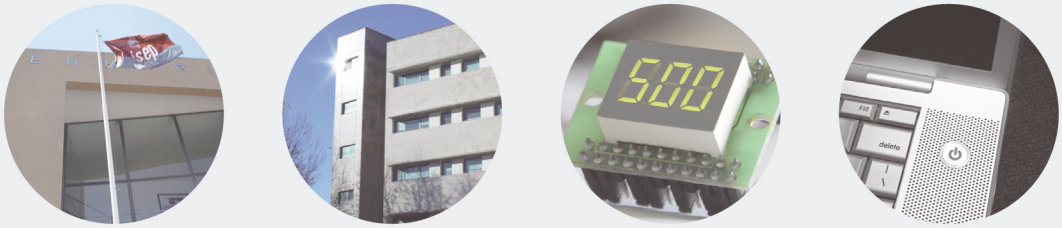
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES



Desenho e desenvolvimento de um Sistema de Informação em Oracle ADF

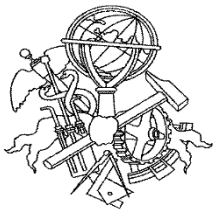
FÁBIO JOSÉ DA SILVA MATOS
Novembro de 2015

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

DESENHO E DESENVOLVIMENTO DE UM SISTEMA DE INFORMAÇÃO EM ORACLE ADF

Fábio José Silva Matos



Mestrado em Engenharia Eletrotécnica e de Computadores

Sistemas e Planeamento Industrial

Departamento de Engenharia Eletrotécnica

Instituto Superior de Engenharia do Porto

2015

Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de Tese/Dissertação, do 2º ano, do Mestrado em Engenharia Eletrotécnica e de Computadores

Candidato: Fábio José Silva Matos, Nº 1100304, 1100304@isep.ipp.pt
Orientação científica: Eng.ª Cecília Reis, cmr@isep.ipp.pt

Mestrado em Engenharia Eletrotécnica e de Computadores
Sistemas e Planeamento Industrial
Departamento de Engenharia Eletrotécnica
Instituto Superior de Engenharia do Porto
2 de novembro de 2015

Agradecimentos

Em primeiro lugar, gostaria de agradecer a toda a minha a família o apoio incondicional dado durante o meu percurso académico e em especial aos meus pais, que providenciaram tudo o foi necessário para que eu pudesse concluir o Mestrado, e ao meu irmão por toda a ajuda e conselhos que me permitiram envergar pelo caminho certo para atingir os meus objetivos.

Por fim, um agradecimento a toda a comunidade do ISEP pelo excelente percurso académico proporcionado e principalmente à Eng.ª Cecília Reis pelos ensinamentos e orientação prestada. Sem esquecer os meus atuais colegas de trabalho que também me proporcionaram um apoio importante nesta etapa da minha vida.

Resumo

Um Sistema de Informação consiste num sistema capaz de armazenar, organizar e estruturar dados para ajudar a responder às necessidades das empresas, passando também pela capacidade de resposta às questões diárias das empresas. Assim, um Sistema de Informação pode ser definido como o *software* que ajuda a organizar e analisar dados, tendo como objetivo fornecer informação útil na altura certa para que possa ser utilizada para a tomada de decisões ou para uma gestão mais eficiente dos diversos fluxos que uma empresa pode conter.

Neste sentido, o projeto apresentado centra-se no desenho e construção de um Sistema de Informação capaz de gerir o negócio de uma empresa do setor alimentar, mais propriamente do setor da transformação de carnes. Foi desenvolvido em Oracle ADF, de forma a aproveitar as vantagens inerentes à tecnologia e ao desenvolvimento *web*. Sendo uma tecnologia relativamente nova no mercado e dominada por poucos, a sua utilização neste momento pode tornar-se uma grande vantagem.

Para o desenvolvimento da aplicação foi realizado o levantamento e análise de requisitos, foi criada a base de dados capaz de suportar o funcionamento do *software* e desenvolvido um sistema de *login*, capaz de gerir as sessões de cada utilizador. Foi implementado um processo de introdução e edição de informação, nomeadamente o registo de entradas, transformações e saídas. Contemplou-se também uma secção com dados mestre da empresa com a possibilidade de inserção, atualização e/ou remoção. Além disso, foram incorporadas validações em todos os processos que são usados pelos utilizadores, de modo a evitar a existência de dados incoerentes ou duplicados.

Relativamente à lógica de negócio, foi embutida na aplicação de forma a permitir consultar a informação de forma clara, rápida e em diversos lugares, reduzindo tempo e tarefas ao colaborador/utilizador, visto que os processos foram automatizados.

Com a implementação deste Sistema de Informação, a empresa pode usufruir de um sistema integrado capaz de gerir e controlar todo o seu processo produtivo, reduzindo custos e desperdícios, aumentando a produtividade e eficiência.

Palavras-Chave

Indústria Alimentar, Sistema de Informação, ADF, *Web Development*.

Abstract

An Information System is a system able to store, organize and structure data to help meeting the companies' needs, including also the ability to solve the daily issues presented by the companies. Thus, an Information System can be defined as software that helps organize and analyze data, aiming to provide useful information at the right time so it can be used for making decisions or for a better management of the various resources that a company can hold.

The presented project focuses on the design and construction of an Information System able to manage the business of a company in the food industry, more specifically the meat processing sector. It was developed in Oracle ADF, in order to take advantage inherent to technology and web development. Being a relatively new technology in the market and dominated by a few, its use at the moment can become a great advantage.

For the application's design and development it was carried a phase for requirements analysis, also the creation of a database model capable of supporting the operation of the software, along with this it was also designed a login system that is able to manage the sessions for each user. The processes to manipulate foundation and transactional data had also been considered. It is also contemplated a section with the company's master data with the ability to insert, update and/or remove. Furthermore, validations have been incorporated in all processes that are used by users, in order to avoid inconsistent or duplicate data.

With regard to business logic, the application allows visualization of the information clearly, quickly and in different places, reducing time and tasks to the employee/user, since the processes were automated.

By implementing this Information System, the company can take advantage of an integrated system capable to manage and control the organization's entire production process, reducing costs and waste, increasing productivity and efficiency.

Keywords

Food Industry, Information System, ADF, Web Development.

Résumé

Un Système d'Information est un système capable de stocker, organiser et structurer des données pour aider à répondre aux besoins des entreprises, aussi par la capacité de réponse aux problèmes quotidiens des entreprises. Ainsi, un Système d'Information peut être définie comme un logiciel qui vous aide à organiser et analyser des données ayant comme objectif de fournir l'information au bon moment pour que celle-ci puisse être utile dans la prise des décisions ou pour une gestion plus efficace dans les différents flux qu'une entreprise peut avoir.

En ce sens, le projet présenté se concentre dans la conception et la construction d'un Système d'Information capable de gérer les affaires d'une entreprise du secteur alimentaire plus particulièrement le secteur de la transformation de la viande. Il a été développé en Oracle ADF, afin de prendre des avantages inhérents à la technologie et au développement web. Sachant que c'est une technologie relativement nouvelle sur le marché et maîtrisé pour quelques uns son utilisation en ce moment peut devenir une grosse avantage.

Pour le développement de la application a été réalisé une enquête et une analyse des besoins, a été créé une base des données capable de supporter le fonctionnement du logiciel et développer un système de connexion, capable de gérer les sessions de chaque utilisateur. Un processus de saisie et édition d'information, y compris les traces des registres, transformation et sorties on été mis en œuvre. Il est aussi envisageable une session avec des données de base d'une entreprise avec une possibilité d'insertion, mettre à jour où le retrait. En outre, les validations ont été intégrées dans tous les processus utilisé par les utilisateurs d'une façon à éviter l'existence des données incohérentes ou en double.

En ce qui concerne la logique du métier à été intégré dans la application pour permettre de consulter l'information clairement et rapide en différents lieux en réduisant le temps et les tâches du employés/utilisateurs vue que les processus sont automatiques. En mettant en place ce Système d'Information l'entreprise peut tirer profit d'un système intégré capable de

gérer et contrôler tout son processus de production tout en réduisant les coûts et gâchis, permettant d'augmenter la productivité et efficacité.

Mots-clés

Industrie alimentaire, Système d'information, ADF, *Web Development*.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

ADF – Application Development Framework

API – Application Programming Interface

AJAX – Asynchronous JavaScript and XML

BD – Base de dados

CSS – Cascade Style Sheets

DEE – Departamento de Engenharia Eletrónica

ERP – Enterprise Resource Planning

HTML – HyperText Markup Language

IDE – Instituto Superior de Engenharia do Porto

ISEP – Integrated Development Environment

JDBC – Java Database Connectivity

JEE – Java Enterprise Edition

MEEC – Mestrado em Engenharia Eletrotécnica e de Computadores

MVC – Model-View-Controller

PME – Pequena ou Média Empresa

SI – Sistema de Informação

SPI – Sistemas e Planeamento Industrial

UML – Unified Modeling Language

XML – Extensible Markup Language

1. INTRODUÇÃO

Este documento pretende descrever a elaboração da Tese/Dissertação, do 2º ano do Mestrado em Engenharia Eletrotécnica e de Computadores (MEEC) na área de especialização em Sistemas e Planeamento Industrial (SPI), do Departamento de Engenharia Eletrotécnica (DEE), do Instituto Superior de Engenharia do Porto.

Neste capítulo faz-se uma introdução ao trabalho de dissertação, assim como a sua contextualização, apresentam-se os objetivos e a respetiva calendarização. Por último, é apresentada a organização deste documento e os temas abordados em cada capítulo.

1.1. CONTEXTUALIZAÇÃO

O tema desta tese/dissertação de mestrado surgiu da necessidade de uma empresa, especialista na transformação de produtos alimentares, adotar um sistema de informação capaz de retirar carga de trabalho aos seus colaboradores e de garantir fiabilidade na informação registada. Aliado com o gosto que o autor deste documento possui por

tecnologia, nomeadamente o desenvolvimento aplicacional, surgiu a motivação para o desenvolvimento do presente trabalho. Além disso, o trabalho poderá vir a ser implementado na realidade, agilizando e melhorando os processos na empresa. A referida empresapertence a familiares do autor, deste modo, o produto sofrerá melhorias contínuas de modo a acompanhar as necessidades e o crescimento da companhia. Salienta-se também que, sendo atualmente a disponibilidade do autor limitada, devido a compromissos profissionais o facto de a empresa se encontrar em crescimento permite um desenvolvimento gradual do projeto, o que facilita o apuramento de novas especificações, funcionalidades e/ou regras a serem implementadas no futuro.

1.2. OBJETIVOS

O ojetivo principal deste projeto é o desenho e desenvolvimento de um sistema de informação em Oracle ADF. Com este projeto pretende-se desenhar um sistema semelhante a um ERP, mas menos extenso e possuindo apenas alguns dos módulos deste, capaz de assegurar a gestão da empresa descrita anteriormente.

Pretende-se que a aplicação permita registar e manipular informação nos domínios de produtos, preços, *stock*, vendas, funcionários, departamentos e clientes. Para tal a aplicação deve incluir as seguintes interfaces:

- Registo das entradas de matéria-prima;
- Alocação da matéria-prima aos diferentes produtos que a empresa comercializa;
- Alocação dos produtos acabados aos diversos clientes;
- Registo das saídas;
- Menu com acesso a diferentes ecrãs para a consulta/adição/edição/eliminação referente a informação de produtos, preços, *stock*, funcionários e clientes;

- Registo de pagamentos/recebimentos das respetivas compras de matéria-prima e vendas de produtos.

Com o intuito de cumprir os objetivos apontados, a execução do trabalho foi dividida em várias tarefasa seguir enumeradas:

- Levantamento e análise de requisitos;
- Construção de uma base de dados capaz de suportar o funcionamento do *software*;
- Desenvolvimento de um sistema de *Login*, capaz de gerir as sessões de cada utilizador;
- Processo de introdução e edição de informação, nomeadamente o registo de entradas, transformações e saídas;
- Secção com dados mestre da empresa e a possibilidade de sua inserção, atualização e/ou remoção;
- Incorporação de validações em todos os processos que possam ser utilizados pelos utilizadores, de modo a evitar a existência de dados incoerentes ou duplicados;
- Construção do SI utilizando a tecnologia ADF da Oracle.

1.3. CALENDARIZAÇÃO

Relativamente à calendarização deste projeto, ela foi dividida em diferentes fases, como é possível identificar na figura 1, de modo a agilizar e facilitar a execução das mesmas. Assim sendo, o projeto foi divido em quatro fases essenciais para garantir a eficiência e qualidade do mesmo:

- Levantamento e análise de requisitos para o desenvolvimento do *software*;
- Desenho de uma base de dados capaz de suportar as funcionalidades e especificidades dos requisitos identificados;
- Construção da interface gráfica;
- Por fim, a fase de testes onde será verificado de forma exaustiva se o funcionamento ocorre como esperado.

As figuras 1 e 2 apresentam o cronograma elaborado no Microsoft Office Project, contêm todas as fases do projeto e o tempo planeado para as mesmas, com o respetivo gráfico de Gantt. O Diagrama de Gantt ilustra o progresso das tarefas existentes no cronograma.

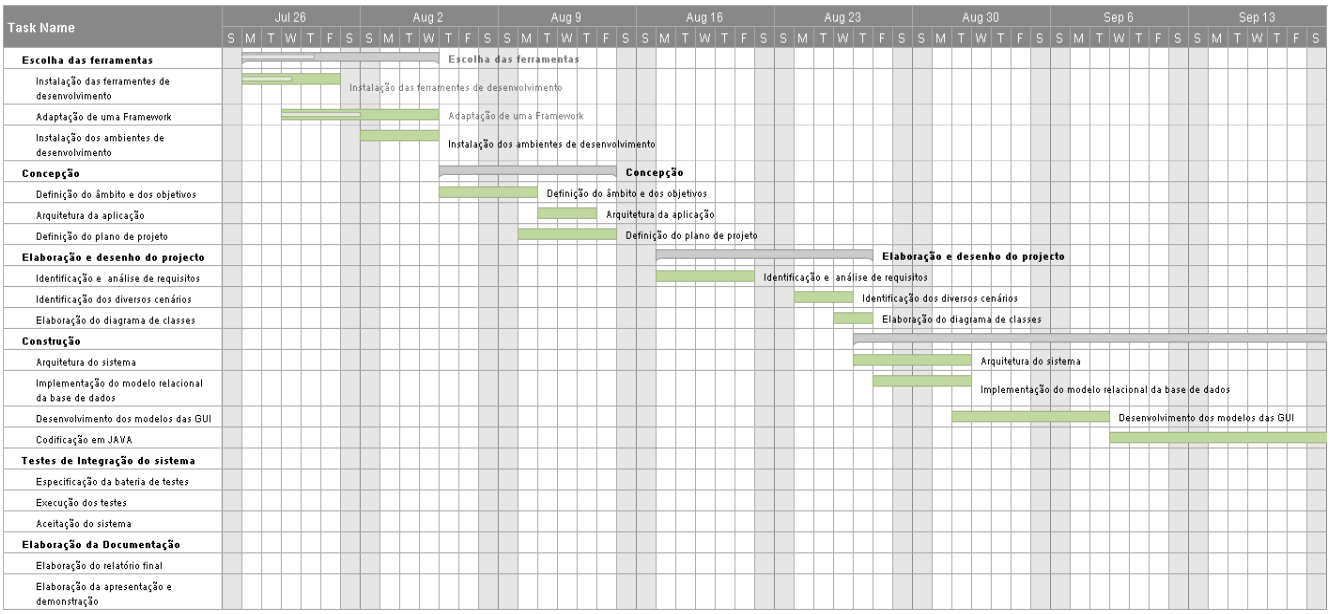


Figura 1- Primeira parte do Diagrama de Gantt

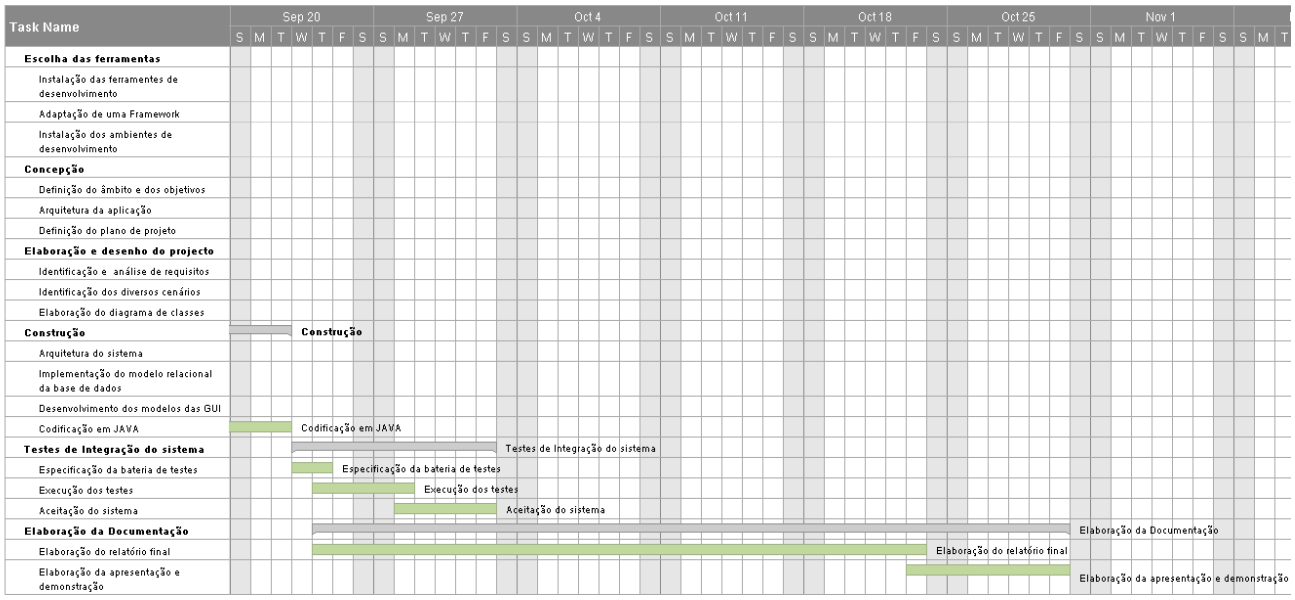


Figura 2- Segunda parte do Diagrama de Gantt

1.4. ORGANIZAÇÃO DO RELATÓRIO

Este documento, encontra-se organizado em cinco capítulos. O primeiro capítulo, Introdução, clarifica o tema do projeto, faz a sua contextualzação, apresenta os objetivos que se pretende atingir, apresenta também a calendarização das tarefas realizadas e a organização do presente relatório. O segundo capítulo apresenta o estado da arte dos sistemas de informação e o terceiro capítulo descreve as ferramentas que serviram de suporte ao desenvolvimento da aplicação. No quarto capítulo, referente aos desenvolvimentos efetuados, são descritas todas as etapas do projeto, desde o desenho até à sua construção. Este capítulo ainda contém a análise efetuada aos requisitos de forma a ser estruturada a arquitetura da aplicação, descrição dos desenvolvimentos realizados na construção da aplicação e os testes realizados para verificar a estabilidade, eficiência e performance da mesma. No último capítulo, capítulo cinco, são reunidas as principais conclusões, identificados os objetivos cumpridos e as perspetivas e desenvolvimentos futuros.

2. ESTADO DA ARTE

Ao longo deste capítulo são referidas e explicadas as principais áreas e tecnologias utilizadas no desenvolvimento deste projeto, desta forma será possível perceber-se o estado atual de cada tecnologia. Começando pelos sistemas de informação, nomeadamente no que diz respeito aos ERP, passando pelo desenvolvimento web - *web development*, visto que a aplicação criada é desenvolvida e executada na plataforma *web*, e a respetiva arquitetura utilizada. Por fim, descrevem-se as capacidades, recursos, características e o estado em que se encontra a principal tecnologia utilizada, a Oracle ADF. Faz-se também uma breve abordagem às bases de dados que estes sistemas contemplam.

2.1. SISTEMAS DE INFORMAÇÃO

Um SI (Sistema de Informação) consiste num sistema capaz de armazenar, organizar e estruturar dados para ajudar a responder às necessidades das empresas, passando também pela capacidade de resposta aos problemas das empresas. Assim, um sistema de

informação pode ser definido como o *software* que ajuda a organizar e analisar dados, tendo como objetivo fornecer informação útil na altura certa para que possa ser utilizada para a tomada de decisões ou para uma gestão mais eficiente dos diversos fluxos que uma empresa pode conter.

Os diversos modelos e tipos de sistemas de informação diferem pela forma de onde e como atuam dentro de uma organização. No entanto, os módulos apresentados a seguir, são comuns a todos: [1]

- *Hardware*: consiste no servidor onde é executado o SI;
- *Software*: permite a interação do utilizador com o SI, para organizar, processar e analisar dados;
- Rede: os diferentes elementos pertencentes ao SI necessitam de se interligar para partilhar informação;
- Base de dados: Os sistemas de informação trabalham com dados que se encontram organizados em tabelas e arquivos, de forma a facilitar o acesso e atualização dos mesmos;
- Procedimentos: Estes são bastantes específicos dependendo do negócio e da organização, pois destinam-se a solucionar os principais requisitos impostos pela especificidade do negócio.

A utilização, em geral, dos sistemas de informação adicionaram várias vantagens ao mercado de trabalho, tais como a globalização da informação, a comunicação tornou-se mais acessível, rápida e mais eficiente. As vantagens proporcionadas pelos sistemas de informação também se verificaram na rentabilização do tempo, uma vez que diminuem o tempo de processamento da informação, rentabilizando o tempo dos colaboradores na realização de estas ações, libertando-os para dedicarem os seus esforços noutras funções. A relação custo-eficácia é igualmente incrementada, visto que a informatização dos processos de negócio agiliza as empresas para torná-las mais rentáveis e eficazes nas suas ações, aumentando também a produtividade que origina lucros. [2]

De seguida será efetuada uma descrição geral sobre um sistema de informação especifico, o *Entreprise Resource Planning*, visto que é o sistema que mais se assemelha ao sistema de informação construído no âmbito deste projeto.

2.2.1- Enterprise Resource Planning (ERP)

Entreprise Resource Planning é um sistema de informação que integra todos os dados e processos de uma organização num único sistema. A integração pode ser vista sob a perspetiva funcional (sistemas de finanças, contabilidade, recursos humanos, produção, marketing, vendas, compras, etc) e sob a perspetiva sistemática (sistema de processamento de transações, sistemas de gestão de dados, sistemas de apoio à decisão, etc). Um bom ERP permite ao utilizador aceder, compreender e responder às informações de forma rápida e eficaz. [3]

Um ERP, sistema transacional focado no nível operacional, é uma plataforma de *software* desenvolvida para integrar os diversos departamentos de uma empresa, possibilitando a automação e armazenamento de todas as informações do negócio. É uma arquitetura de transações que liga todas as funções de uma empresa, como por exemplo, de processamento de pedido de vendas, controlo e gestão de *stock*, planeamento de produção, distribuição e finanças.

Para ser possível beneficiar da utilização de um ERP é necessário ter em consideração os seguintes cinco elementos, incluindo *hardware*, *software*, informação, pessoas e processos. O *hardware* deve ser confiável e deve ser capaz de suportar o volume de dados e transações previstas. O *software* deve ser cuidadosamente projetado e avaliado quanto à sua eficácia. É também necessário ter informação detalhada acerca do negócio para que a construção assente nas necessidades reais da empresa. Além disso, os dados devem ser inseridos com precisão e os utilizadores devem receber formação para trabalhar com o sistema. [4]

As empresas têm necessidade de informação oportuna, de baixo custo, acessível, organizada e precisa. Portanto, os sistemas de informação devem ser constantemente atualizados para atender às expectativas e necessidades das empresas.

A forma como a empresa conduz a escolha e a implementação de um ERP é crucial para a obtenção dos resultados esperados. Alguns cuidados na escolha da solução são imprescindíveis, principalmente quando a solução envolve flexibilidade, escalabilidade e capacidade de atualização em tempo útil para suprir as mudanças do negócio.

Com informações monitorizadas e atualizadas em tempo real, o ERP oferece significativas melhorias ao nível da gestão, padronizando e sincronizando processos de forma segura, além de proporcionar ganho de tempo e redução dos custos, tem impacto direto na qualidade dos produtos, serviços e na gestão de pessoas, tornando a organização mais competitiva. Um sistema ERP combina dados utilizados por diferentes aplicações e elimina a necessidade de interfaces entre aplicações, resolvendo o problema de sincronização de informações mestres.

As vantagens de um sistema desenvolvido de acordo com as necessidades específicas da empresa é o que trará eficiência ao processo, diminuindo custos ao facilitar o trabalho das pessoas envolvidas e diminuindo o tempo na análise e distribuição dos dados. [5]

A principal desvantagem é o seu custo, principalmente o custo do próprio desenvolvimento. Por ser um trabalho altamente sofisticado requer, para o seu projeto e execução, técnicos com formação qualificada e experiência.

O projeto desenvolvido no âmbito desta tese pretende ser um ERP, contemplando para já, apenas alguns dos módulos dos *Entreprise Resource Planning*.

2.2. WEB DEVELOPMENT

O desenvolvimento *web* normalmente conhecido como o desenvolvimento de *sites*, em geral, refere-se às tarefas associadas a desenvolvimentos para alojar via Intranet ou Internet. Consiste na construção e manutenção de *sites* (aplicações em plataformas web) e varia desde a criação de páginas apenas de leitura até aplicações mais elaboradas, como aplicações de redes sociais ou aplicações eletrónicas de negócios. O processo de desenvolvimento *web* inclui *web design*, desenvolvimento do conteúdo, configuração de segurança da rede, entre outras tarefas.

Por *Web Design* entende-se a criação de *layouts* ou *templates* (normalmente recorrendo à linguagem CSS), isto é, trata da apresentação da aplicação de modo a esta ser o mais agradável possível, em termos visuais, para o utilizador final. [6]

O desenvolvimento do conteúdo consiste na programação da aplicação de modo a que esta preencha os requisitos necessários para realizar as funcionalidades previstas e desejadas.

A configuração de rede trata basicamente da segurança da informação contida na aplicação, assim como a segurança dos próprios utilizadores. Isto é, proteje a informação e previne o acesso a utilizadores indesejados ou não autorizados.

Relativamente ao desenvolvimento *Web*, é possível enumerar os seguintes tópicos:

- Codificação do lado do cliente (*cliente-side*);
- Codificação do lado do servidor (*server-side*);
- Tecnologia de bases de dados.

Os *Web Developers* lidam com todos os aspetos de programação e criação de um *site*, incluindo HTML, criação de gráficos, imagens, ligações, e tudo o resto que é pretendido [7].

Em suma, o desenvolvimento *web* permite uma simplificação substancial dos processos, aumentando a produtividade e redução dos custos. Para além de que a integração entre

sistemas, através do desenvolvimento de interfaces *web*, permite a comunicação e partilha de informação, com características técnicas diferentes.

2.3. ARQUITETURA MVC

A arquitectura *Model-View-Controller* (MVC), apresentada na figura 3, é um padrão de arquitetura de *software* que separa a representação da informação da interação com o utilizador. A reutilização de código e a separação de conceitos representam uma vantagem quer para quem desenvolve quer para quem utiliza. É uma *framework* que se divide em três camadas: [8]

- *Model*: é a camada inferior da aplicação que lida com a lógica e funções da mesma, sendo responsável pela manutenção dos dados. Responde às solicitações do controlador para se atualizar ou efetuar ações.

- *View*: é responsável pela disponibilização dos dados, ou parte deles, ao utilizador. Apresenta a informação no formato específico e é acionado pela decisão do controlador para apresentar os dados. A representação dos dados, normalmente é na forma de tabelas ou diagramas.

- *Controller*: é responsável pelas ações efetuadas pelo utilizador e pela mediação da entrada, convertendo-a em comandos para enviar para o *Model* ou *View*. Normalmente, os controladores registam os dados de uma entrada do utilizador, e comunicam com os elementos responsáveis pelas ações a serem efetuadas. O controlador recebe a entrada, valida essa entrada e em seguida, executa a operação de negócio adequada consoante a interação realizada pelo utilizador. A figura 3 ilustra esta interação.

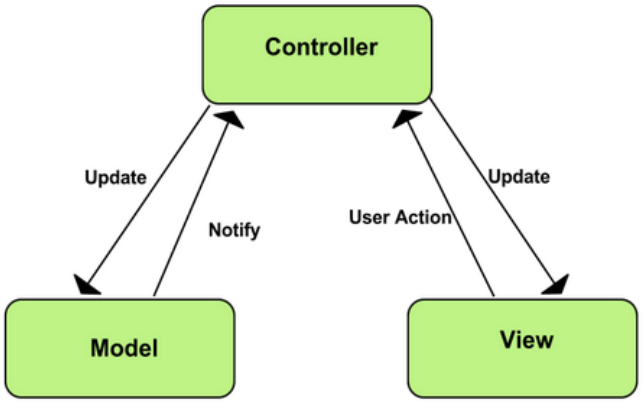


Figura 3- Arquitetura MVC [9]

Esta arquitetura é bastante utilizada pois isola, a camada lógica da aplicação, da interface do utilizador. Aqui, o controlador recebe todas as solicitações da aplicação e, em seguida, trabalha com o modelo de dados de forma a preparar o comportamento do *Model* e da *View*. Este facto é de particular interesse para gerir aplicações complexas e de grande dimensão, visto que se concentra em camadas separadas de acordo com as suas funções. Simplifica também o trabalho de desenvolvimento, uma vez que permite que seja realizado trabalho simultâneo sobre os diferentes níveis existentes. Com o aumento da complexidade das aplicações desenvolvidas, sempre visando a programação orientada aos objetos, torna-se relevante esta separação entre os dados e a apresentação das aplicações. Desta forma, as alterações feitas no *layout* não afetam a manipulação de dados, e estes poderão ser reorganizados sem alterar o *layout*.

2.4. APPLICATION DEVELOPMENT FRAMEWORK (ADF)

No desenvolvimento atual de aplicações, a estrutura desempenha um papel importante, dado que sem o apoio de uma estrutura consolidada e robusta, o desenvolvimento eficiente

e rápido de aplicações seria praticamente impossível. Neste sentido, a Oracle desenvolveu uma estrutura inovadora, robusta e estável a Oracle *Application Development Framework* (ADF).

A Oracle ADF é uma *framework* de aplicação baseada na plataforma Java, padrões Java EE e tecnologias *open source*. Por ser declarativa, a Oracle ADF simplifica e acelera o desenvolvimento, possibilitando a quem desenvolve, focar-se mais na lógica de negócio da aplicação, do que nos detalhes da codificação e da infraestrutura.

Além disso, a Oracle ADF fornece um ambiente que abrange o ciclo de desenvolvimento completo, desde a conceção à implementação, contendo alguns recursos que facilitam o desenvolvimento, tais como: *drag and drop* (arrastar componentes para a página), criação de ligação de dados (*bindings*), *design* de interface visual (*User Interface*), entre outros. [10]

ADF é, desta forma, a tecnologia da nova geração, lançada pela Oracle, está a substituir rapidamente outras tecnologias semelhantes (como OAF, mais conhecido como *Forms*) devido às suas características únicas e eficiência. Com esta tecnologia é possível desenvolver-se aplicações empresariais com qualidade e interfaces *user-friendly*. É também uma tecnologia multiplataforma adequada para o desenvolvimento de aplicações de pesquisa, exibição, criação, modificação e validação de dados usando interfaces *web*, em dispositivos móveis e de *desktop*, com os mesmos recursos e facilidade de utilização.

Possui também uma estrutura de aplicação *end-to-end* construída sobre Java EE e aderente a padrões e tecnologias *open source*, o que simplifica a sua utilização e o desenvolvimento de aplicações, fornecendo serviços *out-of-the-box* de infraestrutura. [11]

2.1.1- Arquitetura

A *framework* ADF implementa a arquitetura MVC com quatro camadas. Isto é, possui mais uma camada que é utilizada de modo a aumentar a sua performance e utilidade, como se pode verificar no diagrama da figura 2:

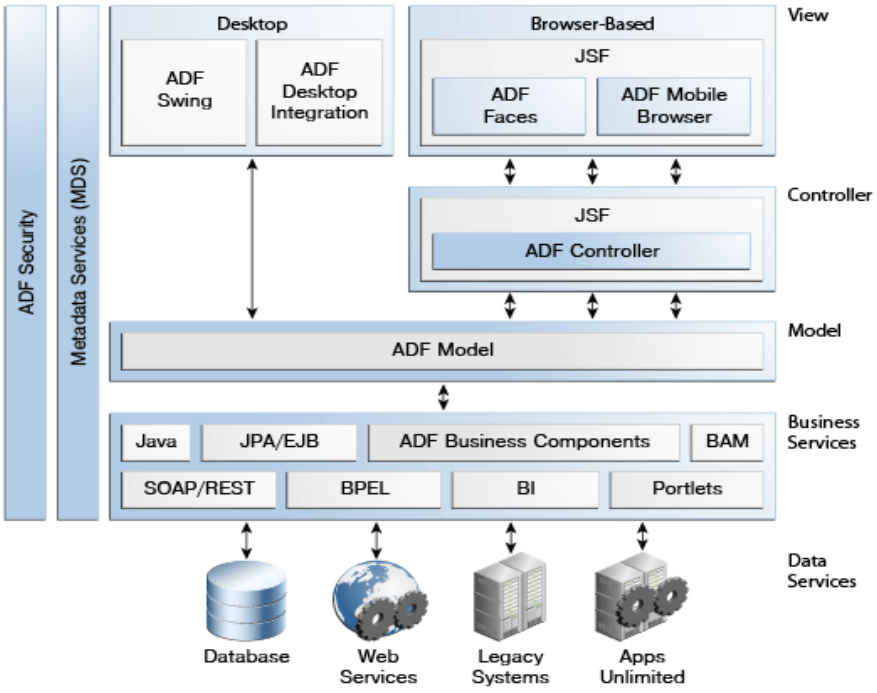


Figura 4 Arquitetura do Oracle ADF [12]

Para se entender melhor, a seguir são apresentados os detalhes sobre as diferentes camadas mostradas no diagrama da figura 2.

ADF *Business Components*:

Esta camada lida com dados de várias proveniências e com a lógica de negócio. A camada de serviço de negócios pode ser Java, *EJB*, *Web*, *Portlet*, etc.

Esta camada proporciona uma flexibilidade de definição de lógica de negócio (regras especificas do negócio) que permite melhorar a aplicação, devido às diversas tecnologias que são inerentes aos *ADF Bussiness Components*. Esta camada gere as seguintes tarefas:

- Interação com os dados que se encontram persistentes na base de dados;
- Mapeamento objeto-relacional;
- Gestão de transações;

- Execução de lógica de negócios;

Esta camada da arquitetura da *framework* ADF possui componentes bastante importantes e úteis que serão descrito de seguida. [13]

Entity objects:

São componentes do domínio de negócio que se relacionam com uma tabela na base de dados, representam essa tabela, ou um conjunto de dados de diversas tabelas, e gerem todas as operações de atualização de dados. Encapsulam a lógica de negócio e garantem a sua consistência. Os *Entity objects* podem ser associados a outros para refletir as associações do modelo de dados.

View objects:

View objects representam uma consulta SQL. Podem ser associados a outros *view objects* para criar uma hierarquia conhecida como pai-filho. Podem referenciar *entity objects* para suportar a atualização e a validação de dados.

Application module:

É um componente transacional que define um modelo de dados atualizável com *procedures* e funções (métodos de serviço que comunicam com a base de dados para receber determinados dados ou procedimentos).

ADF *Model Layer*:

O *Model* abstrai grande parte da implementação do serviço de negócio, fornecendo uma interface de programação para diferentes tipos de serviços, simplificando a construção e posterior utilização do projeto. Os *Data controls*, que existem nesta camada, fornecem às

interfaces do projeto todas as funcionalidades existentes de edição da informação da base de dados, evitando bastante trabalho a quem desenvolve. Assim como as operações do serviço e recolha de dados, incluindo informações sobre as propriedades, métodos e tipos envolvidos, de forma a realizar as validações necessárias sobre as alterações que se pretendem fazer na interface do projeto. Durante a execução, a camada ADF *Model* lê informações de arquivos que descrevem os dados e as suas de ligações, implementando as conexões entre a interface do utilizador e o serviço de negócio, de modo a que a informação disponibilizada pelas interfaces se encontre sempre atualizada durante o *runtime*.

Esta camada não manipula a lógica de negócio diretamente, em vez disso, fornece uma camada abstrata em cima da camada de serviço de negócio. Este padrão de *design* particular ajuda as outras camadas a trabalhar com várias implementações de serviços de negócio.

A camada *Model* fornece uma interface única para se ligar a qualquer serviço de negócios. Esta camada não manipula a lógica de negócios, apenas fornece a abstração no topo dos serviços de negócios. Assim, uma grande vantagem da *framework* ADF é a flexibilidade de implementação de qualquer serviço de negócio. Isto é, fornece abstração em cima de camadas de serviço de negócios, ligações e expõe métodos de controlo de dados para a camada de interface do utilizador.

O ADF *Model* também suporta as seguintes tecnologias:

- *Enterprise JavaBeans* (EJB), *Session Beans* e entidades JPA (Java *Persistence* API);
- JavaBeans;
- Web services;
- XML;
- CSV.

ADF *Controller*:

Na camada de controlo, onde o fluxo de páginas é uma preocupação chave, o ADF *Controller* fornece uma gestão de navegação sobre o modelo JSF. O *jDeveloper* permite criar declarativamente *task flows* onde o controlo de aplicação pode passar entre diferentes tipos de atividades, como páginas, métodos em *managed beans* (classes com código Java), ou chamadas para outros fluxos de tarefas.

Estes fluxos de tarefas podem ser reutilizados dentro da *task flow* e dentro de páginas.

A camada do controlador é responsável por controlar o fluxo da aplicação e a entrada de utilizador. As regras de navegação são apresentadas num diagrama para definir o fluxo da aplicação. Existem dois tipos de controladores utilizados em aplicações baseadas na *web*.

- Controlador JSF;
- Controlador ADF, que estende as funcionalidades dos controladores JSF.

ADF *View Layer*:

Esta camada representa a interface de utilizador da aplicação, isto é, é o *front-end* de qualquer aplicação. A camada de *front-end* pode ser uma aplicação de *desktop,* aplicação móvel ou aplicação baseada na *web*. O suporte multicanal da Oracle ADF é muito útil para o desenvolvimento de qualquer tipo de aplicação e é construído sobre o JSF, que possui muitos recursos extra como gráficos e tabelas, uma *framework* de diálogos, componentes declarativos, data *streaming*, *task flows*, etc.

Suporta, também, acesso multicanal para os serviços de negócio permitindo a reutilização e o acesso a serviços a partir de variadas plataformas, tais como: *Web client*, *client-server*, Microsoft Excel, ou dispositivos móveis como os *smart phones*.

2.1.2- Principais recursos

Como principais recursos pode-se enumerar o ambiente de desencolvimento integrado, o *vendor neutral e* uma solução *end-to-end.*Onde:

Ambiente de desenvolvimento integrado: Integra a Oracle ADF e o *JDeveloper*, o que minimiza o esforço de desenvolvimento e aumenta a produtividade.

Vendor neutral: As aplicações desenvolvidas em Oracle ADF podem ser executadas em qualquer servidor compatível com J2EE.

Suporte tecnológico: A Oracle ADF fornece diferentes camadas para desenvolvimento de aplicações JEE e, estas camadas, podem ser desenvolvidas utilizando diferentes tecnologias.

Solução *end-to-end* : Fornece suporte completo para desenvolvimento de aplicações JEE e suporte de infraestrutura, para que quem desenvolve possa focar toda a sua atenção na construção da aplicação em si. [14]

Alguns dos principais recursos que a Oracle ADF proporciona incluem:

- Várias opções para a construção da camada de serviço de negócio;
- Solução *end-to-end* em todas as vertentes Java EE;
- Tecnologia e independência da plataforma;
- *Meta data*;
- Suporte para desenvolvimento;
- Reutilização e flexibilidade dos serviços de negócio;
- Suporte de componentes de interface Ajax e HTML5;
- Ligação entre a interface do utilizador e os serviços de negócio;

- Camada de acesso a bases de dados relacionais.

2.1.3- Benefícios da Oracle ADF

A Oracle ADF fornece uma completa estrutura de *Model-View-Controller* para desenvolvimento aplicacional e as soluções fornecidas pelas várias camadas desta *framework* reduzem a quantidade código por parte dos programadores, o que resulta em ciclos de desenvolvimento mais rápidos e numa curva de aprendizagem maior.

Além disso a Oracle ADF Essentials oferece funcionalidades avançadas e inovadoras que permitem criar aplicaçoes com funcionalidades mais ricas e eficientes. Existem diferentes fornecedores que oferecem ferramentas de desenvolvimento, *frameworks* baseadas em especificações Java EE. Então, para o desenvolvimento de qualquer aplicação Java EE, os programadores necessitam de conhecer essas ferramentas e a sua utilização. Mas a maioria dessas ferramentas não suportam todas as camadas de desenvolvimento Java EE. Como resultado, os programadores por vezes encontram-se um pouco limitados na escolha de ferramentas ou *frameworks*. A Oracle ADF fornece um ambiente completo de desenvolvimento, código de infraestrutura, padrões de *design* e facilidade de desenvolvimento. Deste modo, esta tecnologia tornou-se muito popular na comunidade de desenvolvimento Java EE.

2.5. BASES DE DADOS

Cada organização tem informação crucial e pertinente que deve armazenar e gerir para cumprir com as exigências e atividades diárias e esta informação deve estar disponível para quem necessitar. A maioria das empresas hoje em dia apoiam-se numa base de dados para automatizar os sistemas de informação. Uma base de dados é a chave para a gestão da

informação porque, para além de normalmente conter regras de negócio bastante importantes, também permite um ambiente multiutilizador, de modo a que os utilizadores possam aceder aos mesmos dados, em simultâneo. Contém, igualmente, funções que impedem o acesso não autorizado e fornece soluções eficientes para a recuperação de falhas.

As bases de dados fornecem uma solução otimizada às organizações, que lhes permite consolidar e gerir melhor as suas infraestruturas, entregar rapidamente resultados, gerir riscos de segurança e reagir rapidamente a oportunidades de negócio. [15]

A escolha do *hardware* que irá suportar a base de dados é de extrema importância, dado que a velocidade de atualização é crucial e, normalmente, as bases de dados guardam grande volume de informação.

3. FERRAMENTAS DE DESENVOLVIMENTO

Este capítulo apresenta as ferramentas utilizadas no decorrer do projeto e que permitiram a sua construção e desenvolvimento. Deste modo, apresentam-se as linguagens de programação utilizadas para a criação e consulta da base de dados, para a codificação do *software* e respetivas regras de negócio e para a construção das interfaces gráficas. Descreve-se também o servidor *weblogic* utilizado para executar a aplicação e os *softwares* utilizados para o desenvolvimento da aplicação.

3.1. LINGUAGEM UML

Unified Modelling Language (UML) é uma linguagem ou uma notação de diagramas de especificação, visualização e documentação de *software* orientada a objetos. A UML não é um método para desenvolvimento mas sim uma linguagem para definir o desenho e

estrutura do projeto de forma padronizada. Neste trabalho foi utilizada na análise e levantamento de requisitos.

A modelação que é possível realizar através da notação UML permite uma melhor gestão da complexidade do SI. Permite também detetar erros e requisitos em falta, perceber o impacto que terá o SI de acordo com os respetivos requisitos, definir como deve ser efetuada a implementação para que a solução preencha os objetivos inerentes ao desenvolvimento do sistema de informação e definir o conjunto de ações da aplicação. [16]

A linguagem UML inclui diversos elementos de modelo que representam as diferentes partes de um SI. Estes elementos são diagramas que representam o sistema e alguns destes, os mais importantes encontram-se mais detalhados a seguir: [17]

- Diagrama de Casos de Uso: são descrições de interações típicas entre os utilizadores de um sistema e esse mesmo sistema. Representam a interface externa do sistema e especificam um conjunto de exigências que o sistema deve cumprir. Descrevem os atores intervenientes no sistema, e as respetivas ações, funcionalidades, dependências e relações de informação que o sistema possui;

- Diagrama de Classes: uma classe define os atributos e os métodos de uma determinada parte do SI e o diagrama de classes descreve as várias classes que compõem o projeto assim como os diferentes atributos pertencentes às classes;

- Diagrama de Entidade-Relação: mostra as diversas classes do sistema, as relações entre as propriedades de cada classe e as restrições entre os dados dessas classes. Estes Diagramas ER ou Modelos ER (Entidade-Relação) mostram o desenho conceitual dos fluxos existentes na base de dados e caraterizam e definem as várias entidades do sistema de informação e as relações e restrições entre as mesmas. Cada entidade possui um conjunto de atributos que descrevem as suas propriedades.

3.2. LINGUAGENS DE PROGRAMAÇÃO

De seguida encontra-se uma breve descrição das várias linguagens de codificação utilizadas e qual o contexto da sua utilização.

3.2.1- SQL

Structured Query Language (SQL) é uma linguagem que permite o acesso e manipulação de bases de dados, sendo utilizada para realizar comunicações com a mesma. É a linguagem padrão para a gestão e desenvolvimento de bases de dados relacionais. As suas instruções são utilizadas para executar tarefas como inserir, apagar, editar ou recuperar dados de uma base de dados. [18] A linguagem SQL para além de ser utilizada na aplicação para realizar consultas e alterações dos dados na base de dados, foi também utilizada para a criação da mesma, respeitando o modelo definido durante a análise de requisitos.

3.2.2- Java

Figura 5- Símbolo da Linguagem Java

Java é uma linguagem de programação orientada a objetos, baseada em classes e com características que a tornam adequada para o desnvolvimento *web*. As aplicações Java de

menor dimensão são chamadas de *Applets* Java, podem ser obtidas a partir de um servidor *web* e executadas num computador por um navegador *Web* compatível com Java. [19]

A programação em Java é bastante intuitiva, eficiente, além disso, permite utilizar funções através das bibliotecas fornecidas, sendo possível aceder a arquivos na internet. Para além de ser uma linguagem segura e robusta, gere a memória automaticamente utilizando um coletor de lixo automático. Este coletor é executado, em segundo plano, como um processo de baixa prioridade e mantém o controlo de todos os objetos e referências a esses objetos num programa Java. [20]

Esta linguagem de programação foi utilizada para codificar toda a lógica de funcionamento do projeto, para alguns métodos de interação do utilizador com a aplicação e para definir algumas regras de negócio para que o *software* se tornasse mais intuitivo e facilitador para os seus utilizadores.

3.2.3- XML

A *Extensible Markup Language* (XML) foi criada para que os documentos estruturados pudessem ser utilizados através da *web*. Originalmente foi concebida como uma linguagem para definir novos formatos de documentos para a *World Wide Web* e deriva do SGML (*Standard Generalized Markup Language*). Pode ser considerada como uma linguagem para definir formatos simples e bastante flexíveis. SGML e XML são formatos baseados em texto que fornecem mecanismos para descrever estruturas de documentos usando *tags* de marcação (palavras entre os simbolos “<” e “>”). A utilização de XML tem crescido, sendo agora, não só útil para descrever novos formatos de documentos para a *Web*, mas também é adequada para descrever dados estruturados, isto é vários dados agrupados numa mesma estrutura, construída em XML. [21]

Atualmente é possível verificar vários documentos de negócios que utilizam XML como um formato de representação.

3.3. MIDDLEWARE

O termo *Middleware,* diz respeito a uma ferramenta que serve para agregar os vários serviços que permitem e constituem o desenvolvimento *web.* É composto por Java EE e pelo servidor utilizado para suportar o desenvolvimento e execução da aplicação, o Oracle *Weblogic*.

3.3.1- Java EE

Java *Platform Enterprise Edition* (Java EE) é um padrão de *software* empresarial, *open source* desenvolvido usando Java *Community Process*, com contribuições de especialistas da indústria e de organizações comerciais. Cada versão integra novos recursos que se alinham com as necessidades da indústria, melhora a portabilidade de aplicações e aumenta a produtividade de quem desenvolve.

Java EE inclui um conjunto de serviços que facilitam o desenvolvimento aplicações estáveis e poderosas de nível empresarial, tem sido bastante utilizado em aplicações importantes e extensas. [22]

Java EE é uma plataforma de programação para servidores na linguagem de programação Java, que fornece diversas API para o desenvolvimento e execução de *softwares* corporativos, incluindo serviços de rede e *web*, outras aplicações de larga escala, multicamadas, escaláveis, confiáveis e seguras. A API permite o mapeamento objeto-relacional, arquiteturas de multicamadas e distribuídas e, também, *web services*.

Possui bibliotecas que fornecem funcionalidades para implementar *software* Java distribuído, tolerante a falhas, sendo baseada amplamente em componentes modulares executados num servidor. A plataforma Java EE é considerada um padrão de desenvolvimento visto que o fornecedor de *software*, para esta plataforma, deve seguir determinadas regras se pretender declarar os seus produtos como compatíveis com Java EE.

JEE é uma coleção de tecnologias e API para a plataforma Java projetada para suportar aplicações empresariais, ou seja, os requisitos de negócio que podem ser geralmente classificados como de grande escala, distribuídos, transacionais e altamente disponíveis. Em geral, aplicações empresariais ou corporativas referem-se a *software* alojado em servidores para suportar a gestão do negócio, ou parte do negócio, da empresa. [23]

O Java EE consiste em uma série de especificações bem detalhadas, especificando como deve ser implementado um *software* que realiza serviços de infraestrutura. Essas especificações, quando implementadas, auxiliam bastante o desenvolvimento de aplicações, pois evita a necessidade de grande parte do código para as infraestruturas, que exige muito trabalho.

As aplicações *web* de hoje em dia já possuem regras de negócio bastante complexas e, além dessas regras, conhecidas como requisitos funcionais de uma aplicação, existem outros, conhecidos como requisitos não-funcionais, tais como: a persistência em base de dados, transações, acesso remoto, *web services*, gestão de conexões HTTP, cache de objetos, gestão de sessões *web*, entre outros. Estes requisitos não-funcionais não necessitam de descrição em código, ou quase nenhuma, o que facilita bastante a tarefa de quem desenvolve. [24]

As tecnologias Core que compõem JEE são:

- Java *Database Connectivity* (JDBC);
- Java *Naming and Directory* Interface (JNDI);
- *Enterprise Javabeans Components* (EJB);
- Java *Remote Method Invocation* (RMI);
- *Java Server Pages* (JSP);
- Java *Server Faces* (JSF);
- Java *Servlets*;
- *Extensible Markup Language* (XML);
- Java *Message Service* (JMS);
- Java *Interface Definition Language* (IDL);
- Java *Transaction Service* (JTS);

- Java *Transaction* API (JTA);
- JavaMail*;*
- JavaBeans *Activation Framework* (JAF);
- Java API para XML *Binding* (JAX-B);
- Java *Autenthication and Authorization Service* (JAAS);
- Java *Persistence* API (JPA);
- Java API para XML *Web Services* (JAX-WS);
- Java API para XML *Binding* (JAX-B);
- Java *Management Extensions* (JMX).

JSP, JSF, JSTL e Servlets são as especificações essenciais para desenvolver em plataforma *Web*.

A tecnologia ADF, fornece um ambiente completo de desenvolvimento, código de infraestrutura, padrões de *design* e facilidade de desenvolvimento. Assim, a combinação da *framework* Oracle ADF com o ambiente de desenvolvimento é bastante utilizada no desenvolvimento Java EE.

3.3.2- Oracle Weblogic Server

Um Servidor *Weblogic* é baseado em *Java Enterprise Edition* (JEE), sendo a plataforma padrão usada para criar aplicações empresariais com base em Java. Como as aplicações JEE são módulos padronizados, o servidor *Weblogic* pode automatizar muitas tarefas de nível de sistema, que de outra forma teriam exigido tempo de programação. Foi desenvolvido para ligar os utilizadores a um ambiente de computação distribuído e para facilitar a integração de aplicações corporativas.

As principais características do servidor *Weblogic* incluem conectores, *Enterprise JavaBeans* (EJB), *pooling* de recursos (gestão de conexões) e partilha de conexão que permite a construção aplicações escaláveis. Possui uma consola de administração com uma

interface para uma gestão mais eficiente dos recursos presentes no *Weblogic* e das aplicações instaladas. [25]

Relativamente à arquitetura, o *Weblogic Server* é um servidor de aplicações, que pode ser descrito como sendo uma plataforma para desenvolvimento e implementação de aplicações corporativas com multicamadas distribuídas. Este servidor centraliza serviços de aplicações, tais como, a funcionalidade do servidor *Web* e componentes de negócios. Usa tecnologias, como *cache* e *connection pool* (área que gere as conexões das aplicações ao servidor) para melhorar a utilização de recursos e o desempenho do projeto. O *Weblogic* opera na camada intermediária de uma arquitetura de múltiplas camadas (*n-tier*). A arquitetura de múltiplas camadas determina onde e como são executados os componentes de *software* que compõem o sistema de computação, assim como *hardware*, rede e utilizadores. Escolhendo a melhor localização para cada componente de *software* permite desenvolver aplicações informáticas mais rápidas, facilita a implementação e administração, e proporciona maior controlo sobre o desempenho, a utilização, segurança, escalabilidade e confiabilidade.

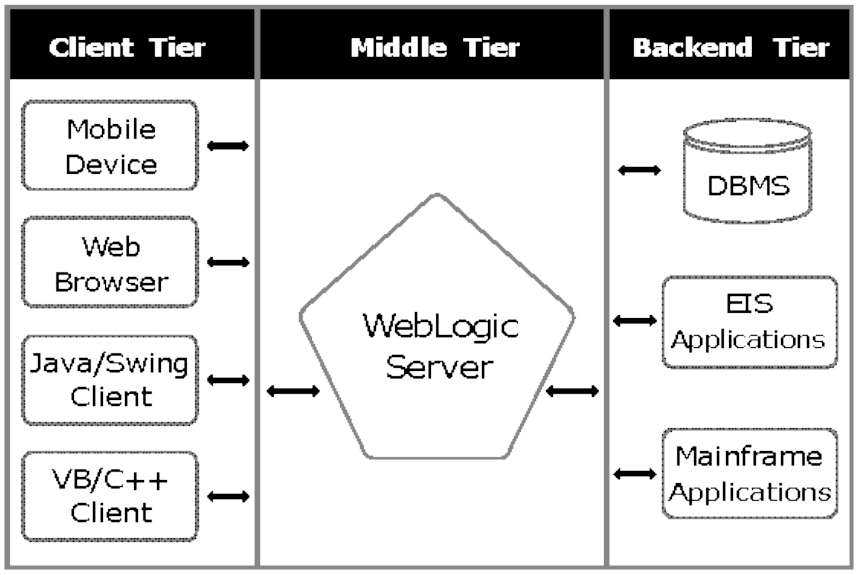


Figura 6- Arquitetura do Weblogic [26]

Os componentes de uma arquitetura de múltiplas camadas (*n-tier*) consiste em três camadas:

- A camada do cliente (*client tier*) contém programas executados pelos utilizadores, incluindo navegadores *web*. Estes programas podem ser escritos em praticamente qualquer linguagem de programação.
- A camada intermediária (*middle tier*) contém *Weblogic Server* e outros servidores que são abordadas diretamente pelos clientes, tais como servidores *web* existentes ou servidores *proxy*.
- A *back-end tier* contém recursos da empresa, como a base de dados e aplicações tipo ERP, ou outras.

A *client tier* permite aceder ao *Weblogic* diretamente, ou através de outro servidor *Web* ou servidor proxy. O *Weblogic Server* normalmente estabelece ligação com serviços de *back-end* em nome de clientes, no entanto, os clientes podem aceder diretamente aos serviços de *back-end* usando um driver JDBC.

O Servidor *Weblogic* facilita os desenvolvimentos devido às funcionalidades padronizadas que já se encontram integradas no mesmo. Além disso, *Weblogic Server* pode gerir até 6.000 utilizadores simultaneamente com milhares de transações e sessões. A Oracle fornece, também, ferramentas para o desenvolvimento, teste e implementação de aplicações. [27]

Com o Oracle *Weblogic Server* é possível executar aplicações de infraestrutura convencional e infraestrutura de computação em *cloud* (nova plataforma da Oracle que tem como vantagem a possibilidade de adesão ao servidor em qualquer parte do mundo, através da Internet).

3.4. SOFTWARE

Neste último subcapítulo das ferramentas utilizadas encontra-se o *software* jDeveloper, utilizado na construção da aplicação através da tecnologia Oracle ADF, e o Oracle *Database* XE, utilizado na construção do modelo de dados que suporta a aplicação.

3.4.1- Oracle jDeveloper

O Oracle jDeveloper (IDE utilizado no desenvolvimento do projeto) é um ambiente de desenvolvimento integrado gratuito que simplifica o desenvolvimento baseado em Java de aplicações Java EE, abrangendo o ciclo completo de vida de desenvolvimento e com suporte embutido para JSF e JDBC. [28]

Figura 7- Logótipo do *software* jDeveloper

Este IDE é usado para o desenvolvimento de aplicações JAVA, XML, SQL, PL/SQL, HTML, JavaScript, BPEL e PHP. Proporciona um editor de JavaScript para o desenvolvimento de aplicações AJAX, entre outros. Reúne características e ferramentas de apoio ao desenvolvimento de *software* com o objetivo de agilizar este processo.

3.4.2- Oracle Database Express Edition

Oracle Database 11g *Express Edition* (*Oracle Database XE*) é uma versão gratuita de *software* para desenvolvimento e manutenção de bases de dados relacionais. É de fácil instalação e apresenta uma interface bastante intuitiva para administrar a base de dados, criar tabelas e outros objetos, importação e exportação de dados, executar consultas e *scripts* SQL. [29]

Oracle Database XE 11g foi o *software* utilizado para a criação da base de dados, assim como o conteúdo da mesma como as tabelas, chaves primárias e chaves estrangeiras. A base de dados construída através deste *software* servirá de suporte à aplicação desenvolvida neste trabalho.

4. PROJETO

Este projeto, desenvolvido no âmbito da unidade curricular de Tese/Dissertação, possui uma vertente académica mas também uma vertente profissional e rigorosa tendo em conta que o objetivo é o desenho e desenvolvimento de um sistema de informação para ser utilizado numa empresa.

Para atingir esse fim, foi necessário desenvolver uma estrutura robusta e que permitisse uma utilização agradável ao utilizador. Assim, foi fulcral realizar um bom planeamento do projeto, principalmente na análise e levantamento de requisitos uma vez que será o ponto de partida do desenvolvimento do *software*.

A análise de requisitos foi feita através da linguagem UML (*Unified Modeling Language*) e do modelo concetual de dados.

4.1. ANÁLISE E ESPECIFICAÇÃO

O tema desta tese resultou da necessidade de uma empresa, pertencente a familiares do autor, necessitar de um sistema de informação capaz de suportar a sua área de negócio e a gestão do seu processo produtivo. Por isso, numa primeira fase, foi ecessário compreender as regras de negócio para determinar as especificações e necessidades da aplicação. Todos estes requisitos encontram-se em notação UML, para posteriormente se construir o Diagrama de Classes e o Modelo de dados.

Os recursos em termos de infraestrutura que são requeridos, por parte da empresa, para um bom funcionamento do projeto desenvolvido encontram-se descritos no anexo A.

4.1.1- Casos de Uso

O diagrama de Casos de Uso apresentado identifica os atores do sistema e as respetivas funcionalidades.

A figura 7 ilustra o diagrama de casos de uso da aplicação desenvolvida, apresentando dois tipos de utilizadores, um utilizador de baixo nível e outro de alto nível, normalmente associados a funcionários e administração respetivamente.

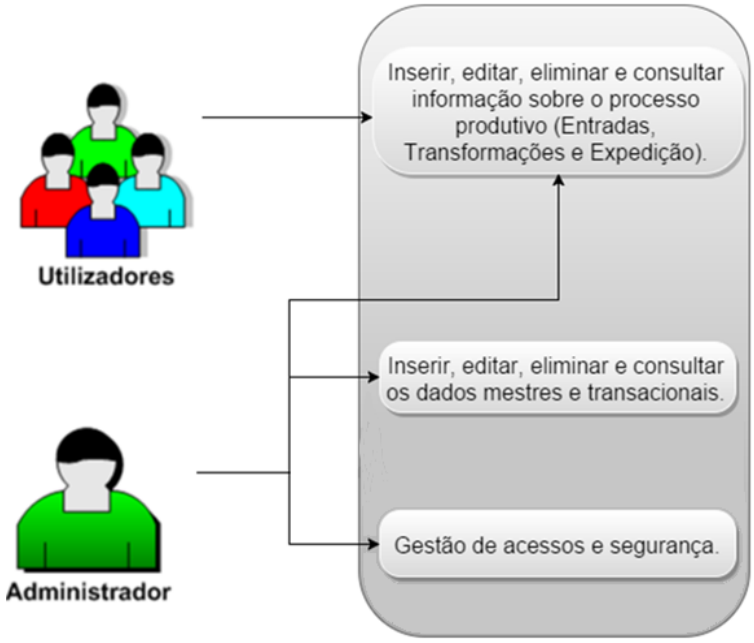


Figura 8- Diagrama de casos de uso

4.1.2- Descrição dos atores

Administrador

A administração possui acesso à base de dados para definir alterações na mesma, gere a segurança e os acessos à aplicação e pode realizar todos os tipos de operações sobre as tabelas que contêm os dados mestres e transacionais. Por dados mestres entende-se fornecedores, clientes, funcionários, matérias-primas, produtos, suplementos e utilizadores, enquanto que os dados globais englobam pagamentos, recebimentos e *stocks*. Para além destas funções pode, também, exercer qualquer tipo de tarefa que seja permitida aos utilizadores.

Utilizador

O Utilizador, normalmente um funcionário, depois de ter efetuado o *login* no sistema, tem permissão para realizar consultas e operações CRUD (*create*, *update* e *delete*) sobre a informação dos diversos departamentos que constituem o processo produtivo da empresa.

- Departamento de Entradas de matéria-prima;
- Departamento de Transformação da matéria-prima;
- Departamento de Expedição de produtos.

4.1.3- Descrição dos casos de uso

Seguidamente são apresentados e detalhados todos os casos de uso da aplicação (tabelas 1 a 24), explicitando as ações, os atores intervenientes e as condições para a concretização da ação.

Tabela 1- Caso de uso para a consulta de informação

Caso de Uso 1	Consultar Entradas/Transformações/Expedição/Stocks
Descrição	O utilizador pode pesquisar e consultar os dados do processo produtivo.
Atores	Administrador/Utilizador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Nenhuma.

Tabela 2- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 1)		
	Ações do ator	Ações do sistema
1	Simplesmente consultar ou pode aplicar filtros sobre as tabelas para reduzir/especificar a informação mostrada.	Caso o utilizador pesquise, a tabela será filtrada consoante os valores introduzidos.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Tabela 3- Casos de uso para a inserção de informação do processo produtivo

Caso Uso 2	Inserir Entradas/Transformações/Expedição
Descrição	O utilizador pode inserir dados no processo produtivo e, depois deste estar inserido, permite a sua consulta, edição e eliminação.
Atores	Administrador/Utilizador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a inserção de informação.

Tabela 4- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 2)		
	Ações do ator	Ações do Sistema
1	Selecionar opção de adicionar registos (botão).	Adiciona uma linha à tabela para se proceder ao seu preenchimento.
2	Preenchimento dos Campos necessários.	Validação dos valores introduzidos.
3	Guardar alterações através do botão Guardar ou do botão Registar, caso esteja na seção de adições rápidas.	Cria um novo registo, na tabela respetiva com identificadores auto incrementais, e atualização do *stock* caso seja uma Entrada ou Transformação.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Tabela 5- Caso de uso para a edição de informação do processo produtivo

Caso Uso 3	Editar Entradas/Transformações/Expedição
Descrição	O utilizador pode editar dados no processo produtivo através das respetivas tabelas existentes no fluxo principal da aplicação.
Atores	Administrador/Utilizador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a atualização da informação.

Tabela 6- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 3)		
	Ações do ator	Ações do sistema
1	Selecionar o registo que se pretende editar.	Define o respetivo registo como selecionado.
2	Edição dos atributos do registo pretendido.	Validação dos valores introduzidos.
3	Salvar alterações através do botão Guardar.	Atualiza informação desse registo, na respetiva tabela, e atualização do *stock* caso seja uma Entrada ou Transformação.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Tabela 7- Caso de uso para a eliminação de informação do processo produtivo

Caso Uso 4	Eliminar Entradas/Transformações/Expedição
Descrição	O utilizador pode eliminar registos das diferentes tabelas do processo produtivo através das respetivas tabelas existentes no fluxo principal da aplicação.
Atores	Administrador/Utilizador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a eliminação da informação.

Tabela 8- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 4)		
	Ações do ator	Ações do sistema
1	Selecionar o registo que se pretende eliminar.	Define o respetivo registo como selecionado.
2	Clicar no botão de eliminar registos.	Elimina a respetiva linha da tabela.
3	Guardar alterações através do botão Guardar.	Guarda as alterações efetuadas respetiva tabela, e atualização do *stock* caso seja uma Entrada ou Transformação.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Administrador:

Tabela 9- Caso de uso para a consulta de dados mestres/transacionais

Caso Uso 5	Consultar dados Mestres/Transacionais
Descrição	O administrador pode pesquisar e consultar os dados mestres e transacionais.
Atores	Administrador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Nenhuma.

Tabela 10- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 5)		
	Ações do ator	Ações do sistema
1	Simplesmente consultar ou pode aplicar filtros sobre as tabelas para reduzir/especificar a informação mostrada.	Caso o administrador pesquise, a tabela será filtrada consoante os valores introduzidos.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Tabela 11- Caso de uso para a inserção de dados mestres

Caso Uso 6	Inserir dados Mestres
Descrição	O administrador pode inserir matérias-primas, produtos, fornecedores, clientes e funcionários.
Atores	Administrador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a inserção de informação.

Tabela 12- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 6)		
	Ações do ator	**Ações do sistema**
1	Selecionar opção de adicionar registos (botão) na tabela que se pretende inserir.	Adiciona uma linha à tabela para se proceder ao seu preenchimento.
2	Preenchimento dos Campos necessários.	Validação dos valores.
3	Guardar alterações através do botão Guardar ou do botão Registar, caso esteja na seção de adições rápidas.	Cria um novo registo, na tabela respetiva, com identificadores auto incrementais.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Tabela 13- Caso de uso para a edição de dados mestres

Caso Uso 7	Editar dados Mestres
Descrição	O administrador pode editar dados mestres através das respetivas tabelas existentes nas páginas acessíveis através das funções administrativas.
Atores	Administrador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a atualização da informação.

Tabela 14- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 7)		
	Ações do ator	Ações do sistema
1	Selecionar o registo que se pretende editar.	Define o respetivo registo como selecionado.
2	Edição dos atributos do registo pretendido.	Validação dos valores introduzidos.
3	Salvar alterações através do botão Guardar.	Atualiza informação desse registo, na respetiva tabela.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema

Tabela 15- Caso de uso para a eliminar de dados mestres

Caso Uso 8	Eliminar dados Mestres
Descrição	O utilizador pode eliminar registos das diferentes tabelas dos dados mestres através das respetivas tabelas existentes nas páginas acessíveis através das funções administrativas.
Atores	Administrador/Utilizador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a eliminação da informação.

Tabela 16- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 8)		
	Ações do ator	Ações do sistema
1	Selecionar o registo que se pretende eliminar.	Define o respetivo registo como selecionado.
2	Clicar no botão de eliminar registos.	Elimina a respetiva linha da tabela.
3	Guardar alterações através do botão Guardar.	Guarda as alterações efetuadas respetiva tabela, e atualização do *stock* caso seja uma Entrada ou Transformação.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema

Tabela 17- Caso de uso para a inserção de dados transacionais

Caso Uso 9	Inserir dados Transacionais
Descrição	O administrador pode inserir pagamentos e recebimentos.
Atores	Administrador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a inserção de informação.

Tabela 18- Fluxo de eventos do caso de uso anterior

	Fluxo básico de eventos (Caso de Uso 9)	
	Ações do ator	Ações do sistema
1	Selecionar o registo que pretende definir como recebido marcando-o como "pago" na última coluna da tabela de pagamentos ou recebimentos.	Adiciona o registo selecionado à tabela de pagamentos ou recebimentos, dependendo da ação.
2	Guardar alterações através do botão Guardar ou do botão Registar, caso esteja na seção de adições rápidas.	Guarda as alterações efetuadas respetiva tabela de pagamentos ou recebimentos.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Tabela 19- Caso de uso para a edição de dados transacionais

Caso Uso 10	Editar dados Transacionais
Descrição	O administrador pode editar os dados transacionais através das respetivas tabelas existentes nas páginas acessíveis através das funções administrativas.
Atores	Administrador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a atualização da informação.

Tabela 20- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 10)		
	Ações do ator	**Ações do sistema**
1	Selecionar o registo que se pretende editar.	Define o respetivo registo como selecionado.
2	Edição dos atributos do registo pretendido.	Validação dos valores introduzidos.
3	Salvar alterações através do botão Guardar.	Atualiza informação desse registo, na respetiva tabela.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema.

Tabela 21- Caso de uso para a eliminação de dados transacionais

Caso Uso 11	**Eliminar dados Transacionais**
Descrição	O utilizador pode eliminar registos das diferentes tabelas do processo produtivo através das respetivas tabelas existentes nas páginas acessíveis através das funções administrativas.
Atores	Administrador.
Pré-condições	Estar registado no Sistema com um utilizador válido.
Pós-condições	Deve confirmar a eliminação da informação.

Tabela 22- Fluxo de eventos do caso de uso anterior

Fluxo básico de eventos (Caso de Uso 11)		
	Ações do ator	**Ações do sistema**
1	Selecionar o registo que se pretende eliminar.	Define o respetivo registo como selecionado.
2	Clicar no botão de eliminar registos.	Elimina a respetiva linha da tabela.
3	Guardar alterações através do botão Guardar.	Guarda as alterações efetuadas respetiva tabela, e atualização do *stock* caso seja uma Entrada ou Transformação.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema

Tabela 23- Caso de uso para a gestão de acessos e segurança

Caso Uso 12	Gestão de acessos e segurança
Descrição	O Administrador gere os utilizador registado no sistema e respetivas permissões.
Atores	Administrador.
Pré-condições	Estar registado no Sistema com um utilizador válido e com permissões de administrador.
Pós-condições	Confirmar as atualizações efetuadas.

Tabela 24- Fluxo de eventos do caso de uso anterior

	Fluxo básico de eventos (Caso de Uso 12)	
	Ações do ator	Ações do sistema
1	Adicionar um utilizador.	Define o respetivo registo como selecionado.
2	Definir o nome de utilizador, palavra-passe e as permissões do novo utilizador.	Envia a informação para a base de dados e regista este utilizador para que este possa aceder à aplicação.
	Utilizador não se encontra com login efetuado.	Necessário efetuar login no sistema e com permissões de administrador.

4.1.4- Diagrama de classes

Na figura 8, está representada uma parte do diagrama de classes em formato UML. Este diagrama contempla todos os atributos que devem estar presentes, tanto nas tabelas da base de dados como nas classes da aplicação. As classes deste diagrama possuem os atributos necessários ao correto funcionamento do sistema de informação, como a identificação dos dados (*id*), nome, *e-mail*, datas, entre outros, de cada dado presente nessa classe. Contém

também atributos de outras classes, dado que é obrigatório a comunicação entre as classes para responder aos objetivos do projeto. No decorrer da realização do projeto, foi necessário adicionar algumas especificações extra, para a obtenção de mais-valias e um bom funcionamento do projeto. O diagrama presente na figura 8 apenas contém algumas das classes que compõem o projeto, o diagrama completo encontra-se no Anexo B.

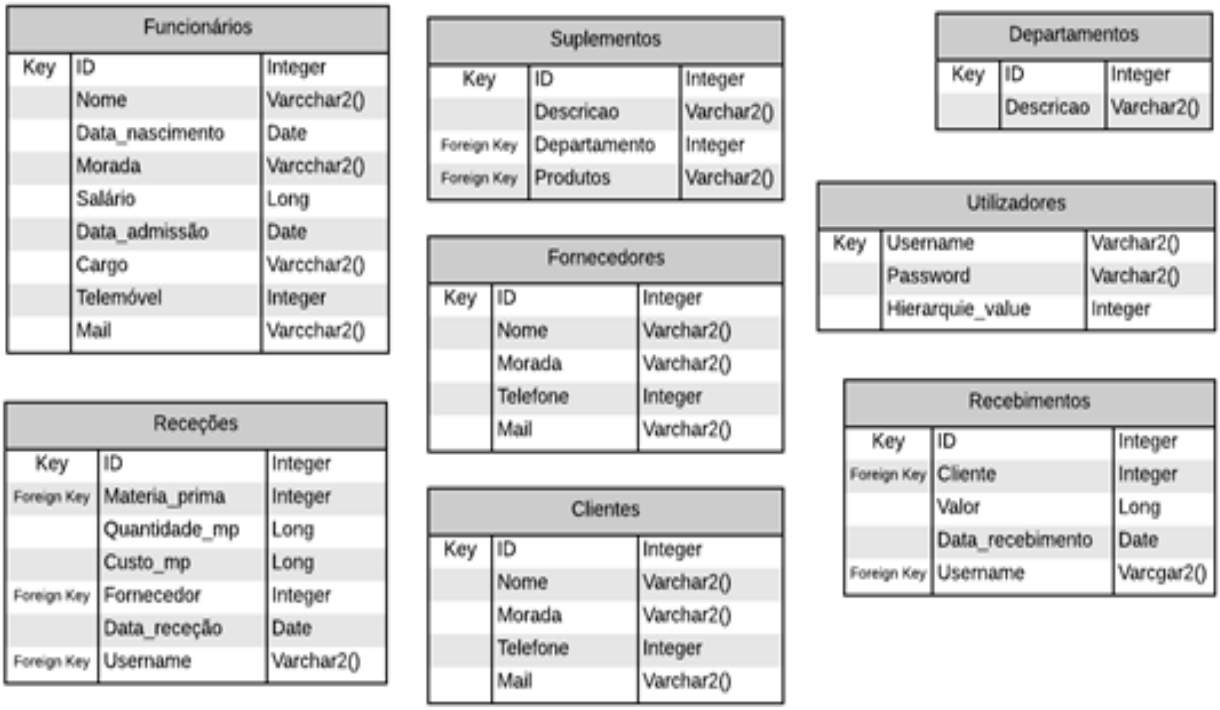


Figura 9- Diagrama de classes

4.1.5- Modelo de dados

O modelo de dados ou diagrama Entidade-Relação tem a finalidade de descrever, de forma concetual, os dados do sistema de informação. Demonstra também a relação entre as entidades, de modo a que se possam definir adequadamente as funções pretendidas no sistema de informação. A construção do *software* assenta também nesta base. No diagrama dafigura 9 de casos de uso podemos encontrar as tabelas da base de dados e os seus

respetivos atributos, assim como o tipo de atributo e quais destes são chaves primárias e chaves estrangeiras.

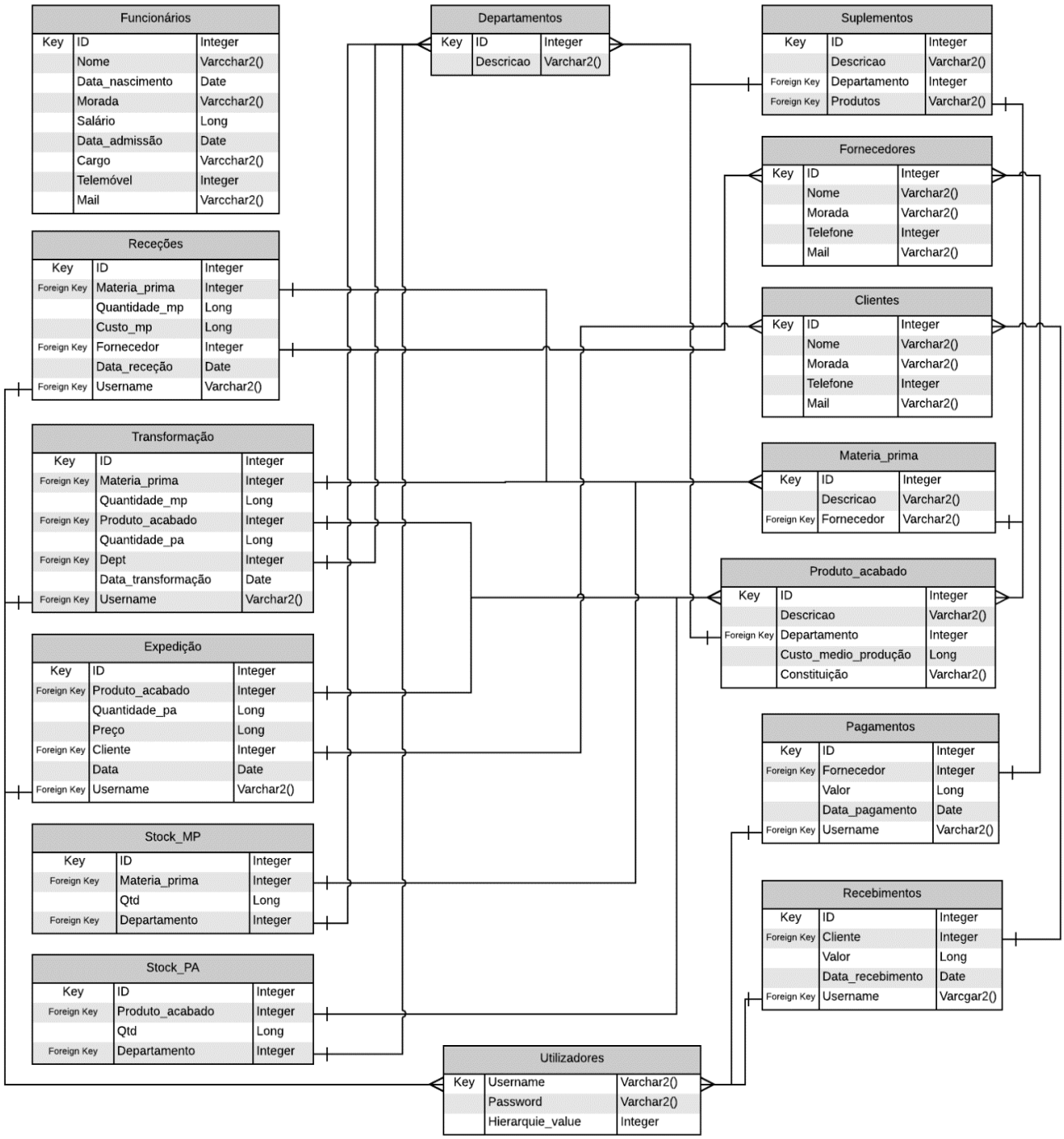


Figura 10- Modelo Físico da Base de Dados (Entidade-Relação)

4.2. DESENVOLVIMENTO

Para uma boa compreensão desta fase de desenvolvimento, todas as classes (codificadas em Java) são apresentadas e explicadas com as respetivas funcionalidades:

- Classe *Index*: Esta classe, codificada em Java tal como todas as seguintes, contém a lógica de programação por detrás da interface gráfica de *login*. Para além de conter todas as declarações do *layout* e dos componentes do respetivo ecrã, contém também as validações necessárias aquando de uma entrada de um utilizador. Trata os erros e consequentes mensagens em caso de *login* inválido e impede o acesso à página principal sem nenhum utilizador registado.

- Classe *Main*: O fluxo principal do *software* assenta nesta classe, isto é, o processo produtivo da empresa (entradas, transformações e expedição) está presente em cima desta classe, embora cada área do fluxo principal usufrua de uma classe exclusiva para o tratamento da lógica. Na classe *Main* está presente a codificação que permite o acesso às funções administrativas e aos reencaminhamentos para as diferentes páginas existentes na seção das funções administrativas.

- Classe Entradas: Esta classe suporta todas as funcionalidades representadas na parte das entradas no fluxo principal da aplicação, composta por um formulário para adição rápida de entradas, tabela das entradas para uma consulta ou atualização dos dados de uma forma mais completa, tabela de stock*s* da matéria-prima para permitir a sua consulta enquanto se adiciona entradas e métodos para controlar a coerência da informação como a adição da quantidade da matéria-prima à tabela dos stocks após a inserção de uma entrada.

- Classe Transformações: A classe Transformações contém exatamente todos os métodos que a classe Entradas, mas os objetos que representam as tabelas correspondem às tabelas de transformações e de *stock* de produtos acabados, ao contrário da classe anterior. Para além destes, possui também funcionalidades para atualizar o campo da quantidade da tabela dos *stocks* de matéria-prima dependendo da quantidade que foi alocada ao produto acabado.

- Classe Expedição: Esta classe permite expedir determinadas quantidades de produtos para clientes a um determinado preço e, após o concluir desta ação, atualiza a tabela de *stocks* de produtos visto que haverá alterações a refletir nesta.

- Classe Cliente, Fornecedores, Funcionários: Estas classes são o suporte das interfaces que se encontram acessíveis através da área das funções administrativas, como o ecrã dos clientes, fornecedores e funcionários. Estes ecrãs permitem uma pesquisa extensa sobre os registos da tabela, sendo que algum dos atributos a pesquisar sugerem qual o registo que queremos pesquisar à medida que vamos escrevendo, ou seja, vai sugerindo quais as hipóteses que coincidem com o que se vai escrevendo no campo para pesquisar. É nestas interfaces gráficas que se trata de informação mestre da empresa, de modo a que os dados permanecem coerentes, fidedignos e apenas alteráveis através destes ecrãs, garantindo a consistência dos dados e evitando a introdução de registos inválidos, como por exemplo a utilização de clientes, fornecedores e funcionários inválidos.

- Classe Produto: Classe que gere as funcionalidades existentes na página que permite tratar da informação dos produtos, isto é, onde se pode criar novos produtos e eliminar produtos já não comercializados pela empresa, posteriormente atualizando a tabela do *stock* caso algum dos produtos seja eliminado e ainda se encontre em *stock*. Permite uma pesquisa alargada por produtos de modo a facilitar possibilitar uma análise destes, definir qual o custo médio da produção dos artigos e qual a sua composição;

- Classe Matéria-Prima: Esta possui as mesmas caraterísticas e funcionalidades da classe anteriormente descrita mas em relação à matéria-prima, em vez de produtos;

- Classe Pagamentos: Classe que trata o ecrã de pagamentos, acessível através do *link* presente nas funções administrativas, que permite definir a encomendas que deram entrada na empresa como pagas. Isto é, permite a consulta das entradas de matéria-prima e registar as mesmas como pagas, sendo adicionado um registo à tabela pagamentos da base de dados. Contém também as funções que se encontram presentes em quase todas as interfaces construídas, como: exportar resultados para Microsoft Office Excel, adição, edição, eliminação, pesquisa de dados, reordenação de colunas da tabela, esconder colunas, anexar ao ecrã inteiro, gravar ou cancelar;

- Classe Recebimentos: Esta possui as mesmas caraterísticas e funcionalidades da classe anteriormente descrita mas em relação aos recebimentos, em vez de pagamentos;

- Classe Stock: Gere as consultas às tabelas de *stock*, quer de matéria-prima quer de produtos, e todas as funções inerentes a este ecrã que também se encontra na seção funções administrativas. Neste ecrã não é possível a adição, edição ou eliminação de *stocks* porque tal ação iria corromper os dados, deste modo, os *stocks* são inteiramente controlados pela aplicação sem necessidade de interferência do utilizador;

- Classe Suplementos: Classe que suporta a interface gráfica dos suplementos, presente na seção das funções administrativas que possibilita a adição, edição, eliminação e consulta destes;

- Classe Utilizadores: Gere as funcionalidades do ecrã de adição de utilizadores, possui métodos que permitem adicionar utilizadores, que permitem consultar os utilizadores existentes, um método para voltar à página principal, entre outros.

Todas as classes anteriormente descritas incorporam métodos que suportam as seguintes funções: adicionar, remover, editar, exportar para Microsoft Office Excel, pesquisar, limpar pesquisa, guardar, cancelar alterações e *logout*.

De seguida são demonstradas e explicadas todas as interfaces da aplicação e respetivas funcionalidades, complementadas com vários *print screen* da aplicação e alguns excertos de código dos métodos utilizados nas classes que compõem a aplicação. De modo a proporcionar uma melhor perceção da aplicação e do seu funcionamento interno.

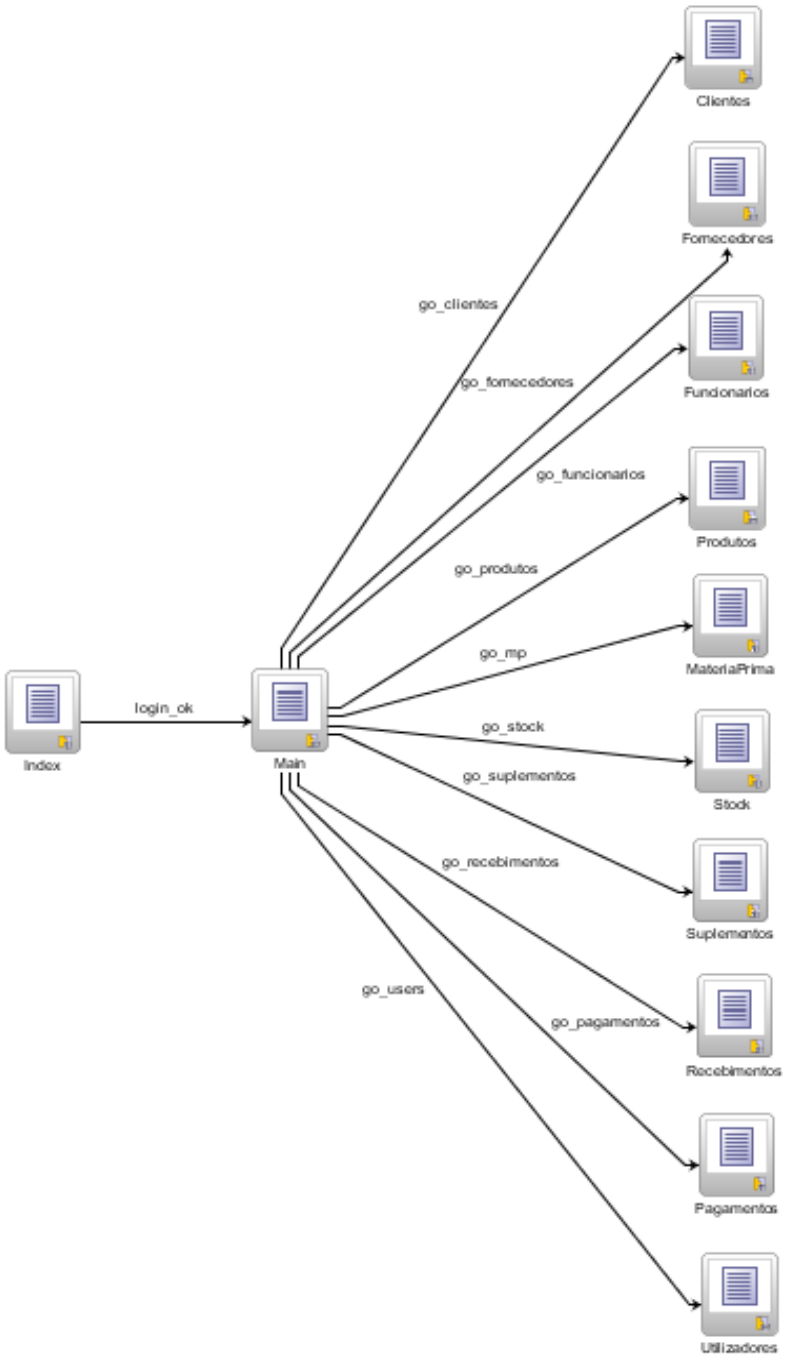


Figura 11- Diagrama dos fluxos da aplicação

Em primeiro lugar, na figura 10, estão descritos os fluxos da aplicação, isto é, a *task flow* que contém todas as páginas da aplicação e os respetivos fluxos entre as mesmas. Estes fluxos entre as páginas JSF (Java Server Faces) permitem navegar entre as páginas, na mesma sessão, para permitir uma melhor performance da mesma quando esta é extensa.

Figura 12- Página inicial da aplicação

No figura 11 está ilustrada a apresentação da entrada no SI, isto é, a página de *login* onde apenas com um utilizador válido possibilita a entrada na aplicação. Esta página mostra mensagem de erros caso algum dos campos a preencher (utilizador e palavra-passe) se encontre vazio ou caso o login efetuado seja invalido, como é possível observar na imagem da figura 12. Senão esta página irá seguir um fluxo e, posteriormente, apresentar a página principal (*Main*) do Sistema de Informação.

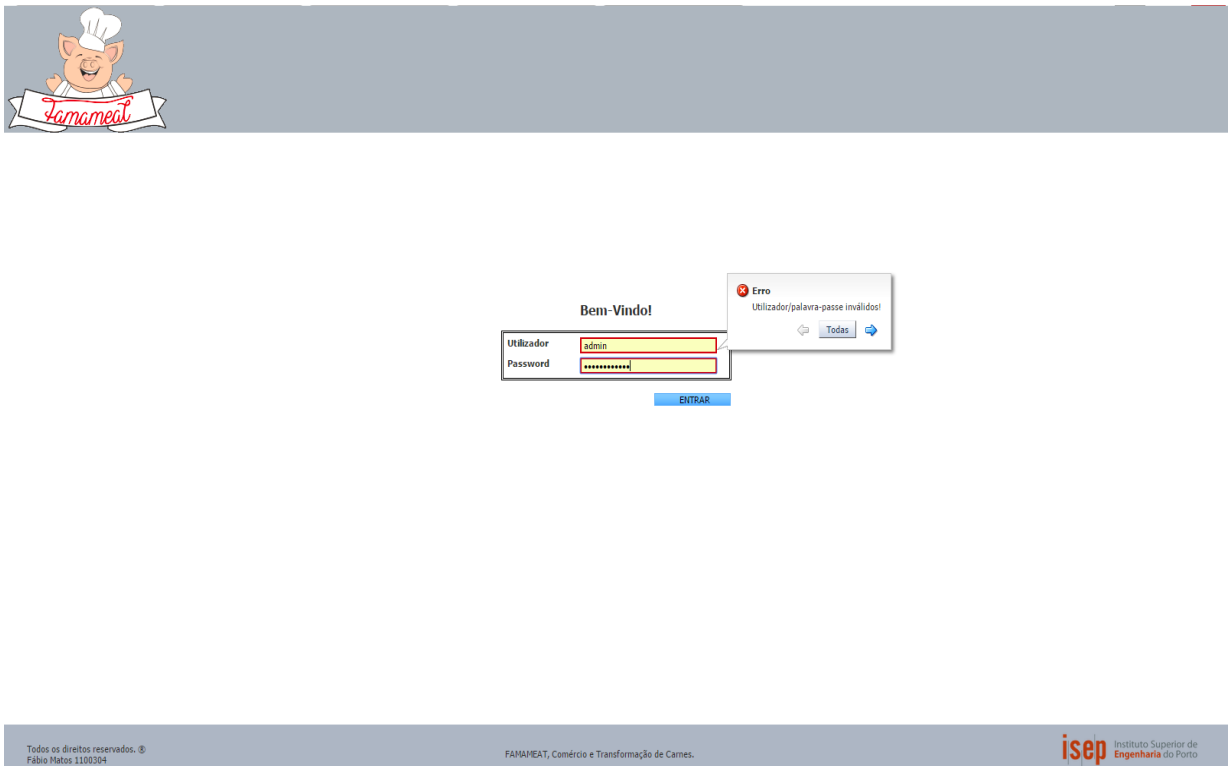


Figura 13- Autenticação inválida

A próxima ilustração, designada como o ecrã principal da aplicação, permite o acesso a todas as páginas que compõem a seção das funções administrativas (lado esquerdo da figura 13) e a todo o processo produtivo. O processo produtivo da empresa está representado no SI divido em três departamentos que correspondem às três *tabs* (Entradas, Transformações e Expedição). Dentro de cada uma das *tabs* que representam os diferentes departamentos do processo produtivo da empresa é possível verificar uma divisão em três seções (adição rápida, atualização detalhada dos registos e consulta de stocks) complementadas com o menu das funções administrativas e o sistema de *logout*. O sistema de *log out* consiste no *icon* existente no canto inferior direito das interfaces que, após quando acionado, termina a sessão do utilizador atual e reencaminha-o para a página de *login*.

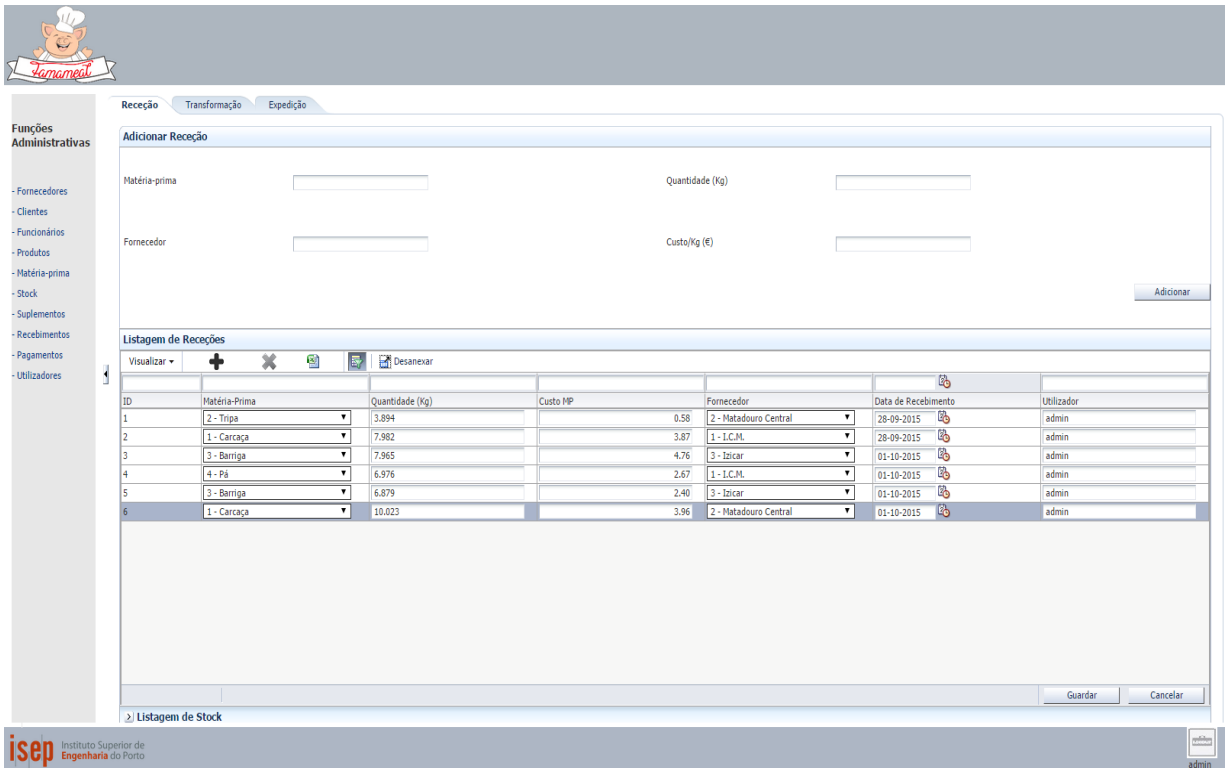


Figura 14- Área de Entradas do processo produtivo

O departamento das Entradas, *tab* que se visualiza na figura 13, permite uma adição rápida de entradas que foi idealizada para quem se encontra a registar os pesos das entradas de matérias-primas, ou seja, existirá um monitor perto das balanças onde os funcionários irão pesar a matéria-prima e registar na aplicação qual a quantidade, matéria-prima e cliente. Mas para a eliminação, edição ou adição mais detalhada, para obtenção dos registos em Microsoft Office Excel de registos ou para uma consulta global dos registos existe uma seção com a tabela que contém a tabela das Entradas e todos os atributos pertencentes. A terceira secção da *tab* das Entradas possui uma tabela com os *stocks* de matéria-prima da empresa, que pode ser bastante útil para uma consulta durante a inserção de entradas ou por permitir um rápido acesso às quantidades das diferentes matérias-primas que, no momento, estão armazenadas à espera de serem transformadas.

Na imagem dafigura 14 verifica-se um erro devido a uma tentativa de entrada forçada na aplicação, ou seja, a utilização da mesma sem nenhum utilizador válido registado. Este erro, mostrado num *pop up*, apenas desaparece após se clicar no botão nele presente, que reencaminha para a página inicial de *log in*.

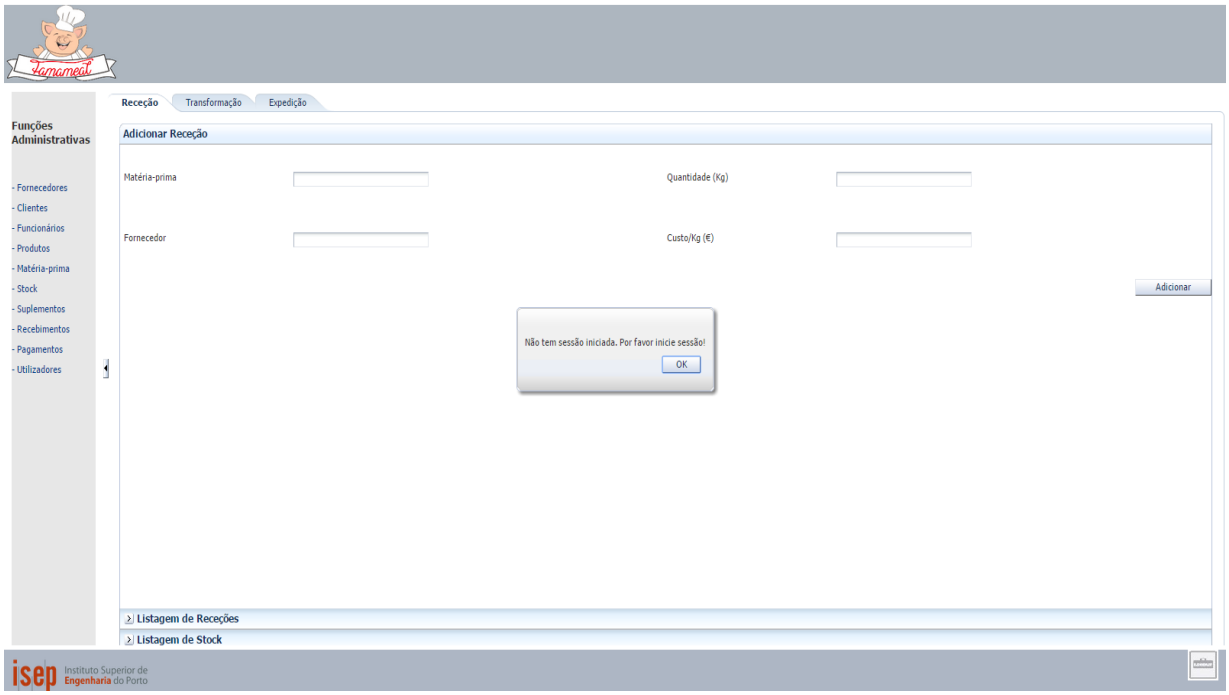


Figura 15- Tentativa de utilização da entrada sem utilizador válido

O separador das Transformações, ilustrado na figura 15, é destinado à alocação de matérias-primas para se obter produtos acabados e é divido em 3 seções, tal como o departamento das Entradas. O funcionamento e as ações das interfaces gráficas destes dois departamentos são bastante idênticas, assumindo que se regista transformações em vez de entradas de matérias-primas e que a consulta do *stock* será de produtos e em vez de matérias-primas. Podemos verificar na seguinte figura que está presente uma mensagem de erro, devido a tentativa de adição de um registo na tabela das transformação com atributos vazios, atributos que são de preenchimento de obrigatório.

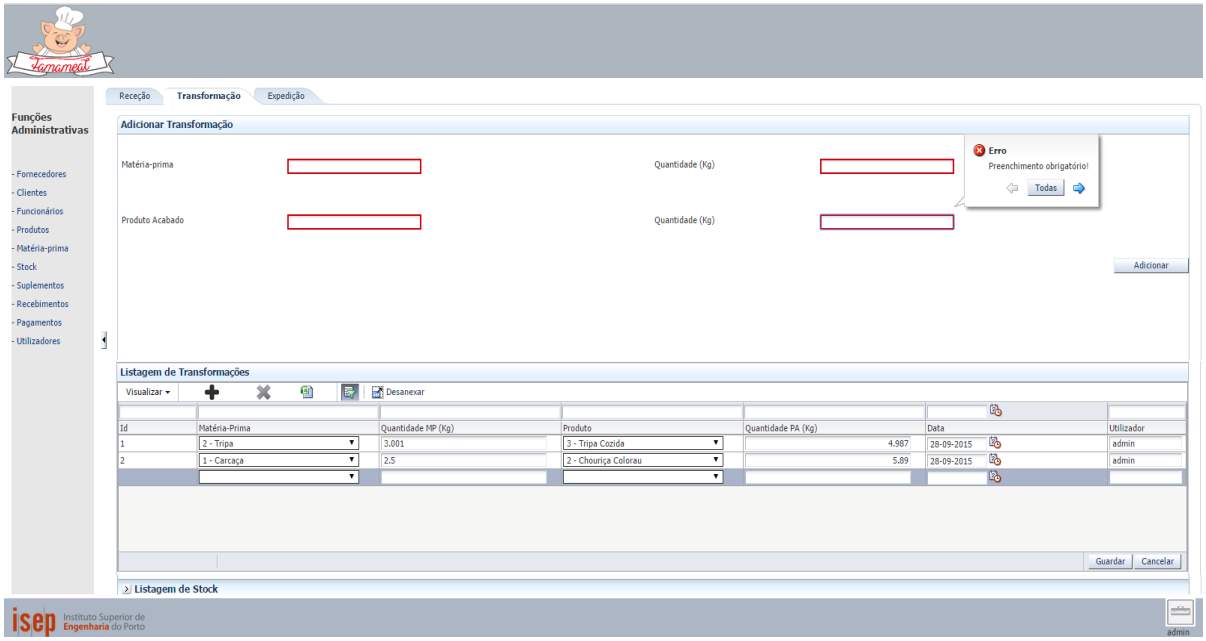


Figura 16- Área de Transformações do processo produtivo

As interfaces da aplicação foram construídas em JSF, que permite um desenvolvimento mais agradável do *layout*/*template* do ecrã. A construção em JSF vai gerando um código fonte em XML que vai representando tudo o que se coloca para ser mostrado no ecrã, permitindo definir todas as propriedades dos componentes e algumas regras de negócio que podem ser incutidas aos componentes, nomeadamente as validações como: mandatório, inativo, apenas de leitura, tamanho do valor, formato do valor e mais opções que aparecem nos atributos dos respetivos componentes. Por exemplo, de seguida está, na figura 16, representado o código XML da tabela presente página dos clientes, neste excerto é possível verificar a declaração da tabela, de uma coluna e do componente presente nessa coluna (componente para entrada de texto) e a definição de algumas das suas propriedades. Como a ligação da informação da tabela ao objeto que representa a tabela da base de dados e algumas validações como a obrigatoriedade de preenchimento.

```
<af:table value="#{bindings.ClientesVO1.collectionModel}" var="row"
          rows="#{bindings.ClientesVO1.rangeSize}"
          emptyText="#{bindings.ClientesVO1.viewable ? 'No data to display.' : 'Access Denied.'}"
          fetchSize="#{bindings.ClientesVO1.rangeSize}" rowBandingInterval="0"
          filterModel="#{bindings.ClientesVO11Query.queryDescriptor}"
          queryListener="#{bindings.ClientesVO11Query.processQuery}"
          filterVisible="false"
          varStatus="vs"
          selectedRowKeys="#{bindings.ClientesVO1.collectionModel.selectedRow}"
          selectionListener="#{bindings.ClientesVO1.collectionModel.makeCurrent}"
          rowSelection="single" id="t1"
          binding="#{backingBeanScope.backing_Clientes.t1}" width="ADFStretchWidth"
          columnStretching="multiple">
    <af:column sortProperty="#{bindings.ClientesVO1.hints.Id.name}" filterable="true"
               sortable="true" headerText="ID"
               id="c2" width="50px">
        <af:inputText value="#{row.bindings.Id.inputValue}"
                      label="Clientes"
                      required="true"
                      columns="#{bindings.ClientesVO1.hints.Id.displayWidth}"
                      maximumLength="#{bindings.ClientesVO1.hints.Id.precision}"
                      shortDesc="#{bindings.ClientesVO1.hints.Id.tooltip}" id="it2"
                      readOnly="true">
            <f:validator binding="#{row.bindings.Id.validator}"/>
        </af:inputText>
    </af:column>
```

Figura 17- Código XML da interface gráfica

Na próxima ilustração, figura 17, está presente o último departamento do processo produtivo representado na aplicação, o departamento que se destina ao registo de saídas de produtos acabados. Isto é, esta interface gráfica permite expedir determinadas quantidades de produtos a determinados clientes com o preço que foi praticado nessa venda. Ao contrário dos ecrãs que continham os restantes departamentos, que se dividiam em três seções, o ecrã das expedições apenas se divide em duas áreas. Uma primeira área para o registo rápido de saídas de produtos acabados e uma segunda área que possui a tabela de expedições, que permite o acesso mais completo a todos os registos para os atualizar, eliminar ou para adicionar novos.

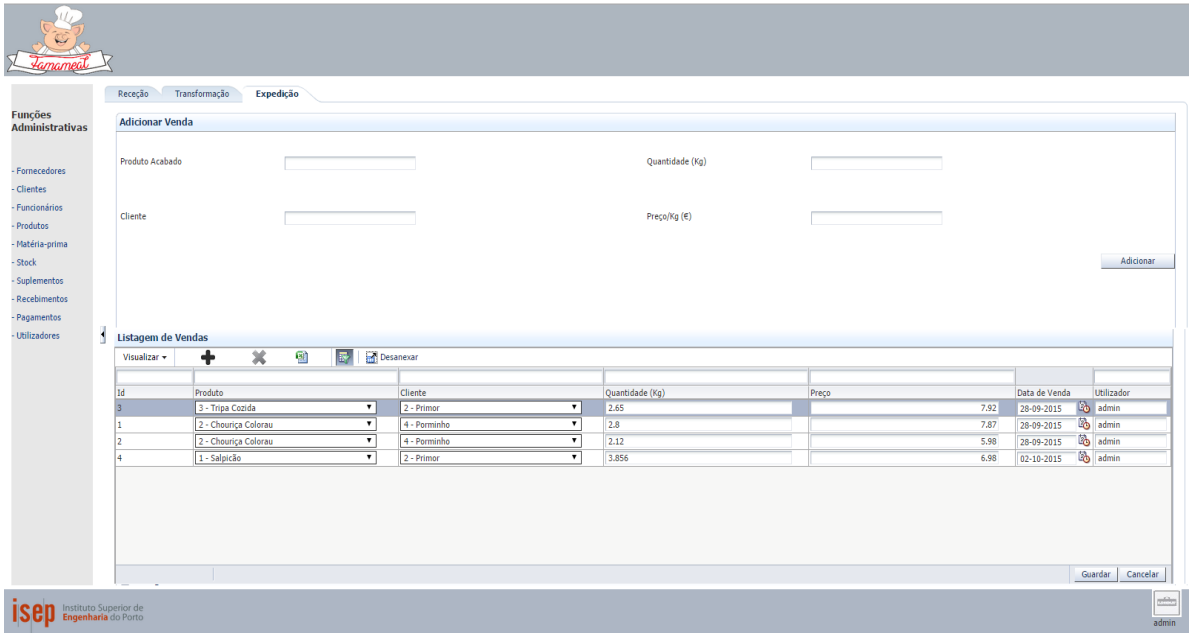


Figura 18- Área de Expedições do processo produtivo

Em todas as tabelas presentes na aplicação e em todos os painéis de adição rápida de registos existe um método chamado no final destas ações de forma a guardar as alterações na informação da base de dados. De seguida, é possível verificar esse método e uma explicação mais extensiva do mesmo.

```
public String guardar_action() {
    BindingContainer bindings = BindingContext.getCurrent().getCurrentBindingsEntry();
    OperationBinding operationBinding = bindings.getOperationBinding("Commit");
    Object result = operationBinding.execute();
    if (!operationBinding.getErrors().isEmpty()) {
        return null;
    }
    return null;
}
```

Figura 19- Método em Java para guardar alterações

Anteriormente à construção do método para gravar as alterações, presente na figura 18, foi necessário criar uma ação de *commit*. Esta ação, simplesmente irá tornar as alterações

presentes na tabela como permanentes mas, anteriormente, irá validar a informação contra as regras da base de dados e mostrar erros ao utilizador caso seja esse o cenário, caso contrário a operação ocorre com sucesso.

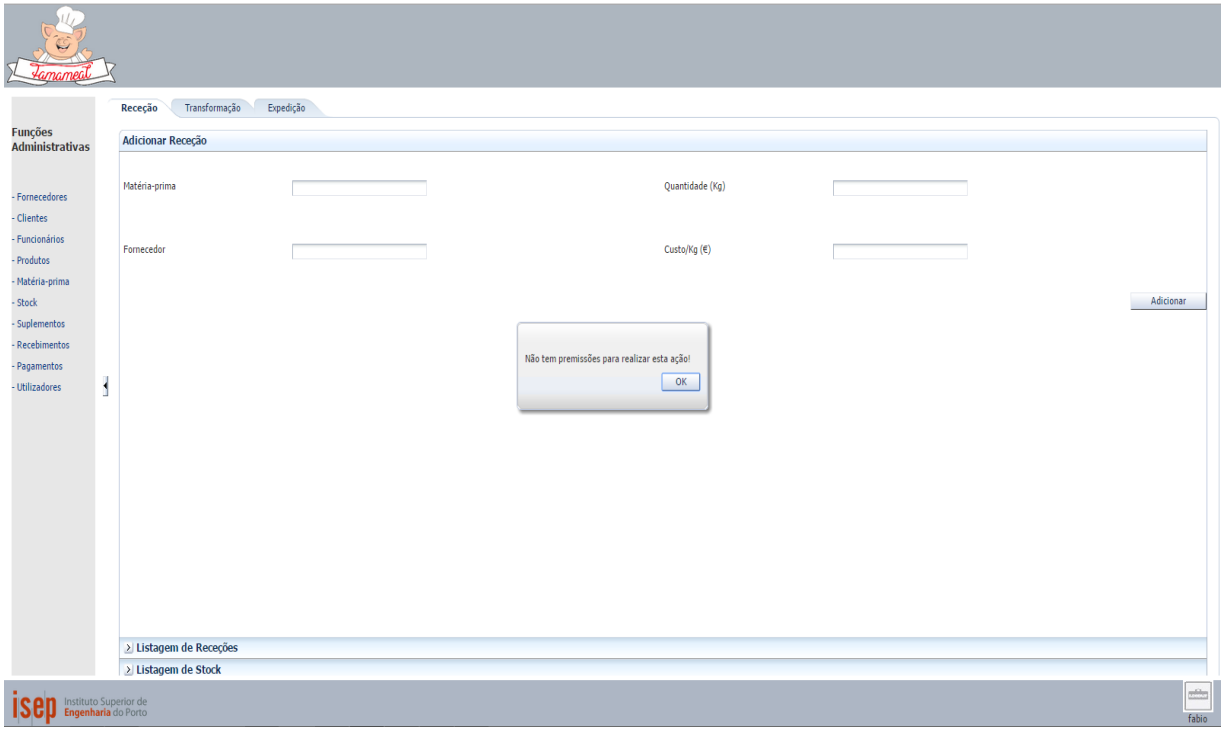


Figura 20- Impedimento de utilização das funções administrativas

Quando se entra no sistema de informação com um utilizador com o nível de permissões baixo, ou seja que não permite o acesso a toda a aplicação, e se tenta aceder à seção das funções administrativas irá surgir um erro para avisar o utilizador de que não tem permissões para realizar essa ação, como é possível confirmar na figura 19.

```
public String GO_CLIENTES() {
    if(User_premissions.equals("1")){

    FacesContext fctx = FacesContext.getCurrentInstance();
    Application application = fctx.getApplication();
    NavigationHandler navHandler = application.getNavigationHandler();
    navHandler.handleNavigation(fctx,null, "go_clientes");
    return null;

    } else {
                        ot11.setValue("Não tem premissões para realizar esta ação!");
                        RichPopup.PopupHints ph = new RichPopup.PopupHints();
                        p1.show(ph);

      }
    return null;
}
```

Figura 21- Método em Java para a navegação entre páginas da aplicação

A aplicação possui vários fluxos que permitem navegar entre as diversas interfaces, para tal é necessário ligar as diferentes páginas através de uma ação de navegação. Deste modo, apenas se executa esta ação e a aplicação irá reencaminhar o utilizador para a página desejada consoante o fluxo escolhido. A figura 20 contém um método da classe *Main* que possibilita a navegação dá página principal para a página dos clientes, através do acionamento do botão "Clientes" na seção de funções administrativas.

De seguida serão mostradas e explicadas as interfaces gráficas pertencentes à área de funções administrativas. Devido à grande quantidade de ecrãs existentes nesta seção, foram escolhidos apenas alguns ecrãs, que contemplam todas as funcionalidades. Assumindo que todas as interfaces possuem a ação de retroceder para a página anterior, através de um botão existente no canto inferior esquerdo das interfaces gráficas.

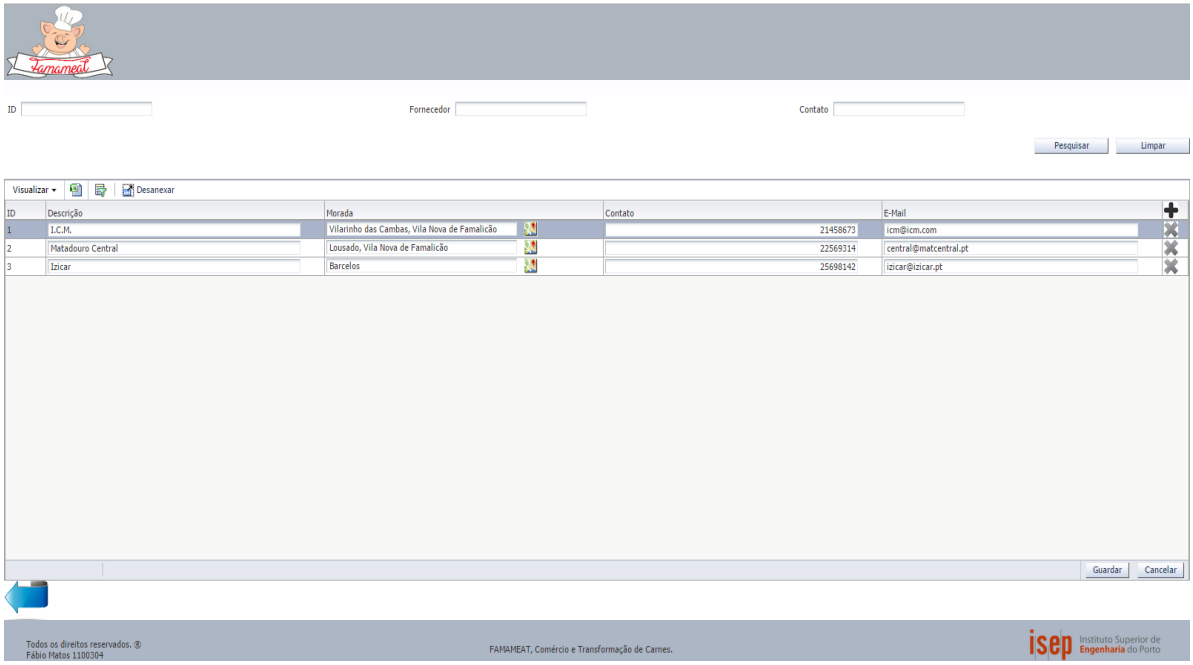


Figura 22- Página com informação dos Fornecedores

Na imagem da figura 21, encontra-se um *print screen* do ecrã dos fornecedores, que permite uma pesquisa alargada e extensiva sobre todos os atributos e todas as outras funções presentes nas tabelas de informação explicadas anteriormente. Somente neste ecrã existirá a possibilidade de adicionar fornecedores e de atualizar a informação destes, de modo evitar informação corrompida e para garantir a integridade dos dados. Através da figura 21, pode-ses verificar que na coluna da tabela que se visualiza as moradas dos fornecedores contém um botão à frente, que aquando do seu acionamento, guia o utilizador para uma página do Google *Maps* já com a pesquisa da morada do respetivo fornecedor efetuada, de modo a facilitar a localização dos fornecedores em relação à localização da empresa.

Todas estas funcionalidades presentes na página dos fornecedores também existem na página dos clientes exatamente com os mesmos efeitos e objetivos mas de forma a evitar repetições, esta não será explicada visto que o fundamental já foi concluído na página dos fornecedores, evitando a necessidade de descrever as mesmas funcionalidades para a interface dos clientes.

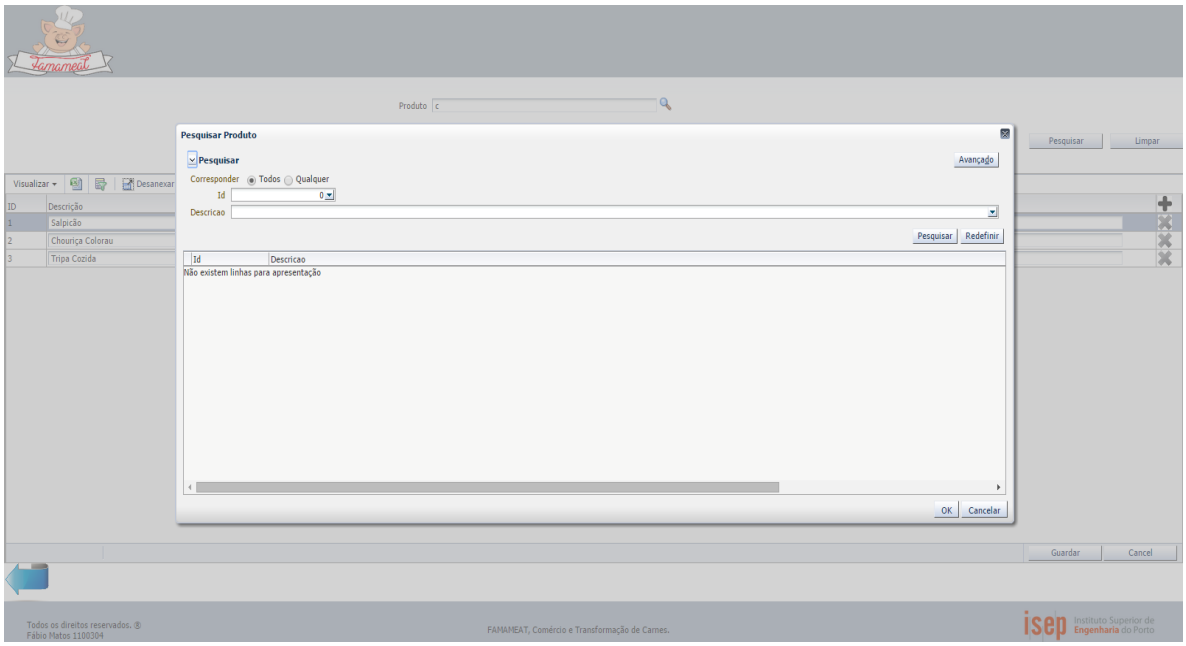


Figura 23- Pesquisa na página dos fornecedores

A ilustração anterior é referente à página dos produtos, acessível através das funções administrativas. Tal como esta interface, existe a interface das matérias-primas que segue o mesmo modelo e engloba as mesmas funcionalidades e componentes. Na figura 22, ecrã dos produtos, é possível verificar uma janela para realizar consultas mais extensivas sobre a tabela dos produtos. Esta janela permite pesquisar com base nos diversos atributos da tabela, e com várias opções do tipo: contém, não contém, diferente, igual, começa com, acaba com, etc. Esta funcionalidade torna-se bastante útil neste tipo de interfaces visto que, aquando da implementação do sistema de informação, irá conter uma enorme quantidade de produtos e este componente facilita a escolha do produto desejado.

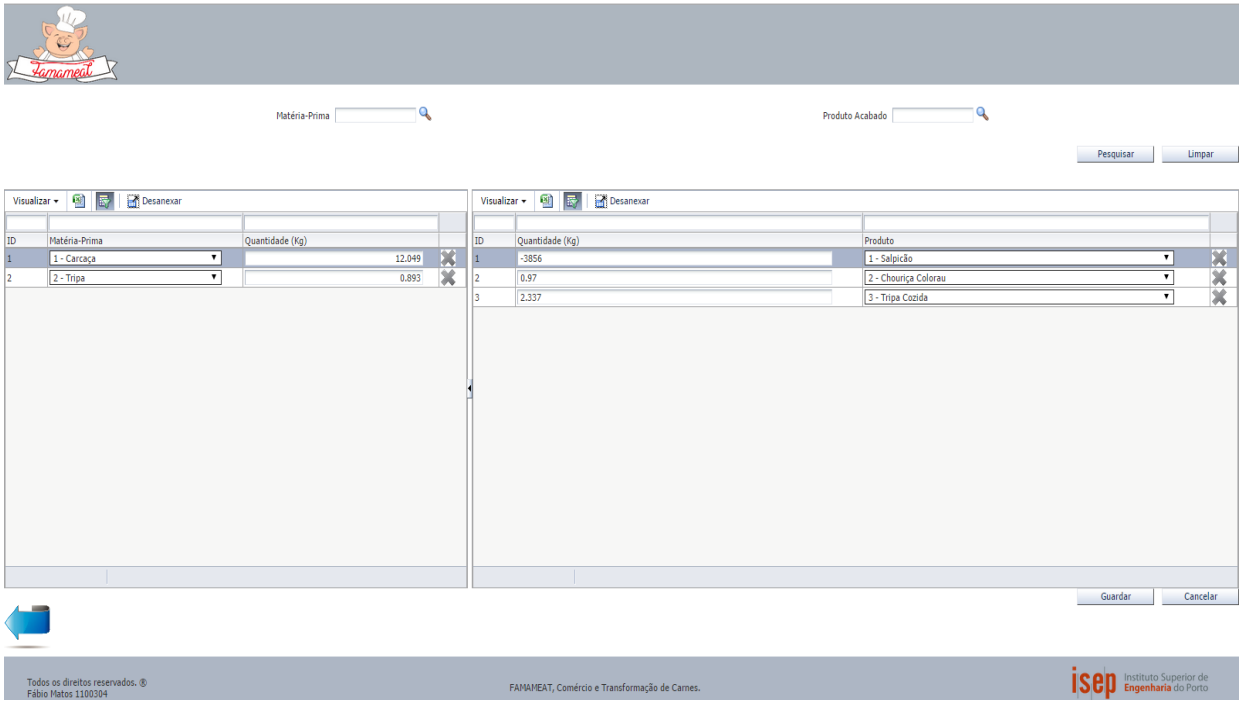


Figura 24- Página com informação de *stocks*

A figura 23 contém a interface através da qual será possível consultar e atualizar os dados dos *stocks*, quer de *stocks* de produtos acabados quer de matérias-primas. Permite, também, a eliminação de registos de *stocks* caso este produto ou matéria-prima não exista ou por outra qualquer razão que justifique esta ação. Visto que este tipo de ações poderá corromper a informação da aplicação ou torná-la incoerente, só utilizadores com permissões poderão aceder às páginas de dados mestres e transacionais.

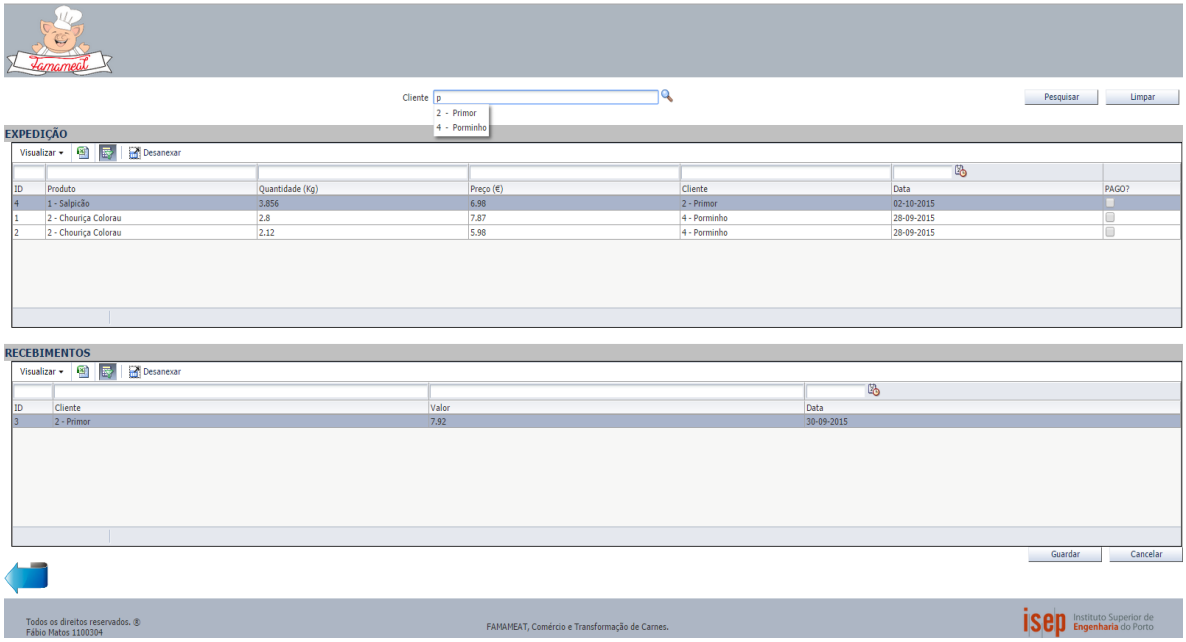


Figura 25- Página onde se regista recebimentos

A figura 24 representa a interface de recebimentos das expedições de produtos acabados, isto é, local onde é possível verificar todas as expedições para clientes e definir estas como já recebidas pela empresa. Este ecrã divide-se em duas áreas, na primeira existe a tabela das expedições com uma coluna extra que possibilita a marcação das expedições como recebidas e, na segunda área, existe a tabela de recebimentos onde é adicionado automaticamente as expedições marcadas como recebidas. Existe uma interface bastante idêntica à ilustrada na figura 24 mas que se destina aos pagamentos, ou seja, permite marcar as entradas de matérias-primas como pagas aos fornecedores. Estas páginas para além de controlarem os pagamentos de entradas e saídas da empresa também permite pesquisar sobre qualquer uma das tabelas presentes no ecrã. Na imagem anterior, também é possível verificar como funciona o componente que permite pesquisar expedições e recebimentos por cliente, para além permitir uma pesquisa avançada sobre todos os atributos (como é possível verificar na figura 22), também indica sugestões para a pesquisa, isto é, à medida que se vai preenchendo o campo de pesquisa vão sendo sugeridas as hipóteses que coincidem com o que foi escrito até então, através de uma *dropdown* como se verifica no topo da figura 24.

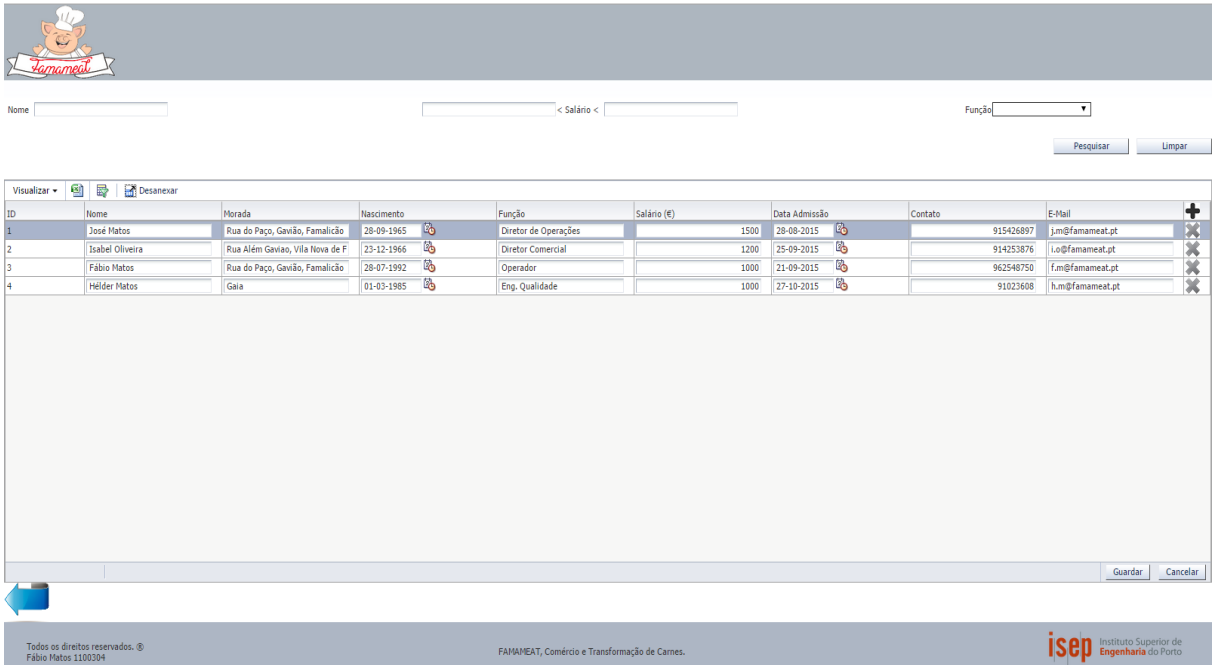


Figura 26- Página com informação dos funcionários

Na figura 25 está refletida a interface dos funcionários que contém todas as funcionalidades descritas até então. Na área de pesquisa desta página é permitido pesquisar por nome, função ou intervalo de salário e na tabela dos dados, para além de permitir filtrar qualquer tipo de coluna também permite a atualização da informação dos colaboradores da empresa.

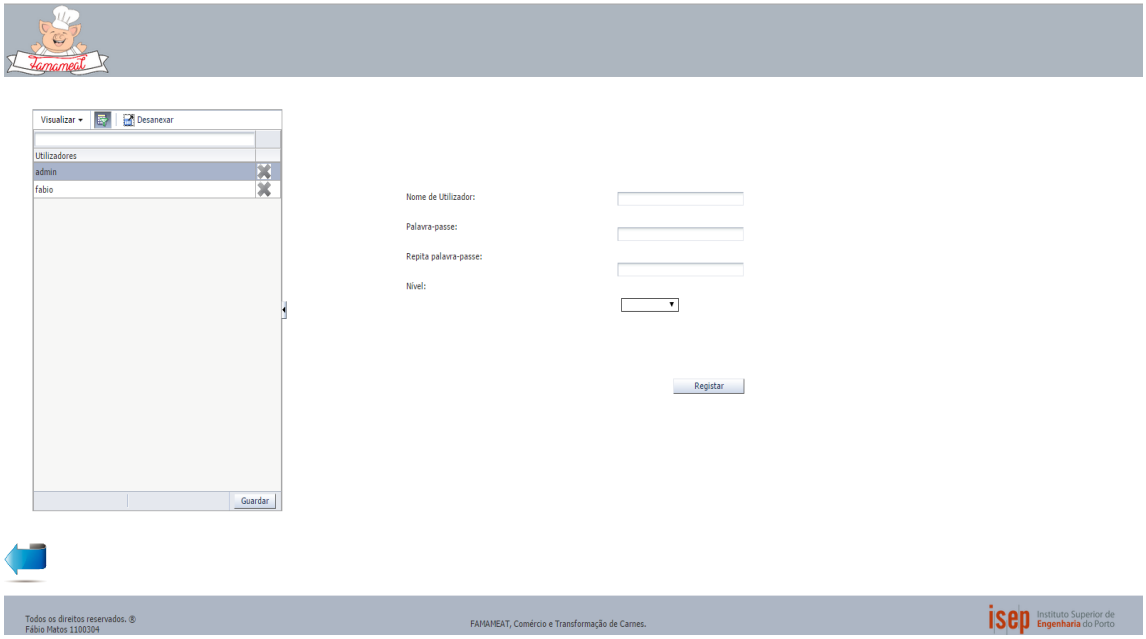


Figura 27- Página com informação dos utilizadores

Na última figura deste capítulo (figura 26), encontra-se o ecrã de gestão de segurança e de acessos que também só é acessível por utilizadores que possuem as devidas permissões. Nesta interface é possível gerir quem pode aceder à aplicação e a quais seções da mesma. Isto é, permite consultar o utilizadores existentes, permite elimina-los ou adicionar novos.

4.3. TESTES

Como o projeto consiste, maioritariamente, em programação foram sendo realizados testes frequentemente para verificar que acontecia o esperado e suposto pela codificação efetuada até então. Como o processo de desenvolvimento do *software* é iterativo, cada parte realizada necessitava de verificação, caso contrário não era possível avançar no desenvolvimento do *software*. É necessário realizar testes em pormenores para ter a certeza de que o que era realizado funcionava, estes testes baseiam-se na verificação do bom

funcionamento do que foi realizado, como por exemplo, se os cálculos que o programa deve efetuar estão corretos, se o que se pretendia está a acontecer ou não. Em baixo, estão representados alguns *print screen* e determinada descrição dos testes de partes mais relevantes.

Ao longo do desenvolvimento da aplicação foram feitas várias validações:

1. *Black Box Tests*;
2. CIT (*Component Integration Testing*;
3. SIT (*System Integration Testing*);
4. UAT (*User Acceptance Testing*).

4.2.1- Black Box Tests

Por *Black Box Tests* entendem-se os testes realizados ao formato dos dados, pesquisas inválidas e funcionalidades básicas como as ações dos botões. Muitos destes possíveis erros são evitados com componentes nas páginas como *droplists* (figura 27) e outros componentes de pesquisa, componentes que permitem uma pesquisa avançada sobre os registos que se pretende escolher, como é possível verificar na figura 28.

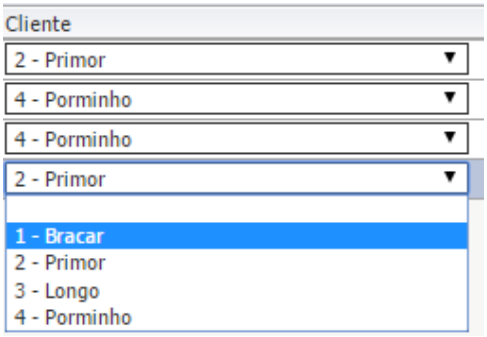


Figura 28- Componente *droplist* da aplicação

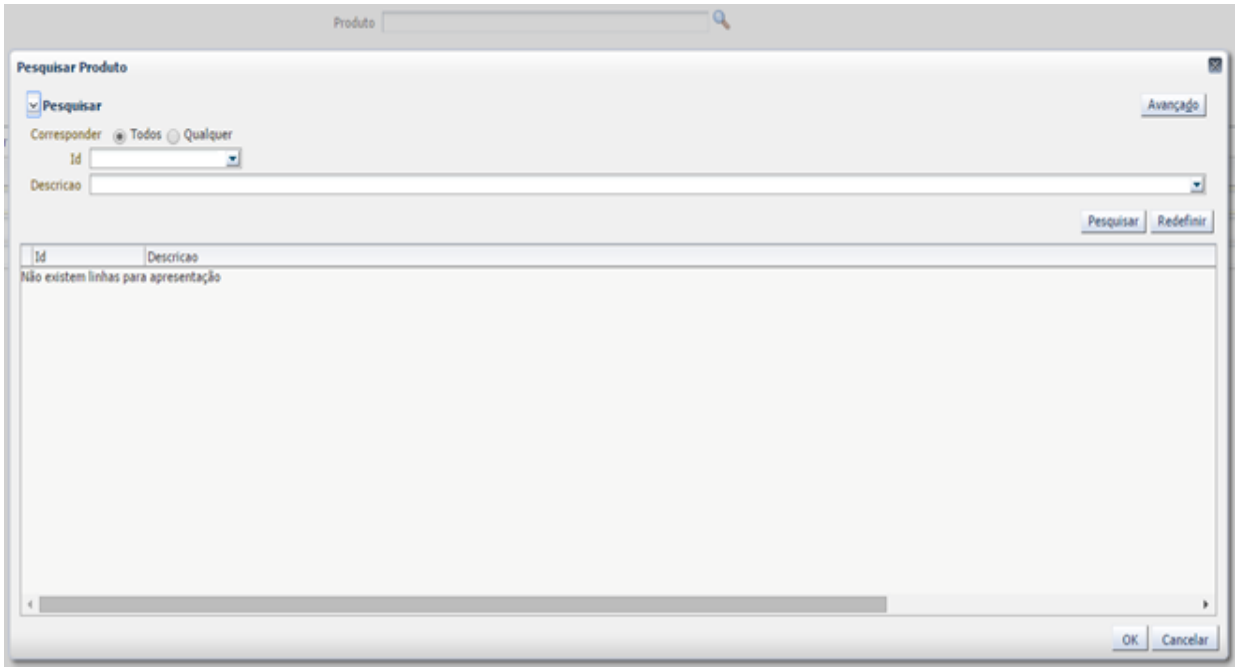


Figura 29- Componente de pesquisa da aplicação

De seguida, na figura 29, encontra-se uma montagem onde é possível verificar vários erros como formato e tipo de dados. Como por exemplo a tentativa de inserção de valores com letras em campos destinados a datas, de letras em campos numéricos, assim como a validação de formatos dos valores introduzidos.

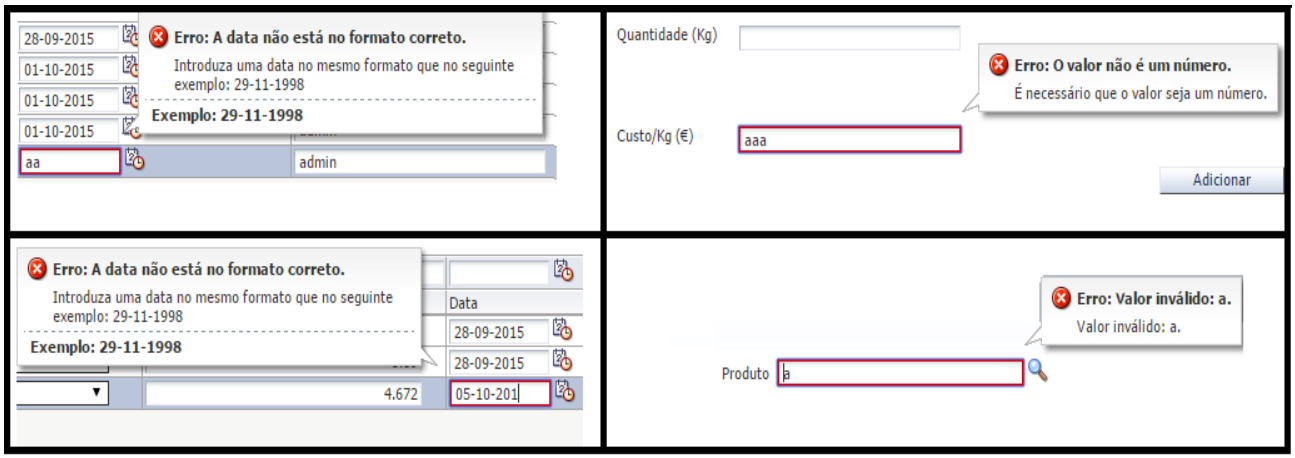


Figura 30- Alguns tipos de erros tratados pela aplicação

4.2.2- CIT (Component Integration Testing)

Os testes efetuados nesta área consistem na confirmação das mudanças da informação na base de dados após se realizar ações na aplicação, como por exemplo a verificação das inserções, atualizações e eliminações, efetuadas nas páginas, na base de dados. Os testes realizados serão descritos através das tabelas que se encontram de seguida.

Tabela 25- Confirmação da interação da aplicação com BD

Ações na aplicação	Verificação do respetivo funcionamento
Adição de Entradas/Transformações e Expedições.	Consulta das respetivas tabelas da base de dados para verificar se os registos eram inseridos com sucesso.
Edição dos dados de Entradas/Transformações e Expedições.	Consulta das respetivas tabelas da base de dados para confirmar se a alterações se verificavam.
Eliminar dados de Entradas/Transformações e	Verificar nas respetivas tabelas da base de dados

Expedições.	se os registos tinham sido eliminados com sucesso.
Adição de dados mestres	Consulta das respetivas tabelas da base de dados para verificar se os registos de dados mestres eram inseridos com sucesso.
Edição de dados mestres	Consulta das respetivas tabelas da base de dados para confirmar se a alterações aos dados mestres se verificavam.
Eliminar dados mestres	Verificar nas respetivas tabelas da base de dados se a informação era eliminada com sucesso.
Adição de Pagamentos/Recebimentos	Verificação se um registo era inserido na tabela da BD dos Pagamentos/Recebimentos com a informação da Entrada/Expedição, respetivamente.
Eliminar Pagamentos/Recebimentos	Verificar nas respetivas tabelas da base de dados se os registos tinham sido eliminados com sucesso.

4.2.3- SIT (System Integration Testing)

System Integration Testing consiste na validação dos fluxos de informação, isto é, se o modelo de dados está a refletir os fluxos de dados, como por exemplo o aumento do *stock* de matérias-primas após a adição de uma entrada. Na Tabela 26 pode-se verificar alguns fluxos da aplicação e quais os testes efetuados para validar o seu funcionamento.

Tabela 26- Verificação do funcionamento dos fluxos de informação

Fluxo de informação	Verificação do respetivo funcionamento
Recalcular *stocks* de matéria-prima, após a inserção de novas entradas no processo produtivo.	Após a adição de uma entrada de matéria-prima, foi necessário verificar se a quantidade desta estava refletida na tabela dos *stocks* de matéria-prima, de modo a validar o correto funcionamento deste fluxo.
Recalcular *stocks* de matéria-prima, após a edição de registos da tabela das entradas no processo produtivo.	Aquando da edição de um registo da tabela das entradas, caso este interfira com a quantidade de matéria-prima, a tabela de *stocks* deve sofrer as alterações de acordo com as efetuadas ao registo de entrada de matéria-prima.
Recalcular *stocks* de matéria-prima, após a eliminação de entradas no processo produtivo.	Após a eliminação de entrada de matéria-prima, foi necessário verificar se essa quantidade era retirada do registo dessa matéria-prima na tabela de *stocks*, validando o correto funcionamento deste fluxo.
Recalcular *stocks* de matéria-prima e de produtos acabados, após o registo de novas transformações de matérias-primas no processo produtivo.	Quando se regista uma nova transformação, era necessário verificar se a tabela de *stocks* de matéria-prima e a tabela de stocks de produto assumiam as quantidades corretas, dado que o *stock* de matéria-prima deveria diminuir consoante a quantidade de matéria-prima a ser transformada e o *stock* de produtos deveria aumentar de acordo com a quantidade resultante do processo de transformação da matéria-prima em produto acabado.
Recalcular *stocks* de matéria-prima e de produtos acabados, após a eliminação de transformações no processo produtivo.	Quando se eliminava um registo da tabela das transformações, tinha que se verificar a tabela de *stocks* de matéria-prima e a tabela de stocks de produto para verificar se estas assumiam as

	quantidades corretas, dado que o *stock* de matéria-prima deveria aumentar de acordo com a quantidade do registo eliminado e o *stock* de produtos deveria subtrair a quantidade resultante de produto acabado que estava contemplada no registo eliminado.
Recalcular *stocks* de produtos acabados, após a inserção de novas expedições no processo produtivo.	Após a adição de uma expedição de produtos, era necessário verificar se a quantidade expedida era retirada da tabela dos *stocks* de produtos acabados, para verificar o correto funcionamento deste fluxo.
Recalcular *stocks* de matéria-prima, após a edição de Expedições no processo produtivo.	Aquando da edição de um registo da tabela de expedições, caso este interfira com a quantidade de produto expedido, a tabela de stocks deve sofrer as alterações de acordo com as efetuadas ao registo de expedição.
Recalcular *stocks* de matéria-prima, após a eliminação de expedições no processo produtivo.	Após a eliminação de uma expedição de produtos, foi necessário verificar se essa quantidade era novamente inserida na tabela de stocks de produtos acabados, validando o correto funcionamento deste fluxo.

4.2.4- UAT (User Acceptance Testing)

Por último foram realizados testes de modo a verificar se as regras de negócio eram cumpridas e respeitadas durante os fluxos da aplicação. Como é possível verificar na seguinte ilustração, a aplicação impede que o utilizador registe uma transformação de matéria-prima em maior quantidade do que a existente em *stock*. O mesmo comportamento é possível ser verificado aquando da tentativa de expedição de uma quantidade de produto

superior à registada na tabela de *stocks* da base de dados, figura 30. Estes erros surgem para que não exista a possibilidade de corromper os dados do sistema por parte do utilizador.

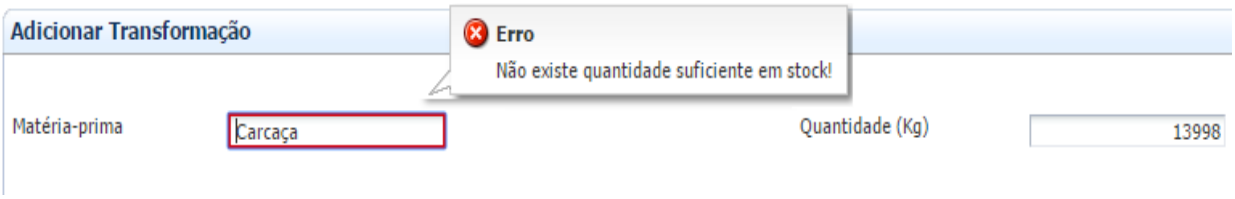


Figura 31- Validação de regras de negócio

As regras de negócio apuradas, anteriormente ao desenvolvimento da aplicação, exigem que alguns atributos sejam de preenchimento obrigatório, como por exemplo a quantidade de matéria-prima na adição de um registo de entradas. No caso destes acontecimentos, um erro semelhante ao da figura 31 aparecerá impedindo o utilizador de continuar a operação até corrigir os dados. Visto que se estes casos fossem permitidos iria impossibilitar tratamento dos dados por parte da aplicação, como cálculo de *stocks*, que por sua vez iria influenciar o bom funcionamento da aplicação e coerência da aplicação.

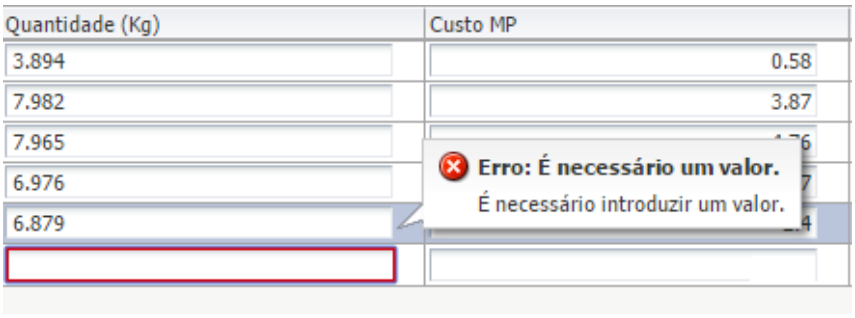


Figura 32- Validação de valores obrigatórios

5. CONCLUSÕES

Neste capítulo são referidas as conclusões obtidas após o término deste projeto. São apresentados os objetivos atingidos e limitações encontradas. São ainda descritas, brevemente, as funcionalidades implementadas. Por fim, é apresentada uma apreciação pessoal em relação ao desenrolar deste projeto.

5.1. RESUMO

O projeto realizado tinha como objetivos suprimir a perda de tempo na visualização, registo e edição de dados e otimizar a gestão e controlo do processo produtivo da e uma empresa no setor da transformação alimentar.

A estrutura do sistema foi construída de raiz, pelo que a empresa não continha qualquer tipo de base para facilitar o trabalho realizado. O relatório aqui apresentado representa todos os passos da realização do sistema, as dificuldades e quais as vantagens e melhorias que se pretende obter da aplicação.

Desta forma, todas as opções foram cobertas e o sistema foi preparado de forma a, no futuro, caso necessário, seja alterado ou otimizado. Sendo que é esperado que a aplicação sofra mais algumas alterações antes da sua implementação, como é descrito em diante.

5.2. OBJETIVOS REALIZADOS

Todos os objetivos que foram propostos, com a construção do sistema de informação, encontram-se refletidos nas funcionalidades que o SI contém. Ou seja, todos os objetivos foram atingidos com sucesso e, dado isto, algumas melhorias e funcionalidades foram acrescentadas, de modo a acrescentar valor o produto. A seguir pode-se verificar quais os objetivos estipulados e de que forma estes foram atingidos e em que funcionalidade estão presentes.

- Levantamento e análise de requisitos:

  Foi apurado o modelo de funcionamento da aplicação para que esta contenha vantagens para a empresa, para a qual se realizou a análise de requisitos.

- Construção de uma base de dados capaz de suportar o funcionamento do *software*:

  Após ser estruturado o modelo de dados, e este ser refletido na base de dados, foi possível verificar que o sistema de informação era suportado pela BD e que esta

concretizava todas as funcionalidades esperadas e todas as operações realizadas na interface do SI.

- Desenvolvimento de um sistema de Login, capaz de gerir as sessões de cada utilizador:

A aplicação permite o registo de utilizadores com diferentes prioridades e também trata sessões de modo a que seja impossível corromper a aplicação ou aceder sem autorização.

- Processo de introdução e edição de informação, nomeadamente o registo de entradas, transformações e saídas:

No fluxo principal da aplicação é possível verificar que permite adicionar entradas e transformações de matéria-prima, assim com a expedição de produtos (tratando também dos respetivos stocks). A informação presente nesta secção da aplicação permite a sua atualização a qualquer momento e a consulta de *stocks* também, para aumentar a produtividade dos colaboradores que utilizarão o SI.

- Secção com dados mestres da empresa e a possibilidade de sua inserção, atualização e/ou remoção:

Consiste numa seção da aplicação, onde apenas os utilizadores com os privilégios máximos poderão aceder para consultar, adicionar, editar ou eliminar os dados mestres da companhia, tais como: fornecedores, clientes, funcionários, produtos, matérias-primas, pagamentos, recebimentos, *stocks* e utilizadores.

- Incorporação de validações em todos os processos que possam ser utilizados pelos utilizadores, de modo a evitar a existência de dados incoerentes ou duplicados:

Todos os campos presentes nas interfaces que podem ser editados pelo utilizador possuem validações de modo a evitar que o utilizador introduza dados inválidos no

formato errado. Assim, como as operações realizadas pelo próprio SI são validadas internamente para evitar interrupções abruptas que podem resultar em erros de dados ou da própria aplicação.

- Construção do SI utilizando a tecnologia ADF da Oracle:

Como foi possível verificar ao longo deste documento, a construção da aplicação e principalmente das interfaces da mesma foi através da *framework* Oracle ADF.

Desta forma, pode-se concluir que a construção desta aplicação ocorreu com sucesso visto que os objetivos foram cumpridos, a aplicação contém as funcionalidades pré-definidas e o seu funcionamento é eficiente e com uma performance alta.

5.3. CONTRIBUIÇÕES DO PROJETO

Este projeto apresenta várias vantagens para o aluno porque este melhorou as suas habilidades técnicas no que toca ao desenvolvimento aplicacional, principalmente em relação à tecnologia utilizada (ADF). Possibilitou que o aluno melhorasse as suas capacidades na análise de requisitos e de regras de negócio, no desenho e na construção de base de dados, na codificação de lógica de negócio e no processo geral de desenvolvimento web, visto que o projeto realizado abrange várias áreas da construção de soluções informáticas. Para além de todas as contribuições que este projeto trouxe ao autor do documento, é esperado que o mesmo aconteça com a empresa na qual será implementado. A companhia poderá usufruir de um sistema integrado capaz de gerir e controlar todo o processo produtivo da empresa, reduzindo custos e desperdícios, aumentando produtividade e eficiência. O *software* desenvolvido permite consultar, registar e atualizar informação corporativa da empresa, como fornecedores, clientes, produtos e matérias-primas que irá permitir uma centralização da informação, logo, esta informação será mais

fidedigna e de mais fácil acesso. Irá liberta mais tempo dos colaboradores visto que esta aplicação irá realizar muitas tarefas que, neste momento, são realizadas por colaboradores e a fiabilidade do resultado destas tarefas será maior. Ou seja, mais qualidade na informação utilizada pela companhia e os colaboradores despenderão de mais tempo para realizar outras tarefas.

5.4. LIMITAÇÕES & TRABALHO FUTURO

Ao longo do projeto foram encontradas várias dificuldades relacionadas com os imprevistos inerentes à programação de *software*, mas a maior limitação foi o tempo. Pois o autor, dado ao seu emprego, dispunha de pouco tempo para se dedicar à realização do projeto, só tendo podido libertar tempo para a sua elaboração a partir de Agosto. O pouco tempo que sobrou para a composição do projeto de tese de mestrado também dificultou as tarefas mas foi possível a conclusão das mesmas devido ao período de férias usufruído durante Outubro.

O projeto realizado já tem definido quais serão os próximos módulos a integrar na aplicação, visto que o projeto irá sofrer alterações e melhorias, como as descritas em baixo, para que possa ser implementado na empresa referida e trazer as funcionalidades e vantagens pretendidas.

- Construção de uma área financeira que irá possuir mais dados desta área e mais concretos, onde será possível faturar as vendas realizadas no SI (através de uma ligação ao software de faturação que a empresa atualmente possui);

- A área de finanças referida irá usufruir de funções como diagramas, mapas e balanços das contas da empresa ou de uma porção dessas contas, como por exemplo balanços semanais e mensais ou mapas de tesouraria;

- Será introduzida mais uma secção na aplicação que irá permitir realizar encomendas de matéria-prima ou pedir orçamentos aos fornecedores, através do envio de um *e-mail*.

O sistema foi feito de forma dinâmica e a pensar no futuro, pelo que encontra-se facilmente alterável. No futuro espera-se que a aplicação seja implementada na empresa para a qual esta foi desenvolvida já com os módulos referidos anteriormente implementados. Desta forma, é possível dar uma contínua manutenção e evolução ao projeto realizado para a Unidade Curricular de Tese/Dissertação, podendo este projeto ser aproveitado para o contexto real e sofrer atualizações de modo a ficar sempre vantajoso para quem o utiliza.

5.5. APRECIAÇÃO FINAL

Considera-se que o trabalho desenvolvido correu da forma esperada embora neste tipo de projetos existam sempre dificuldades e obstáculos, mas foi concluído com êxito, de forma positiva e atendendo os requisitos propostos. Contratempos como os enfrentados acontecem na maioria dos desenvolvimentos de qualquer projeto, principalmente quando envolve programação. Os problemas que surgiram na construção da aplicação tornaram-se em mais-valias, em maior capacidade de resolução de problemas e num maior conhecimento de causa, neste caso, em programação. Devido à pouca de experiência do autor e problemas inerentes à tecnologia, visto que esta ainda não se encontra num estado de maturação muito alto, algum tempo foi gasto com dificuldades porque o pretendido não funcionava ou pelo menos, da maneira pretendida. E quando este obstáculo é ultrapassado provavelmente irão existir outras dificuldades em breve, quer na elaboração quer no funcionamento ou talvez na estrutura que o projeto deve tomar, para que seja possível responder às necessidades propostas.

Todo o período de realização trouxe vantagens a nível profissional, como um maior domínio das tecnologias utilizadas, aumento de conhecimentos e experiência. A nível

pessoal, trouxe vantagens como o enriquecimento do sentido de responsabilidade, do sentido crítico e de conhecimentos que poderão ser úteis no futuro, nomeadamente conhecimento do funcionamento destes sistemas e da indústria para qual se destina.

O facto de a execução do projeto se basear, essencialmente, em programação e análise de requisitos apresentou dificuldades que mais tarde provavelmente irão originar vantagens acrescidas ao aluno, visto que está empregado nesta área. No entanto, tornou-se numa óptima oportunidade visto que, para além dos conhecimentos adquiridos no decorrer da mestrado, a sensibilidade para o apuramento de soluções de modo a cumprir os objectivos e o conhecimento de programação foram reforçados. A unidade curricular de Tese/Dissertação foi, resumidamente, uma experiência bastante gratificante devido ao conhecimento adquirido e ao facto de poder realizar um projeto que não irá ficar parado mas sim em constante evolução e adaptação.

Referências Documentais

[1] http://study.com/academy/lesson/what-are-information-systems-definition-types-quiz.html, consultado em Outubro de 2015.

[2] Wallace, P. (2014). Introduction to information systems (Second ed.). New Jersey: Prentice Hall.

[3] http://searchmanufacturingerp.techtarget.com/essentialguide/Whats-the-state-of-the-art-in-ERP-trends, consultado em Agosto de 2015.

[4] http://russellobrien.hubpages.com/hub/ERP-Explained, consultado em Setembro de 2015.

[5] http://www.informazione4.com.br/cms/opencms/desafio21/artigos/gestao/organizando/0016.html, consultado em Setembro de 2015.

[6] http://techterms.com/definition/web_development, consultado em Setembro de 2015.

[7] http://www.okoone.com/blog/26-today-s-web-development-technologies-explained-briefly, consultado em Setembro de 2015.

[8] Freeman, A., & Sanderson, S. (2011). Pro Asp.net MVC 3 Framework (3rd ed.). New York: Apress.

[9] https://developer.chrome.com/apps/app_frameworks, consultado em Outubro de 2015.

[10] https://adffromhell.wordpress.com/2014/02/24/introducao-ao-fusion-e-adf-notas/, consultado em Setembro de 2015.

[11] http://www.oracle.com/technetwork/developer-tools/adf/learnmore/angels-in-the-architecture-v1-00-2252920.pdf, consultado em Setembro de 2015.

[12] http://docs.oracle.com/middleware/1212/adf/ADFCG/intro.htm#ADFCG327, consultado em Setembro de 2015.

[13] Purushothaman, J. (2012). *Oracle ADF Real World Developer's Guide*. Birmingham: Packt Publishing

[14] http://www.oracle.com/technetwork/developer-tools/adf/adf-11-overview-1-129504.pdf, consultado em Agosto de 2015.

[15] http://publish.uwo.ca/~kgroling/papers/Autonomic_Database_Management.pdf, consultado em Setembro de 2015.

[16] http://www.omg.org/news/meetings/workshops/MDA-SOA-WS_Manual/00-T4_Matthews.pdf, consultado em Outubro de 2015.

[17] https://docs.kde.org/trunk4/en/kdesdk/umbrello/uml-elements.html, consultado em Outubro de 2015.

[18] http://databases.about.com/od/sql/a/sql.htm, consultado em Outubro de 2015.

[19] http://java.about.com/od/gettingstarted/a/whatisjava.htm, consultado em Setembro de 2015.

[20] http://www.oracle.com/webfolder/technetwork/tutorials/obe/java/gc01/index.html, consultado em Outubro de 2015.

[21] https://msdn.microsoft.com/en-us/library/aa468558.aspx, consultado em Outubro de 2015.

[22] https://docs.oracle.com/javaee/7/firstcup/title.htm, consultado em Outubro de 2015.

[23] http://docs.oracle.com/javaee/6/tutorial/doc/bnaaw.html, consultado em Outubro de 2015.

[24] http://docs.oracle.com/javaee/6/tutorial/doc/bnacj.html, consultado em Setembro de 2015.

[25] http://docs.oracle.com/cd/E13222_01/wls/docs81/intro/chap1.html, consultado em Setembro de 2015.

[26] http://docs.oracle.com/cd/E13222_01/wls/docs81/intro/chap1.html#1065864, consultado em Outubro de 2015.

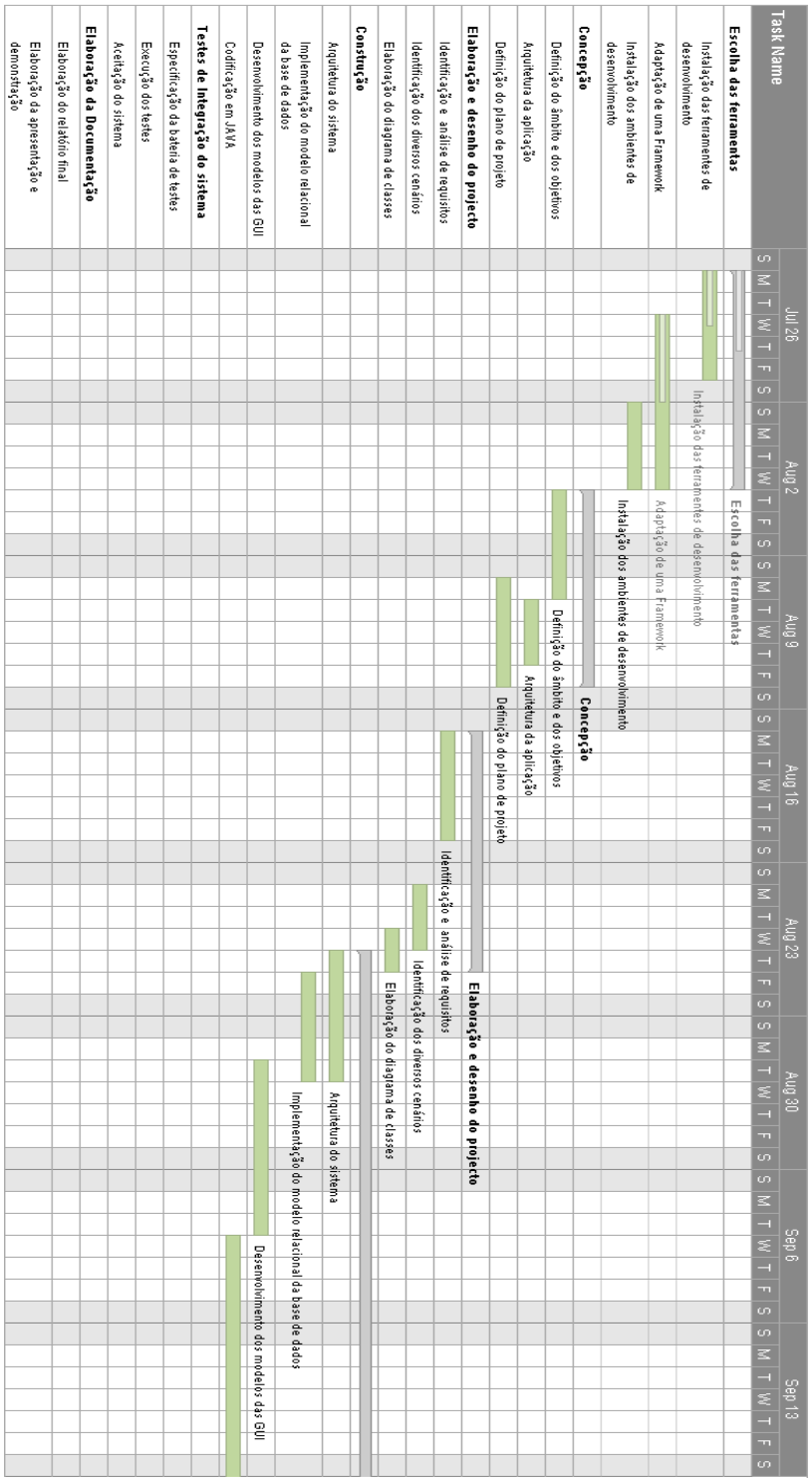
[27] http://docs.oracle.com/cd/E17904_01/web.1111/e13707/toc.htm, consultado em Setembro de 2015.

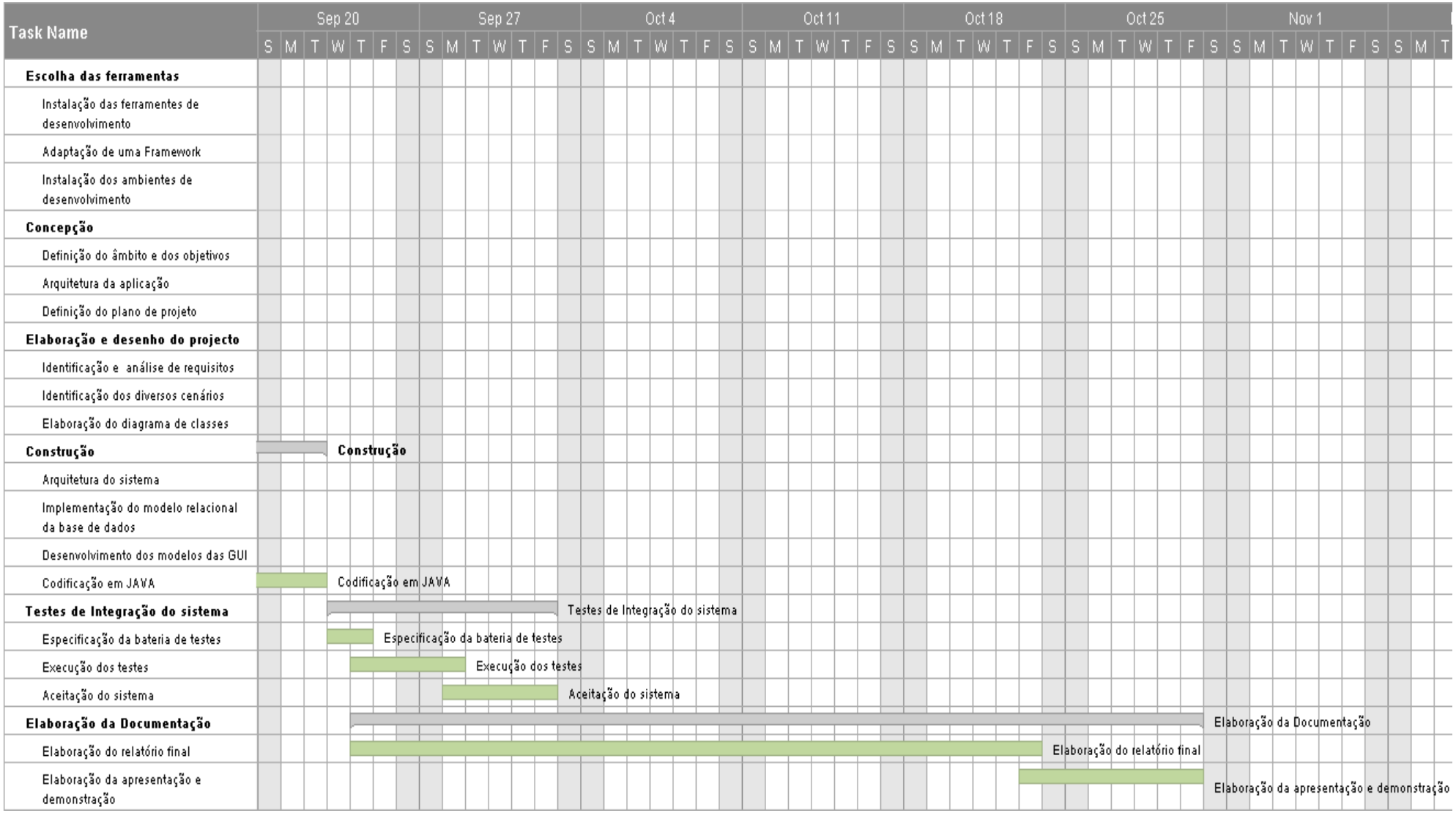
[28] http://www.oracle.com/technetwork/developer-tools/jdev/jdeveloper11g-datasheet-1-133040.pdf, consultado em Outubro de 2015.

[29] https://docs.oracle.com/cd/E17781_01/admin.112/e18585.pdf, consultado em Outubro de 2015.

Anexo A. Diagrama de Gantt

Neste primeiro anexo encontra-se o diagrama de Gantt, em tamanho maior, para facilitar a sua visualização e análise. Contudo, dado a impossibilidade de este ser representado em apenas uma figura, o diagrama encontra-se divido em duas partes para que seja legível ao leitor.




Task Name
Sep 20
Sep 27
Oct 4
Oct 11
Oct 18
Oct 25
Nov 1
S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T W T F S S M T
1 Escolha das ferramentas
2 Instalação das ferramentes de desenvolvimento
3 Adaptação de uma Framework
4 Instalação dos ambientes de desenvolvimento
5 Concepção
6 Definição do âmbito e dos objetivos
7 Arquitetura da aplicação
8 Definição do plano de projeto
9 Elaboração e desenho do projecto
10 Identificação e análise de requisitos
11 Identificação dos diversos cenários
12 Elaboração do diagrama de classes
13 Construção
Construção
14 Arquitetura do sistema
15 Implementação do modelo relacional da base de dados
16 Desenvolvimento dos modelos das GUI
17 Codificação em JAVA
Codificação em JAVA
18 Testes de Integração do sistema
Testes de Integração do sistema
19 Especificação da bateria de testes
Especificação da bateria de testes
20 Execução dos testes
Execução dos testes
21 Aceitação do sistema
Aceitação do sistema
22 Elaboração da Documentação
Elaboração da Documentação
23 Elaboração do relatório final
Elaboração do relatório final
24 Elaboração da apresentação e demonstração
Elaboração da apresentação e demonstração

Anexo B. Diagrama de Classes

No Anexo B é possível consultar o diagrama de classes na sua totalidade.

Funcionários		
Key	ID	Integer
	Nome	Varcchar2()
	Data_nascimento	Date
	Morada	Varcchar2()
	Salário	Long
	Data_admissão	Date
	Cargo	Varcchar2()
	Telemóvel	Integer
	Mail	Varcchar2()

Receções		
Key	ID	Integer
Foreign Key	Materia_prima	Integer
	Quantidade_mp	Long
	Custo_mp	Long
Foreign Key	Fornecedor	Integer
	Data_receção	Date
Foreign Key	Username	Varchar2()

Transformação		
Key	ID	Integer
Foreign Key	Materia_prima	Integer
	Quantidade_mp	Long
Foreign Key	Produto_acabado	Integer
	Quantidade_pa	Long
Foreign Key	Dept	Integer
	Data_transformação	Date
Foreign Key	Username	Varchar2()

Expedição		
Key	ID	Integer
Foreign Key	Produto_acabado	Integer
	Quantidade_pa	Long
	Preço	Long
Foreign Key	Cliente	Integer
	Data	Date
Foreign Key	Username	Varchar2()

Suplementos		
Key	ID	Integer
	Descricao	Varchar2()
Foreign Key	Departamento	Integer
Foreign Key	Produtos	Varchar2()

Fornecedores		
Key	ID	Integer
	Nome	Varchar2()
	Morada	Varchar2()
	Telefone	Integer
	Mail	Varchar2()

Clientes		
Key	ID	Integer
	Nome	Varchar2()
	Morada	Varchar2()
	Telefone	Integer
	Mail	Varchar2()

Materia_prima		
Key	ID	Integer
	Descricao	Varchar2()
Foreign Key	Fornecedor	Varchar2()

Produto_acabado		
Key	ID	Integer
	Descricao	Varchar2()
Foreign Key	Departamento	Integer
	Custo_medio_produção	Long
	Constituição	Varchar2()

Pagamentos		
Key	ID	Integer
Foreign Key	Fornecedor	Integer
	Valor	Long
	Data_pagamento	Date
Foreign Key	Username	Varchar2()

Departamentos		
Key	ID	Integer
	Descricao	Varchar2()

Utilizadores		
Key	Username	Varchar2()
	Password	Varchar2()
	Hierarquie_value	Integer

Recebimentos		
Key	ID	Integer
Foreign Key	Cliente	Integer
	Valor	Long
	Data_recebimento	Date
Foreign Key	Username	Varcgar2()

Stock_MP		
Key	ID	Integer
Foreign Key	Materia_prima	Integer
	Qtd	Long
Foreign Key	Departamento	Integer

Stock_PA		
Key	ID	Integer
Foreign Key	Produto_acabado	Integer
	Qtd	Long
Foreign Key	Departamento	Integer

Anexo C. Requisitos de infraestrutura

Neste anexo estão descritos todos os pressupostos, em termos de infraestrutura, que são necessários para funcionamento eficaz da aplicação desenvolvida e descrita ao longo deste documento.

1. OS: Windows 7 ou Oracle Linux 6;
2. RAM: 16GB;
3. CPU: 8 Cores;
4. Storage: 500Gb;
5. Oracle WebLogic Server 11g;
6. Oracle Application Development Runtime (11.1.1.7);
7. JDK 1.7 (64-bit);