(1)
IsHighlight BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes

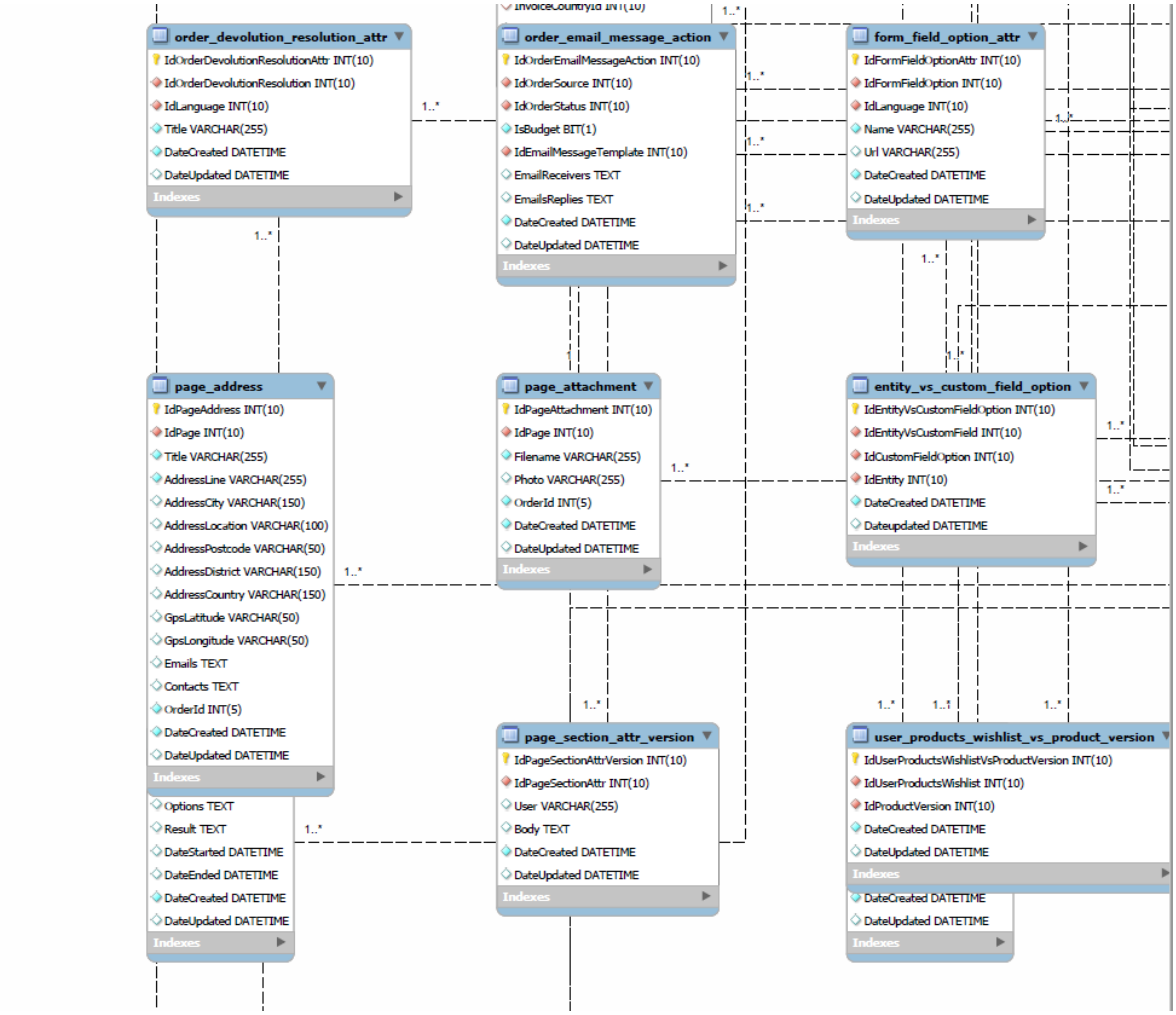

page_vs_product
IdPageVsProduct INT(10)
IdPage INT(10)
IdProduct INT(10)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
custom_field_option
IdCustomFieldOption INT(10)
IdCustomField INT(10)
Photo VARCHAR(255)
Index INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
custom_field
IdCustomField INT(10)
IdCustomFieldType INT(5)
IdCustomFieldGroup INT(10)
IdPageType INT(10)
IdEntityType INT(10)
IdProductCategory INT(10)
Reference VARCHAR(50)
Photo VARCHAR(255)
OrderId INT(5)
Default TEXT
HasMultiLanguage BIT(1)
IsVersionAttribute BIT(1)
IsVisibleInEntities BIT(1)
IsVisibleInProducts BIT(1)
IsVisibleInPages BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_deliver_attr
IdOrderDeliverAttr INT(10)
IdOrderDeliver INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Description TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_photo_attr
IdProductPhotoAttr INT(10)
IdProductPhoto INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Description TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes

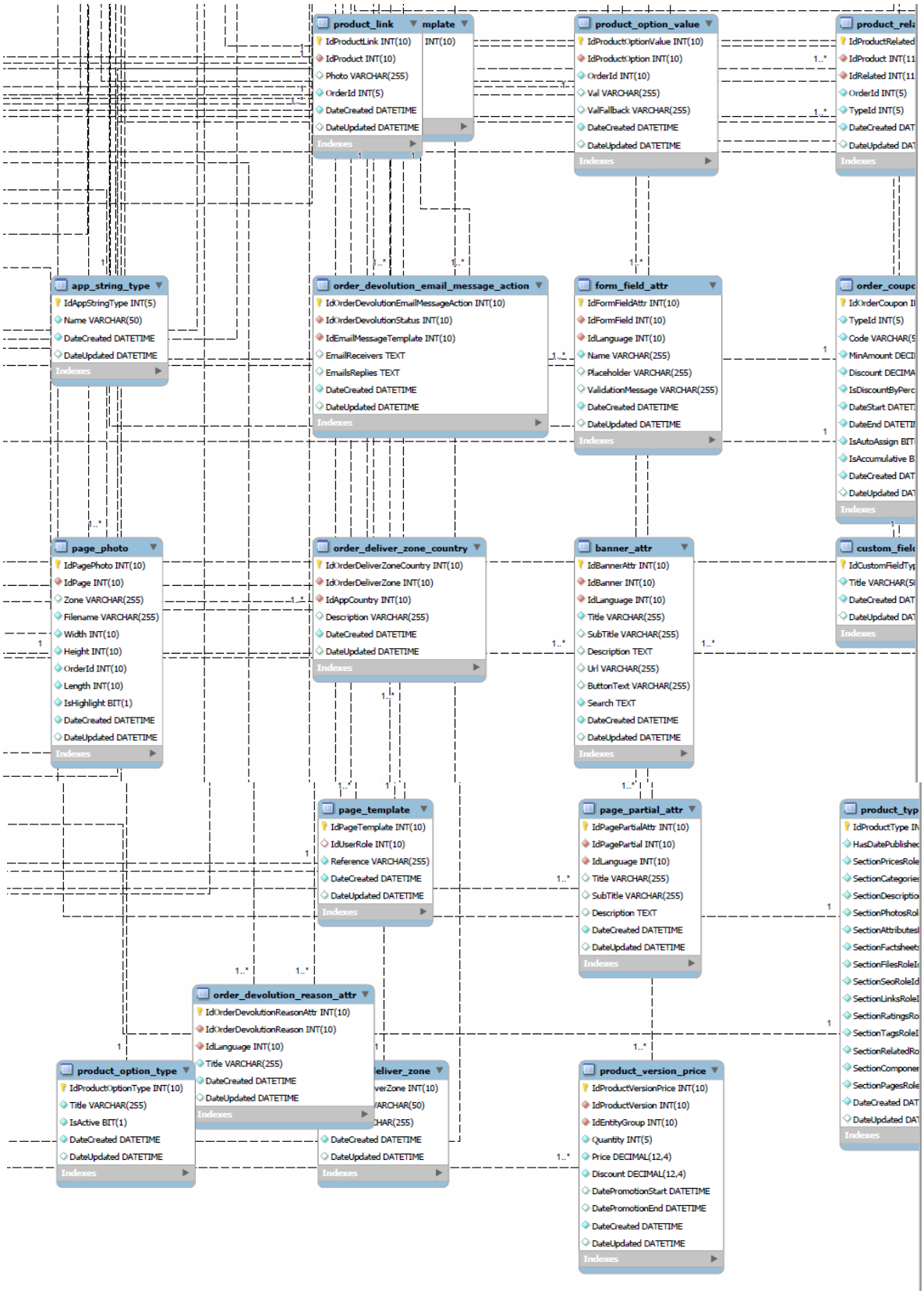

order_devolution
IdOrderDevolution INT(10)
IdUser INT(10)
IdOrder INT(10)
IdOrderDevolutionReason INT(10)
IdOrderDevolutionStatus INT(10)
IdOrderDevolutionResolution INT(10)
Notes TEXT
Comments TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_payment_country_price
IdOrderPaymentCountryPrice INT(10)
IdOrderPayment INT(10)
IdAppCountry INT(10)
Price DECIMAL(12,4)
MinAmount DECIMAL(12,2)
MaxAmount DECIMAL(12,2)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_vs_product_category
IdProductVsProductCategory INT(10)
IdProduct INT(10)
IdProductCategory INT(10)
TypeId INT(5)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
form_submit
IdFormSubmit INT(10)
IdForm INT(10)
IdFormStatus INT(10)
IdLanguage INT(10)
IdPage INT(11)
IdProduct INT(11)
Message TEXT
Notes TEXT
Search TEXT
Validations TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_type
product_file
IdProductFile INT(10)
IdProduct INT(10)
Filename VARCHAR(255)
Photo VARCHAR(255)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_brand_attr
IdProductBrandAttr INT(10)
IdProductBrand INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
TitleSlug VARCHAR(255)
Description TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
form_group
IdFormGroup INT(10)
IdForm INT(10)
Image VARCHAR(250)
OrderId INT(10)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_version_vs_product_photo
IdProductVersionVsProductPhoto INT(10)
IdProductPhoto INT(10)
IdProductVersion INT(10)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_vs_page
IdProductVsPage INT(10)
IdPage INT(10)
IdProduct INT(10)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_devolution_status_attr
IdOrderDevolutionStatusAttr INT(10)
IdOrderDevolutionStatus INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes


product_version
IdProductVersion INT(10)
IdProduct INT(10)
Reference VARCHAR(255)
Identifier VARCHAR(50)
Stock INT(5)
MinQuantity INT(5)
IncrementQuantity INT(5)
Weight DECIMAL(12,4)
SKU VARCHAR(255)
OrderId INT(5)
MovStock BIT(1)
IsActive BIT(1)
Photo VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
IdPage INT(10)
IdPageSection INT(10)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
app_setting
IdAppSetting INT(10)
IdUserRole INT(2)
TypeId INT(5)
Reference VARCHAR(50)
Val TEXT
Description TEXT
Search TEXT
IsSensitive BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
product_category_attr
IdProductCategoryAttr INT(10)
IdProductCategory INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
TitleSlug VARCHAR(255)
Description TEXT
SeoDescription VARCHAR(200)
Seokeywords VARCHAR(200)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
entity
IdEntity INT(10)
OldId VARCHAR(255)
IdEntityType INT(10)
IdEntityGroup INT(10)
Name VARCHAR(255)
FiscalIdNumber VARCHAR(255)
Website VARCHAR(255)
Notes TEXT
Photo VARCHAR(255)
IsActive BIT(1)
IsCompany BIT(1)
Search TEXT
DateSync DATETIME
DateCreated DATETIME
entity_type
IdEntityType INT(10)
Name VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
app_area
IdAppArea INT(10)
SectionId INT(1)
Reference VARCHAR
Icon VARCHAR(25
Color VARCHAR(2
Route VARCHAR(
OrderId INT(10)
IsActive TINYINT
DateCreated DAT
DateUpdated DAT
language
IdLanguage INT(10)
Name VARCHAR(150)
Reference VARCHAR(2)
Code VARCHAR(10)
Direction VARCHAR(10)
IsActive BIT(1)
IsPublished BIT(1)
IsDefault BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_source
IdOrderSource INT(10)
IdBudgetStatus INT(10)
IdOrderStatus INT(10)
Title VARCHAR(255)
IsDefault BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_devolution_line
IdOrderDevolutionLine INT(10)
IdOrderDevolution INT(10)
IdOrderLine INT(10)
Quantity INT(5)


app_link
IdAppLink INT(10)
IdParent INT(10)
Title VARCHAR(255)
Url VARCHAR(255)
Icon VARCHAR(255)
OrderId INT(5)
ZoneId INT(5)
RoleId INT(5)
View VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_link
IdPageLink INT(10)
IdPage INT(10)
Photo VARCHAR(255)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
DateUpdated DATETIME
Indexes
page_link_attr
IdPageLinkAttr INT(10)
IdPageLink INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Url VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_photo
IdProductPhoto INT(10)
IdProduct INT(10)
IdProductVersion INT(10)
OrderId INT(5)
Filename VARCHAR(255)
Width INT(10)
Height INT(10)
Length INT(10)
IsHighlight BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_vs_custom_field_option
IdPageVsCustomFieldOption INT(10)
IdPageVsCustomField INT(10)
IdCustomFieldOption INT(10)
IdPage INT(10)
DateCreated DATETIME
Dateupdated DATETIME
Indexes
order_coupon_attr
IdOrderCouponAttr INT(10)
IdOrderCoupon INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Description TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_status_attr
IdOrderStatusAttr INT(10)
IdOrderStatus INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Description TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_option
IdProductOption INT(10)
IdProductOptionType INT(10)
GoogleMerchantTypeId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_tag_attr
IdPageTagAttr INT(10)
IdPageTag INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
Search VARCHAR(255)
product_category_related
IdProductCategoryRelated INT(10)
IdProductCategory INT(10)
IdRelated INT(10)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_vs_custom_field
IdPageVsCustomField INT(10)
IdPage INT(10)
IdCustomField INT(10)
IdLanguage INT(10)
Value TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
product_option_set_vs_product_option
IdProductOptionSetVsProductOption INT(10)
IdProductOptionSet INT(10)
IdProductOption INT(10)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes


product_type_attr
IdProductTypeAttr INT(10)
IdProductType INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
user
IdUser INT(10)
IdUserRole INT(2)
IdLanguage INT(10)
UserGroupId INT(10)
IdUserGroup INT(10)
IdAppCountry INT(10)
Name VARCHAR(255)
Email VARCHAR(255)
Password VARCHAR(255)
PasswordSalt VARCHAR(50)
Company VARCHAR(255)
PhoneDefault VARCHAR(255)
PhoneOptional VARCHAR(255)
FiscalIdNumber VARCHAR(255)
RelatedFiscalIdNumber VARCHAR(255)
Photo VARCHAR(255)
LastIP VARCHAR(255)
14 more...
DateCreated DATETIME
DateUpdated DATETIME
product_category
IdProductCategory INT(10)
OldId VARCHAR(255)
IdParent INT(10)
OrderId INT(5)
LevelId INT(5)
Photo VARCHAR(255)
PhotoOptional VARCHAR(255)
Filename VARCHAR(255)
IsActive BIT(1)
IsHighlight BIT(1)
Path VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
app_tax
IdAppTax INT(10)
Title VARCHAR(255)
Tax DECIMAL(5,2)
DateCreated DATETIME
DateUpdated DATETIME
page
IdPage INT(10)
IdPageType INT(10)
IdParent INT(10)
IdPopUp INT(10)
OrderId INT(10)
BackgroundFile VARCHAR(150)
OrderMenu INT(10)
OrderFooter INT(10)
IsMenuBackoffice BIT(1)
IsMenuLink BIT(1)
IsFooterLink BIT(1)
IsHighlight BIT(1)
ServerCacheMinutes INT(10)
IdUserRole INT(5)
pop_up
IdPopUp INT(10)
IsAlwaysOpen BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
product_vs_custom_field
IdProductVsCustomField INT(10)
IdProduct INT(10)
IdCustomField INT(10)
IdLanguage INT(10)
Value TEXT
DateCreated DATETIME
DateUpdated DATETIME
user_products_wishlist
IdUserProductsWishlist INT(10)
IdUser INT(11)
DateCreated DATETIME
DateUpdated DATETIME
product_tag
IdProductTag INT(10)
order_devolution_resolution
IdOrderDevolutionResolution INT(10)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
page_rating
IdPageRating INT(10)
IdPage INT(10)
IdUser INT(10)
Rating INT(5)
Comments TEXT
IsApproved BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
product_vs_product_tag
IdProductVsProductTag INT(10)
IdProduct INT(10)
IdProductTag INT(10)
DateCreated DATETIME
DateUpdated DATETIME
user_group
IdUserGroup INT(10)
Name VARCHAR(255)
IsManager BIT(1)
IncludeTaxes BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
entity_address
IdEntityAddress INT(10)
IdEntity INT(10)
Title VARCHAR(255)
AddressLine TEXT
AddressLocation VARCHAR(255)
AddressPostCode VARCHAR(255)
AddressDistrict VARCHAR(255)
Contacts VARCHAR(255)
IdAppCountry INT(10)
IsDefault BIT(1)
DateCreated DATETIME
DateUpdated DATETIME
form_group_attr
IdFormGroupAttr INT(10)
IdFormGroup INT(10)
IdLanguage INT(10)
Name VARCHAR(255)
Description TEXT
DateCreated DATETIME
DateUpdated DATETIME
page_vs_page_category
IdPageVsPageCategory INT(10)
IdPage INT(10)
IdPageCategory INT(10)
DateCreated DATETIME
DateUpdated DATETIME
product_attr
IdProductAttr INT(10)
IdProduct INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
TitleSlug VARCHAR(255)
Introduction TEXT
Description TEXT
Factsheet TEXT
page_type_attr
IdPageTypeAttr INT(10)
IdPageType INT(10)
IdLanguage INT(10)
Title VARCHAR(50)
DateCreated DATETIME
DateUpdated DATETIME


order_devolution_resolution_attr
IdOrderDevolutionResolutionAttr INT(10)
IdOrderDevolutionResolution INT(10)
IdLanguage INT(10)
Title VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
order_email_message_action
IdOrderEmailMessageAction INT(10)
IdOrderSource INT(10)
IdOrderStatus INT(10)
IsBudget BIT(1)
IdEmailMessageTemplate INT(10)
EmailReceivers TEXT
EmailsReplies TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
form_field_option_attr
IdFormFieldOptionAttr INT(10)
IdFormFieldOption INT(10)
IdLanguage INT(10)
Name VARCHAR(255)
Url VARCHAR(255)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_address
IdPageAddress INT(10)
IdPage INT(10)
Title VARCHAR(255)
AddressLine VARCHAR(255)
AddressCity VARCHAR(150)
AddressLocation VARCHAR(100)
AddressPostcode VARCHAR(50)
AddressDistrict VARCHAR(150)
AddressCountry VARCHAR(150)
GpsLatitude VARCHAR(50)
GpsLongitude VARCHAR(50)
Emails TEXT
Contacts TEXT
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_attachment
IdPageAttachment INT(10)
IdPage INT(10)
Filename VARCHAR(255)
Photo VARCHAR(255)
OrderId INT(5)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
entity_vs_custom_field_option
IdEntityVsCustomFieldOption INT(10)
IdEntityVsCustomField INT(10)
IdCustomFieldOption INT(10)
IdEntity INT(10)
DateCreated DATETIME
Dateupdated DATETIME
Indexes
Options TEXT
Result TEXT
DateStarted DATETIME
DateEnded DATETIME
DateCreated DATETIME
DateUpdated DATETIME
Indexes
page_section_attr_version
IdPageSectionAttrVersion INT(10)
IdPageSectionAttr INT(10)
User VARCHAR(255)
Body TEXT
DateCreated DATETIME
DateUpdated DATETIME
Indexes
user_products_wishlist_vs_product_version
IdUserProductsWishlistVsProductVersion INT(10)
IdUserProductsWishlist INT(10)
IdProductVersion INT(10)
DateCreated DATETIME
DateUpdated DATETIME
Indexes
DateCreated DATETIME
DateUpdated DATETIME
Indexes



Anexo B - Nova base de dados