



**DEPARTAMENTO DE CIÊNCIAS E TECNOLOGIAS**  
**MESTRADO EM ENGENHARIA INFORMÁTICA E DE TELECOMUNICAÇÕES**  
**UNIVERSIDADE AUTÓNOMA DE LISBOA**  
**“LUÍS DE CAMÕES”**

**Desenvolvimento de uma aplicação web de gestão dos processos de  
distribuição de bebidas, usando o conceito de sistema responsivo**

Trabalho de Projeto para obtenção do grau de Mestre em Engenharia Informática e de  
Telecomunicações

Autor: Ezequias Silva dos Santos

Orientadora: Professora Doutora Isabel Maria Surdinho Borges Alvarez

Número do candidato: 30003749

**Julho de 2020**

**Lisboa**



## **1. Agradecimentos**

Gratidão a Deus devo por esta fase concluída.

Gostaria de destacar o importantíssimo papel de minha orientadora Sr<sup>a</sup>. Doutora Isabel Alvarez, por todo o trabalho de orientação e pela sua disponibilidade dispensada durante todo o desenvolvimento deste projeto.

Gratidão à instituição de ensino UAL por esta oportunidade em permitir a realização deste projeto na conquista de mais conhecimentos.

Aos meus professores, que contribuíram com os ensinamentos dos quais sempre me lembrarei. Se hoje tenho a fonte do saber e a motivação para aprender, devo muito a eles.

Aos meus pais e familiares que estiveram sempre dispostos a me ajudar e sempre me encorajaram para seguir em frente.

Por fim, jamais conseguiria ter chegado ao cabo deste trabalho de projeto de mestrado que demorou muitos dias e horas de dedicação exclusiva, sem o apoio e paciência de uma pessoa que é toda especial para mim, a minha esposa Aline P. Alves Santos e filhos, Acsa P. Alves Santos, Raquel P. Alves Santos e Asaph P. Alves Santos.

*"O único lugar onde o sucesso vem antes do trabalho é no dicionário."*

Albert Einstein

## 2. Resumo

O presente trabalho de projeto tem como objetivo descrever o desenvolvimento de uma aplicação usando o conceito responsivo para uma empresa distribuidora de bebidas, que permite ao utilizador aceder ao sistema através dos mais diversos tipos de equipamentos desde que o mesmo disponha de uma ligação à internet. Foi criada uma aplicação inovadora capaz de oferecer uma boa experiência ao utilizador com o desenvolvimento de várias funcionalidades e interfaces (para *desktop*, *smartphone*, *tablets*). Do mesmo modo se pretende apresentar um breve conceito sobre a metodologia do desenho responsivo para *a web*, os princípios de usabilidade, as tendências tecnológicas para o desenvolvimento *web*, o desenvolvimento de sistemas *web* acoplados com o modelo de desenvolvimento *MVC*, bem como aspetos de segurança da aplicação *web*. Esta aplicação foi desenvolvida em várias fases. Na primeira fase foi feito um estudo sobre aplicações relacionadas com a aplicação desenvolvida. Na segunda fase foi feita a análise de requisitos que incluiu a descrição dos serviços disponíveis, os requisitos funcionais, os casos de uso e os esboços. Na terceira fase foi feita a implementação da aplicação, que está dividida em três camadas (dados, apresentação e funcionalidades da aplicação). Neste processo de desenvolvimento foi usada a linguagem de programação *JAVA* - uma das linguagens de programação mais utilizadas na indústria de desenvolvimento de *software* [1] - bem como a *framework spring boot* e *spring* com o padrão *MVC*. Na última fase foi feita a implementação, avaliação e testes da aplicação.

A metodologia das tecnologias deste projeto tem sido muito utilizada devido ao seu aspeto ágil e responsivo, tornando estas funcionalidades indispensáveis para os mais diversos dispositivos de comunicação. Os aspetos de segurança também foram considerados através da *framework spring security*.

**Palavras-chave:** Desenvolvimento de *Software*, Sistemas *Web*, *Web* Responsivo, Segurança.

### 3. Abstract

This project aims to describe the development of a project using the responsive concept. This project includes a web application for monitoring the beverage industry, which allows the user to access the diverse types of equipment as long as it has an internet connection. An innovative application was created capable of offering a good user experience with the development of various functionalities and interfaces (for desktop, smartphone, tablets). Likewise, it is intended to present a brief concept on the methodology of responsive design for the *web*, usability principles, technological trends for *web* development, the development of web systems coupled with the MVC development model, as well as security aspects of the web application. This application was developed in several phases. In the first phase, a study was made on applications similar to the developed application. In the second phase, the requirements analysis was carried out, which included the description of the services available, the functional requirements, the use cases, and the sketches. In the third phase, the application was implemented, which is divided into three layers (data, presentation, and features of the application). In this development process, the JAVA programming language was used - one of the most used programming languages in the software development industry [1] as well as the use of spring boot and spring framework with the MVC standard. In the last phase, the application was implemented, evaluated, and tested.

The technology methodology of this project has been widely used due to its agile and responsive aspect, making these functionalities indispensable for the diverse communication devices. Security aspects were also considered through the spring security framework.

**Keywords:** Software Development, Web Systems, Responsive Web, Security.

## 4. Índice

<b>1. Agradecimentos</b> .....	3
<b>2. Resumo</b> .....	5
<b>3. Abstract</b> .....	6
<b>4. Índice</b> .....	7
<b>5. Lista de Tabelas</b> .....	10
<b>6. Lista de Figuras</b> .....	11
<b>7. Lista de Siglas e Acrónimos</b> .....	13
<b>8. Introdução</b> .....	14
8.1 Contextualização .....	14
8.2 Motivação .....	15
8.2 Objetivos .....	15
8.3 Organização do documento .....	16
8.5 Contribuição e Metodologia .....	17
<b>9. Revisão da Literatura</b> .....	20
9.1 <i>Design Web Responsivo</i> .....	20
9.2 <i>Responsive Web Development (RWD) versus Adaptive Web Development (AWD)</i> .....	21
9.3 Princípios de Usabilidade .....	24
9.4 Tecnologias e Tendências dos Sistemas <i>Web</i> .....	26
9.5 Desenvolvimento de Sistemas <i>Web</i> .....	28
9.6 Modelo de desenvolvimento <i>MVC</i> .....	30
9.7 <i>Frameworks Front-End e Back-End</i> .....	31
9.8 Estudo de Trabalho Relacionado .....	32
<b>10. Conceito do Projeto</b> .....	36
10.2 <i>Back-End Frameworks</i> .....	36
10.2.1 <i>Eclipse, IDE</i> .....	36

10.2.2	<i>Java</i>	37
10.2.3	<i>Spring MVC</i>	37
10.2.4	<i>Apache Maven</i>	38
10.2.5	<i>Apache Tomcat</i>	39
10.2.6	<i>Spring Security</i>	40
10.2.7	<i>JPA, Hibernate e Flyway</i>	42
10.2.8	<i>Spring Data JPA</i>	43
10.2.9	Base de Dados <i>MySQL</i>	43
<b>10.3</b>	<b><i>Front-End Frameworks</i></b>	<b>43</b>
10.3.1	<i>HTML - Hypertext Markup Language</i>	44
10.3.2	<i>CSS - Cascading Style Sheets</i>	44
10.3.3	<i>Bootstrap</i>	45
10.3.4	<i>Java Script</i>	45
10.3.5	<i>Thymeleaf</i>	46
<b>11.</b>	<b>Análise e Especificação dos Requisitos do Projeto</b>	<b>47</b>
11.1	Requisitos Funcionais	47
11.2	Requisitos Não Funcionais	49
11.3	Diagrama de Classes	50
11.4	Diagrama de Casos de Uso	52
11.5	Diagrama do Processo de Desenvolvimento do Sistema	53
<b>12.</b>	<b>Implementação do Projeto</b>	<b>55</b>
12.2	Arquitetura do Projeto	55
12.2.1	<i>Database Package Migration Tool Flyway</i>	56
12.2.2	<i>Package Resources JavaScript</i>	58
12.2.3	Segurança da Aplicação	62
<b>13.</b>	<b>Implantação da Aplicação na Cloud</b>	<b>68</b>
13.1	<i>Heroku Plataforma Cloud</i>	68



13.2 Teste da Aplicação <i>Web</i> na <i>Cloud</i> .....	72
<b>14. Conclusão</b> .....	80
14.1 Considerações Finais.....	80
14.2 Limitações do Projeto.....	81
14.3 Trabalho futuro.....	81
<b>15. Bibliografia</b> .....	83
<b>Apêndice A</b> .....	87

## 5. Lista de Tabelas

Tabela 1 Comparação de funcionalidades .....	34
Tabela 2 Requisitos funcionais do sistema - Fonte: Autor .....	47
Tabela 3 Caso de uso login no sistema - Fonte: Autor .....	87
Tabela 4 Caso de uso pesquisar cliente - Fonte: Autor.....	87
Tabela 5 Caso de uso registo cliente - Fonte: Autor.....	88
Tabela 6 Caso de uso alterar registo cliente - Fonte: Autor.....	88
Tabela 7 Caso de uso excluir registo cliente - Fonte: Autor.....	89
Tabela 8 Caso de uso pesquisar produto - Fonte: Autor.....	89
Tabela 9 Caso de uso registar produto - Fonte: Autor.....	90
Tabela 10 Caso de uso alterar registo produto - Fonte: Autor.....	90
Tabela 11 Caso de uso excluir produto - Fonte: Autor.....	91
Tabela 12 Caso de uso pesquisar cidade - Fonte: Autor.....	91
Tabela 13 Caso de uso registar cidade - Fonte: Autor.....	92
Tabela 14 Caso de uso alterar registo cidade - Fonte: Autor.....	92
Tabela 15 Caso de uso pesquisar vendas - Fonte: Autor .....	93
Tabela 16 Caso de uso registar vendas - Fonte: Autor .....	94
Tabela 17 Caso de uso alterar venda - Fonte: Autor.....	95
Tabela 18 Caso de uso cancelar venda - Fonte: Autor.....	96
Tabela 19 Caso de uso gerar relatório venda - Fonte: Autor .....	96
Tabela 20 Caso de uso pesquisar estilo - Fonte: Autor.....	97
Tabela 21 Caso de uso registo estilo - Fonte: Autor.....	97
Tabela 22 Caso de uso alterar registo estilo - Fonte: Autor.....	98
Tabela 23 Caso de uso excluir estilo - Fonte: Autor.....	98

## 6. Lista de Figuras

Figura 1 Padrão <i>MVC</i> - modelo visão e controlo - Fonte: Autor.....	31
Figura 2 Página administrativa do <i>apache-tomcat</i> - Fonte: <i>localhost server apache</i> .....	39
Figura 3 Configuração <i>localhost</i> - página da aplicação - Fonte: Autor.....	40
Figura 4 Diagrama de classe do projeto - Fonte: Autor.....	50
Figura 5 Diagrama de caso de uso - Fonte: Autor.....	52
Figura 6 Diagrama do processo de desenvolvimento do projeto - Fonte: Autor.....	53
Figura 7 Módulos da aplicação - Fonte: Autor.....	56
Figura 8 Base de dados do projeto - Fonte: Autor.....	57
Figura 9 Tabelas - cerveja e cidade.....	57
Figura 10 Tabelas venda e cliente.....	58
Figura 11 Package <i>resources</i> - <i>javascript</i> - Fonte: Autor.....	59
Figura 12 Ficheiro produto.js código - <i>javascript</i> - Fonte: Autor.....	60
Figura 13 Ficheiro produto.html - Evento - Fonte: Autor.....	60
Figura 14 Evento de sucesso - excluir - Fonte: Autor.....	61
Figura 15 Formulário de <i>login</i> - Fonte: Autor.....	62
Figura 16 Dependências de segurança do <i>spring framework</i> - Fonte: Autor.....	63
Figura 17 Base de dados - consulta utilizador - Fonte: Autor.....	64
Figura 18 Configuração de segurança - Fonte: Autor.....	64
Figura 19 Segurança no <i>layout</i> padrão - Fonte: Autor.....	66
Figura 20 Função de segurança <i>javascript</i> - Fonte: Autor.....	67
Figura 21 Código fonte da página de <i>login</i> - Fonte: Autor.....	67
Figura 22 Conexão plataforma - <i>heroku</i> - Fonte: Autor.....	68
Figura 23 Criando base de dados - <i>heroku</i> - Fonte: Autor.....	69
Figura 24 Preparação projeto envio - <i>cloud heroku</i> - Fonte: Autor.....	70
Figura 25 Implantação - <i>cloud heroku</i> - Fonte: Autor.....	71
Figura 26 Teste de usabilidade - Fonte: Autor.....	72
Figura 27 Aplicação funcionando - <i>cloud heroku</i> - Fonte: Autor.....	73
Figura 28 Página de <i>login</i> do sistema para <i>iPad</i> - Fonte: Autor.....	74
Figura 29 Página de <i>login</i> do sistema <i>smartphone</i> - Fonte: Autor.....	75
Figura 30 <i>Dashboard</i> do sistema no dispositivo <i>desktop</i> - Fonte: Autor.....	76
Figura 31 <i>Dashboard</i> do sistema no dispositivo <i>iPad</i> . - Fonte: Autor.....	77

Figura 32 *Dashboard* do sistema no dispositivo *iPad* - Fonte: Autor ..... 78

Figura 33 *Dashboard* do sistema no dispositivo *smartphone*- Fonte: Autor..... 79

## 7. Lista de Siglas e Acrónimos

<i>API</i>	<i>Application Programming Interface</i>
<i>CRUD</i>	<i>Create Read Update and Delete</i>
<i>CSS</i>	<i>Cascading Style Sheet</i>
<i>DAO</i>	<i>Data Access Object</i>
<i>DB</i>	<i>Data Base (Base de Dados)</i>
<i>DCI</i>	<i>Contexts and Dependency Injection</i>
<i>DDL</i>	<i>Data Definition Language</i>
<i>FRAMEWORK</i>	Conjunto de componentes com um objetivo específico
<i>HTML</i>	<i>HyperText Markup Language</i>
<i>HTTP</i>	<i>Hypertext Transfer Protocol</i>
<i>HTTPS</i>	<i>Hypertext Transfer Protocol Secure</i>
<i>IDE</i>	<i>Integrated Development Environment</i>
<i>IT</i>	<i>Information Technology</i>
<i>JAVA</i>	<i>Linguagem de Programação JAVA</i>
<i>JDK</i>	<i>Java Development Kit</i>
<i>JEE</i>	<i>Java Platform, Enterprise Edition</i>
<i>JSF</i>	<i>Java Server Faces</i>
<i>JPA</i>	<i>Java Persistence API é o Padrão de persistência de dados no Java.</i>
<i>LDAP</i>	<i>Lightweight Directory Access Protocol</i>
<i>MVC</i>	<i>Model View Controller</i>
<i>ORM</i>	<i>Object-relational mapping</i>
<i>POO</i>	Programação Orientada a Objetos
<i>POM</i>	<i>Project Object Model</i>
<i>SGDB</i>	Sistema de Gestão de Base de Dados
<i>SKU</i>	<i>Stock Keeping Unit</i>
<i>UI</i>	<i>User Interface</i>
<i>UML</i>	<i>Unified Modeling Language</i>
<i>URL</i>	<i>Uniform Resource Locator</i>
<i>WHATWG</i>	<i>Web Hypertext Application Technology Working Group</i>
<i>W3C</i>	<i>World Wide Web</i>
<i>XML</i>	<i>Extensible Markup Language</i>

## 8. Introdução

Neste capítulo é descrita a contextualização e a motivação para a realização deste Projeto de Mestrado, bem como quais os objetivos a atingir e a organização adotada para a escrita do documento.

Apesar de existirem tecnologias aparentemente semelhantes, a aplicação *web* responsiva apresenta a vantagem que consiste no seu uso independente da versão do sistema operativo que está sendo utilizado. Este projeto compreende o desenvolvimento de uma aplicação *web*, para gerir os processos de uma empresa no âmbito da distribuição de bebidas usando várias soluções tecnológicas, com o objetivo de se criar uma solução digital flexível observando os princípios da usabilidade, conforme seguidamente descrito.

### 8.1 Contextualização

As empresas de distribuições de bebidas são divididas em três segmentos de atividades económicas segundo o banco de Portugal [2] a que correspondem as seguintes características:

**Vinho:** Inclui as atividades de produção de vinho e bebidas alcoólicas, destiladas e não destiladas;

**Cerveja:** Envolve a produção de cidra, cerveja e malte, fabricação de cidra e outras bebidas fermentadas de frutos;

**Refrigerantes e águas:** Produção de refrigerantes, de águas minerais naturais e de outras águas engarrafadas;

Para a gestão da produção dos seus produtos tem sido necessário o uso de tecnologias. Um dos problemas com que as empresas se confrontam, relaciona-se com os pedidos de entrega que, ao serem realizados nos pontos de vendas independentes, têm de ser, em tempo real, do conhecimento do departamento administrativo. Os vendedores, através da aplicação *web*, enviam os pedidos dos produtos, sejam eles diários ou semanais ou até mesmo mensais, para que a empresa se consiga aperceber do volume de produtos a fabricar.

Outra característica interessante é que, a partir do pedido de venda realizada, a empresa consegue também resolver um outro problema que é o controlo das suas matérias-primas para produção, permitindo deste modo gerir as necessidades de acordo com o armazém da distribuidora. Neste contexto, a empresa necessita também saber quais são os produtos que mais

saem para a produção, e cuja informação é proporcionada através da gestão de vendas da aplicação *web*, além do controlo dos fretes envolvidos na venda destas bebidas.

## 8.2 Motivação

Este projeto destina-se a facilitar o trabalho da gestão de um sistema de distribuição de bebidas fora ou dentro de um escritório ou loja física, realizando vendas *on-line*, independentes do local em que esteja o utilizador. Este facto gera facilidade e comodidade para quem está a gerir a aplicação *web* e também segurança porque a aplicação está a correr a partir de uma ligação via *cloud* que pode ser definida de acordo com o interesse da empresa.

A motivação surgiu, assim, de criar uma solução no modelo de uma aplicação *web*, não nativa nem híbrida, mas que pudesse funcionar nos mais diversos dispositivos, fossem eles *smartphone*, *tablets* ou *desktops*, usando uma metodologia tecnológica de desenvolvimento para aplicação *web* mais rápida no seu desenvolvimento, com custos menores, em comparação com soluções nativas e sem estar vinculada através de uma *play store* da *google* ou *app store* da *apple* nem condicionada à versão do sistema operativo como é o caso das soluções nativas.

Com base no resultado da análise e investigação efetuadas, foi criada a aplicação objeto deste projeto para informatizar os processos de vendas, registos de pessoas, geração de relatórios, consultas de informação e vendas e demais funcionalidades e requisitos característicos do âmbito do comércio e distribuição de bebidas.

## 8.2 Objetivos

São três os principais objetivos para a realização deste projeto de mestrado. O primeiro é o desenho e desenvolvimento de uma solução que contemple todo o ciclo de vida de uma aplicação *web* responsiva com a informatização dos processos inerentes, tais como: registo geral envolvendo produtos e pessoas, consultas sobre produtos, vendas e pessoas, gestão de vendas *on-line* e geração de relatórios.

O segundo objetivo é desenvolver a ligação entre o sistema e a camada de dados, usando na sua arquitetura uma das bases de dados usadas na indústria de tecnologia de informação.

O terceiro objetivo é desenvolver a parte de segurança da aplicação usando a tecnologia *framework Spring Security* para a autenticação e autorização dos utilizadores por intermédio de

*login e password*, sendo este o processo de segurança da aplicação para o âmbito e nível do acesso e sobretudo para proteção da aplicação de ataques cibernéticos.

No âmbito do desenvolvimento de *software* existem diversas tendências tecnológicas e metodologias para o desenvolvimento. Neste contexto pretende-se apresentar um breve conceito sobre a metodologia do *design web* responsivo, os princípios de usabilidade, as tendências tecnológicas para o desenvolvimento *web* e o desenvolvimento de sistemas *web* acoplados com o modelo de desenvolvimento *MVC*.

### 8.3 Organização do documento

Este documento possui a seguinte organização, como segue abaixo:

- **Capítulo 9 (Revisão da Literatura):** Neste capítulo apresenta-se a revisão literária sobre os princípios do *design web* responsivo, *responsive web development (RWD vs Adaptive Web Development (AWD))*, princípios de usabilidade, tecnologia e tendência dos sistemas *web*, desenvolvimento de sistemas *web*, modelo de desenvolvimento *MVC*, *frameworks front-end e back-end* e estudos sobre aplicações semelhantes ao projeto desenvolvido.
- **Capítulo 10 (Conceito do Projeto):** Neste capítulo são apresentados os requisitos do projeto desde o estado inicial, abrangendo todas as ferramentas e linguagens de programação no âmbito do *back-end frameworks e front-end frameworks*, e também as soluções tecnológicas usadas na aplicação.
- **Capítulo 11 (Análise e especificações dos requisitos do trabalho de projeto):** Neste capítulo é apresentada a análise sobre a forma de requisitos do projeto, funcionais e não funcionais, incluindo os diferentes diagramas de casos de uso, diagrama de classes e diagrama do processo de desenvolvimento do projeto, bem como cada especificidade do projeto.
- **Capítulo 12 (Implementação, Arquitetura e Desenvolvimento):** Neste capítulo descreve-se os diferentes estados do projeto, desde a implementação, arquitetura e por fim o seu desenvolvimento até à sua conclusão.



- **Capítulo 13 (Publicação e Testes do Projeto):** Neste capítulo será apresentada a fase do empacotamento em formato *JAR* e preparação da aplicação para o envio para *cloud heroku*, ligação à plataforma *heroku* e a sua demonstração ao ser publicado na *cloud* e por fim os testes de usabilidade relativamente ao uso da aplicação *web* responsiva.
- **Capítulo 14 (Conclusão):** Neste capítulo serão apresentadas a conclusão, as considerações finais, as limitações do projeto e sugestão dos trabalhos futuros.

## 8.5 Contribuição e Metodologia

### 8.5.1 Contribuição

A contribuição deste Projeto de Mestrado é a seguinte:

- Desenvolvimento e implementação de uma aplicação *web* responsiva para a gestão de processos no setor da distribuição e venda de bebidas, de modo que o seu funcionamento seja permitido a partir de uma *cloud*, com mecanismo de segurança e controlo de acesso contra ataques cibernéticos.

Todo este trabalho de Projeto também permitiu ao candidato aplicar os seus conhecimentos provenientes da sua experiência profissional, bem como os temas investigados de princípios de *design* responsivo.

### 8.5.2 Metodologia

A técnica metodológica adotada neste projeto foi o *Design Science Research (DSR)*, opção dentro da *Design Science*, ou ciência do projeto [3] [4] que tem o objetivo de estudar, pesquisar e investigar o artefacto e o seu comportamento, tanto numa perspetiva académica como organizacional, criando conhecimento sobre soluções inovadoras para problemas do mundo real.

Este método tem sido muito utilizado no contexto da tecnologia da informação, em projetos envolvendo engenharia elétrica, engenharia de *software* e ciência da computação [5]. Esta abordagem orientada à tecnologia de informação tem como objetivo o desenvolvimento de artefactos que permitam a criação de soluções satisfatórias para alcançar determinados resultados, resolvendo um problema de carácter prático, com um foco em pesquisas literárias

dirigidas ao projeto e que apoiem melhores soluções para os problemas práticos existentes [6]. O *DSR* define um ciclo de etapas para o desenvolvimento da solução abaixo descritas [5]:

- **Consciencialização ou investigação do problema:** Nesta etapa procede-se à compreensão da problemática envolvida, com a definição e formalização do problema a ser resolvido, os seus limites e quais as soluções adequadas. Assim sendo, em relação a este projeto, investigou-se as características e problemáticas de uma pequena e média empresa do setor da distribuição de bebidas que, na sua atividade comercial, obriga o utilizador, muitas vezes, por estar fora da sua empresa, a utilizar um sistema que necessariamente tem que ter comunicação em tempo real com o departamento de administração. O custo do desenvolvimento também seria um elemento importante a considerar.
- **Sugestão ou definição dos resultados esperados:** Nesta etapa considera-se o conjunto de possíveis artefactos e sugere-se a escolha de um, ou mais, a ser usado. Em relação à aplicação a ser desenvolvida, considerou-se: as soluções *web* nativas dependem da versão do sistema operativo do *hardware*, o que restringe o utilizador a um tipo específico de tecnologia, de *hardware*; as soluções híbridas, por outro lado, não estão condicionadas à versão do sistema operativo, mas dependem de um repositório, ou seja, através da *play store* da *google* ou *app store* da *apple*, é que se pode aceder a esta solução. Verificou-se assim a necessidade de uma solução *web* que não fosse nativa e/ou híbrida, que não dependesse na íntegra de um sistema operativo, e que nem estivesse condicionada a uma loja de repositório, ou seja, através da *play store* da *google* ou *app store* da *apple*, que permitisse fazer a gestão da atividade comercial para empresas no setor de bebidas.
- **Desenvolvimento da solução:** Nesta etapa procede-se à construção do artefacto em si. Mediante as investigações e sugestões das etapas anteriores, procedeu-se nesta fase ao desenvolvimento de uma solução *web* responsiva, atendendo a que a responsividade permite uma *interface* amigável ao dispositivo de utilização, sobretudo por ser uma solução *web*, que permite o seu acesso em qualquer lugar, e em qualquer dispositivo, desde que tenha acesso à internet.

- **Demonstração:** Nesta etapa demonstra-se o comportamento do artefacto desenvolvido no ambiente para o qual foi projetado. A aplicação desenvolvida pode ser apresentada já em funcionamento, com base nas especificações e requisitos definidos, onde se pode verificar todo o escopo e arquitetura do projeto.
- **Validação e Conclusão:** Nesta etapa verifica-se e avalia-se a validade do artefacto desenvolvido, através de procedimentos, testes e mecanismos de medição de resultados. Com base nos objetivos deste projeto, a aplicação desenvolvida foi testada em vários dispositivos de acesso à internet, e observou-se que o seu comportamento satisfaz os critérios de usabilidade e responsividade.

## 9. Revisão da Literatura

Neste capítulo veremos o estado da arte para o desenvolvimento de uma aplicação *web* responsiva bem como um estudo sobre outras aplicações *web* existentes.

### 9.1 *Design Web Responsivo*

Os desenvolvedores aplicativos em geral, e os de *sites web* em particular, têm atualmente de lidar com o crescente nível de novas tecnologias e a diversidade de características de *interfaces*. O *design* e desenvolvimento responsivo é uma tendência e um processo crucial no desenvolvimento de um site ou sistema *web de* comércio eletrônico, pois envolve a organização do conteúdo em modelos gráficos que podem ser usados com base para a codificação dos diversos tipos de aplicações *web* [7] satisfazendo a diversidade de dispositivos usados para a navegação da *web*. Contudo, a aplicação de *design* e desenvolvimento responsivo aos sites *web* existentes, envolve frequentemente uma importante reengenharia devido ao inerente conceito de *fluid grid*, que permite usar grade fluida para especificar o tipo de interface de *desktops*, *smartphones*, *tablets* e até mesmo televisor com a dimensão baseada em percentagem e não por pixels; este recurso permite aos desenvolvedores ajustar os *layouts* a dispositivos maiores sem perder a responsividade nos ecrãs de grande resolução.

Do mesmo modo, as aplicações de *design* responsivo estão atualmente limitadas à adaptação de *desktop-para-móvel* [8]. Embora existam várias outras opções para desenvolvimento de soluções *web*, como sejam as soluções nativas e soluções híbridas, a nova tendência é, no entanto, um *design* e desenvolvimento responsivo da *web* o que significa construir o *layout* da interface *web* em *fluid grids* que podem dinamicamente adaptar-se a diversos ambientes de visualização [9] [10].

As *frameworks* e bibliotecas inovadoras da *web* permitem que os produtos de *software* evoluam continuamente [11]. Recentemente, os *designers* de interfaces de computador e unidades móveis, têm tentado fornecer aos utilizadores navegações da *web* de qualidade embora não tenham tido a possibilidade de satisfazer adequadamente as necessidades dos utilizadores expostos aos *layouts* tradicionais dos sites *web*. Verifica-se, portanto, a necessidade de mudar para um *design* da *web* responsivo, capaz de se remodelar dependendo nas várias dimensões e resoluções dos ecrãs desde as unidades de *desktop* até às unidades móveis [11] ajudando a ir de encontro às expectativas criadas na cultura da informação móvel [12] [13].

Para além do conceito de *fluid grids*, as *media queries* também são utilizadas na aplicação do estilo *CSS* padronizando de acordo com a característica de cada dispositivo onde a página será compilada, ou seja, as *media queries* permitem ajustar o *layout* em diferentes dispositivos sem ter que mudar o conteúdo da página, mantendo o mesmo código *HTML*, através das condições que por intermédio das ditas *media queries*, todas as condições estabelecidas serão aplicadas a nível do *CSS* no dispositivo a depender das suas configurações de visualização. As *media queries* também são padrões no *CSS3* (uma versão mais recente da especificação *CSS*) e que possuem compatibilidades com os navegadores mais utilizados a exemplo de:

1. *Google Chrome*
2. *Firefox*
3. *Opera*
4. *Internet Explorer*

## **9.2 Responsive Web Development (RWD) versus Adaptive Web Development (AWD)**

Nos últimos anos tem vindo a surgir cada vez com maior frequência um conjunto de novas tecnologias, dispositivos, e metodologias *web*. No entanto, no início da década de 2010 e devido ao surgimento dos mais diversos tipos de dispositivos, como os *smartphones*, *smartwatch*, minicomputadores e outros, surgiu a necessidade de se criar uma ideologia que permitisse aos engenheiros de *software* criar algo que se pudesse adaptar e também fosse elegível em todos os diversos tipos de resoluções de ecrãs ou dispositivos.

Este conceito foi, inicialmente, desenvolvido pelo *web designer* Ethan Marcotte [10], nos inícios do ano 2011. Marcotte definiu que um *site* ou uma aplicação *web* deveria ter um *layout* que se conseguisse adaptar a qualquer situação, ou seja com um *layout* fluido e que as imagens fossem de tamanhos flexíveis, isto com o objetivo que o utilizador pudesse usufruir de uma melhor e única experiência independentemente do dispositivo por onde estava a aceder [10]. Antes do aparecimento do *Responsive Web Design (RWD)* era necessário criar diversas versões do mesmo *site*, cada uma específica para cada tamanho de dispositivo, na semelhança da metodologia do desenvolvimento *Adaptive Web Design (AWD)* [14]. Esta não seria a melhor abordagem pois era difícil acompanhar a evolução tecnológica e o constante aparecimento de novos aparelhos e novas resoluções.

Além da resolução dependente do tamanho do ecrã, existe também o problema, a nível móvel, de ser possível alterar a orientação com que se vê cada *site*, ou seja, se horizontal ou verticalmente. No caso do *desktop*, nem sempre a página *web* é vista na totalidade do ecrã, vários são os casos em que os utilizadores têm a janela minimizada, tornando impossível depender das resoluções atualmente disponíveis.

A solução então apresentada por Ethan Marcotte e Matthew Harris [10] foi a criação de uma metodologia de *design* que seguisse os dois métodos para os desenvolvedores, para que um *site web*, fosse ele de comércio eletrónico ou um portal de notícias, fosse responsivo.

### *Responsive Web Design (RWD)*

A metodologia de *Responsive Web Design*, é uma abordagem ao *design* dos *sites* na *web*, que visa tornar agradável a experiência no contexto da visualização dos conteúdos digitais independente do tipo de dispositivo em que o utilizador o esteja a usar. Esta experiência de visualização ideal e de fácil leitura e navegação exige o mínimo de redimensionamento, melhor panorâmica do conteúdo e menor *scrolling*, tudo isto para os mais diversos dispositivos móveis, desde *desktops*, *smartphones* e *tablets* [14] [15] O uso do termo *RWD* não é uma tecnologia separada, mas um conjunto de práticas das quais se apresenta pelo menos três técnicas:

1. *Layouts* de grelha fluida: compreende uma combinação de valores relativos a um sistema de grelha em consultas de *media*; desta forma a organização dos conteúdos envolve de uma maneira consistente e sobretudo flexível, o suficiente para redimensionar de modo dinâmico a visualização em qualquer ecrã.
2. A segunda técnica é o conceito de imagens fluidas: esta técnica determina que as dimensões das unidades sejam relativas; desta forma os elementos dentro do contexto *HTML* não ficam fora, perdendo assim a sua formatação.
3. O terceiro componente principal é a utilização de *media queries*: a consulta a *media* permite a mudança e ou alteração de *layout* via *JavaScript* dentro do *CSS*; assim sendo, o *layout* pode ser alterado; em vez de ter *layout* para tamanhos de ecrã diferentes, todo o conteúdo é reposicionado para uma dimensão que está a depender da necessidade do ecrã.

## *Adaptive Web Design (AWD)*

Já esta metodologia do *Adaptive Web Design* difere em muito do modelo *RWD*, visto que não existe um *layout* que sempre muda automaticamente de acordo com a especificidade do dispositivo, mas, ao contrário, existem diversos layouts distintos a depender de cada configuração de ecrã, ou seja, cada dispositivo depende que seja confeccionado um *layout* de acordo com a configuração do respetivo ecrã; tipicamente todas as aplicações que utilizam a metodologia *AWD* necessariamente precisam de produzir *layouts* específicos para cada dispositivo, sejam eles *desktops*, *smartphones* ou *tablets*.

### Comparação entre o modelo *RWD* e *AWD*

- A escolha pelo modelo de desenvolvimento *RWD* é mais difícil de realizar, pois exige mais atenção e domínio no *CSS* bem como a maneira de como organizar os conteúdos *CSS* de forma a garantir que funcione com flexibilidade adaptável em qualquer tamanho possível dos ecrãs.
- A metodologia *AWD* é menos flexível, não o suficiente para continuar funcionando bem por conta própria, enquanto novos dispositivos sejam lançados no mercado tecnológico, ou seja, enquanto no *RWD* o *layout* é decidido pelo navegador do utilizador, por outro lado no *AWD* é pré-configurado no *back-end*, de modo que a interface é exclusiva a cada modelo de dispositivo; no servidor compete detetar o tipo de dispositivo, para deste modo enviar à camada de interface, o *layout*, de acordo com o dispositivo em uso; em resumo, é necessário ter várias versões de *layout* para cada dispositivo e no caso do surgimento de um dispositivo com dimensões de ecrã diferenciadas, uma nova interface terá de ser criada para a aplicação em uso.
- Enquanto os *websites AWD* implicam que seja carregado do lado do servidor todos os *layouts* que se encontram disponíveis, por outro lado em relação ao *RWD* basta apenas carregar um ficheiro *layout* no servidor, o que torna o modelo de *websites AWD* inviável quando o problema identificado é a grande disponibilidade no negócio.

### 9.3 Princípios de Usabilidade

Com o crescimento de novas tecnologias de informação e o acesso a novas soluções *web*, a usabilidade tornou-se condição necessária para quaisquer *websites* ou sistemas *web*. Cada vez mais se observa como tem sido comum a necessidade da usabilidade, notando-se que este novo conceito, no ambiente tecnológico, tem sido alvo de interesse nos mais diversos âmbitos e áreas de negócios, seja entre os *sites* de *marketing online* no contexto do *web designer* ou com qualquer ligação entre os demais diversos tipos de *sites* e sistemas *web*. Em todos estes meios, o assunto usabilidade tem sido discutido e debatido, dado a sua relevância para o momento atual, considerando o valor que se faz no contexto para sistemas ou *sites web* como um componente fundamental indispensável na experiência do utilizador.

O utilizador desempenha um papel muito relevante no sucesso de qualquer solução *web*, seja uma aplicação *web*, aplicação nativa ou aplicação híbrida; os utilizadores tornaram-se exigentes e espera-se que a interface das plataformas de utilização usual esteja de acordo com os requisitos dos seus respetivos dispositivos de comunicação e interação tecnológica [14] [16]. Neste contexto, nota-se que existem muitas aplicações baseados na *web*, disponíveis para diversos públicos-alvo, visando satisfazer as necessidades múltiplas, como no ramo da comunicação, do entretenimento, comércio eletrónico ou outras situações. Nota-se, contudo, vários tipos de diferentes problemas de usabilidade, competindo aos desenvolvedores avaliar novos métodos de engenharia de *software* de forma a contemplar os aspetos da usabilidade, pois permite medir a facilidade de compreensão e a qualidade de interação dos utilizadores. Em 1990 foi pela primeira vez demonstrado por Donald Norman, o conceito de usabilidade através do resultado dos seus estudos. A usabilidade envolve várias partes dentro do contexto da tecnologia, estando associada ou definida por cinco componentes de qualidade [17]:

- 1. Capacidade de Aprendizagem:** dos atributos descritos no conceito de usabilidade, este é pontuado como sendo o principal, visto que se o utilizador não consegue aprender ou compreender, de nada adianta uma determinada solução *web*, por exemplo, ter tantas funcionalidades; a usabilidade é o meio pelo qual os utilizadores podem aprender a interface da solução proposta e em conjunto com a funcionalidade de uma solução *web* são os dois fatores denominados componentes de qualidade que devem gerar satisfação plena no consumo de um serviço *web*.



2. **Eficiência no Uso:** Jakob Nielsen [17] descreve a eficiência como sendo a quantidade de recursos necessários para que o *site* ou sistema *web* providencie a possibilidade ao utilizador de alcançar os seus objetivos finais, tornando-se um grande desafio para os engenheiros de *software* e profissionais *web designer* em que no planeamento de uma solução *web* deverão pensar sempre no consumidor final destes serviços, não bastando, apenas, a beleza no *design*, mas sim que a solução disponível ao utilizador possua uma *interface* agradável e também eficaz para o uso.
3. **Memorização:** destaca a proficiência dos utilizadores que deixam de aceder por um tempo a determinado serviço através de uma solução *web*, mas quando retomado o ambiente segundo o conceito apresentado por Jakob Nielsen [17], restabelece proficiência com facilidade, tendo em vista que uma solução *web* que inclui no seu conceito base estes princípios, permite facilitar uma interação acelerada entre o utilizador e as funcionalidades que compõem a dita solução *web*.
4. **Erros:** No planeamento de um projeto é preciso ter em mente que o utilizador ao gerir a aplicação poderá cometer erros no registar das informações. Neste contexto, como componente de qualidade do *software*, torna-se necessário uma interação por parte da solução proposta com o utilizador, de modo a que este perceba os erros cometidos e que, em resposta a estes erros, a aplicação não responda com erros genéricos de código aleatório [17].
5. **Satisfação Subjetiva:** Este componente de qualidade implica no prazer ou emoção que as plataformas *webs*, *websites*, de redes sociais, comércio eletrónico, sistemas corporativos, sistemas comerciais, portais de notícias, entre outros diversos tipos de sistemas, podem proporcionar ao utilizador no consumo destas soluções *web* para os seus objetivos finais uma vez que essa satisfação é resultado da utilidade da solução em uso, se esta resolver a sua necessidade real.

Comprendemos através dos estudos e análise de Jakob Nielsen [17], que estas características são elementos principais no processo *UX (User eXperience)*. A experiência do utilizador é um facto levado a sério por grandes empresas como *Google* e *Facebook*; elas trabalham criando suas soluções e ambiente, mas totalmente pensando nesta experiência do utilizador, facilitando e tornando dinâmica a interação do utilizador [7] [17]. As perspetivas do

utilizador são sempre levadas em conta, visto ser este o objetivo final quando se trata de criar soluções *web*, bem como a sua experiencia bem sucedida, o que tem levado os *designers* de *UX* a redefinirem as suas ideias sobre *designer*, refinando-as e testando-as com o objetivo do seu melhoramento através de várias iterações - prototipando uma solução antes do seu desenvolvimento permite ter uma visão geral de como será o resultado, compreendendo e identificando erros e problemas subtis de modo que cada interação entre o utilizador com a solução *web* a ser desenvolvida seja qualificada pela experiencia do utilizador final com alta usabilidade em cada passo do processo contido no *designer* da solução *web*.

#### 9.4 Tecnologias e Tendências dos Sistemas *Web*

Com a expansão dos dispositivos móveis, atualmente grande parte das empresas tem-se apercebido que a utilização de aplicações *web*, sejam nativas ou híbridas, é uma mais valia, facilitando a proximidade dos seus negócios aos clientes; esta transformação tecnológica tem sido significativa na maneira de como comunicar com o público-alvo, com impacto em pequenos, médio e grandes empreendedores, levando a pensar em soluções digitais que permitem integrar cada negócio junto ao cliente final [18] [19]. Neste contexto é preciso que as empresas apostem numa estratégia que lhes seja mais hábil, destacando-se algumas tendências e tecnologias:

- **Aplicações Nativas:** Destinadas especialmente para o sistema operativo seja ele *Android* ou *iOS*, sendo estes os mais vulgares no mercado de software, definição que tem que ser especificada antes da sua criação do projeto, dado que a aplicação está condicionada ao sistema operativo, ou seja, não funciona para outro ambiente que não seja aquele para o qual foi projetada. Este facto torna esta metodologia tecnológica mais usada dado que este modelo permite que a solução nativa aceda aos mais variados recursos do dispositivo, sejam eles microfone, câmara, gps, ou outras funcionalidades. Contudo, embora seja esta a atual tendência tecnológica, o preço para desenvolver este tipo de projeto é mais elevado em comparação com uma aplicação *web* que pode ser acedida apenas pelo navegador do utilizador; outro diferencial desta metodologia tecnológica de desenvolvimento é que também permite ao utilizador uma interação com a aplicação mais atrativa por ser nativa, específica para aquele sistema operativo do equipamento em uso.

- **Aplicações Híbridas:** Este modelo de solução *web*, segue em alguns aspetos a metodologia de desenvolvimento das soluções nativas. Ambas podem ser encontradas nas lojas de aplicações, seja através da *play store* da google ou *app store* da *apple* e etc. Do mesmo modo, esta metodologia tem semelhanças com o modelo de aplicações *web*, quanto ao seu desenvolvimento, nas linguagens de programação tal como *CSS*, *JavaScript* e *HTML*, e estilos de programação comuns na indústria de *software* para o âmbito *web* aplicacional. Esta tendência tecnológica para a metodologia no desenvolvimento de aplicações híbridas tem ganho muito espaço na indústria de *software* dado que o seu custo e tempo de desenvolvimento não é elevado em comparação com aplicações nativas, mas, de alguma forma, este modelo híbrido aplicacional está condicionado num repositório de lojas de aplicações tal como já referenciado acima.
- **Aplicações Web:** No caso de uma aplicação *web*, esta solução permite ser acedida através de um *browser*, não exigindo instalação no dispositivo seja ele *desktop*, *smartphone* ou *tablet*, nem sequer está condicionada à versão do sistema operativo do equipamento, e muito menos depende de estar num repositório tal como *play store* da google ou *app store* da *apple* etc.; esta metodologia de desenvolvimento tecnológico compreende o seu funcionamento em qualquer sistema operativo, embora este modelo não contemple a interação com os recursos do dispositivo móvel, tais como câmara, gps, microfone, etc., fator que impossibilita dinamizar a experiência do utilizador. Todavia, o seu desenvolvimento e ou produção, é mais rápido do que ambos os modelos de aplicações acima citados e o seu custo é menor em comparação com as aplicações nativas e híbridas.

Atendendo ao contexto tecnológico e às tendências atuais, cada vez mais estas modalidades de desenvolvimento tecnológico se têm aperfeiçoado, sejam nativas, híbridas ou aplicações, de modo que a metodologia escolhida contemple o seu uso para os serviços digitais através dos mais variados dispositivos, *smartphone*, *tablets*, *notebooks*, visando, sobretudo, permitir muito facilmente a portabilidade [7] [18]. Neste contexto as aplicações *web* têm continuado a aperfeiçoar o seu desenvolvimento, ou seja, uma solução *web* não é mais desenhada apenas para o contexto de equipamentos *desktop* (computadores de mesa), mas para atender aos mais diversos tipos de equipamentos, que é o que se chama de desenvolvimento *web* responsivo na comunidade da indústria de *software*; assim sendo, percebeu-se que ambos os modelos de

aplicações acima citados (nativas e híbridas), possuem vantagens e desvantagens, mas optou-se todavia por usar o modelo de aplicação *web* neste projeto.

## 9.5 Desenvolvimento de Sistemas *Web*

A necessidade de soluções *web* tem cada vez mais aumentado considerando as vantagens que a internet proporciona. Quando falamos de plataformas *web*, *websites*, redes sociais, comércio eletrônico, sistemas corporativos, sistemas comerciais, portais de notícias, entre outros diversos tipos de sistemas, em que dispensam a necessidade de instalar quaisquer programas em computadores ou dispositivos móveis ou até mesmo em computadores de uso doméstico, acedendo-se apenas através de um *browser* seja ele *Firefox*, *Chrome*, *Internet Explorer* ou *Safari*, dentre outros, permitindo aceder de qualquer lugar - providenciando comodidade e flexibilidade comum junto com as exigências dos utilizadores, tem exigido da indústria de *software* modelos inovadores e atrativos de forma que inclua todas as necessidades do utilizador final [19].

Todavia, o desenvolvimento de soluções *web* requer planeamento pois envolve várias fases e a sua estrutura depende da dimensão de várias ferramentas e tecnologias que deverão ser conhecidas e experimentadas pelas indústrias de *software*. Destacamos algumas vantagens no âmbito do desenvolvimento de uma solução *web* [20]:

1. **Backups:** A segurança dos dados tem-se tornado o interesse de todos. Muitas organizações ainda utilizam folhas de cálculo partilhadas, tal como o *excel*, ficheiros que embora fáceis de manipular e práticos no uso, são muito suscetíveis a erros na persistência dos dados e facilmente podem ser corrompidos. Daí a necessidade de se ter uma solução que, de facto, seja resiliente, robusta, ou seja, estas características podem ser tipificadas num sistema *web* quando no seu âmbito se prevê a realização automática de *backups*, possibilitando funcionalidades tais como o teste de recuperação de *backup* (*Backup Recovery Testing*), sem correr o risco de corrupção nas informações, visto se tratar de base de dados e não de cópias de ficheiros em servidores de ficheiros. Uma solução *web* é ainda mais relevante quando o propósito principal na organização é a segurança dos dados; um aspeto como a sensibilidade dos dados, é totalmente possível a sua preservação, quando a política de *backups* compreende toda a organização. Neste caso, o desenvolvimento de uma solução para a *web* permitirá garantir os serviços de *backup* com maior garantia, uma vez que os dados estarão num servidor na *cloud*.

- 2. Flexibilidade:** O que talvez mais se destaca, por assim dizer, quando tratamos de uma solução *web*, é, sem dúvida, a sua característica principal, a capacidade de personalização e flexibilidade tanto para as necessidades direcionadas ao utilizador final como a sua manutenção para o âmbito das suas configurações e da gestão de recursos e funcionalidades que, uma vez disponibilizadas, são feitas a todos ao mesmo tempo - dinâmica esta que facilita a entrega de novos recursos, implementações novas, e, neste caso, ainda mais quando no caso em questão se trata do crescimento da aplicação necessitando-se do melhoramento de recursos. Um sistema bem projetado permite a flexibilidade sem gerar constrangimento aos engenheiros de *software*. Nota-se que a flexibilidade no âmbito de um projeto de *software* compreende ter um código com uma arquitetura amigável e de fácil manutenção de modo que inclua a partilha de várias pessoas trabalhando nas suas novas funcionalidades.
- 3. Atualizações e Instalações:** Para além dos vários benefícios proporcionados pelo ambiente *web*, é importante enfatizar que um projeto de *software web* em muito facilita a sua manutenção, seja a correção de possíveis erros na aplicação, bem como quando há a necessidade de recorrentes atualizações; outra vantagem está na execução de atualizações, necessitando apenas que esta seja feita apenas no servidor, desobrigando a atualização por parte do utilizador final. Desta forma, também é levado em conta a redução dos custos e a mitigação de riscos - assegurada neste modelo.
- 4. Acessibilidade e Mobilidade:** no anseio de ser eficaz e de satisfazer as necessidades do utilizador no uso de uma solução *web*, o desenvolvimento de *software* para o ambiente *web* promove este objetivo, sem a necessidade de uma interação direta com o sistema operativo da máquina, seja um computador *desktop* ou dispositivo móvel, um sistema operativo *Microsoft Windows, Linux, Android* ou *iOS* entre outros; a acessibilidade tem como característica o acesso para todos, possibilitando que qualquer pessoa usufrua de uma determinada solução que foi projetada para o âmbito da *web*, desde que disponha de *smartphones, notebooks, tablets* ou computadores *desktop* com o acesso à internet.

Entre as diversas vantagens de desenvolver uma solução para *web*, e mediante o acima exposto, destaca-se a centralidade de seu funcionamento, permitindo uma manutenção ordenada

com atualizações apenas no servidor de hospedagem, não carecendo de um servidor físico que distribua tais atualizações por cada dispositivo que esteja a utilizar a solução [19] [21].

## 9.6 Modelo de desenvolvimento *MVC*

O *MVC* é uma sigla do termo inglês *Model* (modelo) *View* (visão) e *Controller* (controle) e é um padrão de desenvolvimento que permite a troca de informações entre a interface do utilizador e a camada de base de dados. Desta forma as respostas entre estas camadas *MVC* podem ser mais rápidas e dinâmicas; no caso do *spring framework* trata-se de uma plataforma que fornece condições de infraestrutura abrangentes para o desenvolvimento de aplicações.

Para o padrão do desenvolvimento de *software* com o *MVC* esta separação no Spring *MVC* acontece da seguinte forma: a requisição feita no *browser* pelo utilizador chega à aplicação através do *DispatcherServlet*, *Front Controller* no *Framework Spring MVC*; o *Front Controller* encaminha a requisição para a camada de *Controller* através do *Handler Mapping* que analisa qual a *URL* de acordo com o mapeamento para qual classe *Controller* na camada *Controller* deve ser direcionada. A camada *Controller* por sua vez vai, ao receber os dados, delegar quem é que vai trabalhar os dados e, neste caso, a camada *Model*, que é responsável por trabalhar os dados e fazer a regra de negócio como, por exemplo, salvar os dados na base de dados. A camada *Model* devolve ao *DispatcherServlet* camada *Controller* os resultados dos *inputs* recebidos, como, por exemplo, o resultado de ter guardado um registo na base de dados. Por fim, a camada *Controller* vai chamar a camada *View*, ou seja, um mecanismo de renderização / visualização transformando documentos e recursos numa página de internet em *HTML* com uma representação visual interativa no aparelho do utilizador final [22].

Este processo permite obter imagens digitais resultantes de modelos tridimensionais, por meio de *software* específico de maneira a disponibilizar ao cliente da requisição, as informações no navegador do utilizador utilizando o modelo de *frameworks Action Based*.

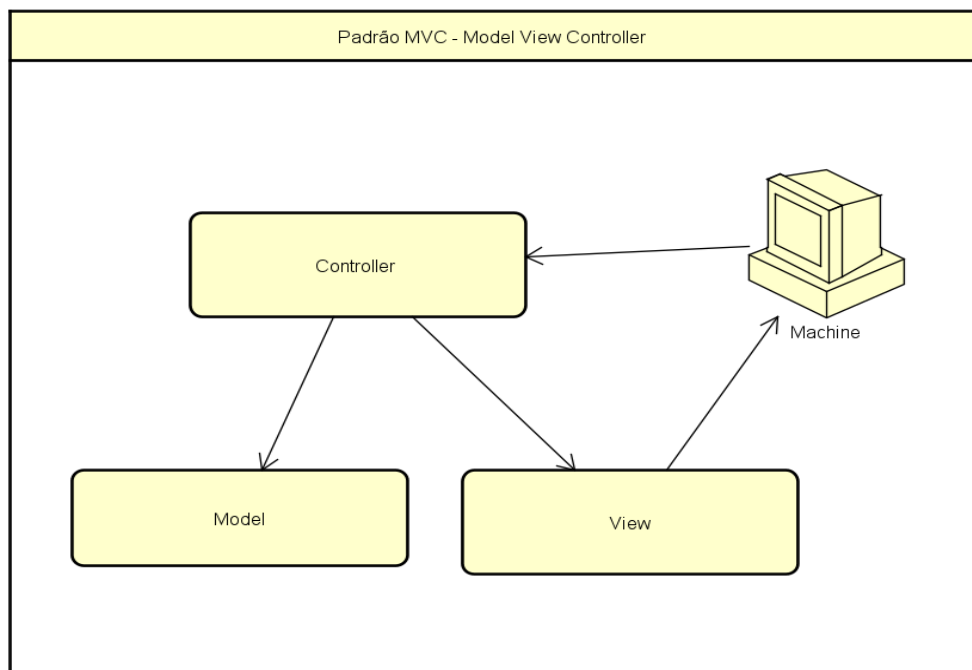


Figura 1 Padrão MVC - modelo visão e controle - Fonte: Autor

O modelo *MVC* [cujos módulos estão representados na Figura 1] tem no seu conceito um desenvolvimento de *software* com algumas vantagens, a seguir referidas:

- Facilidade na manutenção de códigos;
- Permite reaproveitar códigos e métodos;
- Autonomia para uma camada de persistência independente;
- Independência na segmentação e separação para as camadas da aplicação;
- Autonomia e dinamismo para versão.

Entre outras vantagens, este modelo de desenvolvimento é muito aceito e praticado na indústria de *software* [22] [23].

## 9.7 Frameworks Front-End e Back-End

Tipicamente, a denominação *framework* é um termo para caracterizar as interfaces de programas e serviços relativos ao utilizador destas *interfaces*.

Para o desenvolvimento da aplicação *web responsiva* existem vários *frameworks* a nível de *front-end* e *back-end*, tais como, *Dropwizard*, *Struts* da fundação *apache*, *JavaServerFaces (JSF)* *Grails*. Apenas abordaremos os mais populares e que serão usados neste projeto. Ambos são plataformas usadas para o desenvolvimento de *software* que provê soluções para a gestão no desenvolvimento de sistemas e aplicações; da mesma maneira estas plataformas oferecem uma

estrutura com classes e funções para otimizar o desenvolvimento, funcionando numa camada de abstração dentro de uma estrutura de funcionalidades para auxiliar os desenvolvedores nas especificações do sistema a ser desenvolvido.

O *front-end* pode ser classificado como a parte visual de uma aplicação, ou seja, o ambiente que está do lado do cliente; quem trabalha dentro deste ambiente é responsável por codificar a interface gráfica sendo normalmente as tecnologias o *CSS*, *HTML*, *Bootstrap* e *JavaScript*; entretanto, para gerir estas tecnologias, não existem *frameworks* com boas *interfaces* gráficas para facilitar o desenho dos *layouts* e auxiliar os profissionais de desenvolvimento.

O *back-end* compreende a separação no processo de desenvolvimento de *software* do *front-end*, sendo o ambiente do lado servidor; facilita e simplifica o processo de desenvolvimento e organiza, separando para possível manutenção de códigos e classes, objeto e suas propriedades, bem como a segurança na proteção das *APIs* e com amplo âmbito de bibliotecas. Permite gerir a estrutura da base de dados, para lidar com as regras de negócio da aplicação no âmbito da persistência dos dados e desempenho, gerando velocidade no desenvolvimento. As tecnologias mais utilizadas do lado do servidor são *Java*, *C++*, *Python* e *JavaScript*. Neste projeto estaremos utilizando um *framework spring* para tecnologia de programação *Java*.

## 9.8 Estudo de Trabalho Relacionado

Neste estudo foram identificadas aplicações que de alguma forma tenham similaridades com a solução proposta neste projeto. Com o propósito de criar uma solução específica para o setor de bebidas de empresas de pequeno e médio porte, este estudo concentrou-se sobre os aspetos da usabilidade e do *designer* responsivo que as aplicações atuais existentes no mercado podem oferecer. Também foi levada em consideração a possibilidade de o acesso não estar condicionado a um local e equipamento específico; neste sentido, será analisada a flexibilidade de acesso de ambos os *softwares* bem como a sua especificidade ao ramo de atividade, como abaixo se demonstra:

### **Primavera:**

A plataforma primavera é um *software* abrangente e amplamente extensível, proporcionando adequação a cada negócio; não se trata apenas de uma solução voltada para o ramo de bebidas especificamente [24] foi projetado para todo o comércio em geral.



Dos muitos outros processos de atividades comerciais que esta aplicação inclui, destacamos algumas das principais funcionalidades que interessariam para o ramo de bebidas. Permite:

- Registrar produtos;
- Registrar clientes;
- Controlar *stock*;
- Realizar vendas;
- Gerar relatórios;

### **PHC CS:**

Trata-se de um *software* comercial que pode ser instalado no computador, acessido via *desktop*, que também permite, para as suas novas versões, a possibilidade de ser acessido via *browser* [25]. Esta solução no seu escopo foi a princípio desenhada para a plataforma *windows*, mas novos módulos foram sendo implementados, com o fim de atender a todas as atividades comerciais. Outro fator importante a ressaltar é que esta aplicação não contempla a ideia de uma solução totalmente *web*, tal como as demais soluções de *Enterprise Resource Planning (ERP)*. De entre as funcionalidades existentes, destaca-se, que, para o ramo de atividade de bebidas, permite:

- Registrar produtos;
- Registrar clientes;
- Controlar *stock*;
- Realizar vendas;
- Gerar relatórios;

### **CentralGest:**

Este *software* trata de um sistema com uma estrutura aplicacional robusta, contemplando, na totalidade, a ideia de um sistema *ERP*, que permite realizar a gestão de grandes empresas [26]. Esta aplicação é subdividida em módulos que se adaptam a cada seguimento comercial, visando uma instalação prévia no dispositivo do lado do cliente. Destacamos algumas características de um dos módulos do *ERP* que poderia satisfazer o seguimento de bebidas, dado que permite:

- Registrar produtos;
- Registrar clientes;

- Controlar stock;
- Realizar vendas;
- Gerar relatórios;
- Efetuar o controlo de frete na venda;

### 9.8.1 Comparações de funcionalidades

Com base nas comparações realizadas, observou-se que as soluções comerciais acima demonstradas, de alguma forma têm similaridades nos aspetos. Permitem gerir os processos de uma distribuidora de bebidas, quanto ao registo de produtos, registo de clientes, e realização de vendas de produtos com emissão de relatórios.

Notou-se também, neste estudo, que as aplicações estudadas, são *software* de planeamento de recursos empresariais (*ERP*), soluções estas que compreendem a gestão integrada dos principais processos de um negócio empresarial, ou seja, um conjunto de aplicações integradas.

*Tabela 1 Comparação de funcionalidades*

	<b>Primavera</b>	<b>PHC CS</b>	<b>CentralGest</b>
Acedida pelo browser	Apenas algumas versões	Apenas algumas versões	Apenas algumas versões
Contém requisitos de responsividade	Apenas algumas versões	Apenas algumas versões	Apenas algumas versões
Contém os princípios da usabilidade de software	Sim	Sim	Sim
Funciona em qualquer dispositivo	Não	Não	Não
Depende da versão do sistema operativo	Sim	Sim	Sim
Exige instalação prévia no dispositivo	Sim	Sim	Sim
Depende de uma ligação de internet	Sim	Sim	Sim

Pela tabela 1 acima, podemos verificar que, não obstante apresentar similaridades em termos de funcionamento e informatização de processos, algumas das aplicações estudadas, ainda não funcionam em dispositivos móveis, apenas em *desktops* tendo como requisito o ambiente de sistemas operativo *Windows*. Dado que os princípios e métodos do desenvolvimento *web* estão totalmente associados à ideia de uma solução *web*, tipicamente soluções tal como as em estudo dificilmente atenderão à metodologia do *RWD* ou até mesmo do *AWD*.

Como o objetivo deste projeto é uma solução que possa ser acedida por um *browser* sem a necessidade prévia de uma instalação, notou-se, neste estudo, que a depender da versão do *software*, a aplicação está impossibilitada de ser acedida por um navegador *web*, por serem soluções com grandes dimensões de funcionalidades e módulos, dificultando o aspeto da responsividade. Sobre a camada de dados, estas soluções *ERP* tipicamente precisam de uma base de dados local para guardar os dados, muito diferente da solução proposta que permite o seu acesso numa *cloud* qualquer, podendo ser feito lá a persistência dos seus dados, tudo isto sem exigir e/ou gastar muitos recursos do dispositivo em utilização.

## 10. Conceito do Projeto

### 10.1 Requisitos do Projeto

Este trabalho de projeto contempla o desenvolvimento de uma solução *web* responsiva para pequenas e médias empresas nomeadamente na área de distribuição e venda de bebidas, tal como já referido, com a capacidade de se adaptar aos dispositivos *desktops*, *smartphone* e *tablets* compreendendo os princípios de usabilidade e segurança. Em seguida se apresenta as estruturas utilizadas no *back-end* e *front-end*.

### 10.2 Back-End Frameworks

Como enunciado no capítulo 9, secção 9.7 *Frameworks front-end e back-end*, sobre as características e propósito dos *frameworks*, nesta secção estaremos apresentando as *frameworks* do *back-end*. Assim sendo, abordamos as linguagens de programação usadas no *back-end* do presente trabalho com suas *IDEs*, as bases de dados, o aspeto de segurança da aplicação desenvolvida, servidor *web* utilizado e todas as ferramentas que foram usadas neste projeto para o âmbito do *back-end*, bem como as suas características e funcionalidades.

#### 10.2.1 Eclipse, IDE

O *Eclipse, Integrated Development Environment (IDE)* ambiente de desenvolvimento integrado, uma ferramenta para auxiliar no desenvolvimento de uma solução *web* ou *desktop*, possui o conceito de perspectivas e visões para organizar o ambiente de trabalho, permitindo codificar em várias linguagens de programação como por exemplo *Java*, *C++*, *C*, *JavaScript* entre outras. Para criar aplicações *Java* é possível usar qualquer ferramenta de edição de texto comum, como é o caso do *Notepad*, ferramenta padrão da *Microsoft Windows*; entretanto para a execução de programas codificados em *.java* é necessário o uso da *JVM*, ferramenta instalada junto com a *JDK* para compilação dos programas inscrito em *.java* no próprio *prompt* de comando do *microsoft windows*. Deste modo a utilização da ferramenta *Eclipse* veio para facilitar todos estes processos de criação de programas *Java*, permitindo organizar os projetos em desenvolvimento, gerando produtividade.

### 10.2.2 Java

A linguagem de programação *Platform, Enterprise Edition (Java, PEE)* compreende um ambiente de desenvolvimento de *software* corporativo por padrão em *Java*. Esta arquitetura *Java EE* compõe diversas tecnologias que lhes oferecem suporte no âmbito do desenvolvimento *Web*, *Web Services* Componentes distribuídos, componentes de persistências entre outros serviços. De igual modo temos o *Java Platform, Standard Edition (Java, PSE)* com um ambiente mais básico do âmbito *Java*, também utilizada para o desenvolvimento em plataformas *desktop* em modo consola, sendo necessário para o uso uma máquina virtual do *Java*, o *Java Virtual Machine (JVM)* [27]. A linguagem *Java* teve seu início em 1990, aproximadamente, sendo projetada pela empresa *Sun Microsystems*, com o objetivo voltado para dispositivos eletrónicos embebidos, equipamentos eletrónicos e eletrodomésticos, torradeiras, fornos de micro-ondas e sistemas interativos de TV. O seu objetivo principal era fornecer simplicidade e confiabilidade, sendo baseado em *C++* [28]. O uso da linguagem *Java* cresceu rapidamente mais do que qualquer outra linguagem de programação, estando atualmente *Java* presente em mais de 4,5 biliões de dispositivos [29]. A plataforma é utilizada para criar páginas da *web* com conteúdo interativo e dinâmico, grandemente utilizada na construção de documentos *web*, sendo capaz de permitir maior interatividade entre os principais *web browsers* disponíveis com abrangente suporte comercial, apta também para o desenvolvimento de aplicações corporativas de grande porte, aprimorar a funcionalidade de servidores da *World Wide Web*, fornecer aplicações para dispositivos destinados ao consumidor final e para muitas outras finalidades [30]. Neste projeto não usaremos o *J2EE* padrão oficial específico da pilha *Enterprise Application frameworks Java*; ao invés usaremos o *Framework Spring*, uma estrutura que executa vários processos e tarefas nos detalhes e âmbito do *Java EE*, ainda que em seu próprio ambiente ou plataforma.

### 10.2.3 Spring MVC

O *Spring Framework* oferece uma estrutura muito versátil que permite construir desde *APIs Web Services* até projetos com páginas *HTML*, e o facto de ser muito versátil com a junção de outros projetos do eco Sistema *Spring*, como por exemplo, o *Spring Data JPA*, *Spring Boot* com *IDE STS*, sendo possível entregar mais capacidade para esta plataforma para além do poder de produção do desenvolvimento de *software* e site *web*, que por si já o tem [23]. Este *Framework Spring* tem sido utilizado tanto por profissionais que prezam pelo modelo deste ecossistema como

para profissionais que trabalham com *JEE* e que gostam do *Spring* em detrimento de tantas as funcionalidades que é oferecida pelo *Framework Spring* e ao mesmo tempo permite tratar as requisições *web* de modo a intermediar uma requisição *HTTP*, *HTTPS* e o processamento que acontece na aplicação a ser desenvolvida, tratando os dados recebidos da requisição do utilizador no *browser*. Um dos objetivos principais do *Spring Framework* foi permitir gerir o ciclo de dependências do projeto, sendo que o *container Core* do *Spring Framework* é a injeção de dependências e inversão de *Controller* [23] [31]. Muitas funcionalidades que temos hoje no *JEE* foram influenciadas pelo *Spring Framework*. O *Spring Framework MVC* é uma *Framework* baseada em ação; o padrão de desenvolvimento *Action Based* permite a separação entre a camada de controlo e a camada de visão de modo a permitir mais controlo e flexibilidade em cada camada, onde a camada de controlo “empurra” os dados para a camada *View* – Camada de visão, diferente do *JSF* que é baseado em componentes (*Component Based*). O *Spring* é um *framework* de código aberto, uma estrutura de aplicativos para a plataforma *Java*, que suporta o desenvolvimento de aplicações *java*, oferecendo diversas soluções de modo a permitir aos desenvolvedores focar-se apenas no desenvolvimento da aplicação enquanto que o *Spring* se preocupa com a gestão das transações na estrutura de acessos aos dados, oferecendo uma abstração flexível para trabalhar com acesso aos dados [23]. É possível usar o *Spring* para o desenvolvimento de aplicações *Java SE* desktop e também para o desenvolvimento de aplicações *Java EE*, para aplicações corporativas e aplicações *web* como é o caso deste trabalho de projeto de mestrado.

#### 10.2.4 Apache Maven

Uma ferramenta de compilação que visa reunir os princípios atuais da indústria de *software* de forma a garantir as boas práticas para o âmbito da linguagem *Java* fornecida pelo *Apache*, semelhante em funcionalidade à ferramenta *Apache Ant*, possui, porém, conceitos diferentes na sua gestão [32]. O controlador de dependência *Apache Maven* é uma ferramenta para a gestão, construção e compreensão de projetos de *software*. Ferramenta esta que permite aos arquitetos e desenvolvedores gerir a compilação de código e o seu empacotamento, rastreamento de problemas, bem como, atendendo à sua capacidade e maneira fácil, flexível e rápida, proporcionar a gestão de muitas dependências, baixando dinamicamente as bibliotecas *Javas* e *plug-ins* dos respetivos repositórios, bem como as importações de *links*, para o agrupamento de projetos, a criação e implementação de sistemas em um único local. Existem muitos sistemas para a compilação, mas o *Maven*, por ser uma plataforma orientada a projetos,

facilita a partilha de projetos arquitetados em um gráfico de dependências para a plataforma *Java*, permitindo à equipa reduzir o número de ficheiros grandes com os quais é necessário enviar, de modo a facilitar em caso de se ter pessoas usando versões diferentes de uma dependência [33].

## 10.2.5 Apache Tomcat

O *software Apache Tomcat* é um sistema de código aberto das tecnologias *Java*, desenvolvido em um ambiente aberto e participativo com uma estrutura de *container* muito leve em comparação com o *GlassFish* que possui um *container* certificado *Java* de pilha completa; todavia para aplicações *web* comuns é mais popular do que *GlassFish*, consumindo menos memória, embora ambos atuem como a implementação de referência para vários padrões *JAVA*, funcionando como um *WebContainer* que permite executar aplicações da *web* com base em *Servlet* e *JavaServer Pages (JSP)*. A maioria das estruturas modernas da *web Java* é baseada em *Servlets*, *JavaServer Faces*, *Struts*, *Spring*; para exemplo deste projeto em questão, desta forma como um *container* de sublocações, o *Apache Tomcat* pode ser usado como servidor *web HTTP* [34].

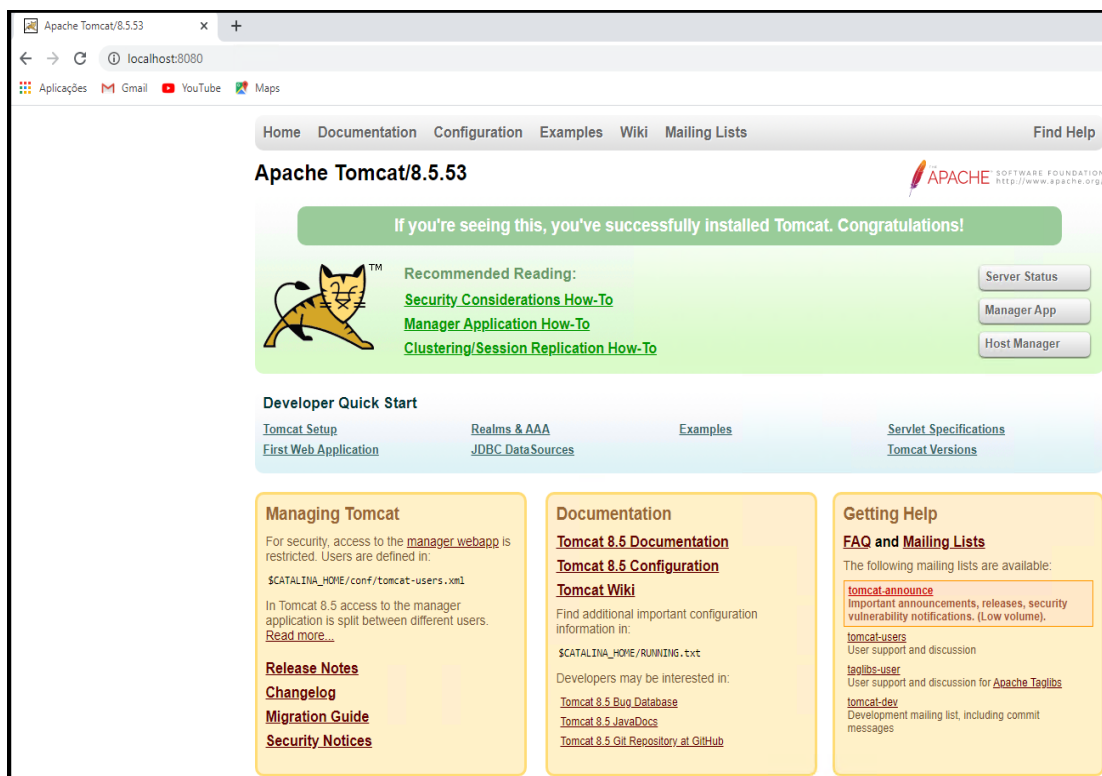


Figura 2 Página administrativa do apache-tomcat - Fonte: localhost server apache

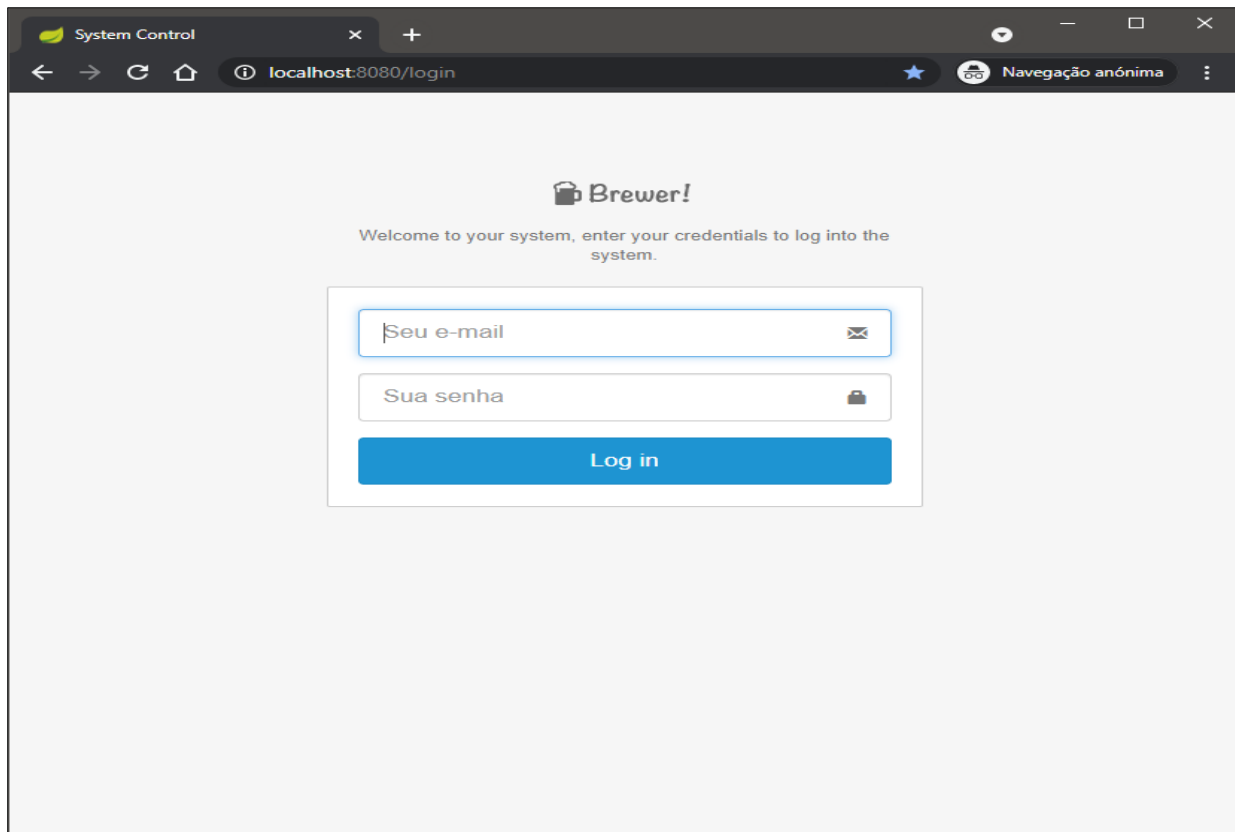


Figura 3 Configuração localhost - página da aplicação - Fonte: Autor

Como apresentado na figura 3, esta é a página de login da aplicação, sendo acedida através do navegador *google chrome*, num equipamento *desktop* configurado para o desenvolvimento do sistema *web*. Como se pode notar, o *link* de acesso a esta aplicação está configurado para funcionar via *localhost*, funcionalidade proporcionada pelo *Apache Tomcat* servidor *web HTTP*, tal como demonstrado na figura 2, a partir do qual as configurações inseridas na aplicação do lado do *back-end* proporcionarão o uso através do *localhost* da máquina.

### 10.2.6 *Spring Security*

O *Framework Spring Security* permite proteger um sistema ou *site* desenvolvido no âmbito da linguagem de programação *Java*. Neste projeto de desenvolvimento de um sistema *web* com linguagem *Java*, para a segurança da aplicação desenvolvida, estaremos utilizando o *Framework Spring Security*, embora que para utilizar o *Spring Security* num projeto de desenvolvimento de *software*, não seja necessariamente preciso ser desenvolvido o projeto sobre o âmbito do eco sistema *Spring Framework*; é possível desenvolver um sistema em *JSF* com *CDI*, por exemplo, e utilizar o *Spring Framework* para proteger a aplicação no seu desenvolvimento, embora o *Spring Security* tenha uma integração forte e natural para o projeto



que é desenvolvido totalmente sobre o âmbito do *Framework Spring*; isto torna-se uma vantagem quando todo o projeto é projetado sobre a base do ecossistema *frameworks*, dado a quantidade de funcionalidade a mais que é disponibilizada do *framework*.

A parte de segurança do *Java* que vem especificado dentro do *JEE* possui um intervalo longo para as novas especificações de segurança e funcionalidades a serem disponibilizadas; uma das vantagens do *Spring Security* é que vai além das especificações, com um tempo bem menor para o lançamento e disponibilização com correções de *bugs*, funcionalidades novas, recursos extras e sobretudo mantendo a compatibilidade com a especificação *JEE*, para além de mais recursos em menos tempo para os respetivos lançamentos de versões com as integrações do recurso, entre eles, o de segurança; da mesma forma o *Spring Security* é um *framework* de segurança portátil, ou seja, permite a flexibilidade de migração de *container* a exemplo do *GlassFish* servidor de aplicações de código aberto - projeto iniciado pela *Sun Microsystems* para a plataforma *Java EE*, para um outro *container* mais leve, a exemplo do *Apache Tomcat*, uma implementação de código aberto das tecnologias *Java Servlet Pages*, *Java Expression Language* e *WebSocket* fornecendo um ambiente de servidor *web* através do protocolo *HTTP* com pura linguagem *Java*. Diferente do *Java EE* que no âmbito da segurança não é totalmente portátil, requiere bem mais configurações para esta mudança de *container* [35]. Ainda mais, o *Spring Security* permite ser altamente customizável, com *APIs* fluentes, que é entregue aos arquitetos e engenheiros de *software*, sendo baseado totalmente em *interface*; isto facilita implementações a determinado projeto com características específicas - outro aspeto dentro do ecossistema *Frameworks Spring Security* é a proteção contra ataques mais conhecidos no âmbito da segurança da informação, como sejam, *Session Fixation*, *Man In The Middle*, *cross site Request Forgery* (esta segurança do *Spring Security* já vem por padrão no *framework*) para os profissionais que trabalham com estas ferramentas de modo a caracterizar tudo isto como grandes vantagens em utilizar estes serviços.

Embora os processos da Autenticação e Autorização possam ser tratados juntos e parecer ser muito semelhantes nos sistemas *web*, o *framework Spring Security* distingue muito bem estas funcionalidades dentro do desenvolvimento de um *software*; a validação das credenciais de um utilizador ou de um outro sistema acontece na autenticação, sendo possível utilizar um formulário *HTTP basic*, ou *HTML* comum, para se informar destas credenciais e através do protocolo *LDAP*, podendo ser feita também com total liberdade a procura das informações e validação via base de dados, como será o caso deste projeto, ou até mesmo uma integração com ferramentas de terceiros não padrão da indústria de *software*; existem muitas maneiras de autenticar as credenciais do utilizador que deseja aceder à aplicação. Por outro lado, temos a autorização, fase que acontece

depois que a autenticação é realizada com sucesso; o processo de autorização é garantir que algum utilizador ou outro sistema tenha permissão para aceder a determinado recurso dentro da aplicação, o que, no caso deste projeto, serão as nossas páginas, telas do sistema *web* desenvolvido. Uma grande vantagem do *Framework Spring Security* é que ele permite aplicar segurança até ao nível de métodos e instâncias de objetos contendo vários recursos relacionados com a autorização de consumo da aplicação.

### 10.2.7 JPA, Hibernate e Flyway

A API de persistência *JPA (Java Persistence API)* é proveniente da especificação *Java* e permite gerir, aceder e definir a persistência entre os objetos *Java* e a respetiva base de dados para as tabelas relacionais ao *DB* [36][20], como por exemplo *MySQL*, *SQL Server* *PostgreSQL*. O *JPA* como especificação, disponibiliza funcionalidades, protótipos comuns para ferramentas *ORM*; temos, portanto, ferramentas *ORM* baseadas em *Java*, tal como *Hibernate*, uma solução de mapeamento objeto-relacional que usamos neste projeto, visto que o *JPA* por si só não realiza operações. Por este modo o *Hibernate* é uma das implementações de *JPA* mais utilizadas com benefícios de acoplamento flexível, enquanto que o *Hibernate*, como já citado, é uma ferramenta *ORM (Object-Relational Mapping)* com o propósito de salvar a condição do objeto *Java* na base de dados para o projeto; por outro lado o API de Persistência *Java (JPA)* apenas define a gestão dos dados nas ferramentas *Java* [37].

Neste projeto trabalhamos com a API *Hibernate Criteria (HCE)*, visto que esta possui um contexto e abordagem orientados a objeto para o âmbito de consultas à base de dados, embora não seja viável a execução de atualizações e/ou exclusão de consultas para quaisquer instruções *DDL*, sendo restritamente praticada para o uso de buscas pelos resultados na base de dados pelo princípio de orientação a objeto [38]. Um dos maiores desafios para os arquitetos e engenheiros da indústria de *software* é a gestão do tempo para a entrega dos produtos; para este fim uma ferramenta essencial, como também de boa prática para o trabalho com base de dados e gestão das requisições, é o *flyway* com *spring* e *hibernate*, que controla as migrações, com integração contínua à base de dados. O *Flyway*, projetado para funções específicas, é leve e fácil de configurar e uma ferramenta de código aberto [39]. A sua principal especificidade é migrar, limpar, validar, desfazer, linha de base, reparar e atualizar as informações e testar os dados sempre que houver uma alteração aplicada, bem como fazer controlo da versão do projeto. O

*flyway* também trabalha com convenção sobre configuração e isto permite agilizar o desenvolvimento [39].

### **10.2.8 Spring Data JPA**

Dentro do ecossistema *Spring Framework* temos o *Spring Data*, sendo um dos seus objetivos o da abstração dentro do repositório colaborar com uma parcela significativa para a redução na codificação para o processo de acesso aos dados dentro dos vários ambientes de persistência na camada de implementação [40]. Em relação ao *JPA*, o *Spring Data* não funciona como provedor desta definição *JPA*, mas funciona como uma biblioteca ou estrutura com camada de abstração extra tal como o *Hibernate* na parte superior do *JPA* provedor deste ambiente, diferentemente de ser o *Hibernate* uma implementação *JPA* [40]. O *Spring Data JPA* é uma abstração para camada de acessos a dados *JPA* da qual permite gerar consultas *JPA* em seu nome a partir da estratégia de nomes de métodos, criando métodos e consultas usando a geração de consultas [36].

### **10.2.9 Base de Dados MySQL**

Para este projeto usamos o *MySQL* visto que esta ferramenta de gestão suporta conectividade com diversas linguagens e ambientes, como *Java*, *PHP*, *Python*, *C*, *C++*, mesmo em ambiente *Windows* como *Linux* entre outros ambientes. O *MySQL* funciona como um gestor de base de dados relacional (*Relational Data base management System - RDBMS*) [41]. Tal como os sistemas de gestão de dados *PostgreSQL*, *SQL Server*, *Oracle*, *DB2* entre outros, é o *MySQL* dentro os quais que permite aceder e manipular dados em suas bases de dados com segurança ante a corrupção e inconsistência para com os meta-dados de maneira a definir os dados que podem ser persistidos de forma rápida, sendo um base escalável [41] [42].

## **10.3 Front-End Frameworks**

Como apresentado no capítulo 9, secção 9.7 *Frameworks front-end e back-end*, as características e propósito dos *framework*, nesta secção estaremos apresentando uma descrição de tecnologias como o *HTML*, *CSS*, *Bootstrap*, *JavaScript* e o *framework Thymeleaf*, soluções estas que permitirão criar do lado do cliente a aplicação proposta.

### 10.3.1 *HTML - Hypertext Markup Language*

O mercado para os sistemas *web* é muito grande; a maioria dos sistemas atuais são desenvolvidos para a plataforma *web*, não apenas *web sites*, redes sociais e comércio eletrônico, mas também sistemas corporativos, sistemas comerciais ou portais de notícias entre outros diversos tipos de sistemas.

O desenvolvimento para o ambiente *web* tem realmente crescido muito; a exemplo de entre os mais conhecidos, temos vários serviços da empresa *Google* que atualmente estão sendo disponibilizados via *browser* do utilizador, como *Google docs*, *Google driver*, permitindo abrir documentos *Word* na própria *URL*, folhas de cálculo, slides, em suma, existem muitos serviços que estão sendo transportados para o ambiente *web* permitindo flexibilidade no acesso e comodidade para o uso e escalabilidade.

Uma grande parte dos sistemas *web* depende de várias tecnologias para o seu funcionamento, principalmente do *HTML*, *CSS* e *JavaScript*.

Neste trabalho de projeto estaremos usando o *HTML5*. Esta nova versão permite escrever mais precisamente o conteúdo com o seu significado, com melhoria na sua semântica, sendo que o estilo e a formatação são definidos por outra tecnologia denominada *CSS*, ficando a cargo do *HTML* o conteúdo e a estrutura da página [43] [44].

### 10.3.2 *CSS - Cascading Style Sheets*

O *Cascading Style Sheets (CSS)*, permite descrever qual o modo da apresentação de páginas na *web*, com uma linguagem de estilo simples e legível e compreensível por humanos e com codificações compactas na maneira de apresentar o conteúdo comparado a imagens ou ficheiros de áudio, que são frequentemente usados, de modo a obter efeitos de renderização; as folhas de estilo geralmente diminuem o tamanho do conteúdo, também facilitam na simplificação da manutenção do site de maneira a manter uma aparência, tal como os elementos *HTML* são apresentados, em uma página, possibilitando a escolha de cores, posicionamento no *layout*, fontes, bordas, entre outros, sendo possível descrever código *CSS* diretamente dentro de um documento *HTML* consistente em todo o sistema ou site [45].

Para gerir a apresentação e a formatação de um documento *HTML* usa-se o *CSS* devido ao seu estilo e linguagem; com o *CSS* podemos aplicar todos os estilos de modo a formatar os documentos *HTML*, como seja, inserir cores, posicionamentos, tamanhos, entre outros tipos de

formatação. Para o desenvolvimento deste projeto de *software* foi utilizada a última versão do *Cascading Style Sheets*, o *CSS3* [45] [46].

### 10.3.3 *Bootstrap*

Este *framework* está estritamente ligado ao *front-end*, mais precisamente na camada *View* do padrão *MVC*; não há qualquer interferência noutras tecnologias ou linguagens de programação, embora este projeto esteja sendo desenvolvido dentro do conceito orientado a objeto com a linguagem de programação *Java* [47]. O *bootstrap* é uma *framework* criada por Mark Otto e Jacob Thornton em 2019 [48] que possui uma estrutura *web* centrada na simplificação para o desenvolvimento de páginas para *web* com o objetivo centrado em ser responsivo de maneira a permitir aplicar as opções de cor, tamanho, fonte e *layout* do *bootstrap* a determinado projeto; umas das características deste *framework* é o fornecimento de definições de estilo para todos os elementos *HTML*, de modo a provisionar vários recursos para o âmbito de classes *CSS*, sendo os componentes de *layout* fatores importantes para o *bootstrap*, visto que isto afeta o estilo inteiro da página da *web* [47] [48]. [32] Outra característica deste *framework* é tornar o desenvolvimento de interfaces responsivas.

### 10.3.4 *Java Script*

Uma linguagem de programação baseada em *scripts* criada por Brendan Eich (Netscape) tendo surgido em 1995 como linguagem de *script* do lado cliente, também conhecida como a linguagem de *scripting* para páginas *web*, mas também utilizada em muitos ambientes fora dos navegadores, padronizada pela *ECMA International*. Com uma sintaxe básica intencionalmente similar à linguagem *C++* e *JAVA*, orientada a objetos, ou seja, trata todos os elementos da página como objetos distintos de maneira a facilitar as tarefas no âmbito da programação, porém é interpretada e dinamicamente tipada, amplamente integrada e processada em páginas *web* devido à facilidade de integração com o *Document Object Model (DOM)* da página. O *JavaScript* é uma linguagem de programação que permite realizar a criação de *scripts* do lado cliente em páginas *web*, possibilitando a criação de conteúdo, que processa dinamicamente, como o controlo de multimídia, imagens animadas, podendo ser inserido numa página *web* de maneira similar ao *Cascading Style Sheet (CSS)*, assim como para aplicação do conceito *CSS* ao uso de folhas de estilo externas o elemento “*link*” e para folhas de estilos internas usa o elemento “*style*”; a

*JavaScript* quando usada no âmbito do *HTML* só precisa do elemento “*script*” para aplicação e criação de efeitos especiais na página *web*. Baseada em protótipos, multi-paradigma e dinâmica, permite programar e projetar comportamentos de páginas *web* a partir de ocorrências de eventos como pode alterar dinamicamente o valor de uma determinada propriedade “cor” de um dado elemento.

### **10.3.5 *Thymeleaf***

Baseado no modelo padrão *MVC* no processo de entrega da camada *Controller* para a Camada *View*, a lista de objetos *JAVA* recebidos precisam de ser trabalhados na *View* para a sua apresentação final ao utilizador; para este projeto será necessário um *template engine* de modo a processar estes dados. Com o *Thymeleaf* é possível pegar nos dados *Java* e transformar em conteúdo *HTML*. O *Thymeleaf* funciona como um mecanismo de modelo para linguagem *Java* moderno, sendo mais adequado para servir *Extensible HyperText Markup Language (XHTML)* e *HTML5*, do lado do *Front-end* na camada *View* para aplicações *web*, mas permite processar ficheiros *XML*, texto, *JavaScript* ou *CSS*; possui uma integração com o *Framework Spring*, em função de permitir usar modelos como protótipos - os mesmos são vistos como ficheiros estáticos, sendo o código do *Thymeleaf* muito idêntico ao *HTML*, bem mais parecido que o *template engine JSP*; no *Thymeleaf* todas as *tags* são *HTML* com atributos diferentes em detrimento ao próprio código *HTML* [31][33].

## 11. Análise e Especificação dos Requisitos do Projeto

Nesta secção especifica-se umas das fases mais importantes a nível de projeto de *software*, visto que o processo de análise visa destacar detalhadamente todas as funcionalidades de uma proposta de desenvolvimento de *software*; o levantamento de requisitos, sendo eles, funcionais ou não funcionais, conforme subentendido que este projeto deve oferecer, será demonstrado nesta secção.

### 11.1 Requisitos Funcionais

Tabela 2 Requisitos funcionais do sistema - Fonte: Autor

Requisito	Descrição
Realizar <i>Login</i>	Autenticação de utilizadores, (Clientes, Operador e Administrador) registado no sistema, permitindo realizar as operações de acordo com seus perfis.
Validar Campo de <i>E-mail</i>	
Registar Pessoa	Deverá permitir registar pessoas de acordo com o grupo que irá pertencer, se Administrador do sistema ou se vendedor.
Ativar Registo Pessoa	Deverá permitir ativar e desativar o registo do utilizador.
Alterar Registo Pessoa	Deverá permitir alterar registo de pessoa, como nome, data de nascimento, <i>email</i> , <i>password</i> , se administrador ou vendedor no sistema.
Pesquisar Pessoa	Deverá permitir a consulta de utilizadores no sistema por nome e por <i>email</i> .
Excluir Registo	Deverá permitir excluir registo de pessoas do sistema, exceto administrador da aplicação.
Registar Cliente	Deverá permitir registar clientes de acordo com o tipo de pessoa, se pessoa física ou se pessoa jurídica.
Pesquisar Cliente	Deverá permitir a consulta de clientes por nome ou por seu número de contribuinte.
Alterar Registo Cliente	Deverá permitir alterar informações do registo, nome, tipo de pessoa, telefone, <i>email</i> , logradouro, número, complemento, código postal, distrito, cidade.

Excluir Registo Cliente	Deverá permitir excluir registo de clientes, gerando um evento de alerta ao utilizador do sistema.
Registar Cidade	Deverá permitir o registo de cidades de acordo com o seu respetivo distrito ou cidade.
Pesquisar Cidade	Deverá permitir a consulta de cidade por nome.
Alterar Cidade	Deverá permitir alterar nome e/ou distrito, concelho.
Excluir Cidade	Deverá permitir excluir o registo cidade.
Registar Produto	Deverá permitir registar o produto de acordo com seu estilo, sabor, origem e sua unidade de armazenamento <i>Stock keeping unit (SKU)</i> , permitindo fazer <i>upload</i> da imagem do produto
Pesquisar Produto	Deverá permitir a consulta por <i>SKU</i> , nome, preço, ordenando pelos respetivos filtros, estilo e sabor
Alterar Produto	Deverá permitir alterar <i>SKU</i> , nome, estilo, sabor, teor alcoólico, origem, se nacional ou internacional, comissão, stock, preço e descrição (campo auxiliar no registo do produto) alterar <i>upload</i> da foto associada ao produto.
Excluir Produto	Deverá permitir remover registo produto do sistema, gerando um evento de alerta da exclusão do produto.
Registar Estilo	Deverá permitir registar um estilo para associar aos produtos.
Pesquisar Estilo	Deverá permitir consultar registo de estilo por nome.
Alterar Estilo	Deverá permitir alterar nome do estilo registado.
Excluir	Deverá permitir excluir registo de estilo.
Registo de Venda	Deverá permitir realizar venda de produtos e com clientes devidamente registado no sistema, campos para inserção de custo frete, descontos, data entrega, horário entrega e caixa de observação ao entregador, no registo de venda permitir a consulta por nome do cliente, consulta por <i>SKU</i> ou nome do produto.
Registar Venda	Deverá permitir após realizar venda, salvar apenas, ou salvar e emitir a venda.
Pesquisar Venda	Deverá permitir consultar venda por código da venda, data de criação, por <i>status</i> de orçamento, emitida



Alterar Venda	Deverá permitir alterar os seguintes campos da venda: cliente, valor do frete, valor do desconto, quantidade produto, produto, local de entrega, exceto se a venda tiver sido cancelada.
Relatórios	Deverá permitir a emissão de relatório de vendas emitidas por data, gerando relatório em formato .pdf
<i>Dashboard</i> do Sistema	Deverá conter neste ambiente gráfico de vendas realizadas por mês, vendas por origem se nacional ou internacional, total de clientes registado no sistema, quantidade de stock no sistema e o valor do stock, total de vendas realizadas por ano e quantidade média de ticket.
<i>Logout</i> do Sistema	O sistema deverá oferecer um botão para poder sair do sistema sem ter de fechar o <i>browser</i> navegado; após realizar o <i>Logout</i> oferecer tela de login novamente para o utilizador.
Configuração de Menus	O sistema deverá aninhar os menus em caso de este sistema estiver sendo usado em dispositivos móveis.

## 11.2 Requisitos Não Funcionais

Para os requisitos restritos à qualidade e exigências técnicas em ambiente do projeto no desenvolvimento deste *software*, seguem os requisitos não funcionais:

- **Confiabilidade** - os recursos do sistema deverão estar disponíveis quando no seu uso, envolvendo todos as suas funcionalidades disponíveis pelo escopo do projeto.
- **Usabilidade** - oferecer interface amigável, com menus bem ordenados facilitando localização e aprendizagem do sistema.
- **Fiabilidade** - este sistema deve possuir tratamento de exceção de erros, apresentando o mínimo de falhas ao utilizador.
- **Segurança** - restrições e limitações em relação a autenticação e autorização para aceder ao sistema e níveis de acessos.
- **Portabilidade** - este sistema *web* deverá funcionar no dispositivo que possua internet, seja dispositivo móvel ou *desktop*.

### 11.3 Diagrama de Classes

O diagrama demonstra como foi organizado o sistema para as classes com os *enum* que são os enumeradores, ou seja, um objeto *string* cujo valor é escolhido a partir de uma lista de valores permitidos, enumerados de forma explícita durante a especificação de uma coluna quando a tabela é criada, que serão gravadas na base de dados *MYSQL*; este esquema trata de um modelo relacional.

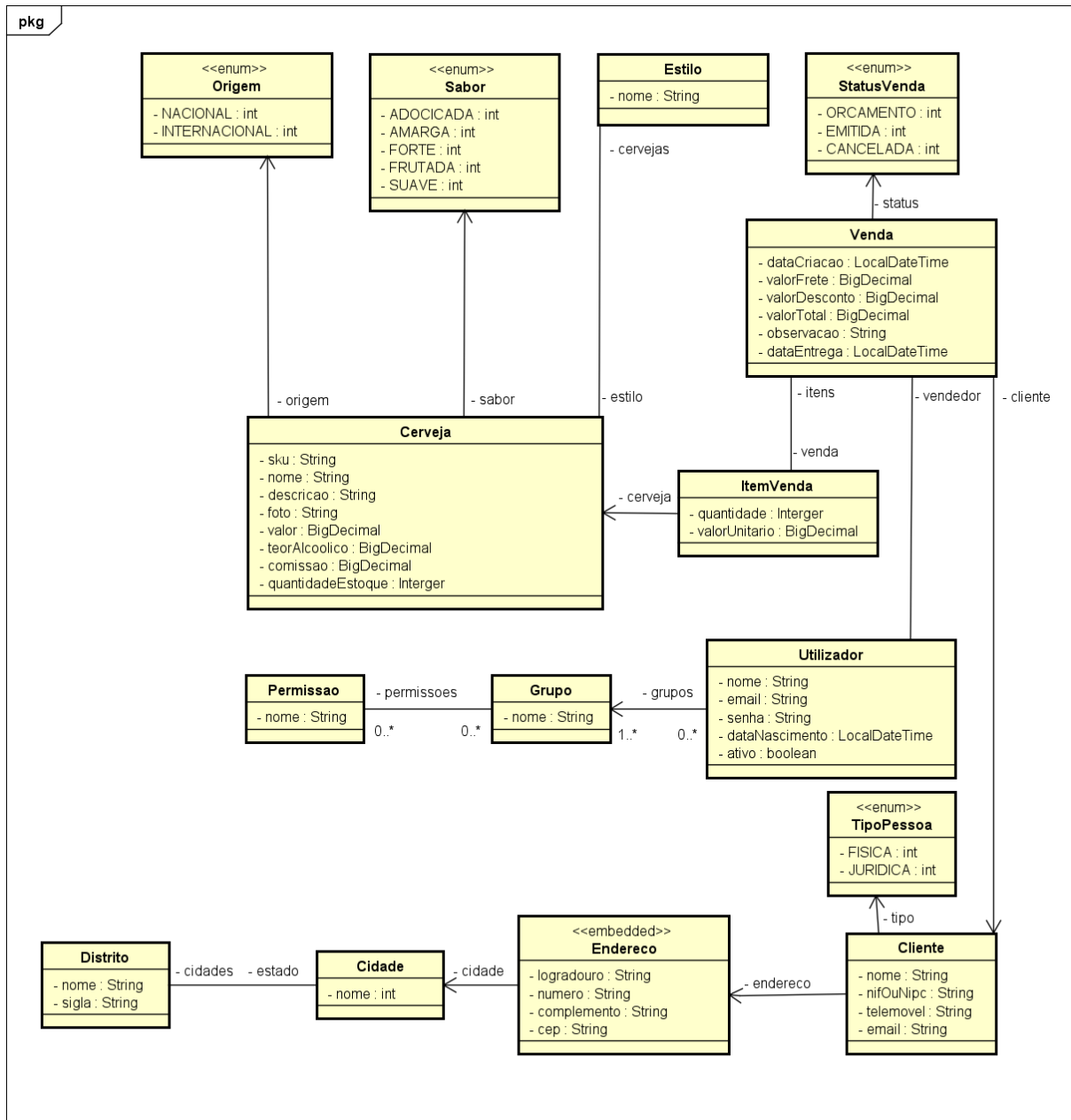


Figura 4 Diagrama de classe do projeto - Fonte: Autor

A figura 4 permite identificar as seguintes classes: classe cerveja, responsável por armazenar os dados dos produtos, que por sua vez possui dois enumeradores tipificando tais dependências, como a origem e sabor, que possuirá uma lista de valores permitidos. Esta lista compreende as nacionalidades dos produtos e os sabores, sendo que a classe cerveja tem uma associação à classe estilo o que permitirá registrar os mais diversos estilos e/ou tipo de cerveja.

Já a classe venda, responsável por registrar as vendas realizadas, também possui um enumerador; tal dependência contém o statusvendas, coluna que será armazenada na própria tabela da classe venda com a lista de valores permitidos, tais como orçamento, emitida e cancelada, tendo esta classe três associações: classe itemvenda o que permitirá registrar quantidade e valores destes itens, classe utilizador e classe cliente.

No caso da classe utilizador, serão registados os dados pessoais, de acordo com a classe grupo e terá os acessos que a classe permissão atribuirá, ou seja, se o utilizador fizer parte de um grupo administrador poderá ter todas as permissões no sistema, se o utilizador for apenas vendedor estará restrito o seu acesso a algumas funcionalidades do sistema. Esta classe possui três associações, tais como, grupo, permissões e venda, estando a classe utilizador associada à classe venda, permitindo desta forma saber a pessoa que realizou a venda.

Também a classe cliente possui um enumerador para o tipopessoa, dependência esta que permitirá listar os valores já pré-definidos no diagrama; possui um agrupamento “*embedded*” para o endereço, e associações para a classe cidade e classe distrito, e uma associação que parte da classe venda. Neste contexto, para a realização da venda é preciso ter um cliente, para que deste modo a informação de vendas na base de dados possua registo do cliente, podendo o cliente ser uma empresa ou uma pessoa comum.

## 11.4 Diagrama de Casos de Uso

Este diagrama de casos de uso traz uma visão geral do relacionamento dos atores principais. Para melhor compressão há uma descrição detalhada no apêndice A.

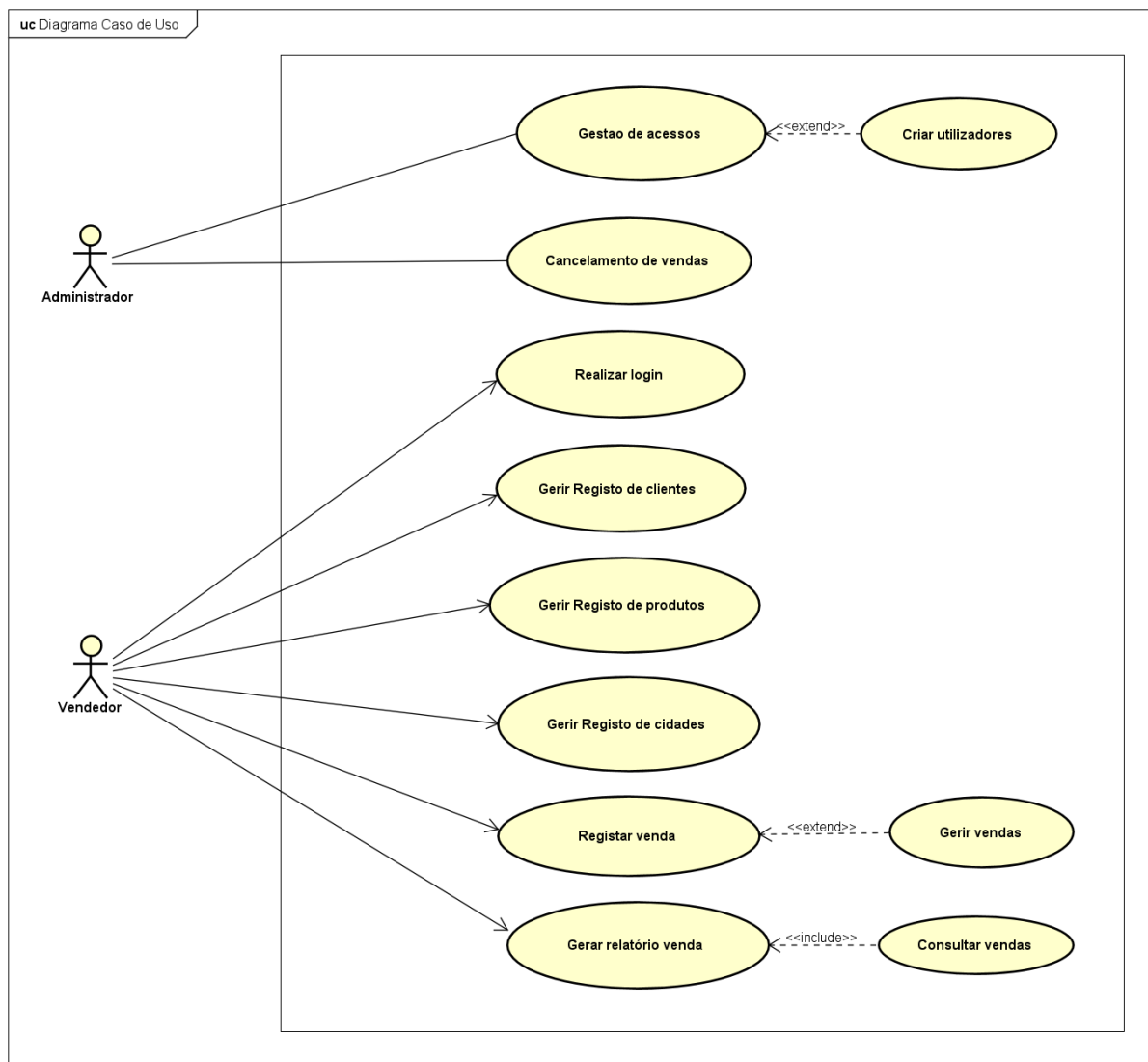


Figura 5 Diagrama de caso de uso - Fonte: Autor

Na figura 5, percebe-se que o ator administrador tem funções de gerir os acessos, registar novos utilizadores, bem como cancelar vendas. No caso do ator vendedor, após lhe ser conferido os acessos pelo administrador, pode aceder à aplicação para gerir os clientes, produtos, cidades, vendas e relatórios (estas descrições são mais detalhas no apêndice aonde consta o caso de uso em detalhe).

## 11.5 Diagrama do Processo de Desenvolvimento do Sistema

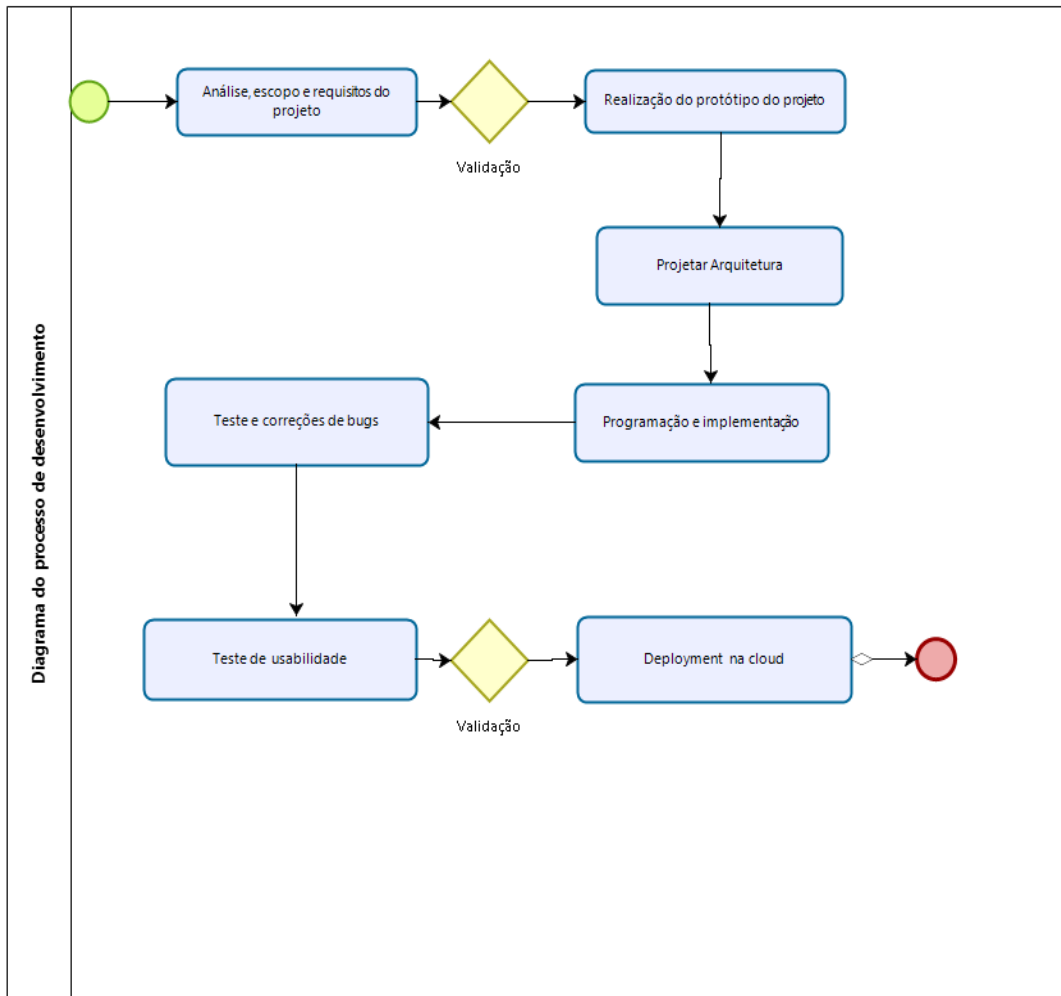


Figura 6 Diagrama do processo de desenvolvimento do projeto - Fonte: Autor

Como ilustrado na figura 6, este projeto de *software* teve várias fases: Este diagrama permite visualizar o ciclo de desenvolvimento, a começar pela fase da análise de requisitos do problema onde foram realizados os diagramas para as especificidades do projeto; para a fase do planeamento da arquitetura foi discutido qual modelo de arquitetura que seria adotado para este projeto, como se pode observar no capítulo 9, secção 9.7 *Frameworks front-end e back-end*, tendo sido da mesma forma alinhado nesta fase quais as tecnologias e ferramentas que seriam usadas para este projeto, informações que podem ser consultadas na documentação, mas especificamente no capítulo 10. Na fase da implementação e programação, foram já criadas as primeiras funcionalidades do *software* proposto e, ao mesmo tempo, aplicados testes e possíveis correções de erros - foi nesta fase que o projeto tanto da parte do *back-end* como *front-end* foi realizado, bem como um teste de usabilidade, o que permitiu identificar o aspeto da usabilidade

e possíveis erros de não funcionamento em algum âmbito do projeto. Entre a fase de teste de usabilidade e implementação do projeto, foram realizadas várias validações, a nível de base de dados: se as tabelas estavam devidamente criadas, se as requisições feitas pelo utilizador de teste estavam sendo comunicadas com a base de dados, como, por exemplo, inserir, alterar, cancelar ou apenas consultar um registo. Todos estes testes foram realizados na fase de validações do projeto. A fase final, como se pode observar no diagrama, trata da implantação da aplicação. No caso deste projeto, a proposta seria realizar este processo de publicação para *cloud* na *heroku* - esta fase consiste em preparar a aplicação para esta finalidade, a configuração da base na *cloud*, bem como a ordenação das páginas e a preparação do projeto, o seu empacotamento de modo que o mesmo possa ser submetido e apto para a publicação da aplicação. Para mais informações sobre esta fase, consultar o capítulo 13 (publicação do projeto) onde trataremos sobre a implementação e publicação do projeto.

## 12. Implementação do Projeto

Neste capítulo apresentamos o relatório do plano para a execução deste projeto, a sua arquitetura e a estrutura por intermédio da utilização da *IDE eclipse* o qual permitirá observar todos os pacotes aninhados.

### 12.1 Calendário do Projeto

Para melhor executar o desenvolvimento deste projeto, foi delineado um plano de trabalho onde se estipulou datas para a entrega do mesmo. Este plano compreende desde a definição da proposta, revisão da literatura, a concepção do projeto, a análise e especificação dos requisitos, sobre a arquitetura, a publicação e demonstração do projeto; estes prazos foram alcançados e o projeto foi realmente publicado na *cloud* onde foi possível realizar os testes da aplicação, testes de usabilidade e testes de persistência a nível de base de dados; o processo de autenticação e autorização está a funcionar corretamente como pré-definido no seu escopo.

### 12.2 Arquitetura do Projeto

A estrutura deste projeto segue o modelo *MVC*, arquitetura de *software* voltada para contribuir na requisição feita pelo utilizador, gerando melhoria e otimização no seu uso; a ferramenta utilizada para desenvolver esta solução *web* foi o *Eclipse Integrated Development Environment (IDE)*. Estando dividida em vários módulos, destaco aqueles que julgo ser mais importantes para esta aplicação e que também demonstram o conceito de modelo controlo e visão do *MVC*, tais como, *model*, *view* e *controller*.

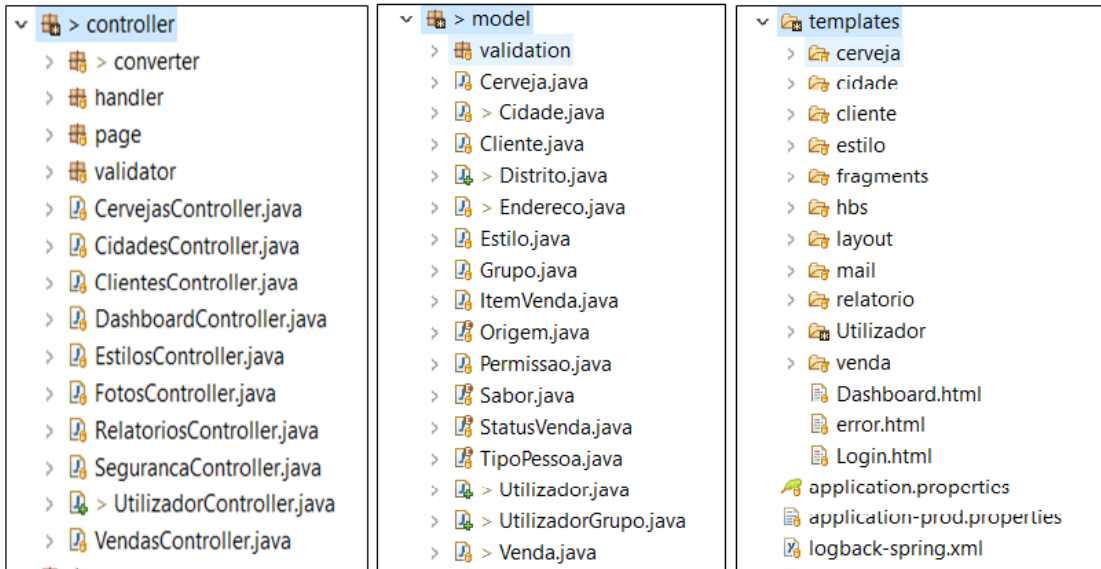


Figura 7 Módulos da aplicação - Fonte: Autor

A figura 7 apresenta os três principais módulos, a *controller* para as classes controladoras, o *model* para as classes de modelo, e a *view* na figura denominada *templates* responsável pela visualização da aplicação. O módulo *controller* é a camada responsável por intermediar as requisições entre a camada *view* e a camada *model*. Ou seja, o utilizador através do seu navegador acede à aplicação, isto é, na camada *view*, quando faz uma requisição, seja ela, por exemplo, registar um produto; esta requisição chega na camada de *controller*. O módulo *controller* através do *handler mapping* que analisa qual a *URL* de acordo com o mapeamento para qual classe *controller* na camada de controlo deve ser direcionada; a *controller*, por sua vez, vai, ao receber os dados, delegar quem é que vai trabalhar estas informações sendo, neste caso, a camada *model*, que é responsável por trabalhar os dados e fazer a regra de negócio como, por exemplo, salvar os dados na base de dados.

### 12.2.1 Database Package Migration Tool Flyway

Na estrutura deste projeto também existe a parte da base de dados em que foi usado o *MySQL* para armazenar os dados. Seguem-se algumas figuras que demonstram a base de dados criada com as respetivas tabelas.



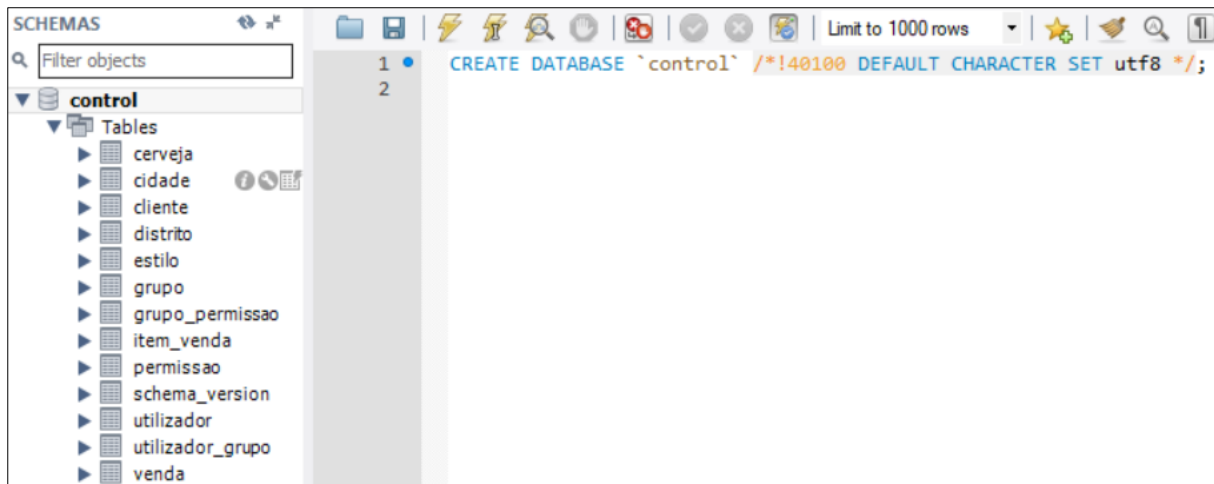


Figura 8 Base de dados do projeto - Fonte: Autor

Para armazenar os dados foram criadas diversas tabelas como tipifica a figura 8. As principais tabelas e suas características são aqui apresentadas. As instruções *SQL* para as bases de dados foram realizadas através da ferramenta *tool flyway*. Esta solução permite facilidade nas migrações para registo no *MySQL*; uma vez que utilizamos o *Eclipse Integrated Development Environment (IDE)* para a criação desta aplicação *web*, dentro deste ambiente de desenvolvimento integrado com *tool flyway* permite a criação de tabelas e inserção de novas colunas ou alterações de registos.

```

CREATE TABLE cerveja (
  codigo bigint(20) NOT NULL AUTO_INCREMENT,
  sku varchar(50) NOT NULL,
  nome varchar(80) NOT NULL,
  descricao text NOT NULL,
  valor decimal(10,2) NOT NULL,
  teor_alcoolico decimal(10,2) NOT NULL,
  comissao decimal(10,2) NOT NULL,
  sabor varchar(50) NOT NULL,
  origem varchar(50) NOT NULL,
  codigo_estilo bigint(20) NOT NULL,
  quantidade_estoque int(11) DEFAULT NULL,
  foto varchar(100) DEFAULT NULL,
  content_type varchar(100) DEFAULT NULL,
  PRIMARY KEY (codigo),
  KEY codigo_estilo (codigo_estilo),
  CONSTRAINT cerveja_ibfk_1
  FOREIGN KEY (codigo_estilo)
  REFERENCES estilo (codigo)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE cidade (
  codigo bigint(20) NOT NULL AUTO_INCREMENT,
  nome varchar(50) NOT NULL,
  codigo_distrito bigint(20) NOT NULL,
  PRIMARY KEY (codigo),
  KEY codigo_distrito (codigo_distrito),
  CONSTRAINT cidade_ibfk_1
  FOREIGN KEY (codigo_distrito)
  REFERENCES distrito (codigo)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;

```

Figura 9 Tabelas - cerveja e cidade

Na figura 9 representam-se as instruções *SQL* para a criação das tabelas *cerveja* e *cidade*, suas características, tal como *AUTO\_INCREMENT* para a coluna código, o *NOT NULL* para campos em que não é permitido valores nulos, a exceção da coluna *content\_type* e *foto* que não

são obrigatórios ser preenchidos pelo sistema; as *PRIMARY KEYS* para ambas as tabelas estão inseridas nas colunas código - na tabela cerveja há duas colunas, sabor e origem, que foram tipificadas no diagrama de classe como *enum*, ou seja os enumeradores, contêm também nas tabelas restrições *CONSTRAINT*, de modo a restringir valores que podem ser armazenados, para as *FOREIGN KEY*, criadas para permitir um *link* entre duas tabelas a exemplo das tabelas cerveja e estilo.

Tal como a figura 9, a figura 10 tipifica instruções *SQL* para a criação das tabelas venda e cliente, destacando-se características como o *AUTO\_INCREMENT* para a chave primária na coluna código para ambas as tabelas e o *NOT NULL* para campos em que não é permitido valores nulos; na tabela venda, tal como foi tipificado no diagrama de classe, existe um enumerador para a coluna status, para as *FOREIGN KEY* ou seja as chaves estrangeiras, criadas para permitir um *link* entre duas tabelas a exemplo da tabela cliente e tabela cidade, e de igual modo para as tabelas item\_venda e codigo\_venda. Como o *Mysql* suporta referências de chave estrangeiras entre colunas ou dentro de uma tabela, neste caso utilizamos a *REFERENCES*, a exemplo da tabela venda, para cruzar dados relacionados com a tabela cliente, bem como com a tabela utilizador.

```

CREATE TABLE venda (
  codigo BIGINT(20) PRIMARY KEY AUTO_INCREMENT,
  data_criacao DATETIME NOT NULL,
  valor_frete DECIMAL(10,2),
  valor_desconto DECIMAL(10,2),
  valor_total DECIMAL(10,2) NOT NULL,
  status VARCHAR(30) NOT NULL,
  observacao VARCHAR(200),
  data_hora_entrega DATETIME,
  codigo_cliente BIGINT(20) NOT NULL,
  codigo_utilizador BIGINT(20) NOT NULL,
  FOREIGN KEY (codigo_cliente) REFERENCES cliente(codigo),
  FOREIGN KEY (codigo_utilizador) REFERENCES utilizador(codigo)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE item_venda (
  codigo BIGINT(20) PRIMARY KEY AUTO_INCREMENT,
  quantidade INTEGER NOT NULL,
  valor_unitario DECIMAL(10,2) NOT NULL,
  codigo_cerveja BIGINT(20) NOT NULL,
  codigo_venda BIGINT(20) NOT NULL,
  FOREIGN KEY (codigo_cerveja) REFERENCES cerveja(codigo),
  FOREIGN KEY (codigo_venda) REFERENCES venda(codigo)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE cliente (
  codigo BIGINT(20) PRIMARY KEY AUTO_INCREMENT,
  nome VARCHAR(80) NOT NULL,
  tipo_pessoa VARCHAR(15) NOT NULL,
  nif_nipc VARCHAR(30),
  telemovel VARCHAR(20),
  email VARCHAR(50) NOT NULL,
  logradouro VARCHAR(50),
  numero VARCHAR(15),
  complemento VARCHAR(20),
  cep VARCHAR(15),
  codigo_cidade BIGINT(20),
  FOREIGN KEY (codigo_cidade) REFERENCES cidade(codigo)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

```

Figura 10 Tabelas venda e cliente

### 12.2.2 Package Resources JavaScript

A estrutura deste projeto também possui várias classes *JavaScript*. O propósito desta figura abaixo é demonstrar a estrutura da aplicação, descrevendo algumas funcionalidades para as principais classes. Como já definido, *JavaScript* é uma linguagem de programação que

permite realizar a criação de *scripts* do lado do cliente, cuja tecnologia oferece um melhor controlo para gerir conteúdo multimídia e animação de imagens; permite também gerar eventos para quem está a usar a aplicação, tudo isto de forma dinâmica, produzindo uma boa interação com o utilizador.

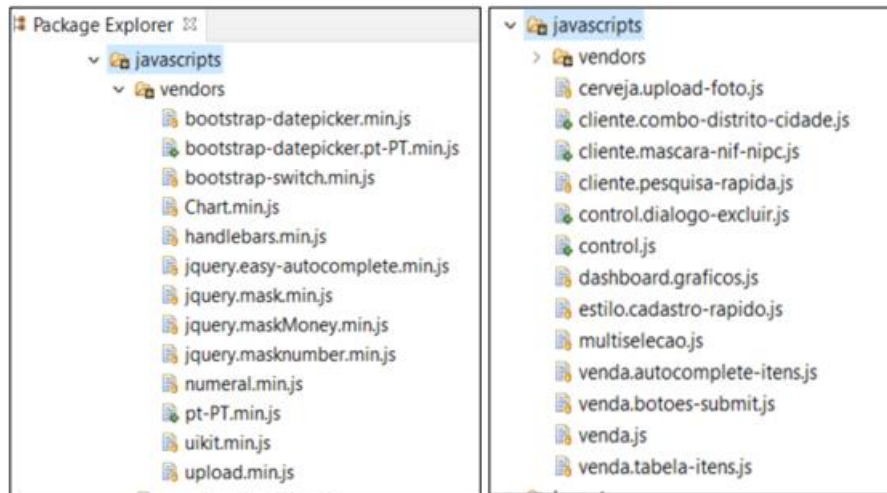


Figura 11 Package resources - javascript - Fonte: Autor

Os ficheiros *JavaScript* da figura 11 estão localizados na pasta *static* no *path*: */control/src/main/resources*. Estas classes tal como *bootstrap-datepicker.min.js* e *bootstrap-datepicker.pt.PT.min.js* têm como objetivo apresentar um calendário em formato móvel e flexível no relatório de vendas emitidas, permitindo a seleção de datas. Já a classe *jquery.easy-autocomplete.min.js*, proporciona o registo rápido do estilo de bebidas no formulário de bebidas, ou seja, caso o utilizador esteja a registar uma bebida e não identifica um determinado estilo, não é preciso sair do formulário cancelando todas as informações já inseridas; com esta funcionalidade é possível registar um novo registo para estilo sem perder as informações do formulário. Outra classe também importante é *dashboard.graficos.js*, responsável pelos gráficos, que fica na página principal da aplicação; as funções inseridas nesta classe são responsáveis por formatar vendas por mês, vendas por origem e se a venda foi para produtos nacionais ou internacionais.

```

1  Produto = Produto || {};
2
3  Produto.DialogoExcluir = (function() {
4
5      function DialogoExcluir() {
6          this.exclusaoBtn = $('.js-exclusao-btn')
7      }
8
9      DialogoExcluir.prototype.iniciar = function() {
10         this.exclusaoBtn.on('click', onExcluirClicado.bind(this));
11
12         if (window.location.search.indexOf('excluido') > -1) {
13             swal('Pronto!', 'Excluído com sucesso!', 'success');
14         }
15     }
16
17     function onExcluirClicado(evento) {
18         event.preventDefault();
19         var botaoClicado = $(evento.currentTarget);
20         var url = botaoClicado.data('url');
21         var objeto = botaoClicado.data('objeto');
22
23         swal({
24             title: 'Tem certeza?',
25             text: 'Excluir "' + objeto + '"? Você não poderá recuperar depois.',
26             showCancelButton: true,
27             confirmButtonColor: '#DD6B55',
28             confirmButtonText: 'Sim, exclua agora!',
29             closeOnConfirm: false
30         }, onExcluirConfirmado.bind(this, url));
31     }
32
33     function onExcluirConfirmado(url) {
34         $.ajax({
35             url: url,
36             method: 'DELETE',
37             success: onExcluidoSucesso.bind(this),
38             error: onErroExcluir.bind(this)
39         });
40     }
41
42     function onExcluidoSucesso() {
43         var urlAtual = window.location.href;
44         var separador = urlAtual.indexOf('?') > -1 ? '&' : '?';
45         var novaUrl = urlAtual.indexOf('excluido') > -1 ? urlAtual : urlAtual + separador + 'excluido';
46
47         window.location = novaUrl;
48     }
49
50     function onErroExcluir(e) {
51         console.log('ahahahah', e.responseText);
52         swal('Oops!', e.responseText, 'error');
53     }
54
55     return DialogoExcluir;
56 }());
57
58
59 $(function() {
60     var dialogo = new Produto.DialogoExcluir();
61     dialogo.iniciar();
62 });
63

```

Figura 12 Ficheiro produto.js código - javascript - Fonte: Autor

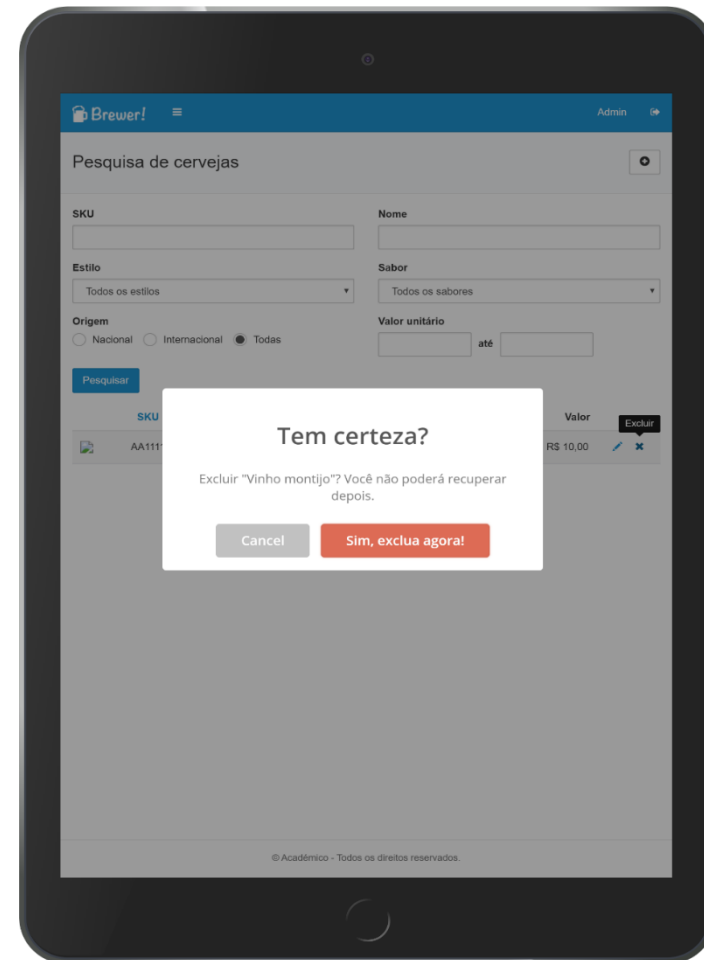


Figura 13 Ficheiro produto.html - Evento - Fonte: Autor

Dentre as funcionalidades fornecidas pelo *JavaScript*, a figura 12 apresenta a codificação feita na classe *produto.dialogo-excluir.js*; estas configurações de funções e métodos vão permitir na figura 13, ao lado, uma interação do sistema *web* com o utilizador - desta forma, quando na exclusão de um registo, a aplicação através da *function DialogoExcluir()*, irá processar o registo para sua exclusão e informar, no formato de uma mensagem, por meio do evento criado no *front-end*.



Figura 14 Evento de sucesso - excluir - Fonte: Autor

Esta figura 14 apresenta a confirmação da exclusão do registo, como resultado das configurações realizadas na classe *produto.dialogo-excluir.js*. Os eventos de interação para quem está a gerir a aplicação, dão este tipo de evento acima; nota-se o comportamento da aplicação em resposta a um registo de exclusão realizada pelo utilizador - esta demonstração está codificada para acontecer nos mais diversos tipos de ação que o utilizador realizar, sejam alterações de registo, cancelamentos de vendas, evento de mensagens de erros em caso de fornecer *inputs* não válidos, tanto para o registo em geral como para gestão de vendas. Este mecanismo traz o conceito de usabilidade na prática que um sistema pode oferecer a quem está a gerir, aplicando-se esta técnica nas diversas ocasiões em que seja necessária interação entre o utilizador e o sistema.

### 12.2.3 Segurança da Aplicação

A segurança como parte essencial e indispensável, tornou-se um desafio regular para os criadores de *software*, preparando estas aplicações, com garantia de integridade, sejam elas *web* ou *desktops*, para estar protegidas com os seus dados, transações e processos realizados no âmbito da internet. Outra questão importante é que, desde o processo de autenticação do utilizador na aplicação, período este que permite validar junto à base de dados a existência das informações inseridas, tal como apresentado na figura 15, o acesso ao sistema compreende duas fases importante: a autenticação e a autorização. Uma vez que a primeira fase de autenticar seja aceite pela aplicação, segue-se para a autorização, fase esta em que o utilizador estará condicionado a gerir as funcionalidades do sistema de acordo com as restrições propostas na sua conta.

O formulário de login da aplicação Brewer! apresenta o seguinte layout:

- Logo "Brewer!" com um ícone de copo de cerveja.
- Texto de boas-vindas: "Welcome to your system, enter your credentials to log into the system."
- Dois campos de entrada de texto empilhados:
  - O primeiro campo contém o texto "Seu e-mail" e possui um ícone de envelope à direita.
  - O segundo campo contém o texto "Sua senha" e possui um ícone de cadeado à direita.
- Um botão azul com o texto "Log in" em branco.

Figura 15 Formulário de login - Fonte: Autor

Este formulário de login, representado na figura 15, é responsável por permitir ao utilizador inserir as credenciais, *email* e senha que serão validados na tabela de utilizadores junto à base de dados.

```

100     <dependency>
101         <groupId>org.springframework.boot</groupId>
102         <artifactId>spring-boot-starter-security</artifactId>
103     </dependency>
104
105     <dependency>
106         <groupId>org.thymeleaf.extras</groupId>
107         <artifactId>thymeleaf-extras-springsecurity4</artifactId>
108     </dependency>

```

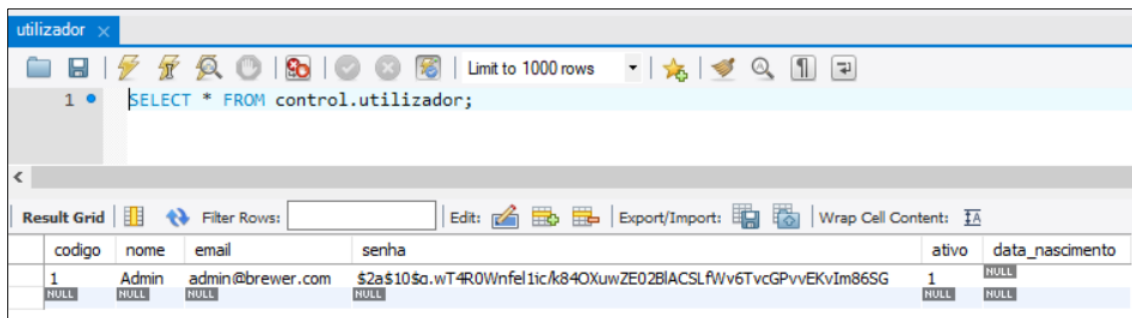
Figura 16 Dependências de segurança do spring framework - Fonte: Autor

Dentro do contexto deste projeto utilizou-se a solução *spring security*. Esta tecnologia ajuda nas configurações de segurança; para o seu funcionamento, basta que seja configurado as dependências, propriedades, repositórios e *plugins*, relacionadas à segurança para este contexto, no ficheiro *POM* o qual possui um formato *XML* que permite gerir estas informações, e que fica armazenado na estrutura do *framework spring*. As dependências apresentadas na figura 16 vão auxiliar na configuração de segurança da aplicação, tais como segurança para formulário de login e configurações de segurança por requisições das páginas do projeto e controlo de acesso tipificado na figura 15.

No seguimento das configurações de proteção da aplicação inseridas no ficheiro *POM*, na figura 15, pode-se observar a anotação ativa de segurança *@EnableWebSecurity*, ou seja, uma técnica que permite configurar proteção na classe *SecurityConfig.java*. Tipicamente o *spring security* consegue perceber que a classe foi anotada; facilitando codificar as regras de autorização e autenticação, esta anotação sinaliza que a aplicação está preparada para o funcionamento seguro.

Tal como referido na figura 15, quando o utilizador inserir as credencias no formulário de login e senha, estas informações de registo são geridas pelos métodos, *encode* e *matches*. O *encode* converte a senha simples na forma codificada e o método *matches*, faz o papel de comparar a senha no formato de texto simples informada pelo utilizador com a senha codificada - o construtor padrão para este contexto é o *BCryptPasswordEncoder*, responsável por gerir internamente na aplicação um *random salt*, ou seja, um valor criptograficamente forte e aleatório para ser usada como entrada e o serviço de codificação da senha que será usado no acesso do utilizador à aplicação. Tal como se representa na figura abaixo, um *SELECT \* FROM control.utilizador* na base de dados *Mysql*, permitirá identificar que o campo de senha não está

com texto simples, mas com um valor que o *SecureRandom* gerou aleatoriamente.

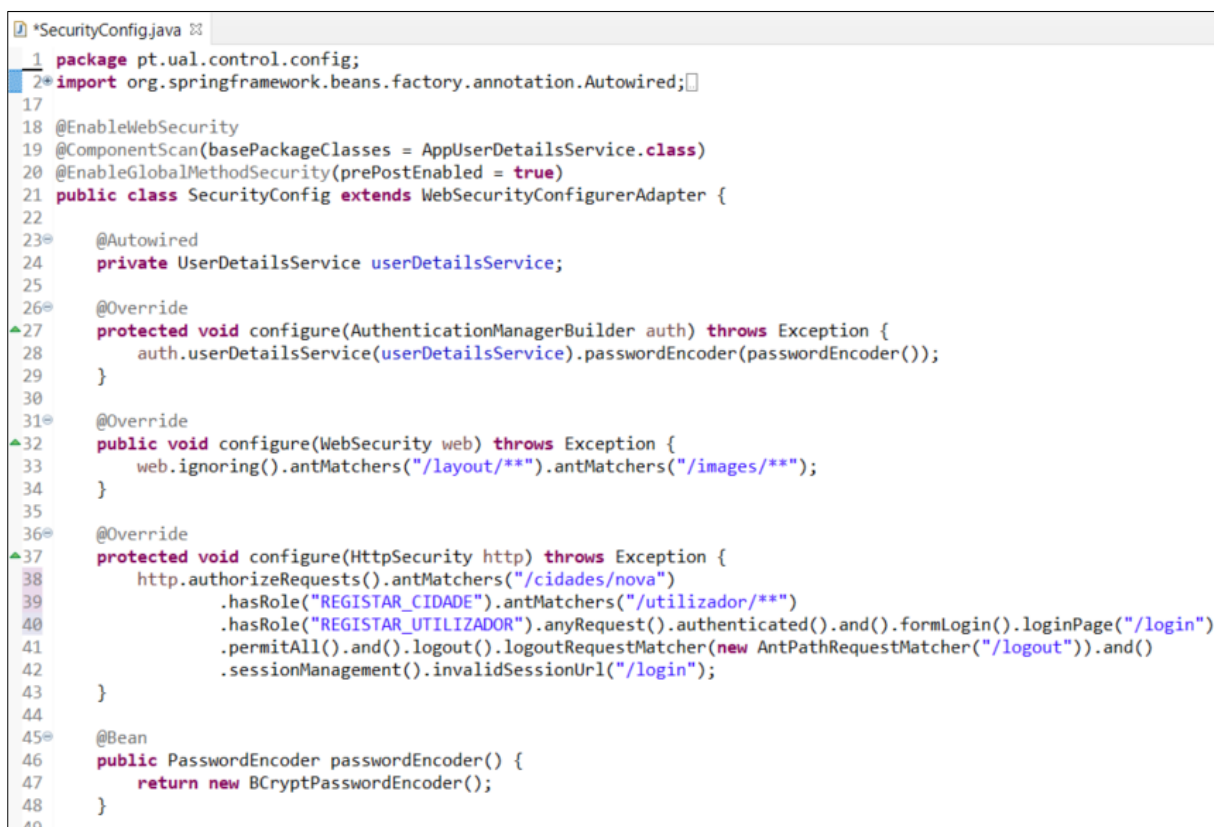


The screenshot shows a database query tool interface. At the top, there's a query editor with the SQL query: `SELECT * FROM control.utilizador;`. Below the query editor, there's a toolbar with various icons. The main area displays a table with the following data:

codigo	nome	email	senha	ativo	data_nascimento
1	Admin	admin@brewer.com	\$2a\$10\$0.wT4R0Wnfe1tic/k840XuwZE02BIACSLfWv6TvcGPvvEKvIm86SG	1	NULL

Figura 17 Base de dados - consulta utilizador - Fonte: Autor

Obviamente não é uma boa ideia guardar as credencias como texto simples na base de dados. É necessário supor que um invasor possa tentar roubar as credencias usando métodos de invasão tal como com o *SQL injection* - este modelo de ataque de injeção de *SQL* permite ao invasor interferir nas consultas que a aplicação realiza à respetiva base de dados, e esta visualização de informações pode facilitar a modificação das informações, o que, para prevenir este efeito, os dados armazenados estão cifrados, tal como o valor tipificado na coluna senha da figura 17, tabela utilizador.



```
*SecurityConfig.java
1 package pt.ual.control.config;
2 import org.springframework.beans.factory.annotation.Autowired;
17
18 @EnableWebSecurity
19 @ComponentScan(basePackageClasses = AppUserDetailsService.class)
20 @EnableGlobalMethodSecurity(prePostEnabled = true)
21 public class SecurityConfig extends WebSecurityConfigurerAdapter {
22
23     @Autowired
24     private UserDetailsService userDetailsService;
25
26     @Override
27     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
28         auth.userDetailsService(userDetailsService).passwordEncoder(passwordEncoder());
29     }
30
31     @Override
32     public void configure(WebSecurity web) throws Exception {
33         web.ignoring().antMatchers("/layout/**").antMatchers("/images/**");
34     }
35
36     @Override
37     protected void configure(HttpSecurity http) throws Exception {
38         http.authorizeRequests().antMatchers("/idades/nova")
39             .hasRole("REGISTAR_CIDADE").antMatchers("/utilizador/**")
40             .hasRole("REGISTAR_UTILIZADOR").anyRequest().authenticated().and().formLogin().loginPage("/login")
41             .permitAll().and().logout().logoutRequestMatcher(new AntPathRequestMatcher("/logout")).and()
42             .sessionManagement().invalidSessionUrl("/login");
43     }
44
45     @Bean
46     public PasswordEncoder passwordEncoder() {
47         return new BCryptPasswordEncoder();
48     }
49 }
```

Figura 18 Configuração de segurança - Fonte: Autor



Outro detalhe importante da segurança neste contexto de acessos, é demonstrado na figura 18, nas linhas de código 32 a 35, em que o método *configure(WebSecurity web)* fará a gestão dos acessos às páginas do projeto, as páginas que podem ser públicas ou ignoradas pelo sistema, ou seja para a estrutura (“/layout/\*\*”) todas as páginas e conteúdo que estiverem depois do *path layout* não serão restringidos pelo método acima, do mesmo modo para (“/imagens/\*\*”). Igualmente temos outro método *configure(HttpSecurity http)* cujo objetivo é trabalhar com as autorizações para as requisições *http*; de notar que nas linhas de código 38 a 42 há duas *hasRole*, (“REGISTAR\_CIDADE”) e (“REGISTAR\_UTILIZADOR”). O primeiro propósito destas instruções de código *java* é garantir que para gerir um registo, o utilizador, como especifica a regra *hasRole*, tem de ter acedido à aplicação com sucesso. Outro facto também importante e que envolve os aspetos da segurança é que se algum invasor tentar aceder à aplicação, colocando no seu navegador o endereço tal como <https://ualcontrol.herokuapp.com/registrarcliente>, sem ter primeiro feito login na aplicação, estes métodos de segurança direcionam o utilizador para o endereço <https://ualcontrol.herokuapp.com/login> aonde terá que passar pelos critérios de autenticação e autorização via base de dados no formulário de login tipificado na figura 15.

Um dos ataques mais utilizados é a falsificação de solicitação entre páginas de aplicações *web*, na qual o utilizador está autenticado no momento. Um invasor através da engenharia social, pode enviar um *link* por email, para enganar o utilizador da aplicação para que o mesmo execute ações à escolha do invasor. Este tipo de ataque, *Cross Site Request Forgery (CSRF)*, é realizado por solicitações *http*. Tipicamente, a solicitação *HTTP* da aplicação *web* do utilizador e a solicitação enviada do site do invasor são exatamente as mesmas. Isto demonstra que não há como negar um pedido vindo de um *site* não credível e permitir pedidos vindo de uma aplicação *web* no contexto do ataque. Desta forma o método de se proteger, é garantir que haja algo na solicitação feita pelo utilizador que não tenha no *site* não credível. Neste contexto, as configurações realizadas nas camadas *back-end* e *front-end*, conseguem diferenciar as duas requisições; neste caso, toda a solicitação *HTTP* quando é enviada, o servidor deverá consultar o *token CSRF* esperado e compará-lo com o *token* real feito na solicitação *http*: se os valores não corresponderem, a solicitação deverá ser negada; caso corresponda, a requisição será aceite e, após processada, entregue ao utilizador.

```
*LayoutPadrao.html
1 <!DOCTYPE html>
2 <html lang="pt" xmlns="http://www.w3.org/1999/xhtml"
3   xmlns:th="http://www.thymeleaf.org"
4   xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">
5 <head>
6   <meta charset="UTF-8"/>
7   <meta http-equiv="X-UA-Compatible" content="IE=edge"/>
8   <meta name="viewport" content="width=device-width, initial-scale=1"/>
9
10  <title>System Control</title>
11
12  <link rel="stylesheet" type="text/css" th:href="@{/layout/stylesheets/vendors.min.css}"/>
13  <link rel="stylesheet" type="text/css" th:href="@{/layout/stylesheets/control.min.css}"/>
14  <link rel="stylesheet" type="text/css" th:href="@{/layout/stylesheets/application.css}"/>
15  <link rel="stylesheet" type="text/css" th:href="@{/stylesheets/brewer.css}"/>
16
17  <link rel="stylesheet" type="text/css"
18    th:href="@{/stylesheets/vendors/bootstrap-datepicker.standalone.min.css}"/>
19
20  <link rel="shortcut icon" th:href="@{/images/favicon.png}"/>
21 </head>
22 <body>
23 <input type="hidden" name="_csrf" th:value="{_csrf.token}"/>
24 <input type="hidden" name="_csrf_header" th:value="{_csrf.headerName}"/>
```

Figura 19 Segurança no layout padrão - Fonte: Autor

Tal como referido anteriormente sobre o método de proteger a aplicação na camada *front-end*, a figura 19 acima permite ver, nas linhas 23 e 24 codificadas, a inserção de algo que irá permitir que a requisição só deverá ser aceite e processada pelo utilizador caso esteja de acordo com as validações do *token CSRF*. Isto acontece após a comparação do *token* esperado com o *token* real feito na solicitação *http*; o facto do campo nas linhas 23 e 24 codificadas estar apresentado como *hidden* é porque na tecnologia de *front-end thymeleaf* esta *tag* assume automaticamente como entrada oculta.

Para os ficheiros *JavaScript* foi criada uma função *control.Security* que irá gerir a leitura e os envios de informações nas requisições realizadas no uso da aplicação, uma vez que a camada *back-end* fica responsável pela emissão do *token* ao *front-end*. Deste modo, todas as páginas que são processadas no navegador pelo utilizador, estão devidamente configuradas com a proteção *CSRF*, tal como aparece na figura 20.

```

78 control.Security = (function() {
79
80     function Security() {
81         this.token = $('input[name=_csrf]').val();
82         this.header = $('input[name=_csrf_header]').val();
83     }
84
85     Security.prototype.enable = function() {
86         $(document).ajaxSend(function(event, jqxhr, settings) {
87             jqxhr.setRequestHeader(this.header, this.token);
88         }).bind(this);
89     }
90
91     return Security;
92

```

Figura 20 Função de segurança javascript - Fonte: Autor

Pelo facto de usarmos em diversos lugares do projeto requisições *Ajax*, a função de segurança criada no ficheiro *JavaScript* principal do projeto, tal como na figura 19, esta classe permite adicionar segurança para todas as requisições neste contexto, de modo que toda requisição *Ajax* feita pelo *jQuery* no *front-end* do projeto será adicionado no *header* a anotada da *tag* de segurança.

```

26 </head>
27 <body>
28 <input type="hidden" name="_csrf" value="4f642d89-c117-4a5d-92a5-1f8e07d7d49d"/>
29 <input type="hidden" name="_csrf_header" value="X-CSRF-TOKEN"/>
30
31 <div class="aw-layout-page">
32     <nav class="navbar navbar-fixed-top navbar-default js-sticky-reference" id="main-navbar">
33         <div class="container-fluid">
34
35             <div class="navbar-header">
36                 <a class="navbar-brand hidden-xs" href="#">

```

Figura 21 Código fonte da página de login - Fonte: Autor

A figura 21, acima apresentada, trata da página de login no formato *.html* que, após o *rendering*, ou seja, representada no navegador do utilizador, tem como características o código fonte. O que se pode observar nas linhas 27 e 28, é o resultado das configurações inseridas na figura 19, o que de facto resultou na geração do *CSRF-TOKEN*, permitindo assim autenticar o conteúdo da página; já neste contexto, as configurações realizadas na camada *back-end* e *front-end* conseguem diferenciar as requisições por conta do *token CSRF* gerido pela camada do servidor.

## 13. Implantação da Aplicação na *Cloud*

Esta secção tem o propósito de apresentar o processo de implantação da aplicação. Para este projeto usaremos uma plataforma *cloud* baseada num sistema de *container* gerido *Heroku Cloud*; nas secções a seguir será demonstrado passo a passo todo o processo, desde a ligação com a plataforma até à fase final dos *commit* - todas estas fases serão realizadas através de comandos no terminal do sistema operacional *Linux Ubuntu*.

### 13.1 *Heroku* Plataforma *Cloud*

Para este processo de publicação estaremos usando os serviços de hospedagem *cloud* da *heroku*, umas das primeiras plataformas para ambiente *cloud* que possui um ambiente orientado a serviço *Platform as a Service (PaaS)* e que suporta várias linguagens de programação (neste caso do projeto, linguagem *Java*), permitindo os desenvolvedores criar, executar e gerir aplicações inteiramente em *cloud*.

```
root@esantos: /mnt/c/Tools Developers/Projeto master/control
esantos@esantos:~$ sudo -s
[sudo] password for esantos:
root@esantos:~# heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/91991d9b-bd21-4884-8bbb-47b44f0c9678
Warning: spawn cmd.exe ENOENT
Warning: Cannot open browser.
Logging in... done
Logged in as arpasistema@gmail.com
root@esantos:~# heroku create ualcontrol
Creating ualcontrol... done
https://ualcontrol.herokuapp.com/ | https://git.heroku.com/ualcontrol.git
root@esantos:~# cd /mnt/c/Tools Developers/Projeto master/control/
root@esantos:/mnt/c/Tools Developers/Projeto master/control# git init
Initialized empty Git repository in /mnt/c/Tools Developers/Projeto master/control/.git/
root@esantos:/mnt/c/Tools Developers/Projeto master/control# ls .gitignore
.gitignore
root@esantos:/mnt/c/Tools Developers/Projeto master/control# cat .git
.git/
.gitignore
root@esantos:/mnt/c/Tools Developers/Projeto master/control# cat .gitignore
target/
.settings/
.classpath
.project
root@esantos:/mnt/c/Tools Developers/Projeto master/control# git add .
root@esantos:/mnt/c/Tools Developers/Projeto master/control# git -am "LastCommit"
```

Figura 22 Conexão plataforma - heroku: - Fonte: Autor

O propósito da figura 22 acima apresentada, é demonstrar a criação do repositório para envolver o projeto na *cloud heroku*; esta fase é composta por etapas, referidas abaixo:

1. *heroku login* - É uma instrução de comando para ligar a plataforma *cloud*
2. *heroku create ualcontrol* - É uma instrução de comando para criar um repositório denominado “*ualcontrol*”

Nota-se que a partir da instrução “*create*”, o ambiente *heroku* disponibiliza dois *links* de acesso, sendo o primeiro para aceder à aplicação no término do processo de publicação do projeto; visto ser esta fase a instância de ligação para a implantação, o segundo *link* refere-se ao local onde o projeto será armazenado no repositório do *GitHub*. Ainda na figura 22 observamos que após o sucesso nas instruções de comando anteriores citadas e representado pela figura já citada, também é feita a inicialização do serviço de repositório através da instrução de comando “*git init*”; outra fase importante é o processo da listagem de quais ficheiros e pasta do projeto desenvolvido que não será necessário ser feito. O seu *commit* para a *cloud*, através da instrução “*cat. gitignore*” permite observar a lista de ficheiros e pasta:

```
. target/  
. settings/  
. classpath,  
. project
```

Estes ficheiros e pasta que não serão enviados para o repositório acima criado, por não ser necessário, uma vez que o projeto foi empacotado pelo *framework* de desenvolvimento e compilação *IDE*, e já com o formato *.JAR*, contêm todas as classes *java* necessárias para o funcionamento na *cloud*. Por fim, a figura acima demonstra a finalização deste processo com a última instrução de comando, o “*git add .*”, *git -am "LastCommit"*; estas duas instruções de comando estão totalmente relacionadas com a figura 23 abaixo, da qual falaremos nas próximas secções com detalhes.

```
root@esantos:/mnt/c/Tools Developers/Projeto master/control# heroku addons:create jawsdb:kitefin -a, --app ualcontrol  
Creating jawsdb:kitefin on @ ualcontrol... free  
Database is being provisioned. Your config_var will be set automatically once available.  
Created jawsdb-rigid-78532 as JAWSDB_URL  
Use heroku addons:docs jawsdb to view documentation  
root@esantos:/mnt/c/Tools Developers/Projeto master/control# heroku config:set AWS_ACCESS_KEY_ID=AKIAJCK22WPIDD70PNTA AW  
S_SECRET_ACCESS_KEY=sy8/3BSyP8p2CDJ1CGUNv9TiTzN998/5wHepfecl  
> Error: Missing required flag:  
> -a, --app APP app to run command against  
> See more help with --help  
root@esantos:/mnt/c/Tools Developers/Projeto master/control# heroku config:set AWS_ACCESS_KEY_ID=AKIAJCK22WPIDD70PNTA AW  
S_SECRET_ACCESS_KEY=sy8/3BSyP8p2CDJ1CGUNv9TiTzN998/5wHepfecl -a, --app ualcontrol  
Setting AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY and restarting @ ualcontrol... done, v6  
AWS_ACCESS_KEY_ID: AKIAJCK22WPIDD70PNTA  
AWS_SECRET_ACCESS_KEY: sy8/3BSyP8p2CDJ1CGUNv9TiTzN998/5wHepfecl  
root@esantos:/mnt/c/Tools Developers/Projeto master/control# git status  
On branch master  
nothing to commit, working tree clean  
root@esantos:/mnt/c/Tools Developers/Projeto master/control# git push heroku master
```

Figura 23 Criando base de dados - heroku - Fonte: Autor

A figura 23 demonstra a continuidade do processo de publicação iniciada na figura 22. Tendo como objetivo a criação da base de dados na *cloud* na plataforma *heroku* através da instrução de comando a seguir: “*heroku addons:create jawsdb:kitefin -a, --app ualcontrol*”. Tal como no ambiente de desenvolvimento na máquina local, a base de dados será o Mysql para o ambiente de produção na *cloud*. Outro aspeto importante nesta fase de migração é o comando, *heroku config:set AWS\_KEY\_ID=\*\*\*\*\*AWS\_SECRET\_ACCESS\_KEY=\*\*\*\*\* -a, --app ualcontrol*, instrução de configuração do serviço *Amazon S3* permitindo a realização de *upload* e *download* e gestão de objetos. Este *ID* e chave gerados pela *Amazon S3* é a chave de acesso ao serviço da *Amazon Simple Storage Service (Amazon S3)* que será aplicado dentro das configurações da *cloud heroku*, funcionalidade que permitirá a realização do *upload* de imagem para o produto no seu registo.

```

root@esantos:/mnt/c/Tools_Developers/Projeto_master/control# git commit -am "LastCommit"
[master (root-commit) 64d1ef4] LastCommit
Committer: root <root@arpasistema@gmail.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

246 files changed, 50619 insertions(+)
create mode 100644 .gitignore
create mode 100644 Procfile
create mode 100644 pom.xml
create mode 100644 src/main/java/pt/ual/control/PrincipalApplication.java
create mode 100644 src/main/java/pt/ual/control/config/MailConfig.java
create mode 100644 src/main/java/pt/ual/control/config/S3Config.java
create mode 100644 src/main/java/pt/ual/control/config/SecurityConfig.java
create mode 100644 src/main/java/pt/ual/control/config/WebConfig.java
create mode 100644 src/main/java/pt/ual/control/config/format/BigDecimalFormatter.java
create mode 100644 src/main/java/pt/ual/control/config/format/IntegerFormatter.java
create mode 100644 src/main/java/pt/ual/control/config/format/LocalDateFormatter.java
create mode 100644 src/main/java/pt/ual/control/config/format/LocalDateTimeFormatter.java
create mode 100644 src/main/java/pt/ual/control/config/format/LocalTimeFormatter.java
create mode 100644 src/main/java/pt/ual/control/config/format/NumberFormatter.java
create mode 100644 src/main/java/pt/ual/control/config/format/TemporalFormatter.java
create mode 100644 src/main/java/pt/ual/control/controller/CervejasController.java
create mode 100644 src/main/java/pt/ual/control/controller/CidadesController.java
create mode 100644 src/main/java/pt/ual/control/controller/CientesController.java
create mode 100644 src/main/java/pt/ual/control/controller/DashboardController.java
create mode 100644 src/main/java/pt/ual/control/controller/EstilosController.java
create mode 100644 src/main/java/pt/ual/control/controller/FotosController.java
create mode 100644 src/main/java/pt/ual/control/controller/RelatoriosController.java
create mode 100644 src/main/java/pt/ual/control/controller/SegurancaController.java
create mode 100644 src/main/java/pt/ual/control/controller/UsuariosController.java
create mode 100644 src/main/java/pt/ual/control/controller/VendasController.java
create mode 100644 src/main/java/pt/ual/control/controller/converter/CidadeConverter.java
create mode 100644 src/main/java/pt/ual/control/controller/converter/EstadoConverter.java

```

Figura 24 Preparação projeto envio - cloud heroku - Fonte: Autor

Nesta sequência de instruções de comando para a fase final do processo da publicação, tanto a figura 24 como a figura 23, tratam das instruções de comandos para a plataforma *cloud heroku*, tal como o comando *git add .*, “*git -am "LastCommit"*” e a instrução “*git -am*”. Esta

tem o propósito de adicionar todas as classes a arquivos dentro do projeto, exceto os arquivos ignorados pelo comando `cat .gitignore`, como representado acima na figura 24; nesta sequência, a instrução `git -am "LastCommit"`, por sua vez, permite iniciar a publicação dando o `commit` para todo o projeto com as suas respectivas exceções já definidas, como se pode ver na figura acima. Nota-se que ao passo que o `commit` é realizado, a figura demonstra as classes `.java` sendo enviadas para o ambiente `cloud`.

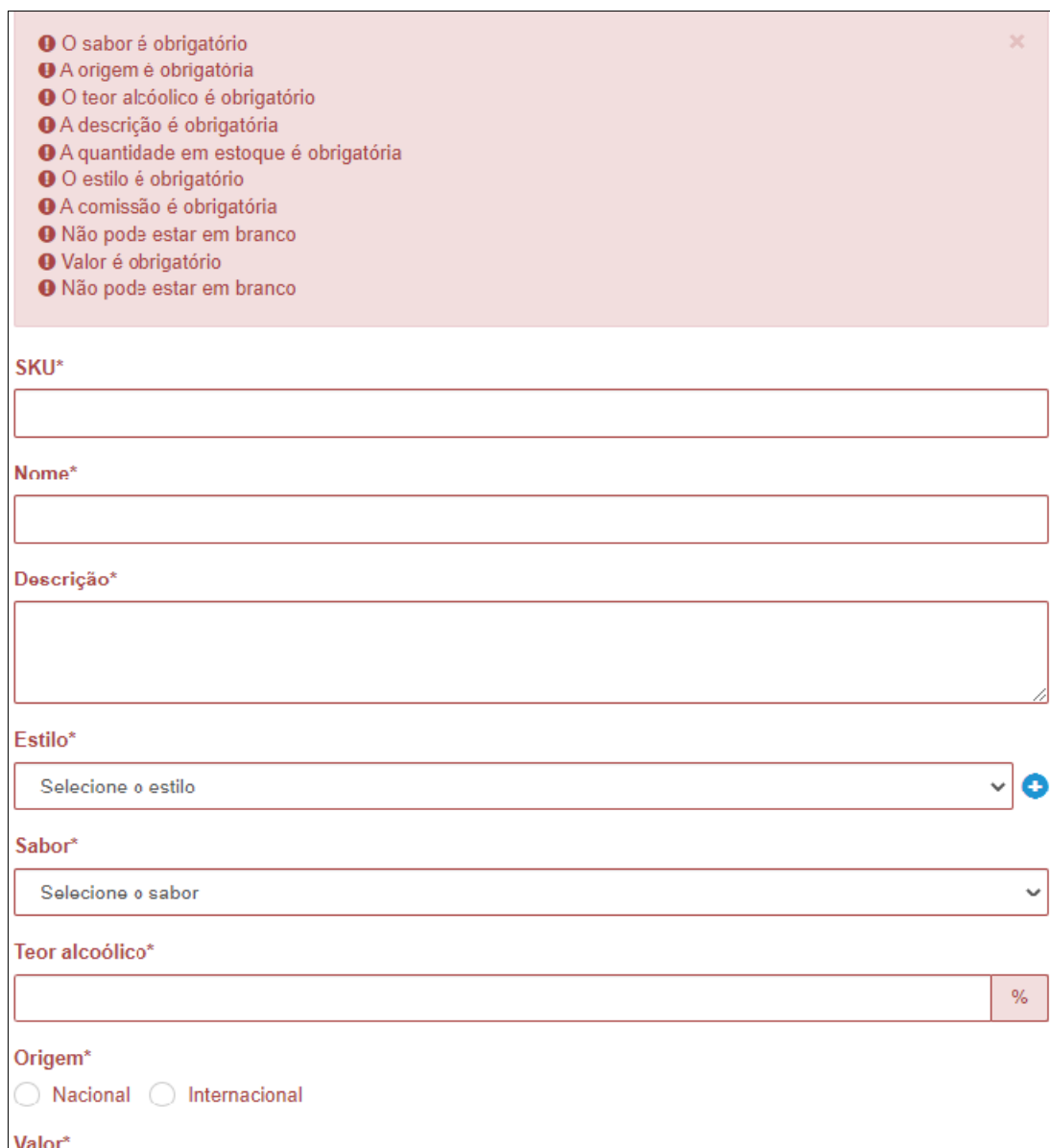
```
remote: [INFO] -----
remote: [INFO] BUILD SUCCESS
remote: [INFO] -----
remote: [INFO] Total time: 01:11 min
remote: [INFO] Finished at: 2020-04-19T22:35:48Z
remote: [INFO] -----
remote: ----> Discovering process types
remote: Procfile declares types -> web
remote:
remote: ----> Compressing...
remote: Done: 126.4M
remote: ----> Launching...
remote: Released v7
remote: https://ualcontrol.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/ualcontrol.git
 * [new branch] master -> master
root@esantos:/mnt/c/Tools Developers/Projeto master/control# heroku open
> Error opening web browser.
> Error: Exited with code 3
>
> Manually visit https://ualcontrol.herokuapp.com/ in your browser.
root@esantos:/mnt/c/Tools Developers/Projeto master/control#
```

Figura 25 Implantação - cloud heroku - Fonte: Autor

A figura 25 é para a fase da publicação do último `printscreen`, visto que nesta figura se permite observar a mensagem de retorno da plataforma e serviços da `heroku cloud` “*BUILD SUCCESS*”, revelando o tempo decorrido para realização da publicação, tamanho da compressão dos arquivos enviados e a versão da `release v7`. Nota-se na figura 24 e na figura 25 a aplicação sendo configurada e provisionada para o funcionamento, disponibilizada com as suas funcionalidades através da plataforma `heroku`. Por fim, nesta fase de publicação, observamos na figura acima, que para além dos `outputs` obtidos, também é retornado o `link` da publicação para a `cloud heroku`; este `link` permite aceder à aplicação de modo a poder utilizá-la onde se dispuser de um acesso à internet.

## 13.2 Teste da Aplicação Web na Cloud

O diagrama da figura 6 onde consta o processo de desenvolvimento, apresenta várias fases que envolvem a conclusão deste projeto; nesta secção abordaremos as fases, teste e correções de bugs e teste de usabilidade. Para os testes de erros do projeto, em primeiro lugar, identificou-se a publicação do projeto na nuvem, após a validação de possíveis erros na transferência da aplicação para *cloud*; outro detalhe importante a nível de base de dados, se as tabelas estavam devidamente criadas, se as requisições feitas pelo utilizador de teste estavam sendo comunicadas com a base de dados, como por exemplo inserir um registo, alterar um registo, cancelar um registo ou apenas consultar um registo. Do mesmo modo para o teste de usabilidade, tal como demonstra a figura abaixo.



The image shows a web application form with a validation error list at the top. The error list contains the following items:

- ❗ O sabor é obrigatório
- ❗ A origem é obrigatória
- ❗ O teor alcoólico é obrigatório
- ❗ A descrição é obrigatória
- ❗ A quantidade em estoque é obrigatória
- ❗ O estilo é obrigatório
- ❗ A comissão é obrigatória
- ❗ Não pode estar em branco
- ❗ Valor é obrigatório
- ❗ Não pode estar em branco

Below the error list, the form fields are:

- SKU\***: A text input field.
- Nome\***: A text input field.
- Descrição\***: A text area input field.
- Estilo\***: A dropdown menu with the text "Selecione o estilo" and a blue plus icon on the right.
- Sabor\***: A dropdown menu with the text "Selecione o sabor" and a downward arrow on the right.
- Teor alcoólico\***: A text input field with a percentage sign (%) on the right.
- Origem\***: Two radio buttons labeled "Nacional" and "Internacional".
- Valor\***: A text input field.

Figura 26 Teste de usabilidade - Fonte: Autor



Tal como tipifica a figura 26, o seu objetivo é mostrar que dentro do conceito da usabilidade de *software*, um dos elementos importante é o relacionamento da aplicação com quem está a gerir; esta interação tem que ser legível até mesmo quando acontece erros tal como a figura 25 - o que ocorre na figura é que o utilizador de teste tentou salvar o formulário de registo de bebidas sem qualquer informação, o que demonstra que esta solução está preparada para tratamento de erros ou, a exemplo de algum campo obrigatório ou valor incorreto, as configurações inseridas no *back-end* apresentarão ao utilizador o erro cometido e como resolver.

A apresentação desta aplicação acontecerá consoante as páginas mais importantes que julgamos ser para este projeto, como veremos abaixo. Esta aplicação foi testada em diversos dispositivos de acesso à internet; para esta secção será disponibilizado figuras de apresentação para os dispositivos, *desktop*, *smartphone* e *iPad*.

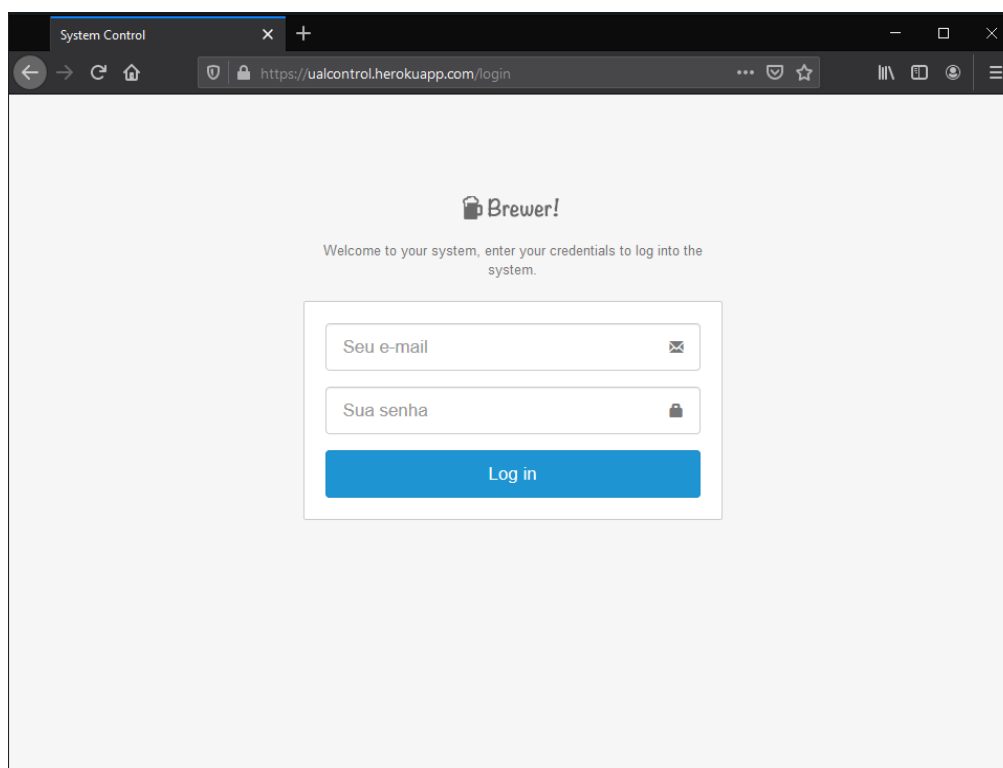
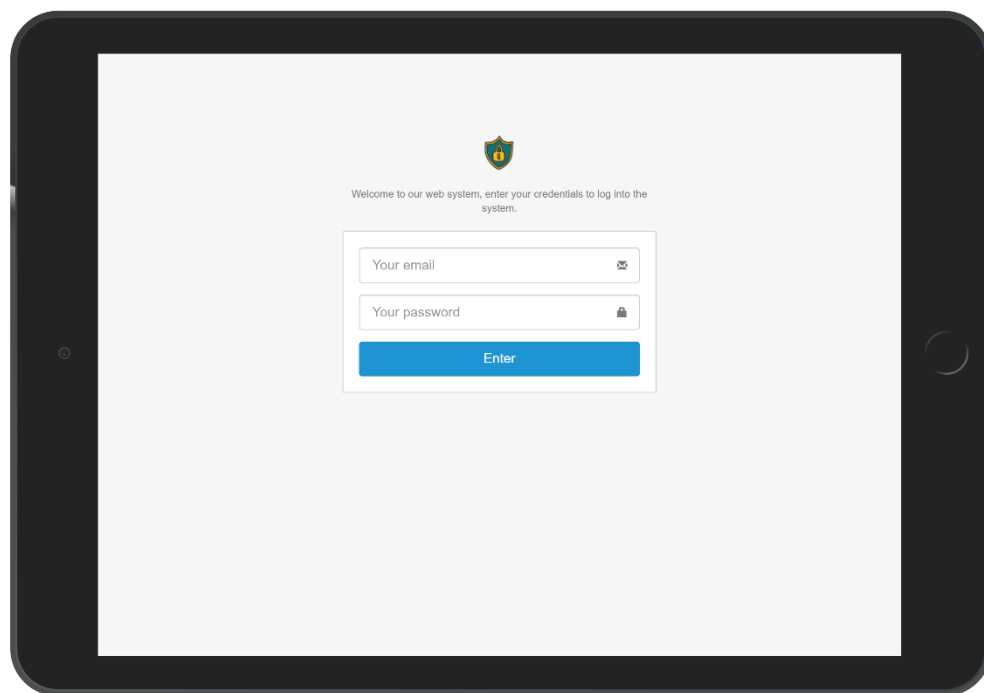


Figura 27 Aplicação funcionando - cloud heroku - Fonte: Autor

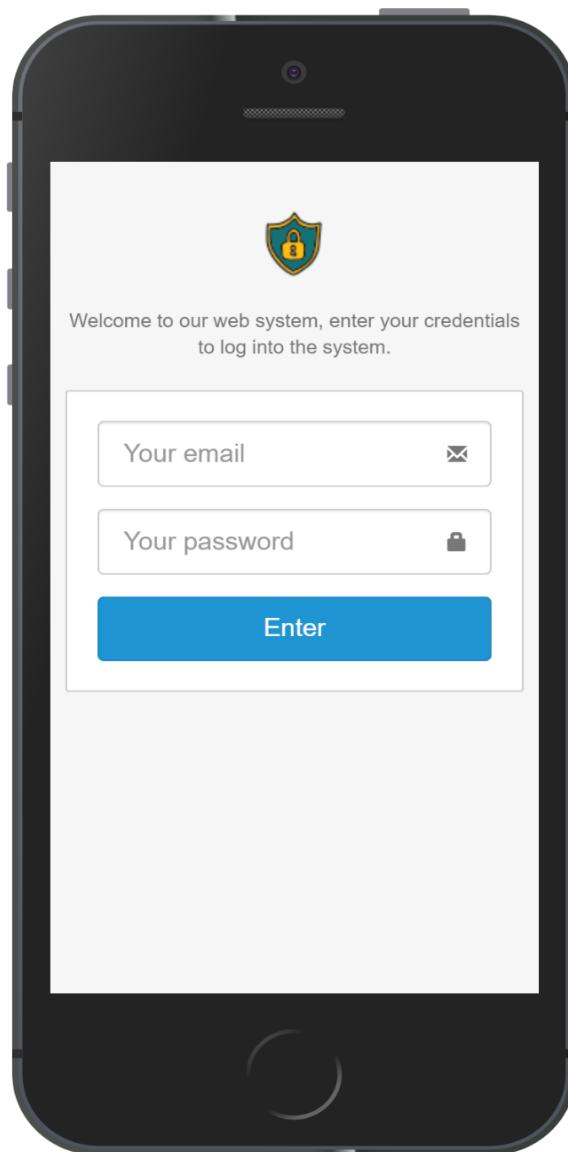
Como pode ser observado na figura 27, a aplicação foi acedida em um dispositivo *desktop*, através do *link* <https://ualcontrol.herokuapp.com/>; este endereço eletrónico estipula, como regra estabelecida nas configurações da aplicação, sempre conduzir o utilizador para o

formulário de login. Neste formulário quem está a gerir poderá aceder com as credencias conferidas, tal como email, e senha.



*Figura 28 Página de login do sistema para iPad - Fonte: Autor*

Na figura 28, é apresentado a tela de *login* do sistema na horizontal para o dispositivo *iPad*. Com o propósito de demonstrar que, para além de ser responsivo, este sistema também permite flexibilidade, seja para o uso na vertical ou horizontal dos dispositivos em utilização.



*Figura 29 Página de login do sistema smartphone - Fonte: Autor*

Tal como na figura 28, esta figura 29 permite apresentar o sistema no formulário de login para o dispositivo *smartphone*; esta demonstração mais uma vez confirma a responsividade que a aplicação oferece. Neste contexto as figuras aqui tipificadas vão além de apresentar esta aplicação, atestar que as configurações inseridas dentro do projeto têm refletido as suas funcionalidades.

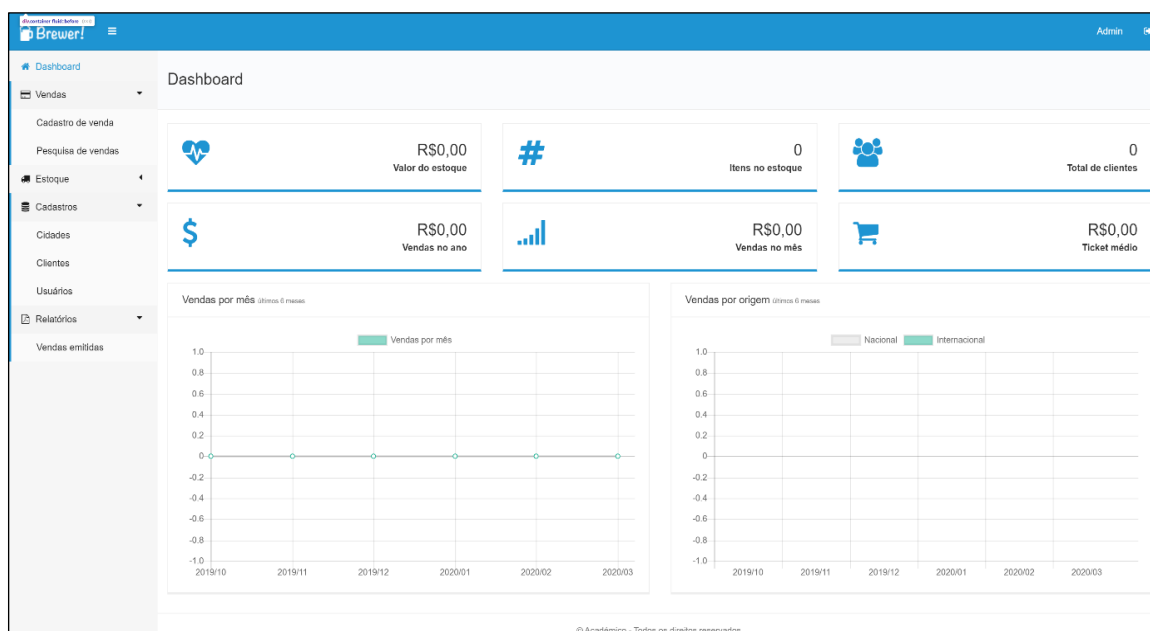


Figura 30 Dashboard do sistema no dispositivo desktop - Fonte: Autor

Já na figura 30 é apresentado o ambiente de trabalho da aplicação, ambiente que permite ao utilizador de acordo com seu nível de acesso, realizar tarefas disponibilizadas no contexto do sistema, como registros, consultas, emissão de relatórios, tal como nos requisitos de casos de usos no apêndice A. Na sequência da figura acima, o sistema disponibiliza ao utilizador um gráfico com diversas informações importantes; estes gráficos, de acordo com a movimentação que é realizada na aplicação, funcionam gerando indicadores, sendo eles: vendas por mês, vendas por origem, valores dos produtos que compõe o stock, quantidade de itens do stock, valores ao mês, quantidade média de ticket e total de clientes, visando desta forma os gráficos facilitar a compreensão do gestor do sistema dos processos realizados.

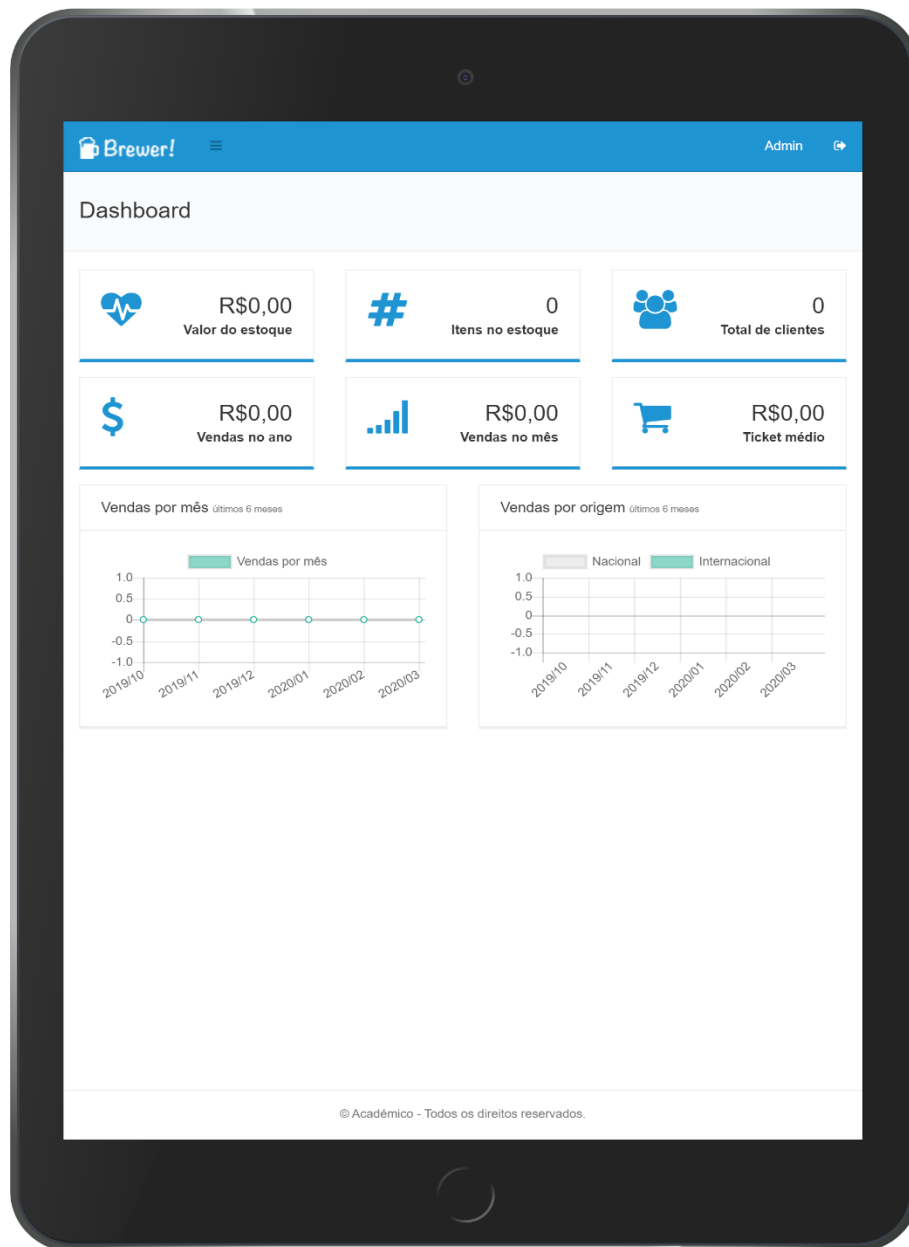


Figura 31 Dashboard do sistema no dispositivo iPad. - Fonte: Autor

Tal como na figura 30, a figura 31 tipifica um dispositivo móvel *iPad*. As funcionalidades do sistema são as mesmas, sendo a diferença a dinâmica da aplicação *web* responsiva que, ao ser executada em dispositivos diferentes, assume comportamentos diferentes. Como neste caso em questão, nota-se que os módulos de menus do sistema estão aninhados no topo, podendo ser expandidos ao clique do rato, ao contrário da figura 30 que, por padrão, apresenta ao utilizador todas as funcionalidades - isto acontece por causa do tamanho da resolução do dispositivo. Trata-se do *designer web* responsivo (*RWD*) para esta técnica.

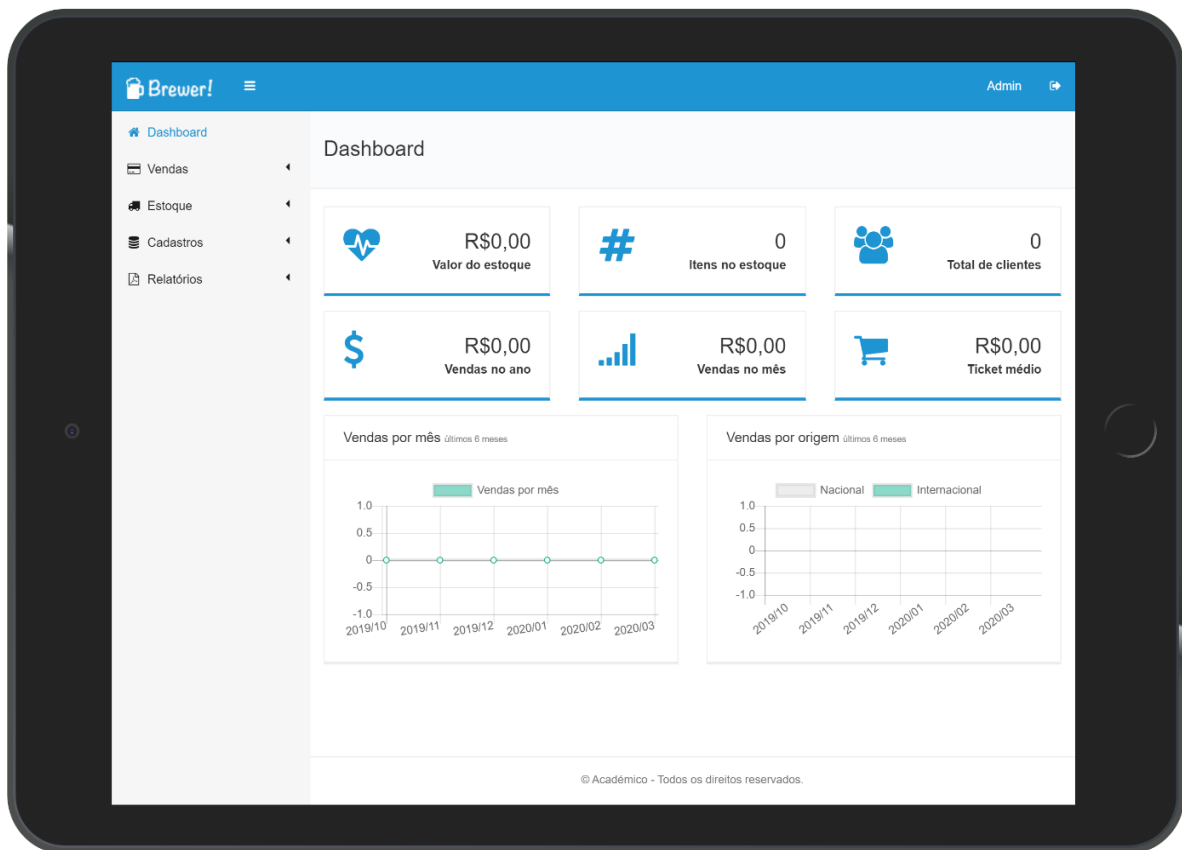


Figura 32 Dashboard do sistema no dispositivo iPad - Fonte: Autor

Tal como as figuras 30 e 31 que apresentam semelhanças nos seus ambientes, o objetivo desta figura 32 é representar o *dashboard* do sistema no dispositivo *mobile iPad*, na forma horizontal. Nota-se que apenas o facto de trabalhar com o equipamento móvel na horizontal o sistema também envolverá esta ação, de maneira responsiva, a disponibilizar o módulo com a lista de menus que outrora na figura 31 estava aninhada; neste caso, como vemos, as funcionalidades do sistema estão todas visíveis para serem executadas.

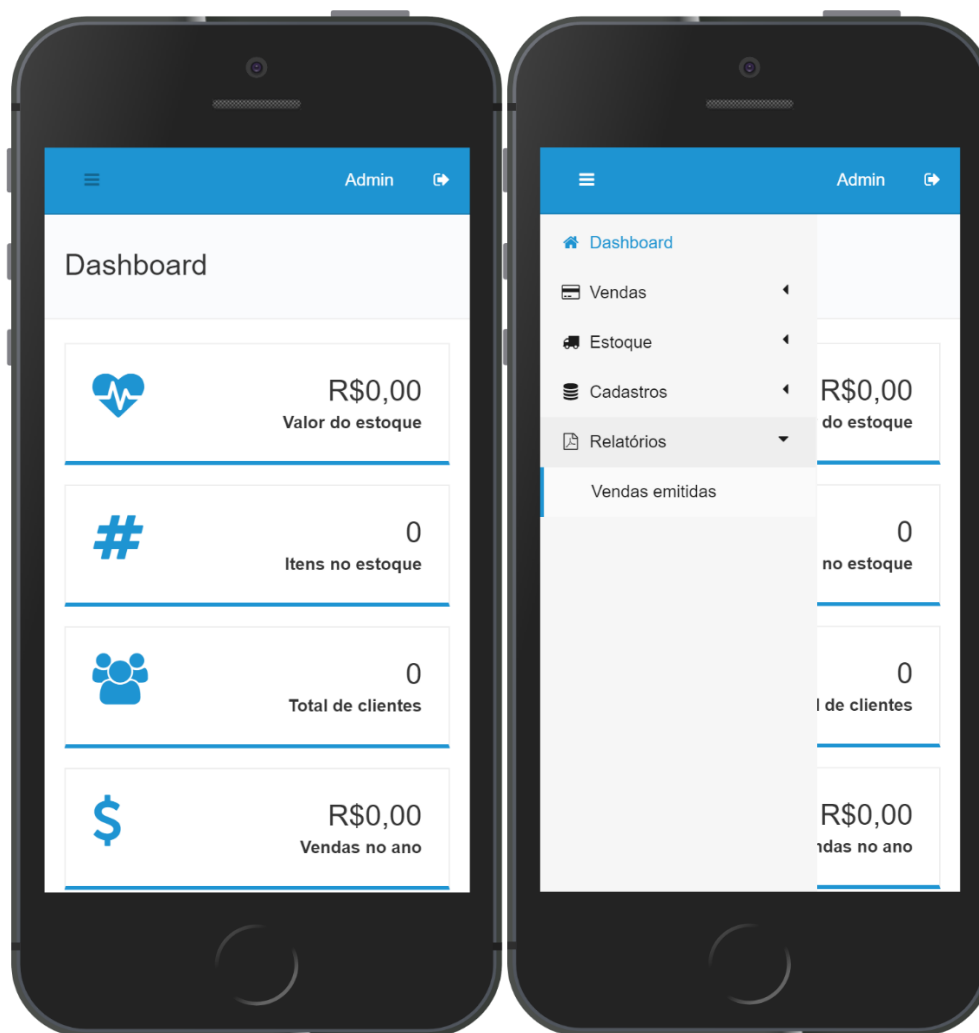


Figura 33 Dashboard do sistema no dispositivo *smartphone*- Fonte: Autor

Finalmente, esta apresentação tipificada na figura 33, refere-se à aplicação em funcionamento através do dispositivo móvel *smartphone*; o que se pode observar nesta figura com dispositivos em paralelo, é que no aparelho do lado esquerdo temos o dispositivo móvel com o módulo de menus aninhados e do mesmo modo no aparelho do lado direito temos já o módulo de menus expandidos permitindo a execução de ambos os menus e submenus. O objetivo desta imagem é demonstrar que embora seja pequeno o ecrã com uma resolução de 320px por 568px ainda assim o sistema desenvolvido por este projeto permite compor em consonância com as diversas resoluções de ecrã. Desta forma, a figura acima embora não esteja visível na imagem, tem a possibilidade de usar o *scroll down* da aplicação e percorrer todo o conteúdo da página.

## 14. Conclusão

Esta secção permite apresentar as considerações finais, limitações do projeto bem como uma proposta para trabalhos futuros do projeto.

### 14.1 Considerações Finais

Ao longo deste trabalho de projeto de mestrado, foi apresentada e implementada uma ferramenta que auxilia na gestão comercial de uma pequena e média empresa, para a área de distribuição de bebidas. Identificou-se que os objetivos propostos no planeamento deste projeto foram alcançados, com o uso dos conceitos e princípios de usabilidade de *software* e com o *designer* responsivo.

Existem muitas tecnologias e soluções na atualidade que apoiam os diversos seguimentos comerciais como aplicações *web* que permitem o seu acesso a partir do *browser*, aplicações nativas otimizadas para sistemas operacionais específicos, aplicações híbridas, sendo, todavia, na sua maioria, específicas aos requisitos de determinados dispositivos em detrimento da versão do seu sistema operativo. Neste caso, excepcional, a solução proposta, traz como contributo o seu uso independente do local e do sistema operativo do equipamento bem como não depender de estar em loja de repositório tal como os da *play store* da *google* ou *app store* da *apple*.

Do ponto de vista da calendarização do projeto, observou-se que os prazos foram cumpridos. No período do desenvolvimento foram identificados recursos e novas tecnologias que facilitaram a agenda para o desenvolvimento do projeto *web*, tal como as novas versões do *framework spring boot*, permitindo o aprimoramento no desenvolvimento.

Compreendeu-se também que este projeto pode ser melhorado e aperfeiçoado para os dois ambientes de desenvolvimento, *back-end* e *front-end*, como tipificado na secção 14.3 dos trabalhos futuros.

O desenvolvimento de um projeto desta índole, com mais de cento e cinquenta classes *java*, mais de vinte e seis ficheiros com instruções *JavaScript*, muitíssimos ficheiros em *HTML* e *CSS*, requer muito tempo e boa metodologia no desenvolvimento.

As dificuldades e limitações encontradas ao longo do desenvolvimento deste projeto serão descritas seguidamente.



## 14.2 Limitações do Projeto

Não obstante as tecnologias e soluções existentes na atualidade, tal como as nativas e híbridas, estas apresentam diversas limitações em desenvolvimento e utilização.

Neste contexto, também se referem algumas limitações para o uso da metodologia escolhida, o da aplicação *web* responsiva, sendo indispensável a sua utilização sem o acesso à internet seja *WiFi* ou até mesmo no uso dos dados móveis; outro problema, é que se apresentar intermitência por um período de tempo de uso, isto causará indisponibilidade para a aplicação, o que impedirá toda a gestão do *software*.

Também é ciente que esta solução por ser independente não é possível disponibilizá-la numa loja de repositório tal como os da *play store* da *google* ou *app store* da *apple* e etc., carece de ser acedida apenas pelo *browser* do dispositivo do utilizador.

Por fim, até ao momento de conclusão deste projeto, não foi possível prever uma forma de ter a aplicação *web* em funcionamento sem estar ligada a uma rede de internet ou dados móveis.

## 14.3 Trabalho futuro

De forma a melhor potenciar esta aplicação e que a mesma se torne mais completa e melhor habilite o utilizador e responda às suas necessidades, foram identificados, como trabalho futuro, os seguintes pontos:

- Implementar uma metodologia que permita o seu uso *offline*, de modo que não sofra intermitência visto que, neste caso presente, esta aplicação *web* não contempla esta funcionalidade.

O desenvolvimento do projeto seguiu as especificações propostas na sua análise, mas serão necessárias melhorias na gestão da aplicação, como se refere nos tópicos abaixo descritos:

- Gestão de envio de *emails* pela aplicação;
- Emissão de documentos fiscais através da aplicação;
- A aplicação *web* não interage com as funcionalidades dos dispositivos - planejar estas melhorias;
- Relatórios de vendas com comissões por vendedores, aplicando o conceito de metas atingidas;
- Relatórios dos clientes que mais realizaram compras mensal e anualmente;

- Dinamizar o módulo de stock, permitindo o inventário dos produtos a serem registados;
- Melhorar a gestão de registo de pessoas, proporcionando mais alternativas no âmbito do registo, como a integração com agenda;

Estes seriam os pontos de maior relevância que permitiriam adicionar valor para esta aplicação *web* responsiva.

## 15. Bibliografia

- [1] B. A. Tate, *Beyond Java: A Glimpse at the Future of Programming Languages*, O'Reilly, 2005.
- [2] Banco de Portugal, “bportugal”, Banco de Portugal Eurosistema, Nov 2019. [Online].Disponível: <https://www.bportugal.pt/comunicado/nota-de-informacao-estatistica-analise-das-empresas-da-industria-das-bebidas-2017>. [Acedido em 23 Abr, 2021].
- [3] D. P. Lacerda, A. Dresch e A. Proença, “Design Science Research: método de pesquisa para a engenharia de produção”, *Gestão & Produção*, vol. 20, nº 4, pp. 741 - 761, 2013.
- [4] J. V. Brocke e A. Maedche, “Electronic Markets”, *The DSR grid: six core dimensions for effectively planning and communicating design science research projects*, pp. 379-385, 29 July, 2019.
- [5] S. A. Gregory, “A Review.of: “The Design Method””, *Taylor Francis Online*, vol. 10, nº 3, pp. 364-365, 2010.
- [6] R. Winter, “DESRIST: International Conference on Design Science Research in Information Systems”, em *5th International Conference, DESRIST 2010, St., Gallen, Switzerland*, 2010.
- [7] F. Almeida e J. A. Monteiro, “Approaches and Principles for UX Web Experiences: A Case Study Approach”, *International Journal of Information Technology and Web Engineering (IJITWE)*, vol. 12, nº 2, pp. 49-65, 2017.
- [8] M. Nebeling e M. C. Norrie, “Responsive Design and Development: Methods, Technologies and Current Issues”, *Web Engineering . ICWE 2013. Lecture Notes in Computer Science*, 2013.
- [9] E. Marcotte, “Responsive-Web-Design”, 25 May 2010. [Online]. Disponível: <https://alistapart.com/article/responsive-web-design/>. [Acedido em 15 Fevereiro, 2020].
- [10] Giurgiu e G. Ilie, "Responsive Web Design Techniques. International conference Knowledge-Based Organization", pp. 68-204, 2017.

- [11] M. H. Baturay e M. Birtane, “Responsive web design: a new type of design for web based instructional content”, *Procedia – Social and Behavioural Sciences vol 106*, pp. 2275-2279, December, 2013.
- [12] B. Kim, “Responsive Web Design, Discoverability, and Mobile Challenge”, *The Library Mobile Experience: Practices and User Expectations vol. 49, no. 6*, pp. 29-39, August /September, 2013.
- [13] N. Singh, M. Giri e S. Mathew, ““Resposive Website" A transformation in web designing”, *International Journal of Engineering Technology, Management and Applied Sciences vol 3*, January, 2015.
- [14] A. Y. Chun, S. G. Heeringa e B. S. , “Responsive and Adaptive Design for Survey Optimization”, *Journal of Official Statistics*, vol. 34, nº 3, pp. 581-597, 2018.
- [15] F. Shahzad, “Modern and Responsive Mobile-enabled Web Applications”, *Procedia Computer Science Vol 110*, pp. 410-415, 2017.
- [16] R. Kaur e B. Sharma, “Comparative Study for Evaluating the Usability of Web Based Applications”, em *4th International Conference on Computing Sciences (ICCS)*, Jalandhar, India, 2018.
- [17] J. Nielsen, Usability engineering, Morgan Kaufmann Publishers Inc., 1993.
- [18] S. . S. Tandel e A. Jamadar, “Impact of Progressive Web Apps on Web App”, *International Journal of Innovative Research in Science*, vol. 7, nº 9, pp. 439-444, 2018.
- [19] C. P. Dinata e Y. N. Marlim, “Application of Dynamic Systems Development Method in WEB-Based Promotion Media”, *Journal of Applied Business and Technology*, vol. 1, nº 3, pp. 196-204, 2020.
- [20] T. Walsh, N. Paciorek e D. Wong, “Security and reliability in Concordia/sup TM/”, *Proceedings of the Thirty-First Hawaii International Conference on System Sciences. Vol. 7*, 1998.
- [21] S. L. Henry e M. Grossnickle, “Accessibility in the user-centered design process”, 2004. [Online]. Disponível: <http://www.uiaccess.com/accessucd/background.html#ref1>. [Acedido em 10 Fevereiro 2020].

- [22] X. Fenga, W. Liu e Y. jiang, “A model-driven system interface design method based on MVC pattern”, *Journal of Physics: Conference Series*, vol. 1549, n° 4, pp. 42-75, 2020.
- [23] S. Kunjumohamed, H. Sattari, A. Bretet e G. Warin, *Spring MVC: Designing Real-World Web Applications*, Packt Publishing, 2016.
- [24] Primavera Software, “primaverabss”, Primavera, 2020. [Online]. Disponível: <https://pt.primaverabss.com/pt/software/solucoes-especializadas/forca-de-vendas/crm/crm/#atividades>. [Acedido em 23 abril, 2021].
- [25] PHC Software, “phcsc”, PHC Software, 2020. [Online]. Disponível: <https://phccs.net/porque-erp-phc-cs/>. [Acedido em 23 abril, 2021].
- [26] CentralGest ERP, “centralgest”, CentralGest ERP, 2021. [Online]. Disponível: <https://www.centralgest.com/software/comercial>. [Acedido em 23 abril, 2021].
- [27] H. Deitel e P. Deitel, *Java Como Programar*, Bookman, 2003.
- [28] D. H. Luckow e A. A. d. Melo, *Programação Java para a Web*, novatec, 2010.
- [29] R. W. Sebesta, *Conceitos De Linguagens De Programação*, Bookman, 2018.
- [30] D. Kramer, “API documentation from source code comments: a case study of Javadoc”, *Proceedings of the 17th annual international conference on Computer documentation*, pp. 147-153, October, 1997.
- [31] Baeldung, “Spring Template-engines”, 2020. [Online]. Disponível: <https://www.baeldung.com/spring-template-engines>. [Acedido em 17 fevereiro , 2020].
- [32] F. P. Miller, A. F. Vandome e J. McBrewster, *Apache Maven*, Alpha Press, 2010.
- [33] J. Lalou, *Apache Maven Dependency Management*, Packt , 2013.
- [34] L. Vogel, “Apache Tomcat”, 2008. [Online]. Disponível: <https://www.vogella.com/tutorials/ApacheTomcat/article.html>. [Acedido em 09 Fevereiro, 2020].
- [35] B. Alex e L. Taylor, “Spring Security”, 19 June 2018. [Online]. Disponível: <https://docs.spring.io/spring-security/site/docs/3.2.0.CI-SNAPSHOT/reference/html/index.html>. [Acedido em 25 Fevereiro, 2020].
- [36] P. Fisher e B. D. Murphy, *Spring Persistence with Hibernate*, Apress, Berkeley, CA, 2016.

- [37] M. Tyson, “JPA-Introduction-to-the-Java-Persistence-API”, Java Developer, JavaWorld, April, 2019. [Online]. Disponível: <https://www.infoworld.com/article/3379043/what-is-jpa-introduction-to-the-java-persistence-api.html>. [Acedido em 09 Fevereiro, 2020].
- [38] Kisman e S. M. Isa, “Hibernate ORM query simplification using hibernate criteria extension (HCE)”, *2016 3rd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS)*, pp. 23-28, 14-16 September, 2016.
- [39] M. Jurisic, “Flyway - Database Migrations Made Easy”, em *JavaCroc Methodologies & Tools*, 2016.
- [40] Spring Pivotal, “Spring Pivotal - JPA Repositories”, 2008. [Online]. Disponível: <https://docs.spring.io/spring-data/jpa/docs/current/reference/html/#jpa.repositories>. [Acedido em 19 Fevereiro 2020].
- [41] R. Sheldon e G. Moes, *Beginning MySQL*, wrox, 2005.
- [42] P. DuBois, *MySQL*, New RidersS, 1999.
- [43] M. B. Hoy, “HTML5: a new standard for the Web”, *Medical Reference Services Quarterly*, pp. 50-55, 24 January, 2011.
- [44] G. Anthes, “HTML5 leads a web revolution vol. 55 No. 7, pp. 16-17,” July, 2012. [Online]. Disponível: <https://cacm.acm.org/magazines/2012/7/151236-html5-leads-a-web-revolution/fulltext>. [Acedido em 24 Fevereiro, 2020].
- [45] D. S. McFarlandv, *CSS3: The Missing Manual*, O'Reilly, 2012.
- [46] E. A. Meyer, *Cascading Style Sheets: The Definitive Guide*, O'Reilly, 2000.
- [47] TutorialRepublic, “Republic Tutorial, Bootstrap”, 2020. [Online]. Disponível: <https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/bootstrap-get-started.php>. [Acedido em 16 Fevereiro, 2020].
- [48] M. Otto e J. Thornton, “Dicas sobre o Bootstrap”, 2019. [Online]. Disponível: <https://getbootstrap.com/docs/5.0/getting-started/introduction/>. [Acedido em 16 Fevereiro, 2020].

## Apêndice A

### 1 Casos de uso do sistema *web*

Tabela 3 Caso de uso login no sistema - Fonte: Autor

<b>Nome do caso de uso:</b> Login no Sistema	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> O objetivo deste caso de uso é permitir o utilizador realizar acesso às funcionalidades do sistema	
<b>Pré-condições:</b> login, senha	
<b>Fluxo de Ações:</b>	
<b>Ator:</b>	<b>Sistema</b>
1. Utilizador acede ao sistema	1. Aguarda os dados: login e senha
2. Utilizador informa os dados	2. Valida as credencias junto a base de dados
<b>Resultado Previsto:</b>	Em caso de sucesso nas credenciais informadas, o sistema permitirá acesso às funcionalidades.
<b>Estado de Erro:</b>	Devolve, por parte do sistema, um alerta ao utilizador em caso de não sucesso.

Tabela 4 Caso de uso pesquisar cliente - Fonte: Autor

<b>Nome do caso de uso:</b> Pesquisar Cliente	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> O objetivo é realizar consultas de clientes registados no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema	
<b>Fluxo de Ações:</b>	
1. Aceder ao sistema	
2. Pesquisar no sistema o menu consultar cliente	
3. Informar as iniciais do nome do cliente a ser consultado	
<b>Resultado Previsto:</b>	Após informar o tipo da consulta o sistema devolverá os detalhes do registo da consulta.
<b>Estado de Erro:</b>	Não existem erros para esta operação, se o utilizador informar nome inexistente o sistema não devolve a informação.

Tabela 5 Caso de uso registo cliente - Fonte: Autor

<b>Nome do caso de uso:</b> Registrar Cliente	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Registrar um cliente no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de registo do sistema</li> <li>4. Efetuar registo do cliente</li> </ol>	
<b>Resultado Previsto:</b>	Neste formulário de registo o sistema permite ao utilizador inserir diversas informações para o registo do cliente.
<b>Estado de Erro:</b>	O erro neste formulário aplica-se apenas se tais campos como: Tipo de pessoa e número de contribuinte não forem devidamente preenchidos. O valor tem de ser válido.

Tabela 6 Caso de uso alterar registo cliente - Fonte: Autor

<b>Nome do caso de uso:</b> Alterar Registo Cliente	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Alteração de registo cliente no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo registo do sistema</li> <li>4. Pesquisar pelo cliente a ser alterado e proceder com a alteração desejada</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 4 do fluxo de ações, o sistema gravará junto à base de dados, o ajuste realizado informando o utilizador que o ajuste foi feito com sucesso.
<b>Estado de Erro:</b>	O erro neste formulário aplica-se apenas se tais campos como: Tipo de pessoa e número de contribuinte não forem devidamente preenchidos. O valor tem de ser válido.



Tabela 7 Caso de uso excluir registo cliente - Fonte: Autor

<b>Nome do caso de uso:</b> Excluir Registo Cliente	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Excluir registo cliente no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de registo geral do sistema</li> <li>4. Pesquisar pelo cliente a ser excluído</li> <li>5. Fazer exclusão do registo desejado</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 5 do fluxo de ações, o sistema apresentará um evento confirmando a exclusão do registo. Após confirmação, o sistema removerá da base de dados o registo selecionado.
<b>Estado de Erro:</b>	Não se aplica a este processo de exclusão de registo.

Tabela 8 Caso de uso pesquisar produto - Fonte: Autor

<b>Nome do caso de uso:</b> Pesquisar Produto	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permite realizar consulta de produtos no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo stock de produtos do sistema</li> <li>4. Pesquisar pelo nome do produto a ser pesquisado</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 4 do fluxo de ações, o sistema apresentará ao utilizador os resultados da pesquisa, em caso de não existir o registo na base de dados o sistema devolverá um evento que nenhum produto foi encontrado.
<b>Estado de Erro:</b>	Não se aplica a este processo de consulta de produto.

Tabela 9 Caso de uso registrar produto - Fonte: Autor

<b>Nome do caso de uso:</b> Registrar Produto	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permitir registrar produtos no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de stock do sistema</li> <li>4. Efetuar registo do produto</li> <li>5. Preencher os campos, <i>SKU</i>, nome, descrição</li> <li>6. Selecionar o estilo e sabor, percentagem do teor alcoólico, origem</li> <li>7. Informar o preço, comissão, stock e <i>upload</i> de foto do produto</li> <li>8. Clicar no botão salvar</li> </ol>	
<b>Resultado Previsto:</b>	Neste formulário de registo, após o passo 7 do fluxo de ações o sistema apresentará ao utilizador uma mensagem de sucesso para salvar os registos na base de dados.
<b>Estado de Erro:</b>	Para este formulário não se aplica erros, exceto se todos os campos não forem preenchidos corretamente.

Tabela 10 Caso de uso alterar registo produto - Fonte: Autor

<b>Nome do caso de uso:</b> Alterar registo Produto	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Alteração de registo produto no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de stock do sistema</li> <li>4. Pesquisar pelo produto informando nome do produto ou o código do produto</li> <li>5. Inserir no registo do produto os novos ajustes e clicar em salvar</li> </ol>	

<b>Resultado Previsto:</b>	Após o passo 5 do fluxo de ações, o sistema gravará junto à base de dados o ajuste realizado com novas informações e também apresentando ao utilizador uma mensagem de sucesso.
<b>Estado de Erro:</b>	Para este formulário não se aplica erros, exceto se todos os campos não forem preenchidos corretamente.

*Tabela 11 Caso de uso excluir produto - Fonte: Autor*

<b>Nome do caso de uso:</b> Excluir Produto	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Excluir registo produto no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de stock do sistema</li> <li>4. Pesquisar pelo produto a ser excluído</li> <li>5. Fazer exclusão do registo desejado</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 5 do fluxo de ações, o sistema apresentará um evento confirmando a exclusão do registo; após confirmação o sistema removerá da base de dados o registo selecionado.
<b>Estado de Erro:</b>	Não se aplica a este processo de exclusão de registo.

*Tabela 12 Caso de uso pesquisar cidade - Fonte: Autor*

<b>Nome do caso de uso:</b> Pesquisar Cidade	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> O objetivo é realizar consultas de cidades registadas no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder ao sistema</li> <li>2. Pesquisar no sistema pelo menu registo e submenu cidade</li> <li>3. Informar as iniciais do nome da cidade a ser pesquisada</li> </ol>	

<b>Resultado Previsto:</b>	Após informar o tipo da consulta, o sistema devolverá os detalhes da consulta pelo registo feito.
<b>Estado de Erro:</b>	Não existe erro para esta operação; se o utilizador informar nome inexistente o sistema não devolve a informação.

Tabela 13 Caso de uso registar cidade - Fonte: Autor

<b>Nome do caso de uso:</b> Registrar Cidade	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> O objetivo é realizar o registo de cidades	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de registo geral no sistema</li> <li>4. Selecionar o módulo registo de cidade</li> <li>5. Efetuar o registo da cidade informando a que distrito pertence</li> <li>6. Salvar o registo realizado</li> </ol>	
<b>Resultado Previsto:</b>	Após o término do passo 6 no fluxo de ações, o sistema devolve uma mensagem de sucesso comunicando que a informação foi salva na base de dados.
<b>Estado de Erro:</b>	O erro neste formulário aplica-se apenas se tais campos como: distrito e nome, não forem devidamente informados.

Tabela 14 Caso de uso alterar registo cidade - Fonte: Autor

<b>Nome do caso de uso:</b> Alterar Registo Cidade	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Alteração de registo cidade no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de registo geral no sistema</li> <li>4. Selecionar o módulo registo de cidade</li> </ol>	

5. Pesquisar pela cidade a ser alterada	
6. Realizar a alteração desejada, clicando no botão disponibilizado para salvar o registo	
<b>Resultado Previsto:</b>	Após o passo 6 do fluxo de ações, o sistema gravará junto da base de dados o ajuste realizado com novas informações e também apresentando ao utilizador uma mensagem de sucesso.
<b>Estado de Erro:</b>	O erro neste formulário aplica-se apenas se tais campos como distrito e nome, não forem devidamente informados.

*Tabela 15 Caso de uso pesquisar vendas - Fonte: Autor*

<b>Nome do caso de uso:</b> Pesquisar Vendas	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permite realizar consulta de vendas realizadas no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de vendas</li> <li>4. Realizar pesquisa de vendas pelo código, estado, se emitida ou em orçamento e canceladas, por datas, valores, nome do cliente, número do contribuinte</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 4 do fluxo de ações o sistema apresentará ao utilizador os resultados da pesquisa; em caso de não existir o registo na base de dados o sistema devolverá um evento que nenhum produto foi encontrado.
<b>Estado de Erro:</b>	Não se aplica a este processo de consulta de produto.

Tabela 16 Caso de uso registrar vendas - Fonte: Autor

<b>Nome do caso de uso:</b> Registrar Vendas	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permite realizar vendas no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de vendas</li> <li>4. Realizar registo de vendas, através do formulário vendas</li> <li>5. Inserir o nome do cliente para que o sistema consulte o cliente na base de dados e associe à venda</li> <li>6. Informar o valor da venda</li> <li>7. Informar o valor do frete, se houver</li> <li>8. Informar o valor de desconto da venda, se houver</li> <li>9. Informar o nome do produto e quantidade para ser pesquisado pelo sistema e ser associado a venda</li> <li>10. Informar o local da entrega com data, horário e informações adicionais ao frete</li> <li>11. Salvar a venda apenas</li> <li>12. Salvar e emitir</li> <li>13. Salvar e enviar por <i>email</i></li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 13 do fluxo de ações, o sistema apresentará ao utilizador os resultados da venda consoante passos 11, 12,13 do fluxo de ações, permitindo visualizá-la em tempos posteriores.
<b>Estado de Erro:</b>	Não se aplica a este processo de gerir vendas, exceto se o produto e o cliente não forem inseridos no processo da venda, o que não permitirá finalizar a venda.

Tabela 17 Caso de uso altera venda - Fonte: Autor

<b>Nome do caso de uso:</b> Alterar Venda	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permite realizar alterações nas vendas	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de vendas</li> <li>4. Consultar pelo módulo pesquisa de vendas</li> <li>5. Inserir no campo pesquisa de vendas, o código da venda ou pela data de criação ou nome do cliente</li> <li>6. Clicar no botão pesquisar para o sistema localizar a venda e disponibilizar na tela do utilizador</li> <li>7. Clicar no botão alterar, para corrigir a venda</li> <li>8. Alterar venda por data, valor, cliente, contribuinte, produto, entrega, observação da entrega, quantidade de itens</li> <li>9. Salvar a venda apenas</li> <li>10. Salvar e emitir</li> <li>11. Salvar e enviar por <i>email</i></li> <li>12. Cancelar</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 9 do fluxo de ações o sistema apresentará ao utilizador os resultados da venda consoante passos 10, 11 do fluxo de ações, permitindo visualizá-la em tempos posteriores, exceto se executar o passo 12 do fluxo de ações.
<b>Estado de Erro:</b>	Não se aplica a este processo de gestão de vendas.

Tabela 18 Caso de uso cancelar venda - Fonte: Autor

<b>Nome do caso de uso:</b> Cancelar Venda	
<b>Ator:</b> Utilizador/Administrador	
<b>Descrição:</b> Realizar cancelamento de vendas	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização do administrador	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de vendas</li> <li>4. Consultar pelo módulo pesquisa de vendas</li> <li>5. Inserir no campo pesquisa de vendas, o código da venda ou pela data de criação ou nome do cliente</li> <li>6. Clicar em pesquisar venda</li> <li>7. Clicar em cancelar</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 7 do fluxo de ações, o sistema apresentará ao utilizador os resultados do cancelamento, indicando que o cancelamento foi realizado com sucesso.
<b>Estado de Erro:</b>	Não se aplica a este processo de gestão de cancelamento venda,

Tabela 19 Caso de uso gerar relatório venda - Fonte: Autor

<b>Nome do caso de uso:</b> Gerar Relatório Venda	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permitir gerar relatório de vendas	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de relatório, relatório de vendas, vendas emitidas</li> <li>4. Inserir data pesquisa a ser realizada consoante o respetivo intervalo de datas</li> <li>5. Clicar em emitir relatório no botão disponibilizado pelo sistema</li> </ol>	



<b>Resultado Previsto:</b>	Após o passo 5 do fluxo de ações o sistema apresentará ao utilizador o relatório das vendas realizadas consoante o filtro de datas informado no processo do fluxo de ações 4.
<b>Estado de Erro:</b>	Não se aplica a este processo de relatório de venda.

*Tabela 20 Caso de uso pesquisar estilo - Fonte: Autor*

<b>Nome do caso de uso:</b> Pesquisar Estilo	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permite realizar consulta de produtos no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo stock, menu estilo</li> <li>4. Pesquisar pelo nome do estilo</li> <li>5. Clicar no botão pesquisar</li> </ol>	
<b>Resultado Previsto:</b>	Após o passo 5 do fluxo de ações o sistema apresentará ao utilizador os resultados da pesquisa; no caso de não existir o registo na base de dados, o sistema devolverá um evento que nenhum estilo foi encontrado.
<b>Estado de Erro:</b>	Não se aplica a este processo de consulta de estilo.

*Tabela 21 Caso de uso registo estilo - Fonte: Autor*

<b>Nome do caso de uso:</b> Registrar Estilo	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Permitir registar estilo no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
<ol style="list-style-type: none"> <li>1. Aceder à aplicação</li> <li>2. Fazer <i>login</i> para autenticação e autorização no sistema</li> <li>3. Aceder ao módulo de stock, menu estilo</li> </ol>	

4. Efetuar registo do estilo 5. Preencher o campo nome do estilo 6. Clicar no botão salvar	
<b>Resultado Previsto:</b>	Neste formulário de registo após o passo 6 do fluxo de ações o sistema apresentará ao utilizador uma mensagem de sucesso para a persistência do registo na base de dados.
<b>Estado de Erro:</b>	Para este formulário não se aplica erros, exceto se o campo nome não for preenchido corretamente.

*Tabela 22 Caso de uso alterar registo estilo - Fonte: Autor*

<b>Nome do caso de uso:</b> Alterar Registo Estilo	
<b>Ator:</b> Utilizador	
<b>Descrição:</b> Alteração de registo estilo no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
1. Aceder à aplicação 2. Fazer <i>login</i> para autenticação e autorização no sistema 3. Aceder ao módulo de stock, menu estilo 4. Efetuar alteração no nome do estilo 5. Clicar no botão salvar	
<b>Resultado Previsto:</b>	Após o passo 5 do fluxo de ações, o sistema gravará junto da base de dados o ajuste realizado com novas informações e também apresentando ao utilizador uma mensagem de sucesso.
<b>Estado de Erro:</b>	Para este formulário não se aplica erros.

*Tabela 23 Caso de uso excluir estilo - Fonte: Autor*

<b>Nome do caso de uso:</b> Excluir Estilo	
<b>Ator:</b> Utilizador /Administrador	
<b>Descrição:</b> Excluir registo estilo no sistema	
<b>Pré-condições:</b> Utilizador autenticado no sistema e possuir autorização	
<b>Fluxo de Ações:</b>	
1. Aceder à aplicação 2. Fazer <i>login</i> para autenticação e autorização no sistema	

<p>3. Aceder ao módulo de stock, menu estilo</p> <p>4. Pesquisar pelo nome do estilo a ser excluído</p> <p>5. Fazer exclusão do registo desejado</p>	
<b>Resultado Previsto:</b>	Após o passo 5 do fluxo de ações, o sistema apresentará um evento confirmando a exclusão do registo; após confirmação, o sistema removerá da base de dados o registo selecionado.
<b>Estado de Erro:</b>	Não se aplica a este processo de exclusão de registo.