



DESENVOLVIMENTO WEB – APLICAÇÃO DE GESTÃO DE EQUIPAMENTO MÉDICO

João Francisco Horta Santos

Mestrado em Engenharia Informática
Especialização em Engenharia de Software

Trabalho de Projeto orientado por:
Prof. Doutor João Pedro Guerreiro Neto

Agradecimentos

Em primeiro lugar, gostaria de agradecer ao meu orientador, professor doutor João Neto, por me ter esclarecido todas as dúvidas e apoiado durante a realização do projeto final de mestrado e pelos seus conselhos durante a escrita do relatório do mesmo.

Gostaria também de agradecer à empresa ARTSOFT por me ter recebido e prestado todo o apoio para que eu me integrasse o mais rapidamente possível e me sentisse confortável durante todo o projeto, e em particular ao meu supervisor João Catalão, e ao líder da equipa de desenvolvimento *web* Flávio Reis.

Um obrigado aos meus amigos por todos os bons momentos e gargalhadas durante todo o percurso académico.

Finalmente, um obrigado à minha família por sempre terem apoiado e acreditado em mim.

À minha família e amigos.

Resumo

Este projeto teve como principal objetivo desenvolver e implementar uma aplicação que irá permitir a digitalização e a automatização de processos internos, assim como a melhoria da comunicação interna entre os vários departamentos da empresa. Tal como todas as outras aplicações desenvolvidas na ARTSOFT, também esta é suportada pela plataforma ARTSOFT Online que oferece uma forma fácil de gerir, sincronizar e configurar as várias aplicações *web* que nela são instaladas. A arquitetura desta plataforma é baseada num conjunto de módulos que podem ser utilizados pelas várias aplicações nela instaladas, o que permite diminuir o tempo gasto em manutenção e um maior foco na melhoria e desenvolvimento de novas funcionalidades. Esta nova aplicação é composta por duas componentes fundamentais, a aplicação *web* desenvolvida com a *framework* AngularJS em JavaScript, e a aplicação *mobile* desenvolvida com versão 11 da *framework* Angular em TypeScript. Ambas as aplicações partilham a mesma API, desenvolvida em PHP.

Todos os desenvolvimentos realizados durante o projeto foram primeiro enviados para a equipa de controlo de qualidade para serem testados antes de serem disponibilizados aos clientes. Por questões de eficiência e organização, todos os dias ocorre uma reunião entre os vários membros da equipa responsável pelos desenvolvimentos das aplicações *web* e *móveis*, onde cada membro refere o que fez no dia anterior e o que irá fazer no presente dia.

Finalmente, os dados utilizados pelas aplicações do ARTSOFT Online podem, por um lado, estar guardados numa base de dados MySQL local ou, por outro, estar guardados na base de dados utilizada pelo ARTSOFT ERP que pode ser acedida através dos serviços ARTSOFT.

Palavras-chave: Aplicação *web*, Módulos, Arquitetura, Base de Dados

Abstract

The aim of this project was the development of a web application that allows the digitization and automation of the client's internal processes, as well as the improvement of internal communication between the multiple organization's departments. Just like every other application developed by ARTSOFT, this one is also supported by the ARTSOFT Online platform which provides an easy way to manage and set up the different web applications. This platform's architecture is based in a set of modules that can be used by any application that is installed in this platform. This allows the software developers to focus on improving and developing new features and spend less time fixing bugs. This new application, has two core components: a web application developed in JavaScript with the AngularJS framework, and a mobile application developed in TypeScript with the Angular framework version 11. Both of these applications share the same API, developed in PHP.

Every feature developed during the project was first sent to the quality control team so that it was tested, before it got sent to the final client. To keep the developers organized and the work efficient, every day there was a meeting where each member explains what he has done the day before and what he will be doing during the current day.

Finally, data used by the ARTSOFT Online's applications can be stored in two different tools. On the one hand, data can be stored in a local MySQL database, and, on the other hand, it can also be stored in a ARTSOFT ERP database which can be accessed using the ARTSOFT's services.

Keywords: Web Application, Modules, Architecture, Database

Conteúdo

Lista de Figuras	xii
Lista de Tabelas	xiii
1 Introdução	1
1.1 Motivação	1
1.2 Objectivos	2
1.3 Metodologia	2
1.4 Estrutura do documento	3
2 Trabalho Relacionado	5
2.1 MySql	5
2.2 PHP	5
2.3 AngularJS	6
2.3.1 HTML	6
2.3.2 JavaScript	6
2.4 Tortoise SVN	7
2.5 Jenkins	7
2.6 SonarQube	7
2.7 Trello	7
2.8 MockFlow	8
2.9 Software ARTSOFT	8
2.9.1 ARTSOFT ERP	8
2.9.2 Serviços ARTSOFT	9
2.9.3 ARTSOFT Online	11
3 Integração na Empresa	15
3.1 Traduções no eCommerce	15
3.2 Módulos de Pagamento	17
3.2.1 MBWay	17
3.2.2 Cartões de Crédito	19
3.3 Módulo de Troca de Horários	20

3.4	Módulos de Gestão de Projetos	21
4	Aplicação de Gestão de Equipamento Médico	27
4.1	O Problema	27
4.2	Infraestrutura de Apoio ao Desenvolvimento	28
4.2.1	Mocks	28
4.2.2	Gestão	30
4.3	Arquitetura	31
4.4	Implementação	35
4.4.1	Consulta de Clientes e Referências	35
4.4.2	Criação e Edição de Pedidos	37
5	Conclusão	45
5.1	Análise	45
5.2	Trabalho Futuro	46
	Abreviaturas	47
	Bibliografia	49

Lista de Figuras

2.1	Menus disponíveis no ARTSOFT ERP	8
2.2	Evento no ARTSOFT ERP	9
2.3	Interface do XMLClient	10
2.4	ARTSOFT Online	12
2.5	Módulos específicos do eCommerce	12
2.6	Módulos disponíveis para todas as aplicações	13
2.7	Lista de aplicações disponíveis	13
2.8	Instalação do eCommerce	14
3.1	Página de Configuração de Traduções	15
3.2	Configuração do MBWay	17
3.3	Pagamento com MBWay	18
3.4	Configuração do módulo de cartões de crédito	19
3.5	Pagamento com cartão de crédito	20
3.6	Troca de horário entre dois utilizadores	21
3.7	Troca simples de horário	21
3.8	Página das Timesheets	22
3.9	Página de Ajudas de Custo	22
3.10	Opção - Com Histórico	23
3.11	Definições do Módulo de Despesas	23
3.12	Definições do Módulo de Despesas	24
3.13	Definições do Módulo das Timesheets	25
4.1	Mock da página de criação de pedido	29
4.2	Página de criação de pedido funcional	29
4.3	Estrutura do Trello	31
4.4	Arquitetura da aplicação	32
4.5	Arquitetura <i>frontend</i> representada em diagrama UML	33
4.6	Esquema de diretivas representada em diagrama UML	34
4.7	Página de listagem de referências	36
4.8	Modal do filtro de franquias	37
4.9	Página de criação de pedidos	39

4.10	Página de criação de pedidos - Referências selecionadas	41
4.11	Página de criação de pedidos - Processos	42
4.12	Página de seleção de favoritos	43

Lista de Tabelas

4.1 Datas de entrega	28
--------------------------------	----

Capítulo 1

Introdução

A ARTSOFT, que está inserida em vários mercados internacionais como Cabo Verde, África do Sul, Espanha, Brasil, Polónia, Portugal e Angola, é uma empresa especializada no desenvolvimento e comercialização de soluções tecnológicas de apoio à gestão empresarial, que acompanham as necessidades do mercado empresarial. Existem soluções adaptadas ao contexto não só de grandes empresas, mas também de micro, pequenas e médias empresas.

Entre as várias soluções desenvolvidas, destacam-se três tipos de aplicações: *desktop*, *web* e móveis.

1.1 Motivação

Num mundo cada vez mais competitivo e digitalizado, é necessário encontrar novas formas de aumentar a eficiência e competitividade em todos os níveis de um negócio, e com o aumento exponencial da capacidade de computação dos sistemas informáticos na atualidade, a melhor forma de atingir estes objetivos é utilizando soluções tecnológicas que automatizem processos que de outra forma seriam muito menos eficientes.

Consciente destes factos, a ARTSOFT apostou no desenvolvimento de soluções tecnológicas de apoio à gestão empresarial, o que resultou em sistemas como o ARTSOFT ERP que oferece um grande leque de funcionalidades com estes objetivos em mente. Desde que a *World Wide Web* foi criada no século XX, o número de utilizadores não tem parado de aumentar, atingindo em outubro de 2020, um total de 4.66 mil milhões de utilizadores, ou seja, 59% da população mundial [5]. Indo ao encontro destas tendências, a ARTSOFT desenvolveu, e continua a desenvolver, um conjunto de aplicações *web* que visam disponibilizar as funcionalidades oferecidas pelo ARTSOFT ERP de uma forma mais global. Apesar do ARTSOFT ERP oferecer este vasto leque de funcionalidades, muitas vezes as empresas que pedem uma solução *web* para um certo problema de gestão empresarial não pretendem utilizar todas elas. Para tal pretendeu-se que se desenvolvesse uma solução focada no problema a resolver, que apenas faça uso de um conjunto restrito

de funcionalidades do ARTSOFT ERP, mas que pudesse ser disponibilizada a utilizadores espalhados por uma grande área geográfica através da *World Wide Web*. Entre as soluções disponibilizadas, será incluída a aplicação de gestão de equipamento médico desenvolvida para este projeto.

1.2 Objectivos

Este projeto teve como objetivo a minha integração numa equipa de desenvolvimento para a implementação de uma aplicação *web* de gestão de equipamento médico que automatiza processos internos da empresa que pediu o desenvolvimento da mesma. Para tal, numa primeira fase foi pedido que se realizassem pequenas implementações e que participasse no processo de manutenção em aplicações já implementadas na ARTSOFT, para que me familiarizasse com as tecnologias e processos utilizados pela empresa. Após este período de integração e após a adjudicação do projeto, pretendeu-se que se desenvolvesse a aplicação referida no início da secção.

1.3 Metodologia

Num ambiente tão diverso como é o caso das aplicações *web* desenvolvidas na ARTSOFT, onde é necessário desenvolver, de forma contínua, novas funcionalidades e melhorias para cada nova versão, é preciso ter uma metodologia de desenvolvimento bem definida. Para isso é utilizada a *framework* de desenvolvimento ágil Scrum e Kanban, à qual foram feitas algumas alterações que vão ao encontro dos objetivos da empresa.

O processo inicia-se com o *Product Owner* a decidir o que vai ser desenvolvido no *sprint* que se vai realizar. Esta fase inicia-se com a definição de um conjunto de ideias que, no futuro, irão dar origem às funcionalidades a desenvolver no *sprint*. Estas ideias podem ter várias origens, sendo algumas delas a análise da concorrência e sugestões internas dos *stakeholders* que participam no planeamento do *sprint*. Após todo este processo, será o *Product Owner* a decidir que ideias podem avançar e quais necessitam de mais análise. As ideias aprovadas são colocadas num *Product backlog*, chamado *roadmap*, que é um documento excel onde são definidos vários parâmetros para cada uma delas, por exemplo, a prioridade que vai ser dada a um certo desenvolvimento.

A partir do *Product backlog* são criados um ou mais projetos, dependendo da complexidade do desenvolvimento. Cada projeto tem um conjunto de tarefas, chamadas novidades. Estas novidades podem ser novidades que precisam de um projeto à parte pelo facto de serem desenvolvimentos com grande impacto para o cliente, novidades que necessitam de uma análise mais técnica e novidades que não necessitam de uma análise técnica tão exaustiva, ou seja, são pequenos desenvolvimentos de melhorias, como por exemplo melhorias de usabilidade. Estes projetos são criados no ARTSOFT ERP, com ajuda do

módulo Gestão de Projetos. Este permite simular a *framework* Kanban, utilizando os vários estados da tarefa, como por exemplo: Desenvolvimentos atribuídos, Desenvolvimentos em *Sprint*, Desenvolvimentos Terminados, entre outros. De modo a ser mais fácil gerir as várias "novidades" dentro de um projeto, é possível também atribuir um tipo que indica qual o departamento que irá ser responsável por uma certa "novidade" (por exemplo: departamento de qualidade, departamento de financeiro, entre outros).

A cada dia, a equipa de desenvolvimento faz uma reunião diária, intitulada por Daily Scrum. Ela tem como objetivo disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho a ser realizado no dia que se inicia. Durante o Daily Scrum, cada membro da equipa provê respostas para cada uma destas três perguntas: O que foi feito ontem? O que vai ser feito hoje? Há algum impedimento no processo?

Apesar de, na ARTSOFT, não haver um indivíduo que tem como responsabilidade ser *Scrum Master*, semanalmente ocorrem duas atividades em que é fulcral o envolvimento de um responsável com estas funções. Todas as sextas-feiras, durante o *Sprint Planning*, o *Product Owner* irá planear as tarefas a atribuir e a apresentar na atividade que se segue, o *Sprint Backlog* e *Sprint Review Meeting*. Esta é uma atividade que, normalmente, ocorre à segunda-feira onde o *Product Owner* tem reuniões individuais, por questão de eficiência, com cada um dos *stakeholders* envolvidos, onde é apresentado o que foi planeado durante o *Sprint Planning*. Todo este processo está em constante evolução e pode ser adaptado caso os objetivos da empresa mudem.

1.4 Estrutura do documento

Este documento está organizado da seguinte forma:

- Capítulo 2: Este capítulo descreve as tecnologias utilizadas durante o desenvolvimento do projeto, assim como a descrição dos sistemas ARTSOFT que são fulcrais para o funcionamento das implementações descritas no capítulo 3.
- Capítulo 3: Neste capítulo são descritas as implementações realizadas durante a fase de integração na empresa e como estas alterações afetaram o comportamento das aplicações em que foram inseridos.
- Capítulo 4: Neste capítulo são descritos o problema, a arquitetura e as funcionalidades desenvolvidas durante o desenvolvimento da aplicação de gestão de equipamento médico.

Capítulo 2

Trabalho Relacionado

A ARTSOFT utiliza um conjunto base de tecnologias para o desenvolvimento das suas aplicações *web*. Nas secções seguintes são apresentadas estas tecnologias e, de um modo geral, como elas são integradas nestas aplicações.

2.1 MySql

O MySQL é um *software* que foi originalmente criado por uma empresa sueca chamada MySQL AB em 1994. Esta empresa foi comprada pela Sun Microsystems em 2008 que, por sua vez, foi comprada pela Oracle em 2010, sendo esta última quem detém os direitos do *software* desde então.

O MySQL é um sistema de gestão de bases de dados relacional que utiliza SQL ou *Structured Query Language* como linguagem de desenvolvimento. No contexto das aplicações *web* desenvolvidas pela ARTSOFT, esta base de dados não só é utilizada para guardar dados relativos às várias aplicações, mas também para automatizar a construção de algumas páginas genéricas que contêm opções de configuração.

2.2 PHP

A linguagem PHP, cujo acrónimo corresponde a "PHP: Hypertext Preprocessor", foi criada por Rasmus Lerdorf em 1995 e é, atualmente, mantida por uma organização chamada "The PHP Group". No contexto dos projetos *web* desenvolvidos pela empresa, esta tecnologia é principalmente utilizada para o desenvolvimento de APIs ou *application programming interfaces* que permitem a comunicação entre o *frontend*, desenvolvido com a *framework* AngularJS, e os serviços ARTSOFT. Para facilitar a implementação destas APIs é utilizada a micro *framework* Slim. Esta *framework* permite a definição de *endpoints*, e fornece objetos que facilitam a receção de pedidos HTTP, assim como a resposta aos mesmos. O funcionamento das várias aplicações *web* que a ARTSOFT desenvolve, assumem a utilização da versão 7 ou superior.

2.3 AngularJS

O AngularJS é uma *framework web open-source* mantida pela Google direcionada ao desenvolvimento *frontend* com as linguagens JavaScript e HTML. A utilização desta *framework* no contexto dos projetos *web* da empresa deve-se ao facto de fornecer uma forma fácil e eficaz de apresentar informação ao utilizador e manipulá-la de forma dinâmica através do uso de diretivas. Algumas das diretivas mais importantes desta *framework* são:

- **ng-app**: Diretiva utilizada para definir qual o módulo que vai determinar o comportamento de uma certa página HTML ou *view*. Um módulo é, resumidamente, uma coleção de serviços, diretivas, controladores, filtros e dados de configuração.
- **ng-controller**: Esta diretiva permite criar um controlador que fica associado a uma certa página HTML ou *view*. Dentro de um controlador é possível manipular, de forma dinâmica, os dados inseridos pelo utilizador através de um objeto chamado *scope*. Com este objeto é possível criar variáveis para as quais será possível estabelecer uma ligação direta bidirecional com a *view* a que o controlador está associado. Esta ligação é estabelecida com a diretiva **ng-model**.
- **ng-model**: Esta diretiva é invocada na forma de atributo de um elemento HTML e permite a criação de uma ligação bidirecional entre os dados contidos nesse elemento e a variável correspondente no objeto *scope* definido no controlador da *view*.

2.3.1 HTML

O HTML é uma linguagem de marcação que é utilizada para o desenvolvimento de páginas na *web*. De modo a facilitar o desenvolvimento das várias páginas *web* utilizadas nas aplicações, é utilizado uma das *frameworks* mais populares para CSS ou *Cascading Style Sheets* e HTML, o Bootstrap. Esta *framework* fornece propriedades CSS e elementos HTML já implementadas que ajudam na construção de páginas responsivas.

2.3.2 JavaScript

O JavaScript é uma linguagem de programação criada por Brendan Eich e é a linguagem que permite gerar quase todo o comportamento dinâmico e interativo na maioria das aplicações *web*. O AngularJS tira partido deste facto e permite a implementação de diretivas em JavaScript que podem ser invocadas diretamente no HTML como elementos ou como atributos de elementos nativos.

2.4 Tortoise SVN

O Tortoise SVN é um *software* de controlo de versões para o Windows e que é baseado no cliente do Apache Subversion (SVN). Este *software* é *open source* e está licenciado sob o GPL, o que permite a sua utilização de forma gratuita e o desenvolvimento de versões adaptadas ao ambiente em que vai ser utilizado. É este *software* que é utilizado para controlo de versões na ARTSOFT, não só para as aplicações *web*, mas também para as aplicações móveis, *desktop* e serviços ARTSOFT.

2.5 Jenkins

Utilizado em conjunto com o TortoiseSVN, o Jenkins é um servidor de automatização grátis e *open source*, que automatiza tarefas relacionadas com teste, construção e distribuição de *software*. Na ARTSOFT, este *software* tem acesso aos repositórios do Tortoise SVN e permite, por exemplo, compilar o *software* contido nestes em intervalos regulares de tempo.

2.6 SonarQube

O SonarQube é um produto *open-source* desenvolvido pela SonarSource que permite acelerar o processo de *Code Review* durante o desenvolvimento de projetos, ao analisar a qualidade do código de forma automática através de regras pré-definidas, apesar de ser possível inserir novas regras de forma manual. Ao analisar um projeto, o SonarQube oferece reportes de *bugs*, código duplicado, vulnerabilidades de segurança, entre outras, mas também permite guardar o histórico destas métricas assim como gráficos de evolução para que seja possível fazer uma análise global dos problemas que ocorreram no fim de um projeto.

2.7 Trello

O Trello, que atualmente pertence à empresa com o mesmo nome, foi criado pela Fog Creek Software em 2010, e lançado em setembro do ano seguinte. Esta é uma ferramenta *online* de gestão de projetos que permite a equipas de desenvolvimento de *software* e não só, planear de uma forma visual e fácil as tarefas que são necessárias desenvolver. Para isto, a ferramenta utiliza um sistema de "cartas" dispostas num tabuleiro de listas, onde cada lista contém um conjunto de cartas que representam as tarefas a serem realizadas. Numa tarefa, ou carta, é possível ainda definir um conjunto de parâmetros como a descrição, uma ou mais etiquetas que podem, por exemplo, definir a categoria da tarefa, uma *checklist* onde se definem subtarefas a serem executadas na tarefa pai, entre outros.

2.8 MockFlow

O Mockflow é uma ferramenta que permite planejar esboços de interfaces de alta fidelidade, ou seja, esboços que se vão assemelhar à interface funcional final. Esta ferramenta permite tornar o processo de planeamento de novas interfaces mais eficiente, já que o tempo despendido a planejar estas interfaces agiliza bastante o desenvolvimento das interfaces funcionais das aplicações.

2.9 Software ARTSOFT

Nas secções anteriores foram apresentadas todas as tecnologias externas utilizadas pela ARTSOFT no desenvolvimento das aplicações, mas estas aplicações também utilizam *software* desenvolvido na empresa que permitem a integração das mesmas com as restantes tecnologias *desktop* utilizadas. Nas restantes subsecções são apresentadas estas tecnologias e como estas permitem a integração das aplicações *web* com o restante *software*.

2.9.1 ARTSOFT ERP

O ARTSOFT ERP é um *software* de apoio à gestão empresarial desenvolvido em C++ e que oferece várias ferramentas que automatizam, por exemplo, processos relacionados com gestão de recursos humanos ou de gestão de projetos, entre outros. Na figura 2.1 é possível ver mais algumas opções que este *software* oferece.

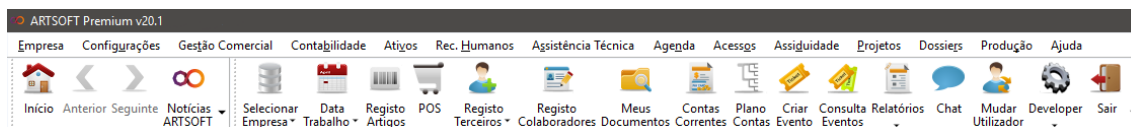


Figura 2.1: Menus disponíveis no ARTSOFT ERP

Existem duas versões deste *software*, uma versão que é disponibilizada aos programadores com as funcionalidades mais recentes, onde é possível fazer configurações de dados que são usadas noutras aplicações, mas que ainda não foram testadas, e uma versão que está em "produção" e que é utilizada tanto pelos clientes, como pela própria empresa para automatizar tarefas como facilitar a distribuição de tarefas pelos vários funcionários através de eventos. Como se pode ver na figura 2.2, a funcionalidade de gestão de eventos deste *software* permite incluir reportes de erros e pedidos de implementação de funcionalidades em eventos que são transmitidos entre as várias equipas, aumentando assim a eficiência de trabalho a nível global.

Além do referido anteriormente, este *software* utiliza um sistema de gestão de base de dados (SGBD) chamado Actian Zen para gerir as bases de dados utilizadas não só pela

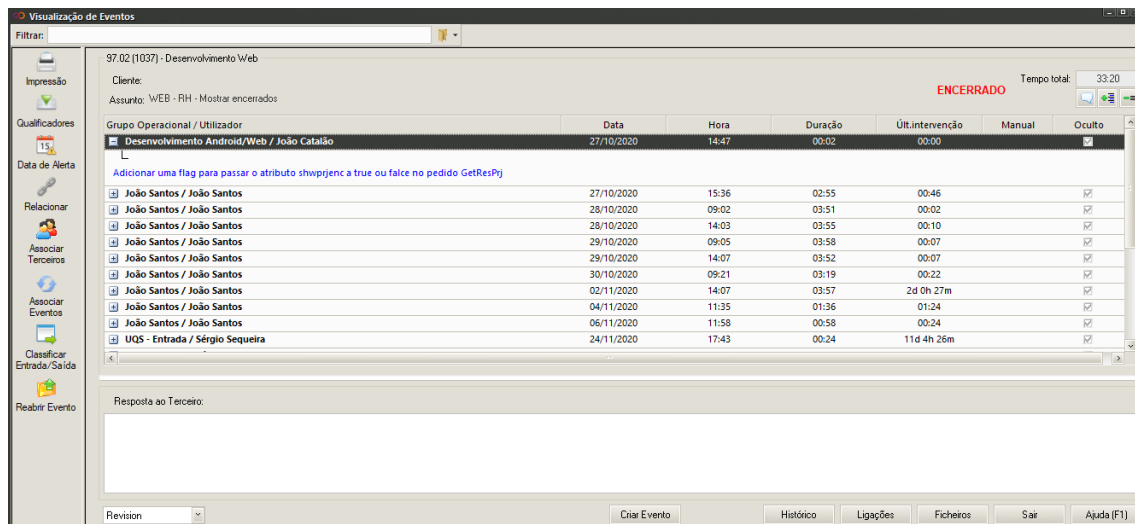


Figura 2.2: Evento no ARTSOFT ERP

própria empresa mas também pelos clientes. A utilização de um sistema deste tipo não só permite abstrair toda a lógica de gestão de base de dados, mas também ter acesso a medidas de segurança já pré-implementadas.

2.9.2 Serviços ARTSOFT

Como o ARTSOFT ERP é a base de qualquer aplicação desenvolvida na ARTSOFT, foi necessário criar uma forma de comunicação entre ambas. Assim, foram criados um conjunto de serviços que comunicam com o ARTSOFT ERP e que podem ser invocados por qualquer aplicação, seja ela *web* ou móvel.

Uma particularidade destes serviços é que tanto pedidos como respostas contém *strings* em formato XML. De modo a simplificar o processo de desenvolvimento de *software*, existe uma ferramenta chamada XMLClient que permite visualizar que serviços, tabelas e funções estão disponíveis, assim como testar estes serviços.

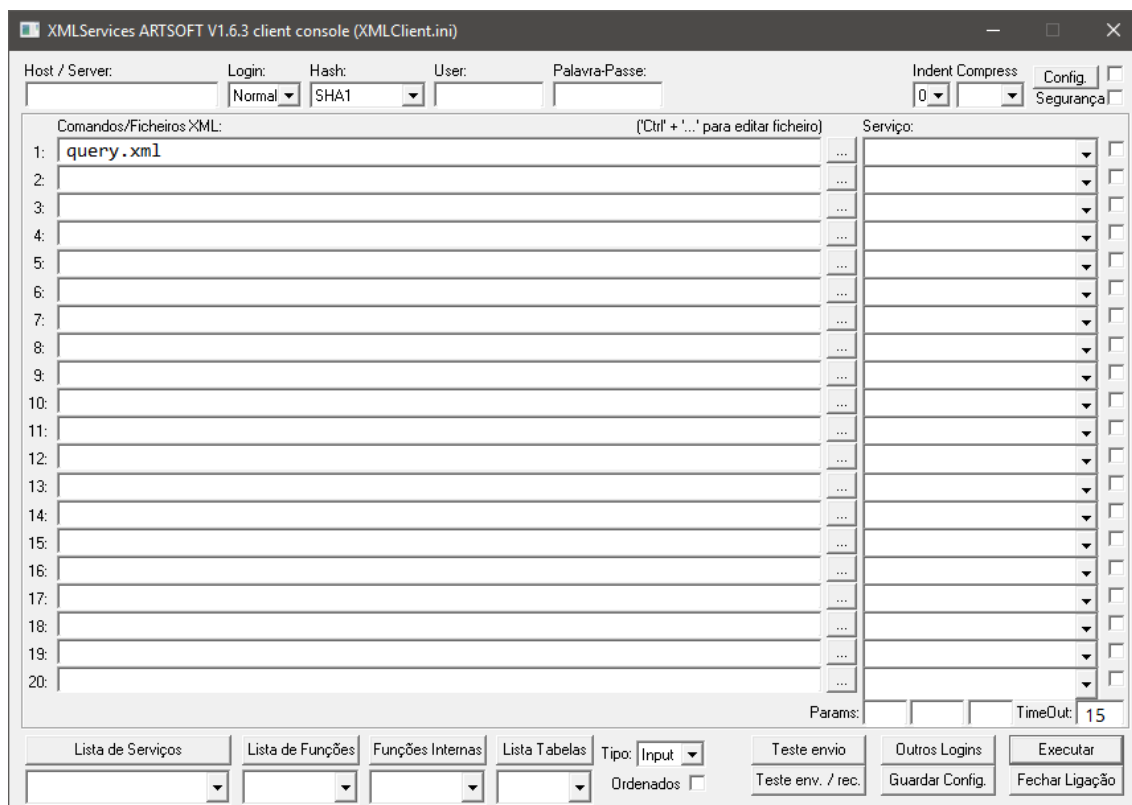


Figura 2.3: Interface do XMLClient

Para testar um serviço neste *software*, apenas se tem de definir, no campo 'Comandos/Ficheiros XML', um caminho válido para um ficheiro de extensão XML com uma *query* válida, e seleccionar no campo 'Serviço', o serviço para onde se pretende fazer o pedido e executar. A maioria dos serviços abstrai a necessidade de saber a que tabela se está a aceder e retorna apenas um conjunto de parâmetros restrito, que é definido internamente no serviço. Abaixo é possível ver um exemplo genérico de um pedido e resposta a um serviço deste tipo:

- **Pedido:**

```
<?xml version='1.0' encoding='utf-8'?>
<root Parametro1='Valor1' Parametro2='Valor2' .../>
```

- **Resposta:**

```
<root rc='codigo'>
  <rec Atributo1='Valor1' Atributo2='Valor2' .../>
  <rec Atributo3='Valor3' Atributo4='Valor4' .../>
</root>
```

Mas caso seja mesmo necessário aceder a uma tabela específica, o formato do pedido e resposta passa a ser bastante diferente. Num pedido deste tipo, no campo 'Tabela' define-se o nome da tabela a que se quer aceder, e no campo 'Params' define-se quais os registos

que se pretende receber, ou seja, este campo é o equivalente à cláusula 'WHERE' numa query SQL. Caso se pretenda atribuir um nome específico ao elemento que irá agregar os valores de um registo, esse nome é definido no atributo 'name', caso contrário, não é necessário incluir esse atributo na *string* e o nome será, por omissão, 'row'. Abaixo está um exemplo de um pedido e resposta genéricos, ao invocar este serviço:

- **Pedido:**

```
<?xml version='1.0' encoding='utf-8'?>
<Registos type='list' name='nome' query='Tabela | Params'>
  <defcol>
    <Coluna1 form='%Tabela.Atributo1'>
    <Coluna2 form='%Tabela.Atributo2'>
    <Coluna3 form='%Tabela.Atributo3'>
  </defcol>
</Registos>
```

- **Resposta:**

```
<Registos>
  <nome>
    <Coluna1>Valor1</Coluna1>
    <Coluna2>Valor2</Coluna2>
    <Coluna3>Valor3</Coluna3>
  </nome>
</Registos>
```

Ao contrário dos pedidos anteriores, que são utilizados para pedir dados, o formato dos pedidos para inserir, atualizar ou apagar dados pode variar bastante, ou seja, não existe um modelo genérico para pedidos deste tipo. Mas as respostas a estes pedidos podem ser definidas pelo seguinte modelo, onde o 'id' corresponde ao identificador do elemento alterado:

```
<root rc='codigo' id='id'></root>
```

2.9.3 ARTSOFT Online

O ARTSOFT Online (figura 2.4) é uma plataforma onde é possível instalar, configurar e gerir várias aplicações *web* e *plugins*. Entre as aplicações oferecidas, destacam-se o eCommerce e o Portal do Colaborador já que foi este o foco no período de integração. Cada aplicação oferece um conjunto de funcionalidades de base que são iguais para todos os clientes. Caso o comportamento de uma certa funcionalidade não se adeque ao contexto em que vai ser utilizada, é possível que os clientes peçam que se desenvolvam *plugins* que, resumidamente, consistem na adaptação do comportamento destas funcionalidades ao contexto em que vai ser utilizado.

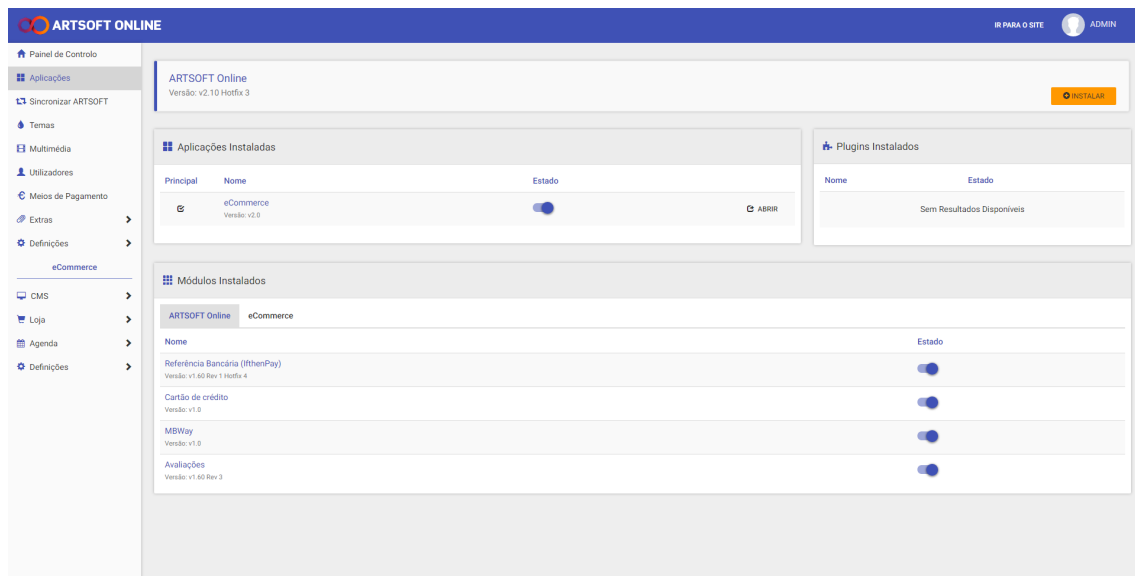


Figura 2.4: ARTSOFT Online

Arquitetura

A aposta inicial da ARTSOFT passou por desenvolver um conjunto de aplicações *web* individualmente, mas à medida que se foi adicionando funcionalidades a estas aplicações o processo de manutenção foi-se tornando cada vez mais complexo. Funcionalidades como o *login*, apesar de terem o mesmo comportamento em todas as aplicações, tinham de ser alteradas em cada uma delas sempre que se alterava o seu comportamento. Tendo isto em conta, o desenvolvimento destas funcionalidades passou a ser feito através de módulos que podem ser utilizados pelas várias aplicações instaladas no ARTSOFT Online, o que fez com que o tempo gasto em manutenção fosse reduzido significativamente.

Um módulo é, resumidamente, um conjunto de classes que oferecem uma funcionalidade específica e que pode estar ou não ativo. Um módulo pode estar disponível apenas para uma aplicação específica (figura 2.5), como é o caso dos utilizadores no "eCommerce", ou pode estar disponível para todas as aplicações (figura 2.6), como é o caso dos meios de pagamento.



Figura 2.5: Módulos específicos do eCommerce

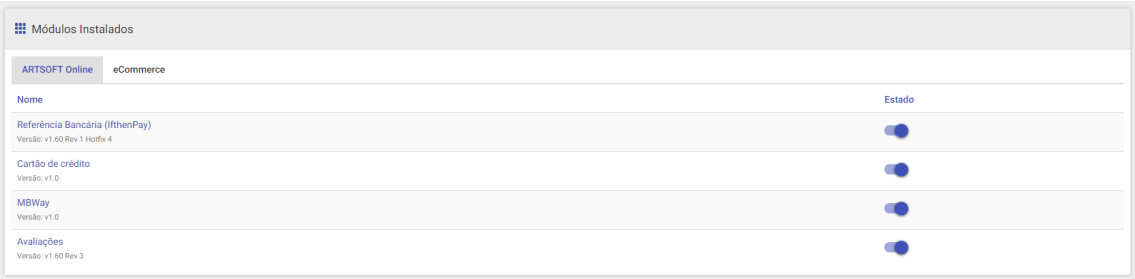


Figura 2.6: Módulos disponíveis para todas as aplicações

Tal como se pode ver na figura 2.7, a instalação de módulos ou aplicações é feita através do ARTSOFT Online, que obtém a informação do que está disponível para instalação através da conexão com um repositório.

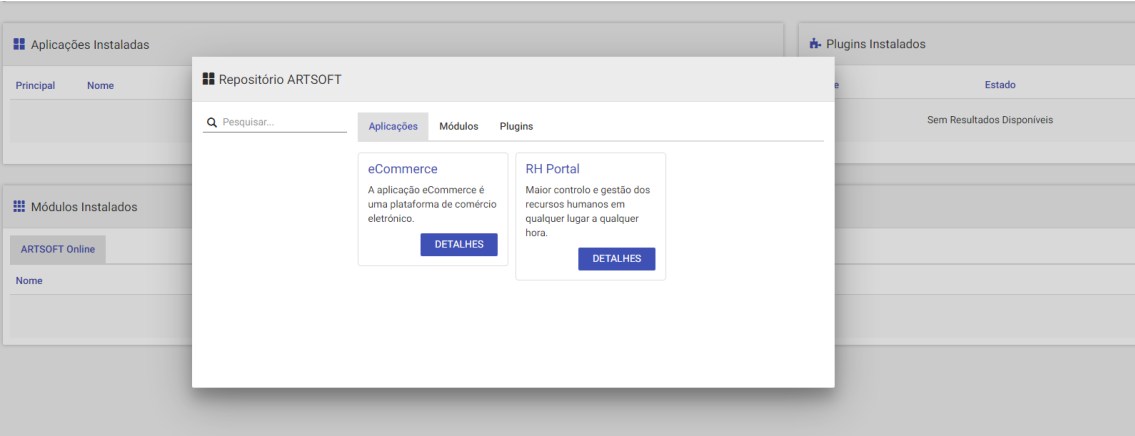


Figura 2.7: Lista de aplicações disponíveis

Ao proceder-se com a instalação de um módulo/aplicação, o ARTSOFT Online acede a este repositório e descarrega um zip da versão seleccionada que contém todos os ficheiros da aplicação, um arquivo que descreve todas as dependências da aplicação/módulo em questão e um *script* de instalação que é executado automaticamente.

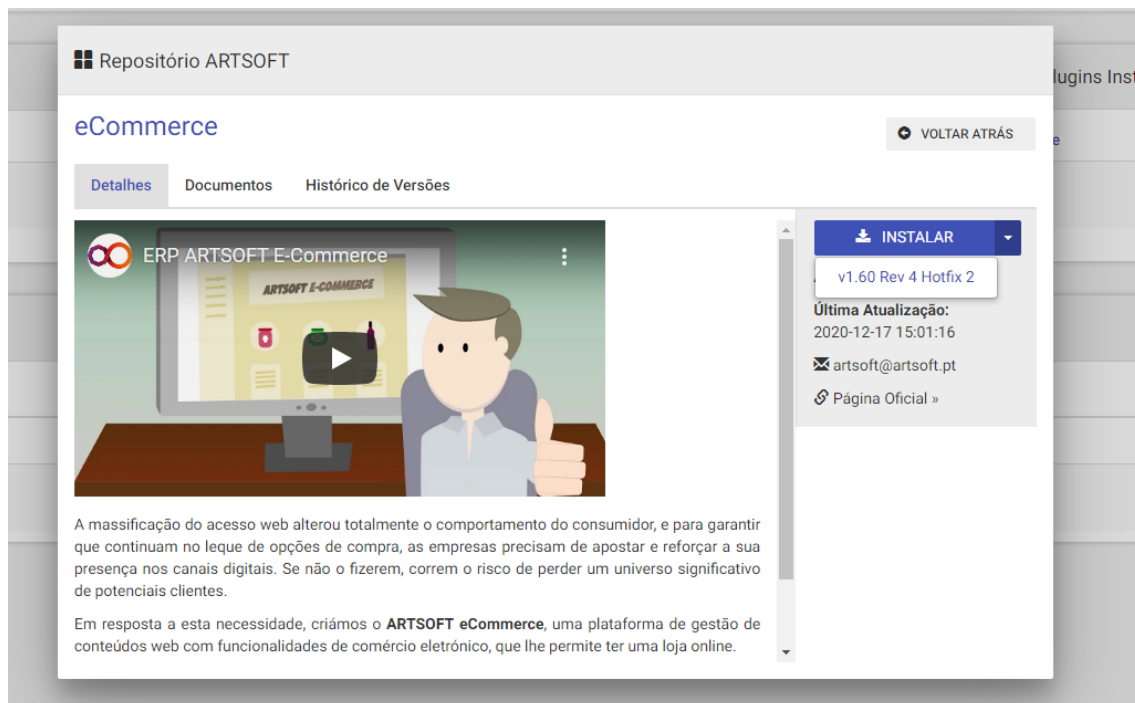


Figura 2.8: Instalação do eCommerce

Como foi referido anteriormente, para as várias aplicações *web* comunicarem com o ARTSOFT ERP é necessário a utilização dos serviços ARTSOFT, mas primeiro é preciso estabelecer uma ligação segura. Para isso, é enviado um pedido de *login*, que contém o nome de utilizador e a palavra-passe que são configurados durante a instalação do ARTSOFT Online, para o servidor onde os serviços estão em execução. Na resposta é enviado um "challenge" (uma *string* gerada aleatoriamente) que é utilizado para gerar um "digest" que é por sua vez enviado para o servidor. Este "digest" é gerado através do algoritmo SHA-1 que utiliza o nome de utilizador e palavra-passe, referidos anteriormente, e o "challenge" como parâmetros. Finalmente, utilizando o nome de utilizador e palavra-passe da primeira comunicação, o servidor consegue ou não validar o "digest" recebido ao executar o processo inverso ao utilizado para o criar.

Capítulo 3

Integração na Empresa

Durante o período de integração na empresa, foram atribuídos um conjunto de desenvolvimentos de modo a permitir a familiarização de, por um lado, a arquitetura das aplicações já implementadas pela ARTOSFT e, por outro, com as tecnologias utilizadas para o seu desenvolvimento. Neste capítulo são apresentados todos os desenvolvimentos feitos durante este período.

3.1 Traduções no eCommerce

A primeira implementação, tal como mostrado na figura 3.1, consistiu na criação de um menu de configuração no ARTSOFT Online, onde o utilizador pode inserir traduções para os produtos disponíveis na loja, nos vários idiomas possíveis.

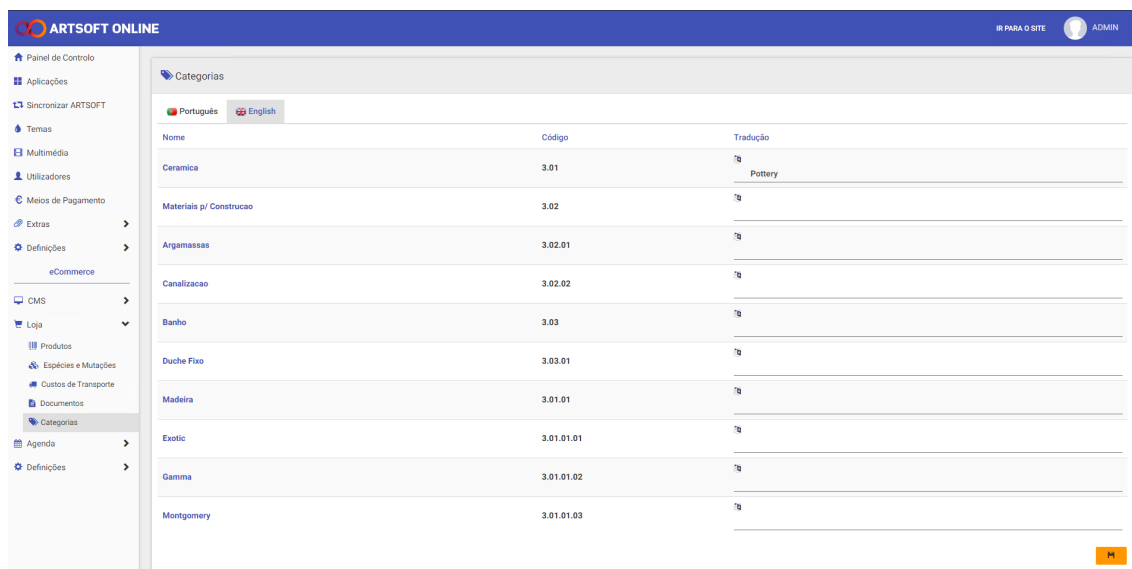


Figura 3.1: Página de Configuração de Traduções

As traduções destas categorias serão depois mostradas no eCommerce caso o idioma correspondente esteja selecionado. Caso não seja inserida nenhuma tradução, será mostrado

o nome original da categoria.

Inicialmente, a solução proposta foi aceite e colocada em produção, ou seja, foi disponibilizada aos clientes, mas após algumas queixas de alguns destes clientes descobriu-se que, após cada nova sincronização, todas as traduções inseridas anteriormente desapareciam. No caso específico do eCommerce, o sincronizador é executado no ARTSOFT Online e transfere dados como produtos, categorias de produtos, preços de produtos, entre outros, da base de dados que está a ser utilizada pelo ARTSOFT ERP e os serviços para a base de dados local da aplicação *web*. Durante a sincronização, de modo geral, todos os dados referidos anteriormente são eliminados da base de dados local e voltam a ser transferidos, o que, quando são utilizadas bases de dados com dezenas de milhares de produtos, tornam a sincronização muito lenta. Para dar resposta aos reportes apenas se alterou o bloco do sincronizador que transfere as categorias, mas em trabalho futuro este é um ponto que terá de ser analisado e possivelmente reestruturado.

O sistema de traduções tem como elemento principal um tabela SQL que guarda qualquer tradução de qualquer elemento que tenha possibilidade de ser traduzido na aplicação, seja ele um produto ou uma categoria, assim como o código da linguagem da tradução e o tipo do elemento traduzido. O tipo de elemento é distinguido através de um conjunto de valores fixos definidos num enumerado, já que um produto e uma categoria podem ter o mesmo id internamente. Para sincronizar as categorias, é feito um pedido aos serviços ARTSOFT que retorna uma lista das categorias existentes com os seus ids internos, assim como a designação e os códigos que identificam a sua família, isto porque categorias podem ter subcategorias. É nesta fase que o novo e o antigo algoritmos diferem. Em vez de se apagarem todos os registos, para cada categoria adiciona-se o id interno da categoria num *array*, e verifica-se se esta já se encontra inserida na base de dados da aplicação *web*. Caso se verifique que já está presente na base de dados, apenas se faz uma atualização ao registo, caso contrário, insere-se um novo registo. Após se percorrerem todas as categorias, apenas se executam mais duas *queries* SQL que, com operador "NOT IN" e o *array* referido anteriormente, permitem remover todos os registos que já não existem nesta nova sincronização

Apesar desta solução ter satisfeito os clientes, ela tem o inconveniente de manter traduções incorretas para categorias que têm o mesmo id de uma categoria diferente que estava presente anteriormente. Isto ocorre pois é possível duas bases de dados ARTSOFT conterem duas listas completamente diferentes de categorias com ids exatamente iguais, o que, caso se queira mudar de base de dados nesta situação, irá manter as traduções incorretas. Isto não foi considerado um erro, já que os clientes que utilizam o eCommerce nunca alteram o tipo de loja ao ponto de reproduzir esta situação.

3.2 Módulos de Pagamento

O ARTSOFT Online oferece vários meios de pagamento, nomeadamente o PayPal, pagamento por referência bancária, transferência bancária, pagamento à cobrança e pagamento a crédito, o que dá uma grande flexibilidade aos utilizadores no momento de efetuar um pagamento. No entanto, como a ARTSOFT não implementa estas soluções no seu *software*, é necessário utilizar serviços fornecidos por terceiros. Como o formato e os meios de comunicação que estes serviços usam pode variar bastante, cada meio de pagamento é um módulo individual que implementa uma interface comum.

Assim, para aumentar a biblioteca de meios de pagamento oferecidos, foi pedido que se adicionasse dois novos módulos: MBWay e cartões de crédito.

3.2.1 MBWay

O MBWay, um dos meios de pagamento mais utilizados em Portugal na atualidade, é uma solução MULTIBANCO que permite fazer compras online, e não só, através de *smartphones*. A API utilizada para integrar o MBWay com as aplicações do ARTSOFT Online é fornecida pela ifthenpay, que é a empresa que também fornece a API para uma das soluções de pagamento por referência bancária já integrada no sistema. Antes de estar pronto para os utilizadores usarem é necessário inserir dois valores que são fornecidos pela própria ifthenpay no ARTSOFT Online: o **MBWayKey** e a **chave anti-phishing** (figura 3.2). Para além disso, o cliente também necessitará de informar a ifthenpay qual será o URL para o qual pretende que seja enviado o *callback*, que será explicado mais abaixo.

The screenshot shows the 'Meios de Pagamento' (Payment Methods) configuration page in the ARTSOFT ONLINE system. The left sidebar contains a navigation menu with options like 'Painel de Controlo', 'Aplicações', 'Sincronizar ARTSOFT', 'Temas', 'Multimédia', 'Utilizadores', and 'Meios de Pagamento'. The main content area is titled 'Meios de Pagamento' and includes sections for 'Referência Bancária (IfthenPay)', 'Cartão de crédito', and 'MBWay'. The 'MBWay' section contains a form with the following fields:

- MBWAYKEY**: A text input field.
- CHAVE ANTI PHISHING**: A text input field.
- URL Callback**: A text input field with a placeholder URL: `http://www.site.com/public/modules/ifthenPayMBWay/core/v1/ifthenPayMBWayAPI/paymentConfirmation?chave={CHAVE_ANTL_PHISHING}&referencia={REFERENCIA}&idpedido={ID_TRANSACAO}&valor={VALOR}&datahorapag={DATA_HORA_PAGAMENTO}&estado={ESTADO}`.
- MEIO DE PAGAMENTO BIC**: A dropdown menu.
- MEIO DE PAGAMENTO B2B PT**: A dropdown menu.
- MEIO DE PAGAMENTO B2B INTRA**: A dropdown menu.
- MEIO DE PAGAMENTO B2B EXTRA**: A dropdown menu.
- MEIO DE PAGAMENTO CC PT**: A dropdown menu.
- MEIO DE PAGAMENTO CC INTRA**: A dropdown menu.
- MEIO DE PAGAMENTO CC EXTRA**: A dropdown menu.

A blue 'OK' button is located at the bottom right of the form.

Figura 3.2: Configuração do MBWay

Após esta configuração estar concluída, o MBWay passará a funcionar corretamente nas aplicações em que está a ser utilizado. Utilizando como exemplo o eCommerce, ao fazer-se o pagamento de uma compra, é pedido um conjunto de informações (figura 3.3) ao utilizar o MBWay como meio de pagamento. Após estes dados serem confirmados pelo utilizador, é enviado um pedido, formatado de acordo com o protocolo SOAP, para um dos serviços disponibilizados na API, cuja resposta ou *callback* será enviada para o URL acordado com a ifthenpay.

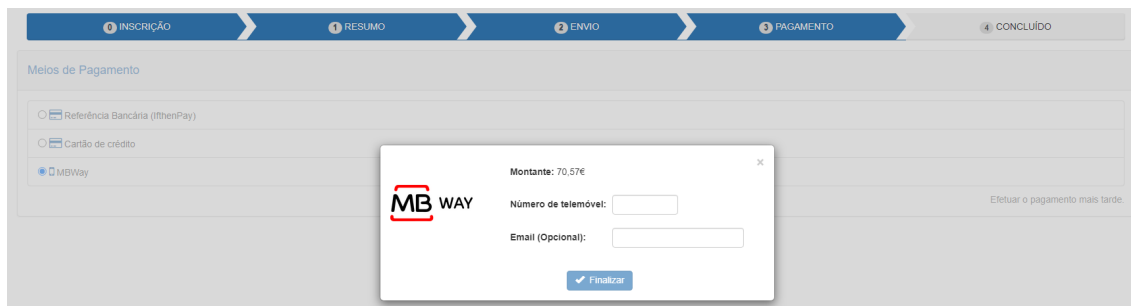


Figura 3.3: Pagamento com MBWay

Como o pagamento por MBWay é assíncrono, antes de criar as faturas da compra no eCommerce é necessário esperar por uma resposta que só é recebida caso o pagamento seja feito. Assim que a resposta é recebida, por questões de segurança a chave *anti-phishing* que está configurada no ARTSOFT Online é comparada com a chave que vem na resposta, e, caso sejam iguais, procede-se à criação das faturas do pagamento.

O protocolo SOAP ou *Simple Object Access Protocol*, referido anteriormente, é utilizado para troca de informações estruturadas em interações com serviços *web*, que são descritos num documento WDSL ou *Web Service Definition Language*. Este protocolo utiliza mensagens em formato XML que são, normalmente, enviadas utilizando o protocolo de transporte HTTP ou *Hypertext Transfer Protocol*, apesar de ser possível utilizar outros.

Finalmente, é necessário referir uma limitação desta API, quando comparada com implementações como o PayPal ou mesmo com o módulo explicado na secção seguinte. Ao contrário das soluções referidas anteriormente, a solução fornecida pela ifthenpay não oferece forma de testar num ambiente local, ou seja, no *localhost*, nem de fazer pagamentos de teste que não descontem da conta bancária de quem está a testar. Para contornar este problema, foi fornecido à ifthenpay um URL que tinha como destino um servidor de testes da ARTSOFT onde, após cada pagamento, era necessário ir buscar os parâmetros do *callback* recebido e inseri-los diretamente num *browser* aberto no computador em que se queria testar. Além disso, não havia forma de simular um pagamento, ou seja, o valor pago era realmente descontado da conta bancária utilizada para testar, sendo assim necessário enviar um email à ifthenpay para que devolvessem o valor pago após cada

compra de teste. Por estas razões, o método de teste foi bastante limitado, não podendo assim ser considerado representativo de um ambiente real.

3.2.2 Cartões de Crédito

Ao contrário do MBWay, os serviços utilizados neste módulo para se proceder ao pagamento com cartões de crédito são fornecidos pela rede Unicre. A Unicre disponibiliza serviços que permitem o pagamento com algumas das marcas mais utilizadas de cartões de crédito como Visa, Mastercard e American Express. Mas apesar de ser uma empresa diferente a fornecer os serviços, também neste módulo é necessário configurar dois valores no ARTSOFT Online (figura 3.4) que são fornecidos pela Unicre, neste caso: o *Access Token* e o *Entity ID*.

Figura 3.4: Configuração do módulo de cartões de crédito

Para que seja fácil testar a integração destes serviços com o sistema, a Unicre disponibiliza dois URLs distintos para invocação de serviços: um para um ambiente real, em que é necessário inserir dados de um cartão válido e cujo os montantes das compras serão descontados das contas bancárias respetivas, e um para ambiente de testes cujos dados a inserir estão disponíveis na documentação fornecida pela Unicre e cujos montantes não serão descontados de nenhuma conta bancária, já que os dados inseridos não correspondem a um cartão real. Para que seja fácil para o administrador do sistema decidir qual utilizar, existe a opção "TEST MODE" (figura 3.4) que, caso esteja selecionada, fará o sistema utilizar o segundo URL referido e, caso contrário, fará o sistema utilizar o primeiro URL referido. Utilizando mais uma vez o eCommerce como exemplo, ao selecionar-se os cartões de crédito como meio de pagamento, são pedidos os dados habituais em pagamentos deste género (figura 3.5). Após o utilizador inserir estes dados e antes de se invocar os serviços para efetuar o pagamento, é verificado o número do cartão inserido com o algoritmo de Luhn [13]. Este algoritmo foi desenhado com o intuito de detetar erros acidentais

e não travar ataques maliciosos. Caso o número do cartão passe o teste, é enviado um pedido HTTP, com os dados necessários ao pagamento, para o URL que fornece os serviços. Para enviar este pedido é utilizada a biblioteca cURL do PHP, que permite conectar e comunicar com diferentes tipos de servidores através de diferentes tipos de protocolos, um dos quais é o HTTP. Por este meio de pagamento ser síncrono, o pagamento é feito instantaneamente o que permite ao serviço invocado responder imediatamente. Caso esta resposta indique que o pagamento foi feito com sucesso, os documentos do pagamento são criados e o pagamento é dado como completo.

Figura 3.5: Pagamento com cartão de crédito

3.3 Módulo de Troca de Horários

A implementação deste módulo teve como objetivo a automatização do processo de troca de horários de trabalho no "Portal do Colaborador". Esta é uma funcionalidade que já estava disponível no ARTSOFT ERP, mas que ainda não existia em nenhuma das soluções *web*. Este módulo permite a um utilizador consultar tanto as suas próprias trocas de horário como as dos utilizadores de quem é coordenador, mas apenas permite fazer pedidos de novas trocas caso tenha permissões para tal, as quais são definidas no ARTSOFT ERP.

Existem dois tipos de pedidos neste módulo. Por um lado, é possível fazer um pedido para que dois colaboradores façam uma troca direta de horários durante um período de tempo, ou seja, o colaborador 1 ficará com o horário do colaborador 2 durante esse período e vice-versa (figura 3.6). Por outro, pode ser um pedido de troca simples, ou seja, um colaborador passará a ter um horário de trabalho diferente no período especificado, sem afetar o horário de nenhum outro colaborador (figura 3.7).

The screenshot shows a web form titled "Pedido de Troca" with a close button (X) in the top right corner. Below the title, there are two tabs: "Troca Simples de Horário" and "Troca com Outro Colaborador", with the second tab being active. Under the "Validade" section, there are two date pickers: "Data Início" (2020-12-23) and "Data Fim" (2020-12-23), each with a calendar icon. The "Colaboradores a Trocar" section contains two rows of input fields. The first row has "Código" (1), "Colaborador 1" (Raul Rafael Resende), and "Horário" (Noturno). The second row has "Código" (17), "Colaborador 2" (Laurinda Loureiro Laranjeiro), and "Horário" (Geral). A blue "GUARDAR" button is located at the bottom right.

Figura 3.6: Troca de horário entre dois utilizadores

The screenshot shows the same "Pedido de Troca" form, but with the "Troca Simples de Horário" tab active. The "Validade" section remains the same. The "Troca de Horário" section has four input fields: "Código" (10), "Colaborador" (Lara Luísa Lapa), "Horário Antigo" (Geral), and "Novo Horário" (24h). A blue "GUARDAR" button is at the bottom right.

Figura 3.7: Troca simples de horário

3.4 Módulos de Gestão de Projetos

Ao contrário dos capítulos anteriores que consistiram na adição de novas funcionalidades no eCommerce e no Portal do Colaborador, este desenvolvimento teve como objetivo alterar o comportamento de dois módulos de gestão de projetos que se encontram disponíveis neste último. No primeiro módulo, designado por *Timesheets*, é possível consultar e registar quanto tempo um utilizador esteve a trabalhar numa tarefa (figura 3.8).

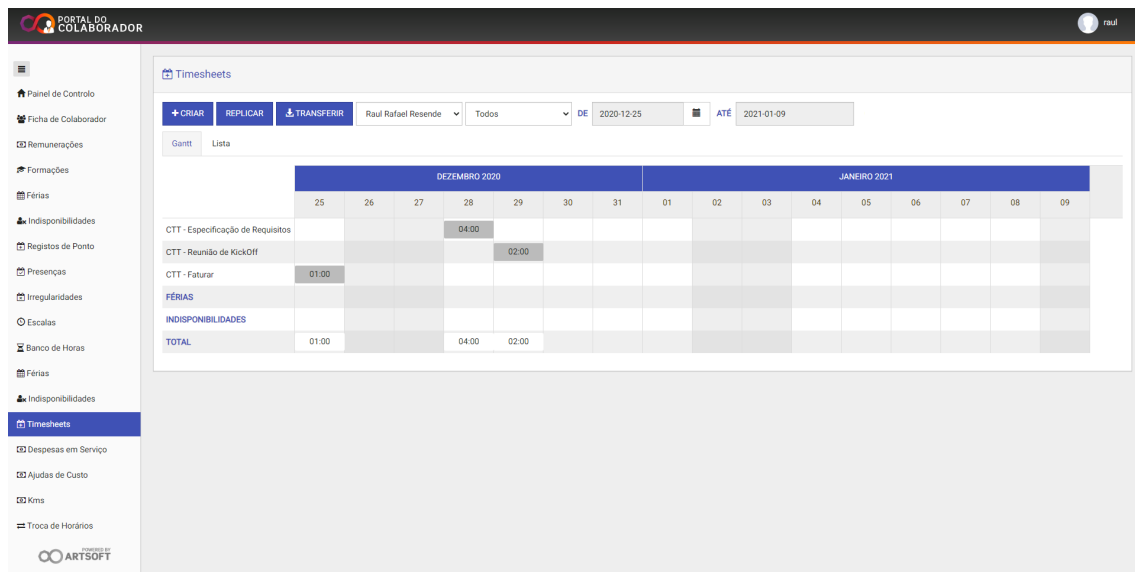


Figura 3.8: Página das Timesheets

O segundo módulo, que está dividido em "Despesas em Serviço", "Ajudas de Custo" e "Kms", permite consultar, editar e, caso seja administrador, aprovar ou rejeitar despesas relacionadas com projetos (figura 3.9).

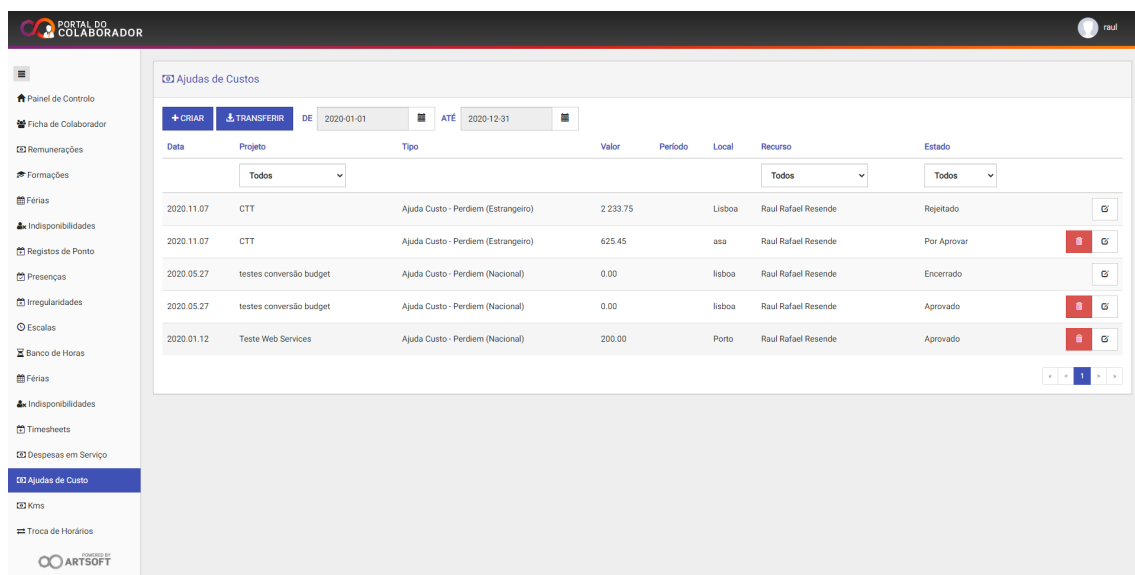


Figura 3.9: Página de Ajudas de Custo

Em ambos os módulos, todos os registos eram mostrados, independentemente se o projeto a que pertenciam estava encerrado ou não. Assim, foi pedido que se adicionasse uma opção que filtre os registos que pertencem a projetos que ainda se encontram abertos, dos registos que pertencem a projetos já encerrados. Uma vez que todos os dados mostrados nesta aplicação estão guardados no ARTSOFT ERP, foi necessário que os

responsáveis pelo desenvolvimento dos serviços ARTSOFT adicionassem uma opção ao serviço responsável por obter esses registos, que permita filtrá-los. Quando essa opção foi adicionada ao serviço, apenas foi necessário alterar o formato de invocação desse serviço e adicionar a opção "Com Histórico" (figura 3.10) às páginas desses módulos para que o utilizador a possa desativar caso queira apenas os registos dos projetos aberto, ou ativar caso queira ver todos os registos.

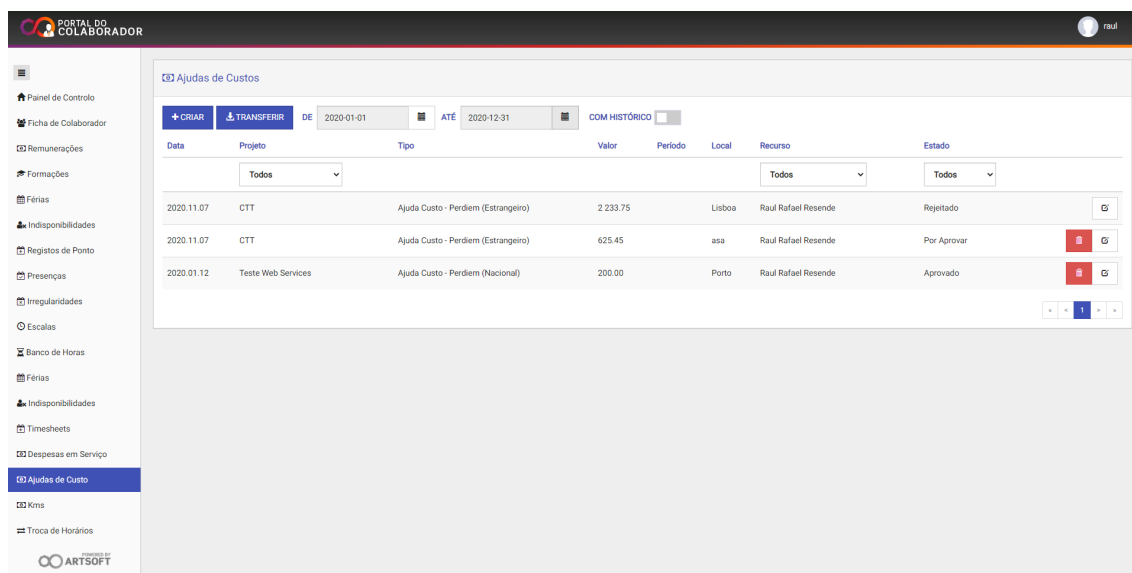


Figura 3.10: Opção - Com Histórico

Além disso, foi também pedido que se adicionasse um conjunto de configurações que permitem ao administrador do sistema definir que campos serão obrigatórios, entre os campos opcionais, ao criar ou editar registos e o nível de compressão das imagens ao guardá-las.

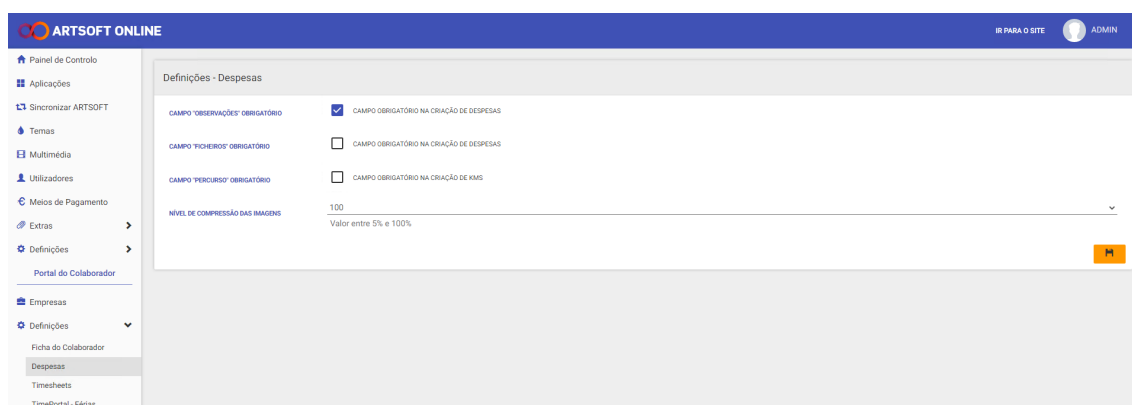


Figura 3.11: Definições do Módulo de Despesas

Assim, ao inserir uma nova despesa nos "Kms"(figura 3.12) utilizando as definições da figura 3.11, apenas o campo das observações será obrigatório e caso se insira uma imagem, esta será guardada com o tamanho original.

The image shows a web form titled "Criar Kms" with a close button (X) in the top right corner. The form is organized into several sections:

- Projeto ***: A dropdown menu.
- Tipo ***: A dropdown menu.
- Data ***: A date input field with a calendar icon.
- Observações ***: A large text area for notes.
- Kms**: A section header.
- Total ***: A dropdown menu.
- Custo sem IVA**: A text input field showing the value "0".
- Percurso**: A text input field.
- Ficheiros**: A section header.
- Escolher Ficheiros**: A button next to the text "Nenhum ficheiro seleccionado".

At the bottom right of the form is a blue button labeled "GUARDAR" with a save icon.

Figura 3.12: Definições do Módulo de Despesas

Finalmente, foi criada uma página de configurações semelhante para o módulo das *timesheets* (figura 3.13), onde se pode definir se o campo observações será obrigatório ou não e se a descrição das tarefas terá o nome do projeto e da tarefa ou apenas o nome da tarefa. Como exemplo para esta segunda configuração, caso a opção selecionada fosse "Só Tarefa", na figura 3.8 não apareceria o nome "CTT" na descrição da tarefa.

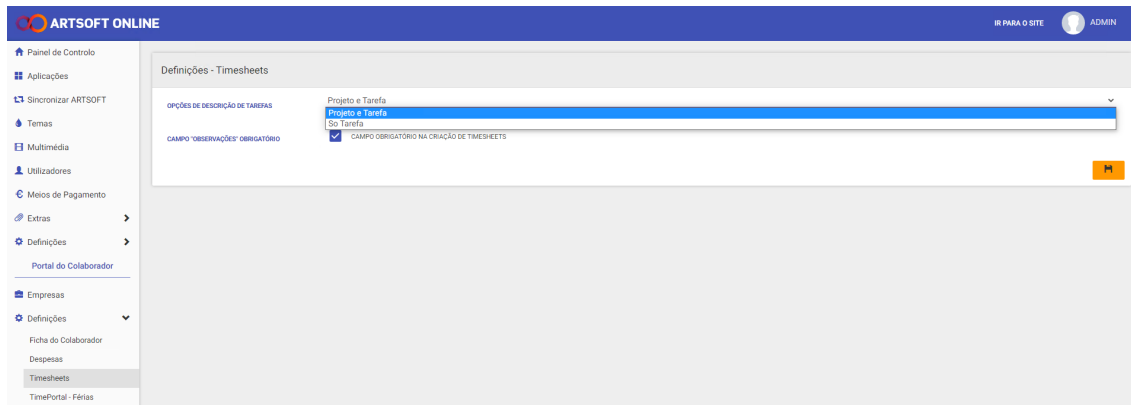


Figura 3.13: Definições do Módulo das Timesheets

Capítulo 4

Aplicação de Gestão de Equipamento Médico

4.1 O Problema

O cliente que requisitou o desenvolvimento da aplicação é uma empresa especializada na venda e distribuição de produtos médicos cirúrgicos em Portugal, e de modo a se manter competitiva no mercado foi pedido que se desenvolvesse uma aplicação que, por um lado, automatize alguns processos internos mas, principalmente, que aumente a eficiência de todo o fluxo de distribuição de produtos. Anteriormente ao desenvolvimento desta aplicação, quando era necessário efetuar uma encomenda de produtos, o pedido dessa encomenda tinha de ser feito por *email*, o que não é escalável quando começam a chegar muitos pedidos de, possivelmente, muitos clientes diferentes.

Assim, esta aplicação teve como principais objetivos padronizar a criação e aprovação de pedidos, visualizar pedidos existentes, facilmente visualizar todos os produtos que estão disponíveis e visualizar todos os clientes localizados na mesma área geográfica que o cliente que está a aceder à aplicação, tendo o seu desenvolvimento ficado dividido em quatro *sprints*. Na tabela abaixo é possível visualizar as datas de entrega que inicialmente foram planeadas para cada *sprint*, que sofreram algumas alterações desde que foram definidas, devido a atrasos no desenvolvimento que resultaram de *bugs* introduzidos durante o desenvolvimento que demoraram a ser corrigidos, mas também de pedidos de alterações por parte de cliente de funcionalidades já implementadas. As três primeiras entregas foram as entregas onde foi planeado desenvolver a aplicação base, enquanto que a quarta entrega será para o desenvolvimento de funcionalidades extra que não estavam previstas na aplicação inicial. Esta última sofreu alguns atrasos e as funcionalidades previstas ainda estão em desenvolvimento.

	Início	Fim
1ª Entrega de pacote de requisitos	08/04/2021	12/04/2021
2ª Entrega de pacote de requisitos	10/05/2021	14/05/2021
3ª Entrega de pacote de requisitos	07/06/2021	11/06/2021
4ª Entrega de pacote de requisitos	11/07/2021	16/07/2021

Tabela 4.1: Datas de entrega

4.2 Infraestrutura de Apoio ao Desenvolvimento

4.2.1 Mocks

O documento de requisitos inicialmente acordado com o cliente, apenas continha imagens de páginas de aplicações que já estavam implementadas, e cujo template não se enquadrava no contexto da aplicação a ser desenvolvida. Assim, ainda antes do começo do desenvolvimento da aplicação, foi acordado que se iria utilizar a ferramenta MockFlow (capítulo 2.8) para se desenvolverem os *templates* adequados ao contexto da funcionalidade em questão, tanto para a aplicação *web* como para a aplicação móvel.

Para desenvolver estes *templates*, a equipa de desenvolvimento teve um conjunto de reuniões de *brainstorming* onde cada elemento propunha um conjunto de sugestões. Após todos os elementos proporem todas as sugestões, as melhores ideias eram aproveitadas e o *template* era desenhado no momento. Após todos os *templates* estarem desenvolvidos, estes foram validados com o cliente e caso o cliente pretendesse alguma alteração, o processo era repetido.

Todo este processo permitiu poupar bastante tempo no desenvolvimento da aplicação, já que existia um modelo onde basear a página a ser desenvolvida já validado pelo cliente, apesar de pudermos haver pequenas diferenças entre o *template* inicial (figura 4.1) e a página final (figura 4.2).



Figura 4.1: Mock da página de criação de pedido

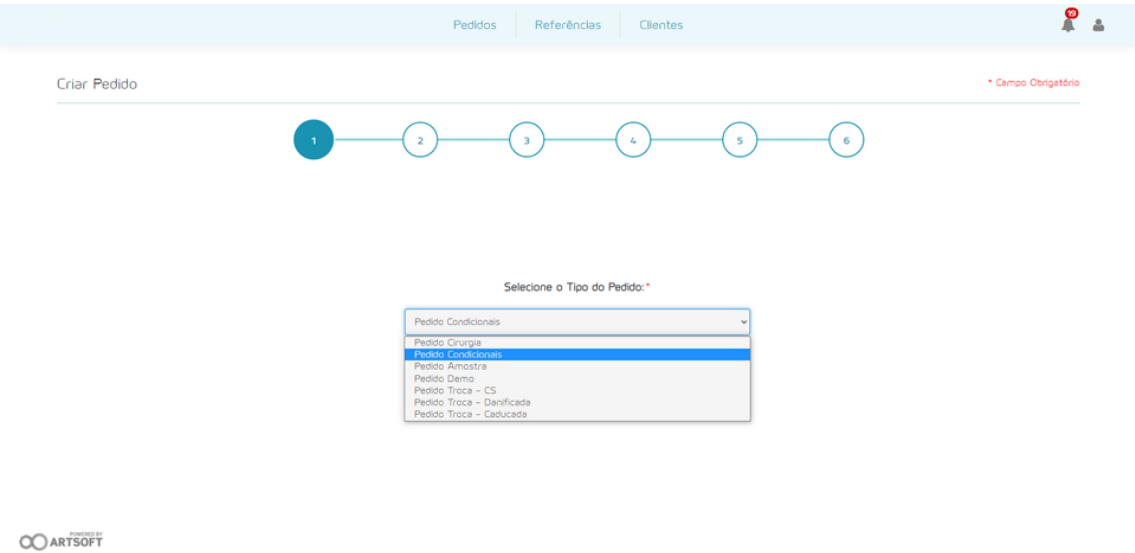


Figura 4.2: Página de criação de pedido funcional

4.2.2 Gestão

Tal como foi referido no capítulo 2.9.1, todos os pedidos de desenvolvimento e reportes de erros são comunicados através de eventos. Numa aplicação nova, inevitavelmente irão acabar por haver bastantes reportes, principalmente os reportes de erros. Apesar do ARTSOFT ERP ter esta funcionalidade de eventos, quando começam a haver muitos eventos, fica um pouco complicado perceber ao que cada evento se refere. Por esta razão, utilizou-se a ferramenta Trello onde se criaram cinco categorias, e pelas quais se distribuíram todos os eventos de cada membro da equipa de modo a que toda a equipa esteja ciente do que estava e do que não estava feito. As categorias estavam divididas da seguinte forma (figura 4.3):

- **BACKLOG:** Contém todos os eventos que apenas estavam dependentes do *frontend*, tanto da aplicação *web* como da aplicação móvel, para serem dados como concluídos;
- **TODO (Backend):** Contém todos os eventos que estavam dependentes dos desenvolvimentos da API e dos serviços ARTSOFT, para os programadores do *frontend* puderem avançar com o que estava pedido no evento, ou para que pudesse ser dado como concluído;
- **TODO (Extras):** Contém todos os eventos referentes a desenvolvimentos que não estavam previstos na entrega da aplicação base, e que estão planeados para uma entrega futura;
- **DONE (Backend):** Contém todos os eventos que os programadores de *frontend* podem avançar, já que a API e/ou os serviços ARTSOFT já suportam o desenvolvimento em questão;
- **IN PROGRESS:** Contém todos os eventos que estavam a ser tratados no momento em que se acedeu ao Trello;
- **FINISHED:** Contém todos os eventos que podem ser dados como concluídos já que todos os desenvolvimentos estão terminados;

Adicionalmente, cada "carta" que representa um evento tem um conjunto de *tags* referentes à área da aplicação a que se refere e uma *checkbox* referente a cada uma das *tags*. Assim, um evento só é dado como concluído quando cada programador, responsável por cada "área", selecionar a sua *checkbox*, ou seja, quando todas as *checkboxes* estiverem selecionadas.



Figura 4.3: Estrutura do Trello

4.3 Arquitetura

O desenvolvimento desta aplicação teve como principal requisito, ter uma aplicação móvel que funcionasse em conjunto com uma aplicação *web*, podendo todos os dados utilizados por elas ser configurados através do ARTSOFT ERP. Para tal foi utilizada uma API partilhada por ambas as aplicações de modo a uniformizar todos os pedidos feitos, já que ambas têm exatamente as mesmas funcionalidades e fazem os mesmo pedidos à API. Finalmente, a API comunica com os serviços ARTSOFT da mesma forma que todas as outras aplicações desenvolvidas pela ARTSOFT, através de pedidos XML. Assim, uma visão global da arquitetura da aplicação pode ser dada pela figura [4.4](#).

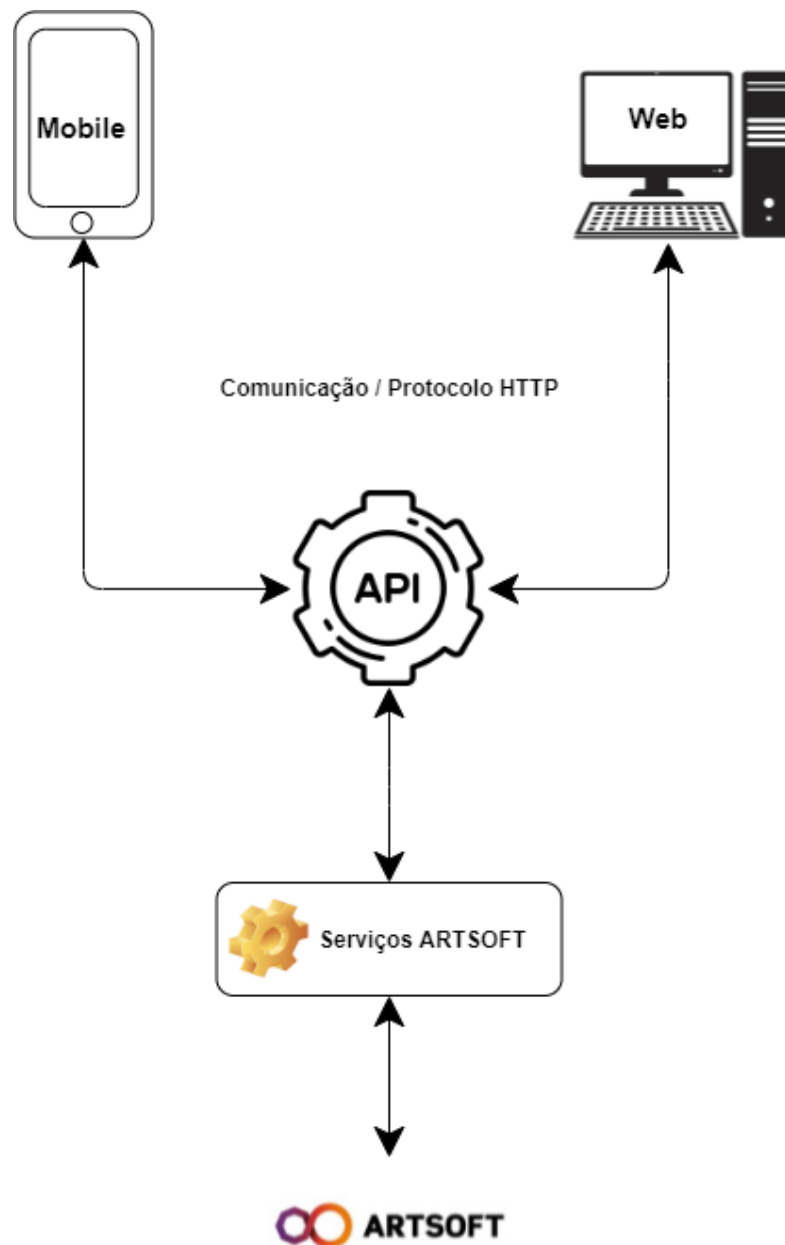
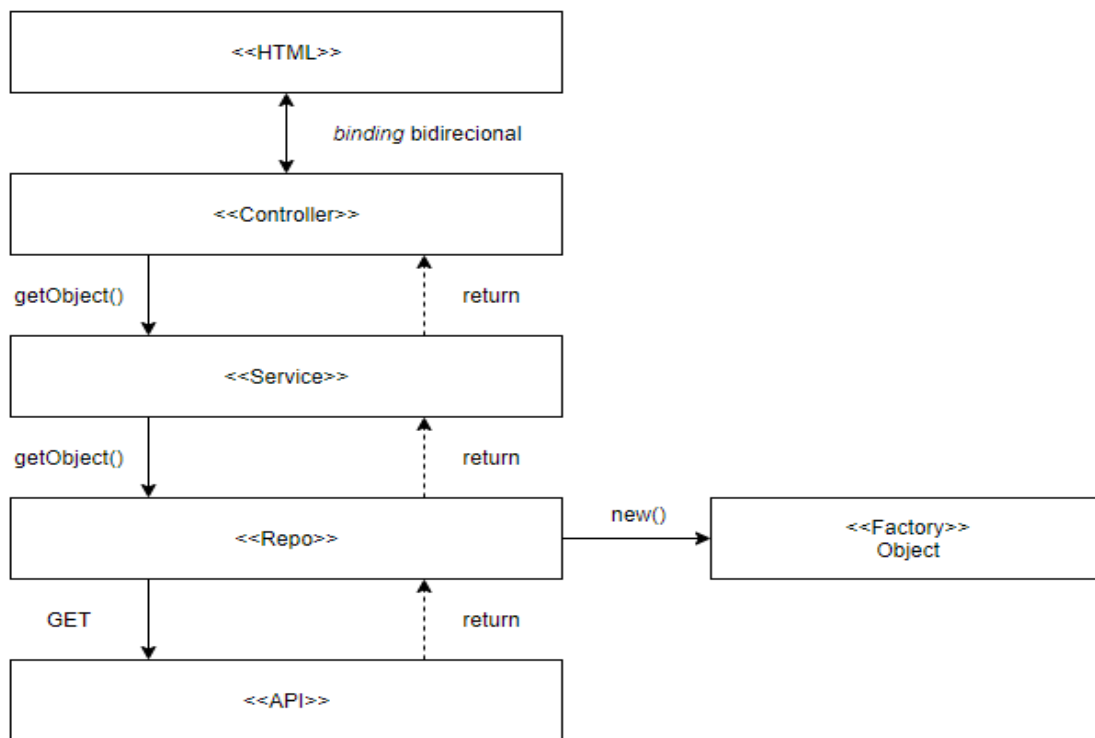


Figura 4.4: Arquitetura da aplicação

Com o foco do projeto na componente *web* da aplicação, foi pensada uma arquitetura com o mínimo de dependências possíveis entre módulos, onde cada módulo fosse unicamente dependente de si próprio. Assim, um módulo na aplicação está dividido em quatro componentes principais (figura 4.5):

- **Controller:** O *Controller* corresponde ao controlador AngularJS da página principal do módulo mas apenas tem a função de fazer o chamado *binding* bidirecional de informação entre o controlador e a página, ou seja, transfere todo o tratamento de lógica que seja necessário efetuar para a camada do *Service*;

- **Service:** O *Service* corresponde a um serviço AngularJS e que faz o tratamento de toda a lógica do módulo. Assim, sempre que é necessário reformatar os dados para serem enviados para a API, ou, eventualmente, haver comunicação entre módulos, é responsabilidade do serviço fazer esse trabalho;
- **Repo:** O *Repo* corresponde a um serviço AngularJS e que é responsável por toda a comunicação com a API;
- **Factory:** A *Factory* corresponde a uma *Factory* AngularJS e é utilizada para representar um modelo na aplicação. Este elemento contém um construtor e um conjunto de *setters* e *getters* para todos os atributos possíveis do modelo em questão. Finalmente foi definido como nomenclatura padrão o nome do modelo que representa;

Figura 4.5: Arquitetura *frontend* representada em diagrama UML

Após a estrutura geral dos módulos estar bem definida, foram pensados e desenvolvidos um total de nove módulos: WebUI, Auth, Clients, Dashboard, Documents, Files, Notification, Shortcut e Sku. O módulo WebUI não segue a estrutura definida anteriormente, já que foi pensado com um objetivo muito específico. O WebUI é um módulo que apenas contém diretivas AngularJS genéricas, que podem ser usadas por toda a aplicação. Estas diretivas não têm uma função específica, e podem ir de uma simples verificação que apenas permite inserir dígitos numa *input* até a uma *input* estilizada que representa uma barra de pesquisa e que permite receber uma função de pesquisa dinâmica para fazer a

procura. Os restantes módulos seguem a estrutura referida anteriormente e também podem conter diretivas que apenas façam sentido no contexto do módulo em questão. Cada um destes módulos apenas acede aos *endpoints* da API relativos ao modelo que representam e, caso necessitem de informação referente a outro módulo, necessitam de utilizar os serviços desse módulo. Assim, a figura 4.6 pode ser utilizada para representar a utilização de diretivas por parte das páginas HTML.

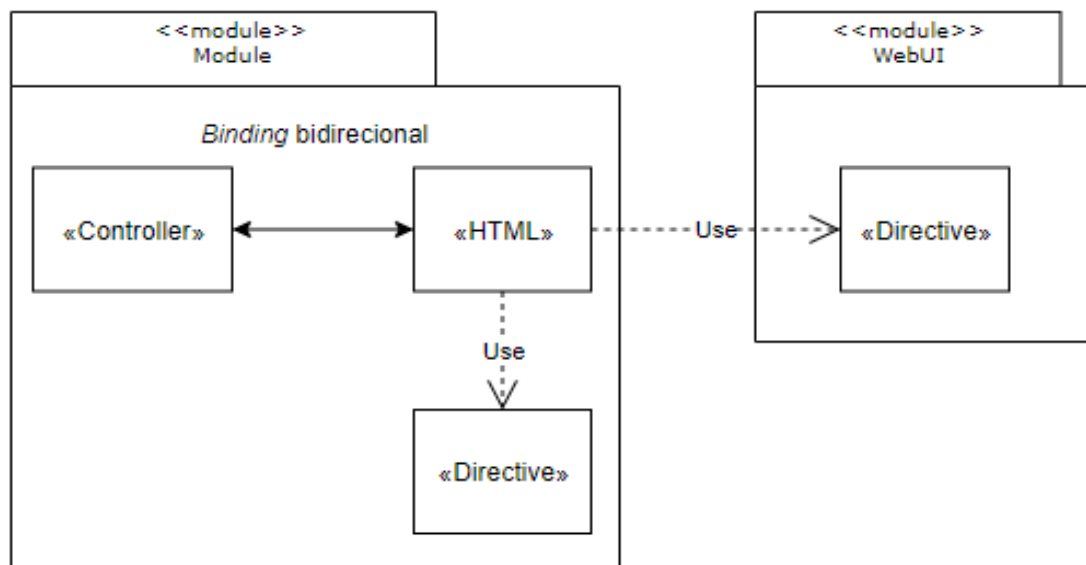


Figura 4.6: Esquema de diretivas representada em diagrama UML

4.4 Implementação

Para a implementação da aplicação, foi reunida uma equipa de cinco programadores, que ficou dividida da seguinte forma:

- Um programador responsável pelo desenvolvimento da aplicação móvel;
- Dois programadores responsáveis pelo desenvolvimento da aplicação *web*, onde eu fui incluído;
- Um programador responsável pelo desenvolvimento da API;
- Um programador responsável pelo desenvolvimento dos serviços ARTSOFT com que a API comunica;

Nas restantes sub-secções serão apresentadas todas as funcionalidades desenvolvidas durante o projeto.

4.4.1 Consulta de Clientes e Referências

Tal como referido no capítulo [4.1](#), o projeto foi dividido em quatro *sprints*, e no primeiro *sprint* foi pedido que se implementassem as páginas que permitem visualizar as listagens de clientes e referências, assim como as páginas que permitem visualizar as informações específicas de cada uma. Assim, após a distribuição de trabalho por parte do gestor de projeto, fiquei responsável pelo desenvolvimento das páginas de listagem, enquanto que o outro colega da equipa da aplicação *web* ficou responsável pelas páginas de visualização de informações de referências e clientes.

Numa fase inicial, a aplicação estava a ser desenvolvida com uma visão baseada em tabelas, o que acabou por ser uma limitação em fases mais avançadas da aplicação. Apesar disso, estas duas páginas foram das poucas que mantiveram um aspeto baseado em tabelas, já que a informação que era necessária mostrar era bastante simples e este elemento HTML adequava-se ao contexto das páginas.

Referências

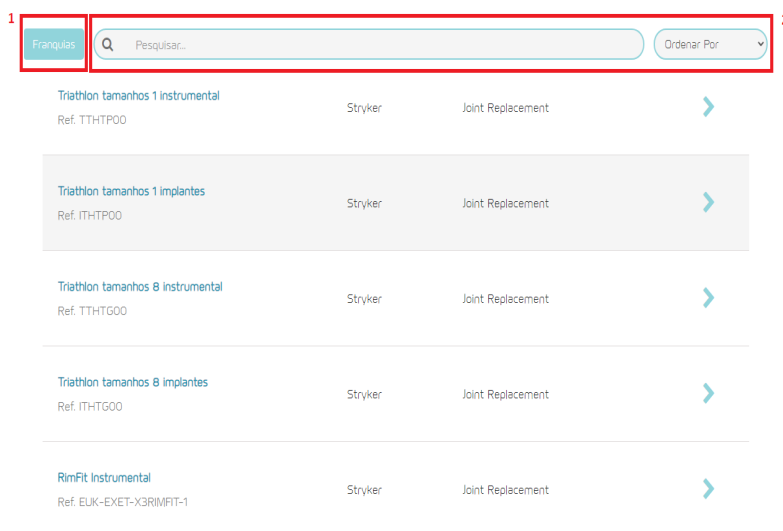


Figura 4.7: Página de listagem de referências

Apesar do elemento principal destas páginas ser a tabela, uma funcionalidade que era significativamente importante para o cliente era haver a possibilidade de fazer pesquisas e filtros, já que estas listagens podem ter centenas de registos. Para tal, e de modo a seguir a arquitetura definida no capítulo 4.3, foi criada uma diretiva genérica que soluciona o problema ao fornecer uma barra de pesquisa e uma *dropbox* onde é possível escolher que parâmetro será usado para ordenar os registos (por exemplo o nome), tal como se pode ver no elemento identificado por 2 na figura 4.7. No caso da página de referências, foi necessário criar um filtro extra para filtrar por franquias, que, resumidamente, são categorias de referências.

Para este último caso das franquias, foi adicionado um botão que abre uma *modal* com todas as franquias que existem, que se pode ver em 1 na figura 4.7. Resumidamente, uma *modal* é uma página HTML que se sobrepõe e desativa o *scroll* do *body* da página HTML principal. Uma das principais preocupações foi como apresentar estas franquias, já que estas têm uma estrutura hierárquica, com "pais" e "filhos". Assim, o primeiro passo consistiu em criar uma *factory*, tal como referido no capítulo 4.3, para representar as franquias, e criar um *array* destes elementos recursivamente a partir da informação enviada pela API. Cada elemento criado a partir desta *factory* contém a designação da própria franquias, o seu id, um *array* com as suas franquias filho, assim como alguns atributos utilizados para controlar o elemento na página HTML. Após se ter um *array* destes elementos, bastou apresentá-los na modal com *checkboxes*, já que é possível filtrar as referências por múltiplas franquias ao mesmo tempo.

Apesar destes dois elementos estarem separados, eles têm de funcionar em "conjunto" já que sempre que se filtra por um conjunto de franquias, é necessário manter o

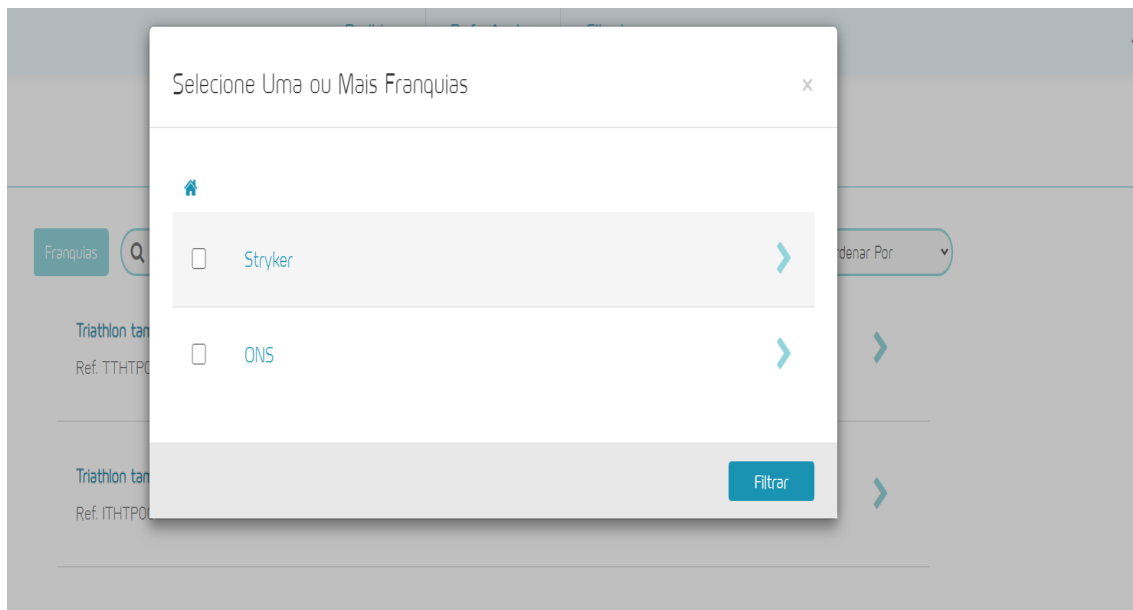


Figura 4.8: Modal do filtro de franquias

filtro da barra de pesquisa e *dropbox*, e vice-versa. Assim, os controladores que utilizam o botão das franquias e a diretiva da barra de pesquisa inicializam um objeto que guardam esses três filtros, ou seja, um *array* de franquias, uma *string* que corresponde(m) à(s) palavra(s) da barra de pesquisa e um inteiro que corresponde à opção selecionada na *dropbox*. Tanto este objeto, como a função de pesquisa que vai aplicar o filtro são enviados para a diretiva da barra de pesquisa e sempre que se deteta alguma alteração na palavra inserida na barra de pesquisa ou *dropbox*, esta função de filtragem é invocada com todos os filtros, incluindo as franquias.

4.4.2 Criação e Edição de Pedidos

Após o fim do primeiro sprint, o trabalho foi apresentado ao cliente que validou o que foi desenvolvido até ao momento e reportou *bugs* detetados durante os testes realizados. Assim, no segundo sprint, foi pedido que se corrigissem os *bugs* detetados e que se implementassem as páginas de criação e edição. Assim como no primeiro *sprint*, o trabalho foi dividido entre a equipa de desenvolvimento do *frontend* da aplicação *web*, ficando eu responsável pelo desenvolvimento da página de criação, apesar de também ter ajudado em certos pormenores da página de edição já que os campos apresentados na página de criação são os mesmos da página de edição.

Inicialmente, o aspeto da página de criação pedido estava baseado numa página de uma outra aplicação já implementada que apresentava todos os campos de *input* na mesma página, o que era impossível de implementar pelo facto do número de campos pretendido na criação de um pedido ser demasiado grande para ser apresentado numa única página. Assim, e após validação com o cliente, decidiu-se por uma página de criação por "pas-

sos”, onde em cada ”passo” se mostravam campos relacionados entre si, ou seja, todos os dados relacionados com datas e moradas eram apresentados num ”passo”, e os dados relacionados com o material do pedido eram apresentados num ”passo” diferente. Para tal foram desenvolvidos um total de sete ficheiros HTML, um principal que contém a barra de passos (figura 4.9) e onde vão ser injetados os ficheiros HTML de cada um dos passos, e os restantes ficheiros, onde cada um apresenta a página de um ”passo” específico.

Como se pode observar na figura 4.9, para esta página foi definido um padrão visual para ajudar os utilizadores navegarem neste sistema de passos:

- No passo que se está a visualizar atualmente, a cor do fundo do círculo é um azul escuro e o número desse círculo branco, como se pode visualizar no passo 3 da figura referida;
- Um passo que ainda não foi acedido pelo utilizador apresenta um círculo com um fundo branco com bordas azuis e um número azul, como se pode visualizar nos passos 4, 5 e 6 da figura referida;
- Um passo que já foi acedido pelo utilizador, mas que não está selecionado pelo utilizador no momento apresenta um círculo com fundo azul claro e com bordas e o número com um azul um pouco mais escuro.

O padrão deste caso não é só aplicado em passos anteriores ao selecionado atualmente, já que o utilizador pode já ter completado todos os passos e apenas estar a rever os dados que inseriu. Neste caso, todos os passos apresentam este padrão com a exceção do passo selecionado atualmente.

- No caso de o utilizador passar o cursor por cima de um passo que já acedeu anteriormente, foi decidido que o círculo teria um diâmetro igual ao do círculo do passo selecionado atualmente, mas com um aspeto mais ”acinzentado”. No caso de utilizador fazer a mesma ação num passo que ainda não acedeu anteriormente, o aspeto não muda já que não lhe é possível selecionar esse passo;

A escolha das cores para este padrão não foi aleatória. O azul foi utilizado neste padrão e, de modo geral, em toda a aplicação, pois esta cor está fortemente ligada à imagem comercial do cliente, e foi decidido que esta seria a cor base em toda a aplicação.

Além deste padrão, foram tomadas algumas precauções de modo a diminuir ao máximo a possibilidade de um utilizador fazer ações erradas nesta página. Uma dessas precauções já foi parcialmente explicada no padrão falado anteriormente, que foi não permitir ao utilizador selecionar um passo posterior sem introduzir e confirmar os dados no passo atual, já que existem campos de *input* obrigatórios que, mesmo sendo feita essa verificação na API, poderia provocar erros no processo de criação de pedidos. Isto foi aplicado tanto na navegação pela barra de passos, como nos botões ”Voltar” e ”Continuar”. Foi também

apresentado um asterisco vermelho ao lado dos títulos dos campos de *input* obrigatório para complementar as verificações explicadas anteriormente, e que permite ao utilizador distinguir rapidamente os campos que precisa e que não precisa de preencher.

Finalmente foi criada uma diretiva simples que apenas permite ao utilizador introduzir dígitos em *inputs* que pedem, por exemplo, números de telemóvel ou preços. Apesar de no HTML nativo ser possível definir o tipo de input pretendido, como *number* ou *text*, estes *inputs* permitem a inserção de caracteres diferentes do tipo definido, apenas desativando os botões quando os caracteres introduzidos são diferentes do tipo esperado.

A interface de criação de pedidos condicionais apresenta uma barra superior com abas para 'Pedidos', 'Referências' e 'Clientes', e um ícone de notificação. O título da página é 'Criar Pedido Condicionais' e há uma legenda para '* Campo Obrigatório'. Um progressor de 6 etapas está no topo, com a etapa 3 selecionada. O formulário contém os seguintes campos:

- Data de Condicionais ***: Campo de data com o valor '30-08-2021' e ícone de calendário.
- Local de Entrega ***: Campo de texto vazio.
- Horário de Entrega ***: Campo de texto vazio.
- Horas do Cliente**: Campo de texto com o valor '08:30 / 10:00 / 13:00 / 19:00'.
- Transporte Urgente**: Campo de seleção com o botão desativado.

Na base do formulário, há dois botões: 'Voltar' (desativado) e 'Continuar' (ativo). No canto inferior esquerdo, há o logótipo 'POWERED BY ARTSOFT'.

Figura 4.9: Página de criação de pedidos

Alterações ao passo 4 da criação de pedidos

O aspeto das páginas dos vários passos manteve-se praticamente igual desde o momento da sua implementação, apenas sendo necessário remover e adicionar campos de *input* à medida que o cliente ia dando *feedback*, mas houve uma página que teve de ser reestruturada devido à quantidade de informação que era necessário mostrar em cada elemento. Inicialmente, o passo 4 era uma tabela com todas as referências seleccionadas, com um aspeto parecido à figura 4.7 com a diferença que apenas eram mostrados os campos mais à esquerda dessa figura e que era possível inserir a quantidade desse elemento que era suposto estar no pedido. Mas à medida que o cliente foi adicionando campos de *input* a cada referência no processo de criação de um pedido, não foi possível manter este aspeto, já que era impossível mostrar tanta informação em cada elemento da tabela. Apesar de se seleccionar um conjunto de referências na criação/edição de um pedido, dentro da aplicação e neste contexto específico, uma referência é considerada como uma "linha" que, resumidamente, consiste no id da referência, quantidade e os restantes parâmetros que se podem editar e que dependem do tipo de pedido. Para tal, foi criada uma diretiva que representa cada referência seleccionada, e apesar de que o problema do "espaço" estar na edição de pedidos, e não na criação, esta diretiva foi utilizada em ambos já que o elemento criado através desta diretiva é mais dinâmico do que uma tabela. Para este elemento foi criado um ficheiro HTML que contém uma *div* colapsável com todos os campos que uma referência pode ter quando seleccionada na criação/edição de um pedido, e que podem variar dependendo do tipo de pedido em questão. Mais uma vez, foi criada uma *factory* para representar este modelo, e sempre que uma referência é seleccionada, uma nova "linha" é criada e colocada num *array* do controlador da página de criação/edição de pedido. Após se ter este *array* disponível, na página HTML inseriu-se um ciclo onde, para cada elemento deste *array*, se criava um novo elemento da diretiva referida anteriormente ficando com o aspeto final visualizado na figura 4.10.

Pedidos Referências Clientes

Criar Pedido Demo * Campo Obrigatório

1 2 3 4 5 6

★ Criar Favorito

▼ Triathlon tamanhos 1 instrumental
Referência TTHTP00 1 + [Red X]

▼ Triathlon tamanhos 1 implantes
Referência ITHTP00 1 + [Red X]

Voltar + Adicionar Referência Continuar


ARTSOFT

Figura 4.10: Página de criação de pedidos - Referências selecionadas

Alterações ao passo 6 da criação de pedidos

O passo 6, é um passo específico aos pedidos do tipo "Cirurgia", ou seja, este passo só é mostrado quando, no passo 1, se seleciona o tipo referido anteriormente. Neste último passo no processo de criação de um pedido, são criados processos, onde cada representa uma cirurgia que será realizada e na qual serão utilizadas as referências e não só, definidas nos passos anteriores. Assim como no passo 4, também no passo 6 foi, inicialmente, utilizada uma tabela para representar os processos onde, para se inserir uma entrada, se define a data e o id (chamado número stryker pelo cliente) do novo registo. Mas após se ver os benefícios da nova diretiva criada para o passo 4, e apesar de ser possível incluir todos os campos do processo numa linha da tabela sem desformatar a página, criou-se uma diretiva similar à referida anteriormente, já que esta permite que sejam adicionados mais campos no futuro caso seja necessário.

Criar Pedido Cirurgia * Campo Obrigatório



Processo 4234

Data 02/09/21 - 12:56 🗑

VoltarCriar ProcessoGuardar

Figura 4.11: Página de criação de pedidos - Processos

Favoritos

Como é possível verificar na figura 4.10, ao se selecionar um conjunto de referências, é possível definir essa lista como favorita. Esta é uma funcionalidade simples, onde, quando se tem todas as referências pretendidas selecionadas, se clica no botão "Criar Favorito" que abre uma *modal* para definir a designação do novo favorito. Inicialmente, caso a lista de referências selecionada já fosse um favorito, era possível voltar a clicar neste botão para o remover, já que se assumiu que uma lista de referências era única no caso dos favoritos, ou seja, que não era possível ter duas listas com exatamente as mesmas referências em dois favoritos diferentes. Após um esclarecimento, verificou-se que este último caso não estava correto, e que era possível ter dois favoritos com listas exatamente iguais, mas com nome diferente. Assim, foi decidido que o botão referido só seria utilizado para a criação dos favoritos e, depois de criado um, para informar o utilizador que um favorito estava selecionado, estando o botão desativado neste caso. Assim, a eliminação de um favorito apenas pode ser feita pela página que apresenta a listagem de todos os favoritos já criados e onde é possível selecionar um deles (figura 4.12).

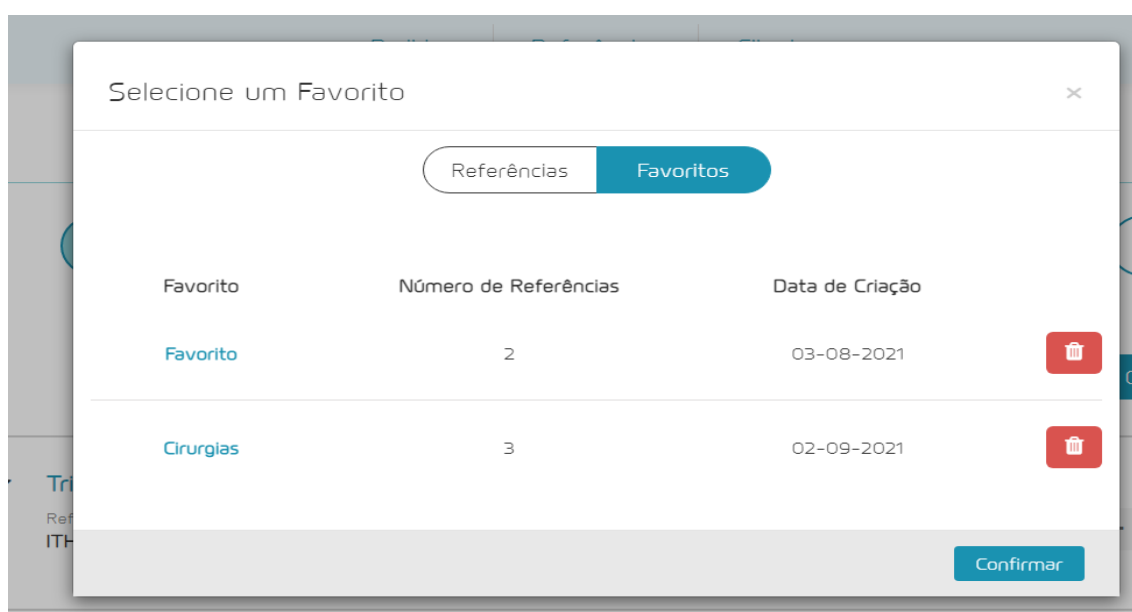


Figura 4.12: Página de seleção de favoritos

Capítulo 5

Conclusão

Por fim, segue-se uma análise de todo o trabalho desenvolvido, as dificuldades sentidas e todos os desenvolvimentos que estão planeados serem integrados na aplicação desenvolvida, no futuro.

5.1 Análise

O início do estágio para o desenvolvimento do projeto teve início em setembro de 2020, estando dividido em duas fases principais. A primeira fase consistiu na minha integração na empresa, onde foram apresentadas as tecnologias utilizadas pela empresa e atribuídos pequenos desenvolvimentos nas aplicações da mesma que permitiram familiarizar-me com o modo de utilização destas tecnologias. A segunda fase consistiu no desenvolvimento do projeto apresentado que teve um período de desenvolvimento de 5 meses (de março a julho). Durante todo o estágio participei no desenvolvimento de novas funcionalidades e correção de *bugs* de aplicações como o eCommerce e o Portal do Colaborador que ajudaram a integrar-me na empresa e a perceber como a mesma opera. Todo este trabalho permitiu identificar as limitações das *frameworks* utilizadas pela empresa, assim como perceber como é que estas permitem implementar ações como enviar ficheiros através do protocolo HTTP. Além disso também me permitiu identificar pontos nestas aplicações que podem resultar em erros ou simplesmente que dificultam a manutenção das mesmas. Todo este trabalho realizado durante a fase de integração na empresa, em conjunto com a experiência dos elementos mais seniores da equipa, permitiram um planeamento eficiente da aplicação que resultou em muito menos erros durante o desenvolvimento da aplicação do projeto do que em desenvolvimentos de outras aplicações.

Apesar de o desenvolvimento ter decorrido quase sem percalços, uma das principais dificuldades encontradas durante o desenvolvimento da aplicação foi perceber que *end-points* da API já estavam implementados já que, ainda antes do início do desenvolvimento da aplicação, foi desenvolvida uma listagem de todos os *endpoints* que estavam planeados serem desenvolvidos ao longo das várias fases do projeto, com todos os campos que

cada um esperava receber, e com o formato previsto da resposta. Como estavam dois programadores alocados para o desenvolvimento do *frontend* da aplicação *web* e apenas um alocado para o desenvolvimento da API, houve situações onde o desenvolvimento do *frontend* ficou terminado ainda antes dos *endpoints* utilizados neste estarem terminados. Como, antes de enviar o novo desenvolvimento para a equipa de controlo de qualidade, é necessário realizar uns teste preliminares para verificar se realmente o desenvolvimento está funcional, foi necessário estar constantemente em contacto com o programador responsável pela API de modo a perceber quando é que os *endpoints* estavam terminados. Como tudo isto estava a atrasar o calendário do projeto, começou-se a utilizar o Trello (capítulo [2.7](#)).

5.2 Trabalho Futuro

No futuro, estão planeados dois desenvolvimentos adicionais para esta aplicação, que foram requisitados à parte da aplicação base. O primeiro desenvolvimento pedido foi o registo e posterior possibilidade de consulta de todas as alterações efetuadas a um pedido. Sempre que uma alteração é feita, uma notificação será enviada ao utilizador que o criou e o histórico de todas as alterações feitas poderá ser acedido através da página de detalhes de um pedido. Finalmente, foi também pedido que se implementasse uma funcionalidade, chamada "BookExtension", que permita criar um novo pedido a partir de um pré-existente, e que fique associado a este último. O novo pedido terá que ser obrigatoriamente do mesmo tipo que o pedido a partir do qual foi gerado e quase todos os campos serão editáveis, estando pré-preenchidos com os dados desse pedido.

Abreviaturas

API Application Programming Interface. vii, 5, 17, 18, 30, 31, 33–36, 38, 45, 46

CSS Cascading Style Sheets. 6

GPL General Public License. 7

HTML Hypertext Markup Language. 6, 34–36, 38–40

HTTP Hypertext Transfer Protocol. 5, 18, 20, 45

PHP PHP: Hypertext Preprocessor. vii, 5, 20

SGBD Sistema de Gestão de Base de Dados. 8

SOAP Simple Object Access Protocol. 18

SQL Structured Query Language. 5, 11, 16

URL Uniform Resource Locator. 17–20

WDSL Web Service Definition Language. 18

XML Extensible Markup Language. 9, 10, 18, 31

Bibliografia

- [1] About mockflow. <https://support.mockflow.com/article/5-about-mockflow>. (Acedido a 08/2021).
- [2] About sonarqube. <https://www.sonarqube.org/about/>. (Acedido a 08/2021).
- [3] About tortoisefvn. <https://tortoisefvn.net/about.html>. (Acedido a 08/2021).
- [4] About trello. <https://trello.com/about>. (Acedido a 08/2021).
- [5] Global digital population as of october 2020. <https://www.statista.com/statistics/617136/digital-population-worldwide/>. (Acedido a 08/2021).
- [6] Jenkins user documentation. <https://www.jenkins.io/doc/>. (Acedido a 08/2021).
- [7] Preface. <https://www.php.net/manual/en/preface.php>. (Acedido a 08/2021).
- [8] Site artsoft. <https://www.artsoft.pt/>. (Acedido a 08/2021).
- [9] What is angularjs? <https://docs.angularjs.org/guide/introduction>. (Acedido a 08/2021).
- [10] What is mysql? <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>. (Acedido a 08/2021).
- [11] What is soap? https://www.ibm.com/support/knowledgecenter/en/SSMKHH_9.0.0/com.ibm.etools.mft.doc/ac55770_.htm. (Acedido a 08/2021).
- [12] World wide web (www). <https://www.britannica.com/topic/World-Wide-Web>. (Acedido a 08/2021).
- [13] Hans P. Luhn. US patent 2950048A. Computer for verifying numbers (1960).