MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA FARROUPILHA
***CAMPUS* SANTO ÂNGELO**

SISTEMA PARA CONTROLAR E VISUALIZAR VALIDADES

TRABALHO DE CONCLUSÃO DE CURSO

Mateus Morais de Oliveira

Santo Ângelo, RS, Brasil.
2021

SISTEMA PARA CONTROLAR E VISUALIZAR VALIDADES

por

Mateus Morais de Oliveira

Monografia apresentada ao Instituto Federal de Educação Ciência e Tecnologia Farroupilha, como requisito parcial para obtenção do título de Tecnólogo em Sistemas para Internet.

Orientador: Juliano Gomes Weber

Santo Ângelo, RS, Brasil

2021

Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal de Educação Ciência e Tecnologia Farroupilha

A Comissão Examinadora, abaixo assinada,
aprova a Monografia

SISTEMA PARA CONTROLAR E VISUALIZAR VALIDADES

elaborada por
Mateus Morais de Oliveira

como requisito parcial para obtenção do título de
Tecnólogo em Sistemas para Internet

COMISSÃO EXAMINADORA

_____
Juliano Gomes Weber, Me.

(Presidente/Orientador)

_____
Fábio Novaski, Me. (IFFAR)

_____
Fábio Weber Albiero , **Me.** (IFFAR)

Conceito Final: _____

Santo Ângelo,10 de Dezembro de 2021.

AGRADECIMENTOS

À Deus por proporcionar até aqui forças para continuar...

Ao meu orientador que não mediu esforços para ajudar a mim e a meus colegas, com sua imensa sabedoria, conhecimento e paciência nas diversas áreas para esse projeto, vindo contribuir para o ambiente acadêmico, profissional e pessoal.

A todos os outros professores das outras disciplinas, que também com paciência nos ajudaram a sempre ir em frente, fazendo nós não desistir pelo caminho.

Aos meus colegas, que estão comigo nesse desafio, por ter ajudado, compartilhando experiências ao longo do curso de Sistema para Internet, que veio acrescentar nessa caminhada.

LISTA DE TABELAS

LISTA DE ABREVIATURA E SIGLAS

XHTML - eXtensible Hypertext Markup Language

JSF - Java Server Faces

IDE - Integrated Development Environment

SGBD - Sistema Gerenciador de Banco de Dados

MVC - Model, View, Controller

PHP - Hypertext Preprocessor

UML - Unified Modeling Language

SQL - Structured Query Language

XML - Extensible Markup Language

EAN - European Article Number

FIFO - First in First out

PEPS - Primeiro que entra Primeiro que sai

PVPS - Primeiro que vence Primeiro que sai

LISTA DE FIGURAS

RESUMO

SISTEMA PARA CONTROLAR E VISUALIZAR VALIDADES

AUTOR(A): MATEUS MORAIS DE OLIVEIRA
ORIENTADOR: PROF. ME. JULIANO GOMES WEBER

Data e Local da Defesa: Santo Ângelo, ...de .....de 2021.

As perdas de produtos são um problema recorrente, seja por motivo de manuseio errado, furto, entre outros motivos. A perda de produtos por data validade vencida é um problema recorrente que existe desde que o formato de comércio Varejo e Atacado existe, pois neste momento, houve a necessidade natural de geração de Estoques. Devido a unidade do código de barras, que é a principal interface de controle dos produtos, independente da data de fabricação, consequentemente a validade, a automatização deste controle, se torna uma tarefa difícil, a prova é, a falta de ferramentas computacionais no mercado para este fim. A solução proposta por este trabalho é uma solução relativamente simples, porém abrange todo e qualquer estabelecimento comercial que possui estoque que necessite de controle de datas de vencimento. A ideia é original e estará presente em um caso de uso real, alterando pouco ou nada do fluxo administrativo já existente na empresa, pois apenas irá utilizar informações já existentes no fluxo de compras, gerando informações adicionais que permitirão gerar por sua vez, relatórios e alertas inteligentes de maneira automática. Todo o processo de modelagem, desenvolvimento e testes de software, utiliza softwares livres, bem como, o estudo de caso, se dará em um ambiente comercial real.

Palavras-chave: Estoque; Hipermercado; Fornecedor; produtos; Banco de dados; validades; sistema de informação

ABSTRACT

SYSTEM TO CONTROL AND VIEW VALIDITIES

Product losses are a recurring problem, whether due to mishandling, theft, among other reasons. The loss of products due to an expired date is a recurring problem that has existed since the Retail and Wholesale trade format exists, because at this moment, there was a natural need to generate Inventories. Due to the barcode unit, which is the main interface for controlling products, regardless of the date of manufacture, consequently the validity, the automation of this control, becomes a difficult task, the proof is, the lack of computational tools in the market to this end. The solution proposed by this work is a relatively simple solution, but it covers any and all commercial establishments that have stock that needs to control due dates. The idea is an original idea and will be present in a real use case, changing little or nothing of the administrative flow that already exists in the company, as it will only use information that already exists in the purchase flow, generating additional information that will allow to generate in turn, reports and intelligent alerts automatically. The entire software modeling, development and testing process uses free software, as well as the case study, which took place in a real commercial environment.

Keywords: Inventory, Hypermarket, Supplier, products, Database, validity, information system

SUMÁRIO

1 INTRODUÇÃO

O controle de validade de produtos em mercados de todos os tamanhos, sejam minimercados ou até hipermercados é um desafio o qual ainda, não possui um processo totalmente automático. O principal motivo para isso, é que o código de barras incluído nos produtos, é um código de barras único por produto, durante toda sua vida comercial, por exemplo, “barra de chocolate marca X, com x gramas possuirá sempre o código de barras *0809892842093”*, independente da data de fabricação e data de vencimento. A data de vencimento do produto está vinculada ao lote comprado e não ao código de barras original do produto em si.

Tendo em vista, que o código de barras dos produtos é a principal interface de comunicação entre os produtos e os sistemas de automação comercial, este código de barras fixo, limita o controle de produtos por data de fabricação, consequentemente por data de validade.

Este trabalho propõe o desenvolvimento de um sistema para controlar e gerenciar validades de produtos, o qual o ponto de partida inicial seria o código de barras original do fabricante, contudo, a partir deste, gera-se outro código de barras próprio, que contém a informação *CodBarrasOriginal+DataValidade+Lote.* Com isso, é gerado então um novo código de barras que permite saber exatamente a validade de determinado produto ou lote dos mesmos.

Entre vários benefícios de ter este controle mais eficiente, estão: diminuir os preços desses itens para antecipar a venda e proporcionar a este, um melhor destino, além de claro, evitar o descarte por vencimento dos produtos.

O sistema proposto foi desenvolvimento com ferramentas livres, como: a IDE NetBeans, o Sistema gerenciador de banco de dados MySql e alguns frameworks, que serão descritos na sequência, usando tecnologias atuais e futuramente com atualizações poderá ter novas funcionalidades, conforme venha querer o usuário.

1.1 OBJETIVOS

1.1.1 OBJETIVO GERAL

Desenvolver um sistema para auxiliar de maneira efetiva e automática no controle de validade de produtos, que permita controlar e visualizar validades dos produtos, e também relatórios específicos e sob demanda, com validade fora do prazo e ou falta de vendas, além de tantos outros relatórios específicos que sejam necessários para atender o objetivo do sistema, tanto analíticos quanto sintéticos.

1.1.2 OBJETIVO ESPECÍFICOS

- Fazer lançamentos de produtos novos
- Cadastrar lotes de produtos
- Alterar dados dos produtos
- Consultar produtos
- Emitir relatórios, com os produtos com as datas e quantidades

1.2 JUSTIFICATIVA

A perda de produtos, seja pelo motivo que for, gera prejuízos econômicos para qualquer empresa. As vezes esta perda pode ocorrer por acidente, furto, má manipulação, contaminação, entre outros.

Contudo, um dos motivos existentes para perda ou descarte de produtos é a questão do prazo de validade excedido. Este último motivo citado, é baseado na informação de data de validade, que nos sistemas atuais, não permite um gerenciamento eficiente.

O sistema descrito neste trabalho, permite que, juntando, o código de barras original, que é uma informação estática, mais a data de validade do lote de fabricação, mais o lote a qual pertence o produto, seja gerado um novo código de barras, e a partir deste, seja feito o gerenciamento automatizado, de forma simples, mas muito eficiente e descomplicada, da validade dos produtos, auxiliando na perda de produtos, que por si só, já é um evento indesejado, mas também evitando prejuízos econômicos para a empresa que utilizará o sistema.

A demanda surgiu de uma situação real, vivenciada, onde o próprio desenvolvedor do sistema percebeu a demanda junto à sua gerência administrativa, ambos, irão ajustar e testar o sistema em um domínio efetivo, conseguindo, ao final da implementação do mesmo, trazer resultados factíveis do uso do sistema, fazendo assim considerações conforme a realidade e também propondo continuidade neste trabalho se for necessário.

Observou-se nesta demanda recém explicada, a oportunidade de colocar em prática os conhecimentos adquiridos durante o Curso em Sistemas para Internet.

1.3 TRABALHOS CORRELATOS

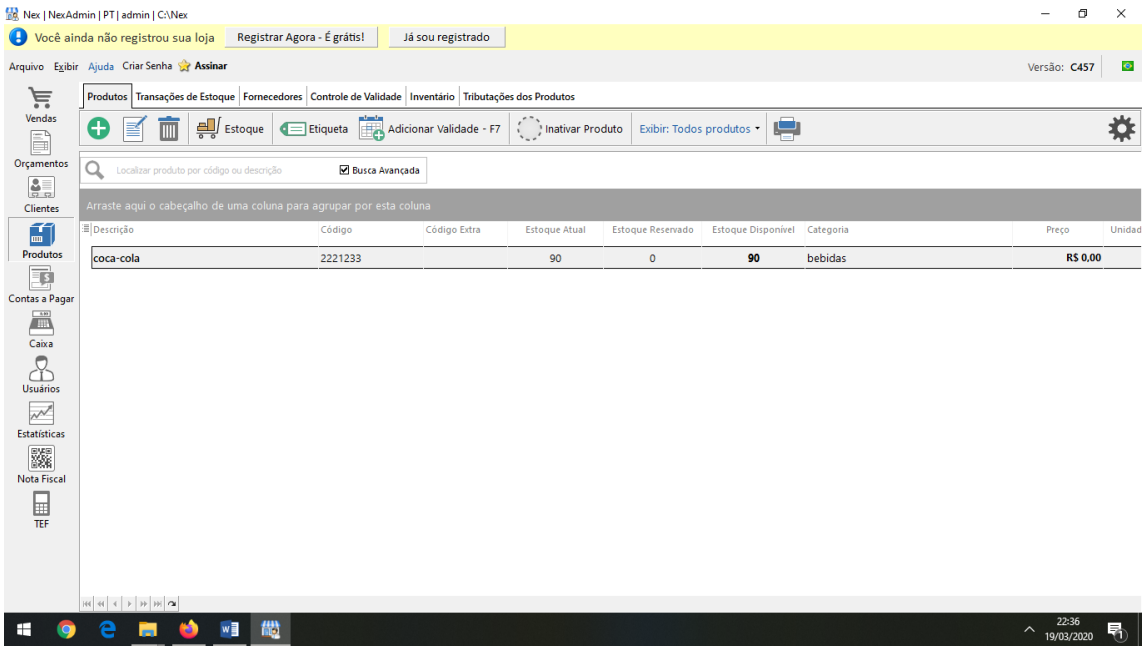
Nessa seção serão apresentados trabalhos semelhantes ao proposto, levando em consideração que a maioria dos sistemas são pagos, o que limita o acesso a estes.

1.3.1 NEX

O NEX é um sistema que controla seus produtos perecíveis por data de validade. Ele é um sistema completo de gestão da sua loja. Controla o estoque de perecíveis por Data de Validade e conta com recursos como Controle de Estoque e Vendas, PDV, Controle Financeiro, Controle de Caixa e muito mais.

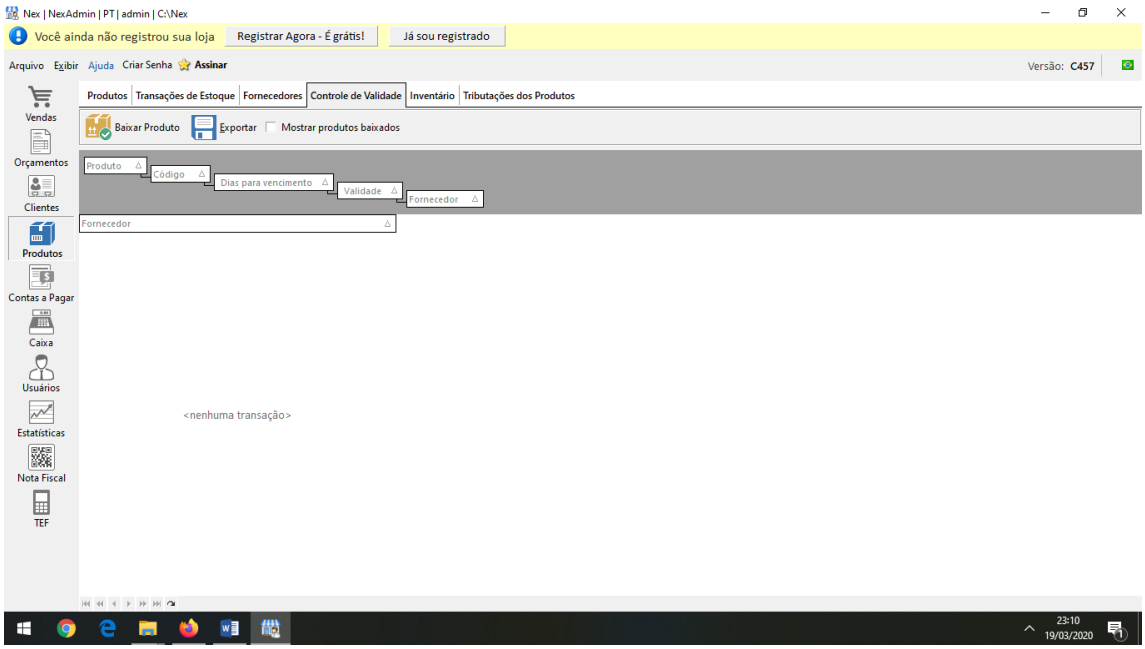
A seguir nas figuras 1 e 2 é possível ver o painel administrativo e o painel da parte da validade do programa NEX.

Figura 1: Painel Administrativo



Fonte: imagem capturada do programa NEX.

Figura 2: painel da parte de validade



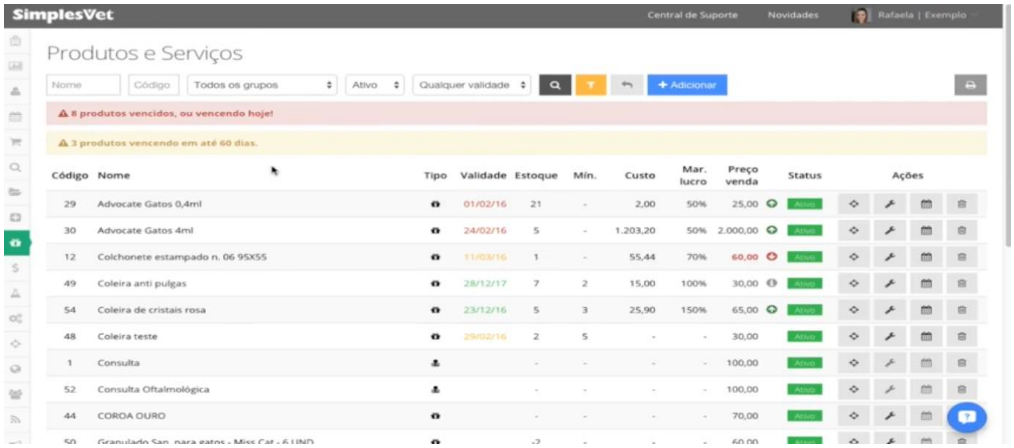
Fonte: imagem capturada do programa NEX.

1.3.2 Simples VET

Sistema de gestão de petshop, também tem a finalidade de controle de validades dos produtos da linha veterinária, podendo fazer as modificações necessárias ao produto. Esse sistema é pago por isso não foi possível obter mais informações.

Porém conseguimos visualizar o painel do programa a seguir na figura 3.

Figura 3: Painel do Programa



Fonte: imagem capturada do sistema

1.3.3 Web Jasper

Aplicativo móvel e web, destinado para controle de produtos de validade, esse aplicativo também é pago sendo necessário sua compra. Gera relatórios e indicadores de controle de forma automática, envia produtos com data curta para a área comercial de forma automática em tempo real, recebe notificações no seu celular quando os produtos estiverem próximo ao vencimento, utiliza relatórios de controle compartilhado entre filiais opção de cadastro para mais de 100.000 produtos Aplicativo mobile, controla diretamente de seu Smartphone ou Tablet.

A baixo na figura 4 é possível ver a logo do aplicativo Web Jasper.

figura 4: logo do site

Fonte :https://www.webjasper.com.br/#home

1.4 FUNDAMENTAÇÃO TEÓRICA

Com base no ambiente de trabalho, no dia a dia das funções exercidas, com a equipe de trabalho, não conseguimos controlar um mix com ‘n’ variedades de itens, para isso temos que ser mais assertivos.

Contudo, leva-se a muito sério essa questão de validades, pois está amparado na Lei Nº 8.078, DE 11 DE SETEMBRO DE 1990, (Brasil, 1990).

Assim, para se ter uma boa prática nesse ambiente utilizamos técnicas para os melhores manejos dos produtos. Dias (2010, p. 133) explica que o método PEPS (FIFO) significa: Primeiro a entrar, Primeiro a sair (First in, First out). A avaliação por este método é feita pela ordem cronológica das entradas. Sai o material que primeiro integrou o estoque, sendo substituído pela mesma ordem cronológica em que foi recebido, devendo seu custo real ser aplicado. Quando o giro dos estoques ocorre de maneira rápida ou quando as oscilações normais nos custos podem ser absorvidas no preço do produto, ou quando se

dispõe de material que esteja mantido por longo prazo, esse tipo de avaliação serve também para valorização dos estoques.

E como nos relata Madeira e Ferrão (2002, p323) “A disposição dos produtos deve obedecer a data de fabricação, sendo que os produtos de fabricação mais antiga são posicionados para serem consumidos em primeiro lugar (PEPS-primeiro que entra primeiro que sai ou pode-se utilizar o conceito de PVPS-primeiro que vence primeiro que sai).”

Já no setor varejista temos diversos segmentos, no entanto o segmento supermercadista tem grandes preocupações, então o primeiro trabalho é desenvolver através de informações reais, as quais setores têm mais problemas com vencimentos. Use o princípio de Pareto, conhecido como 80-20. Setores com mais incidência que ocorre validades:

- Perecíveis lácteos – queijos, iogurtes, manteigas, sobremesas lácteas e etc.
- Perecíveis Congelados e Resfriados – peixes, aves, alguns cortes bovinos, afiambrado, presuntos e etc.
- Bebidas não alcoólicas – refrigerantes, sucos, refresco em pó e etc.
- Mercearia Doce – chocolates, biscoitos, balas e etc.
- Matinais – cereal matinal, cappuccino, café e etc.

Outra parte que está atrelada às empresas mercadistas é o conceito de “**AGING**” (forma genérica de dizer que o produto está muito tempo parado, sem vender). O Aging de Estoque, ou o controle de estoque, é uma área muito importante, senão a mais importante para a pequena ou para a grande empresa, pois geralmente representa boa parte do ativo e capital de giro de uma empresa.

Segundo Sérgio Grünbaum:

> Estabelecer um sistema de controle sistemático para a não depreciação do estoque. Monitoramento constante de produtos em estoque para que os mesmos não venham entrar na linha de aging estabelecida de 25 dias. Reuniões periódicas para melhor posicionamento de ideias para o controle das mercadorias a serem compradas quando “sazonalidade”, quantidade e vendas.

1.4.1 Códigos De Barras

> “Em *1974 foi escaneado o primeiro código de barras nos Estados Unidos* e hoje cerca de *6 bilhões de códigos de barras são lidos por dia no mundo”* [1].

Neste trabalho será abordado o uso de códigos de barras, os quais possuem inúmeros tipos, aqui será executado os padrões EAN 13 e EAN 8. EAN=European Article Number (número de artigos europeu) mas foi renomeado (número de artigo internacional). Esses códigos possuem uma lógica onde cada bloco de números representa uma informação diferente. Esses blocos estão padronizados assim:

- O bloco 1: a sequência dos 3 números identifica de qual país os produtos foram cadastrados a do (Brasil é o 789).
- O bloco 2: 4 a 7 dígitos faz a identificação da empresa que cadastrou o produto.
- Bloco 3: ordem numérica que identifica os produtos, daí vem as variações dos produtos e suas especificações individuais.
- Segundo o site da área:

Por ser um padrão, o EAN facilita na identificação e na informação a respeito dos produtos, o que torna as relações comerciais mais fáceis e ágeis, otimizando o tempo em todos os processos — seja com clientes, fornecedores ou colaboradores.

Na figura 5 podemos ver essa exemplificação dos blocos de forma ilustrada e explicativa e compreender como se da a composição dos códigos de barras.

[1] Disponível em https://www.gs1br.org/servicos-e-solucoes/beneficios/Paginas/Automacao.aspx acesso em: 12 jul. 2020.

Figura 5: identificação do código

Fonte: http://www.code128.com.br/codigo-de-barras/como-conseguir-o-nmero-do-cdigo-de-barras-gs1-ean-para-seu-produto-2/

1.4.2 IDE Netbeans

Essa ferramenta criada é um ambiente de desenvolvimento integrado (IDE) Java desenvolvido pela empresa Sun Microsystems, lançada em 1 de dezembro de 2000, de acordo com o website de tecnologia:

> Ambiente de desenvolvimento multiplataforma, uma ferramenta que auxilia programadores a escrever, compilar, debugar e instalar aplicações, foi arquitetada em forma de uma estrutura reutilizável que visa simplificar o desenvolvimento e aumentar a produtividade pois reúne em uma única aplicação todas estas funcionalidades. Totalmente escrita em Java, mas que pode suportar qualquer outra linguagem de programação ou linguagem que desenvolva com Swing, algumas das linguagens que o NetBeans suporta são o C, C++, Ruby, PHP, XML e linguagens HTML.

1.4.3 Linguagem Java (Ee)

Java é uma linguagem de programação orientada a objetos desenvolvida na década de 90 por uma equipe de programadores chefiada por James Gosling, na empresa Sun Microsystems. A sintaxe utilizada deriva do C++, porém com

um modelo mais simples. Como sua principal característica, todo o código é escrito dentro de uma classe e tudo é um objeto.

De acordo com website:

> O **Java EE** simplifica o desenvolvimento do aplicativo e diminui a necessidade de programação e de treinamento do programador, criando componentes modulares reutilizáveis padronizados e ativando a camada para tratar de muitos aspectos da programação de forma automática.

1.4.4 Glass Fish

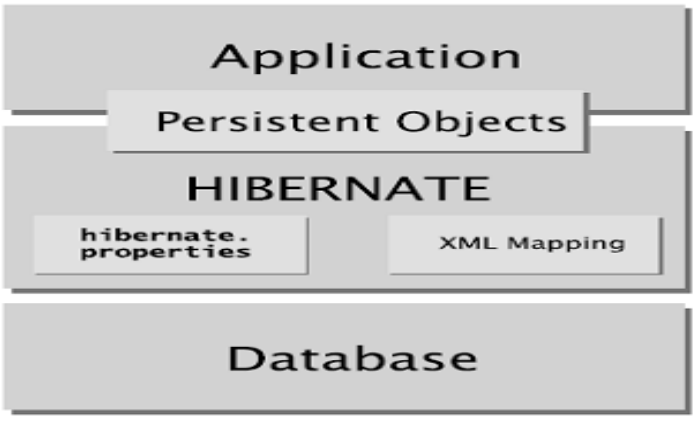
Servidor de aplicação. A própria Sun/Oracle desenvolve uma dessas implementações, o Glassfish que é open source e gratuito.

1.4.5 Hibernate

O Hibernate é um framework de mapeamento objeto relacional para aplicações Java, ou seja, é uma ferramenta para mapear classes Java em tabelas do banco de dados e vice-versa, através do conceito de anotações o que facilita ainda mais o mapeamento objeto-relacional, que pode agora ser feito diretamente na classe.

Para melhor compreensão na figura 6 é possível ver o funcionamento do Hibernate dentro da aplicação, fazendo o mapeamento de classes Java em tabelas do BD.

Figura 6: Demonstração do hibernate dentro da aplicação



Fonte: capturada do site https://www.devmedia.com.br/desenvolvendo-com-hibernate/14756

1.4.6 Java Server Faces

Um framework web baseado em Java que tem como objetivo simplificar o desenvolvimento de interfaces (telas) de sistemas para a web, através de um modelo de componentes reutilizáveis. (EGÍDIO).

1.4.7 Prime Faces

É uma coleção de componentes de UI ricos para Java Server Faces. Todos os widgets são open source e gratuitos para uso sob licença Apache. O Primefaces é desenvolvido pela PrimeTek Informatics, um fornecedor com anos de experiência no desenvolvimento de soluções de UI de código aberto.

1.4.8 MYSQL

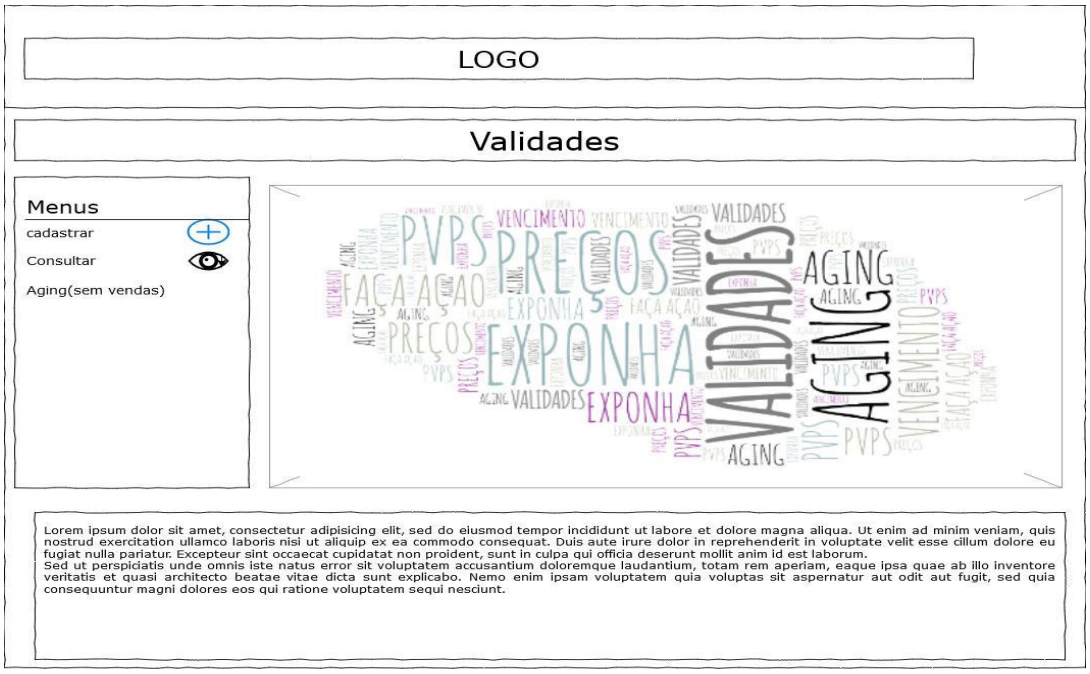
O programa MySQL é um servidor robusto de bancos de dados SQL (Structured Query Language – Linguagem Estruturada para Pesquisas) muito rápido, multitarefa e multiusuário, servidor MySQL pode ser usado em sistemas de produção com alta carga e missão crítica bem como pode ser embutido em programa de uso em massa, é atualmente um dos sistemas de gerenciamento de bancos de dados mais populares da Oracle Corporation, com mais de 10 milhões de instalações pelo mundo (ORACLE CORPORATION, 2019).

1.4.9 Wireframe

Desenho básico, seria o esqueleto do site propriamente dito, são usados no início do processo de desenvolvimento para estabelecer a estrutura básica de uma página antes de acrescentar o design visual e conteúdo, auxiliar o desenvolvedor no entendimento dos requisitos que foram recolhidos junto ao cliente com relação às funções e objetos que um sistema deve conter.

Na figura 7 é possível ver um demonstrativo de esboço da página inicial do WIREFRAME.

Figura 7: Demonstrativo de esboço da página inicial.



Fonte: Própria (2020)

1.4.10 Jasper Reports

Segundo website de tecnologia:

Jasper Reports Server é um servidor de relatório independente e incorporável. Ele fornece relatórios e análises que podem ser incorporados a um aplicativo da Web ou móvel, além de funcionar como um hub central de informações para a empresa, fornecendo informações.

1.4.11 Técnica MVC

Técnica usada para para você pensar, em uma divisão de tarefas ou responsabilidades do projeto, no ambiente web. Na figura 8 pode se ver o demonstrativo do padrão MVC.

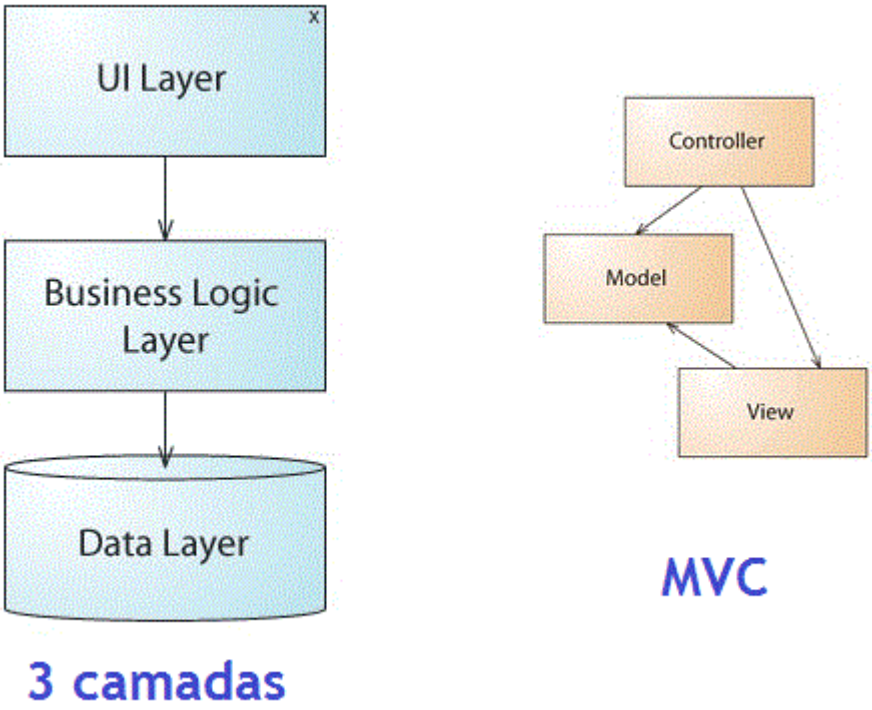
A manipulação dos dados temos:

- View, essa camada responsável pela aparência, interface que é apresentada ao usuário.
- Controller, nessa camada ela fica responsável de ligar o model e a view,

e vice-versa.

- Model, fica como responsável por representar o negócio do projeto, fazendo o acesso e as manipulações dos dados.

Figura 8: Demonstrativo do padrão mvc



fonte:http://www.macoratti.net/vbn_mvc.htm

2 DESENVOLVIMENTO

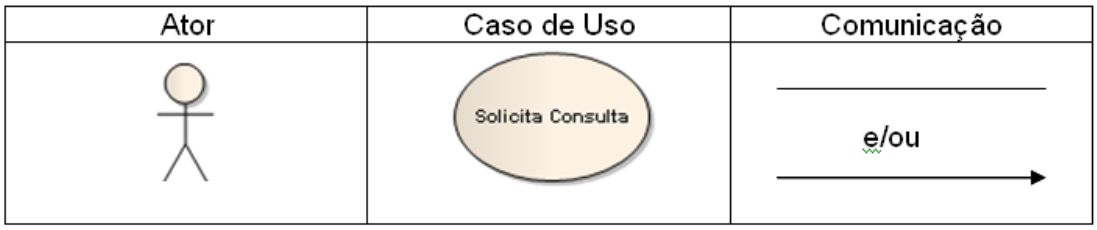
2.1 CASOS DE USO

Casos de usos são uma demonstração de como o sistema irá se comportar num todo representado com atores, os casos de uso podem ser vistos como a ponte entre a Análise e a Especificação de Requisitos. Na figura 9 podemos ver uma ilustração de símbolos dos casos.

De acordo com Gilleanes T. A. Guedes:

> Diagrama mais geral e informal da UML, utilizado normalmente nas fases de levantamento e análise de requisitos do sistema, embora venha a ser consultado durante todo o processo de modelagem e possa servir de base para outros diagramas. Apresenta uma linguagem simples e de fácil compreensão para que os usuários possam ter uma ideia geral de como o sistema irá se comportar. Procura identificar os atores (usuários, outros sistemas ou até mesmo algum hardware especial) que utilizarão de alguma forma o software, bem como os serviços, ou seja, as funcionalidades que o sistema disponibilizará aos atores, conhecidas neste diagrama como casos de uso (2009. p31).

Figura 9: Ilustração de símbolos dos casos

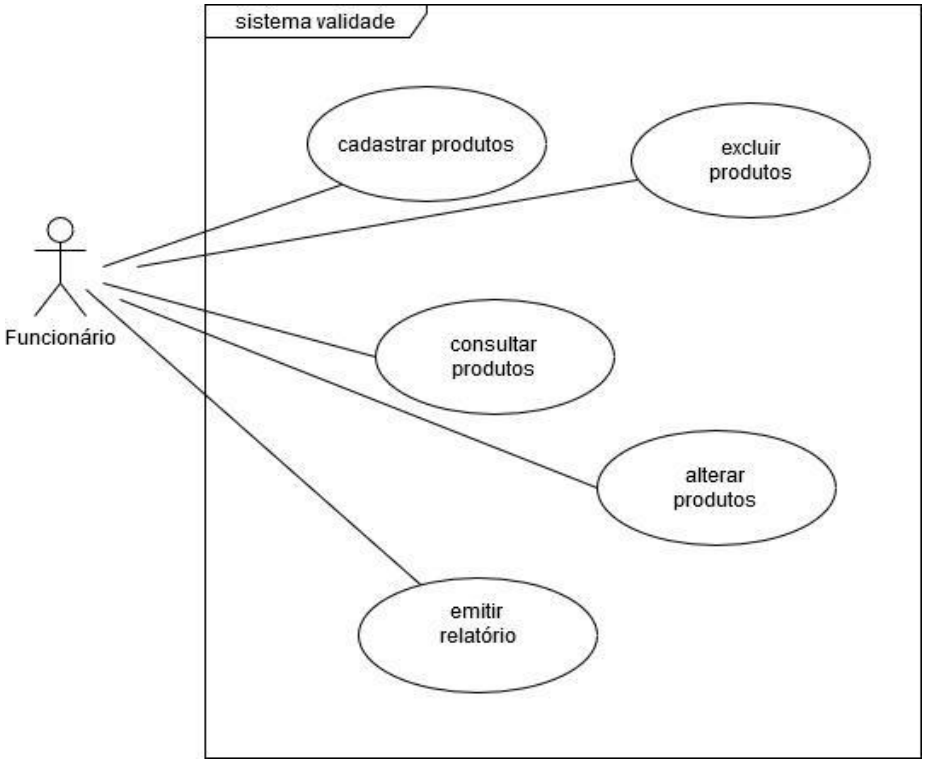


Fonte: Capturado do site https://www.devmedia.com.br/o-que-e-uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408

2.1.1 Diagrama De Caso De Uso

Na figura 10, temos a demonstração das funcionalidades do sistema com relação ao ator (funcionário), irá desempenhar essas funções, uma representação visual de como se comporta e mostrando também as comunicações entre eles.

Figura 10: Caso de uso no sistema



Fonte: Própria (2020)

2.1.2 Especificações De Caso De Uso

As especificações de casos de usos, são uma representação de como os passos dentro do sistema irá se conduzir.

Técnica que descreve a sequência de uma determinada ação que o sistema deve realizar, assim o sistema dará uma resposta para o ator.

A seguir na tabela 1, é possível ver a especificação do caso de uso e o cadastro de produtos. Já na tabela 2 é possível ver a especificação de alterar produto. E na tabela 3, a especificação de emitir relatório.

Tabela 1: Especificação do caso de uso, cadastro de produtos.

Identificação: CP01	**Nome**: Cadastrar Produtos
Atores: Colaborador	
Resumo: Descreve como o colaborador irá conduzir os passos para efetuar o cadastro dos produtos	
Pré-Condições: Estar com o sistema aberto.	
Pós-Condições: Cadastro efetuado	
Sequência típica de eventos	

Sequência Alternativa	
1a. todos os campos devem ser preenchidos 2a. campo com dados invalido	Retorna na opção faltante. Retorna com dizer que só aceita campo com numeração ou escrita.

Fonte: Própria (2020)

Tabela 2: Especificação do caso de uso alterar

Identificação: AL01	**Nome**: Alterar Produtos
Atores: Colaborador	
Resumo: Descreve como o colaborador irá conduzir os passos para alterar os dados já inseridos dos produtos	
Pré-Condições: Estar com o sistema aberto.	
Pós-Condições: Alteração efetuada	
Sequência típica de eventos	
Sequência Alternativa	
1a. todos os campos devem ser preenchidos 2a. campo com dados invalido	Retorna na opção faltante. Retorna com dizer que só aceita campo com numeração ou escrita.

Fonte: Própria (2020)

Tabela 3: Especificação caso de uso emitir relatório

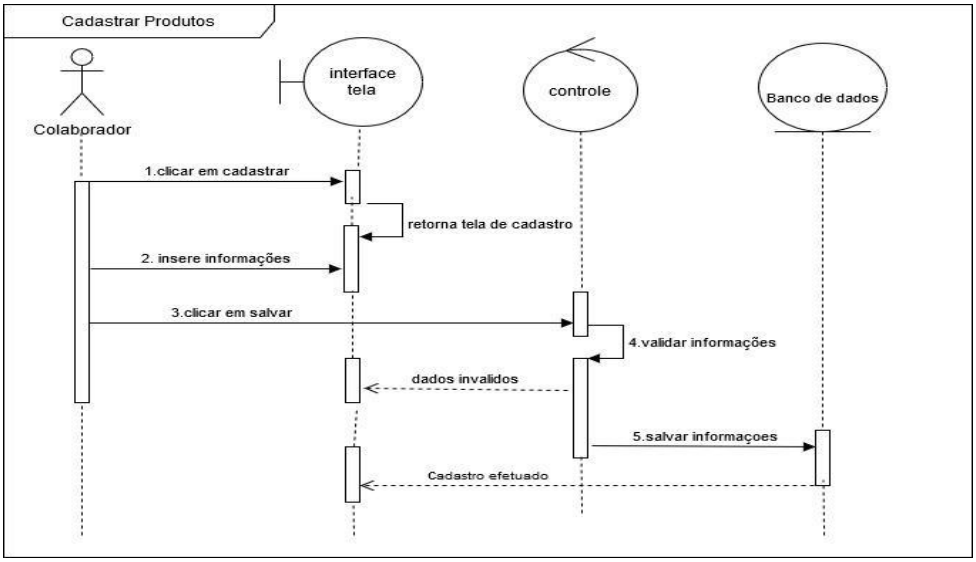
Identificação: ER01	**Nome**: Emitir Relatório
Atores: Colaborador	
Resumo: Descreve como o colaborador irá conduzir os passos para gerar um tipo de relatório	
Pré-Condições: Estar com itens cadastrados	
Pós-Condições: Gerar relatório com as informações	
Sequência típica de eventos	
Sequência Alternativa	
	Informa que não tem relatório a ser gerado

Fonte: Própria (2020)

2.2 DIAGRAMA DE SEQUÊNCIA

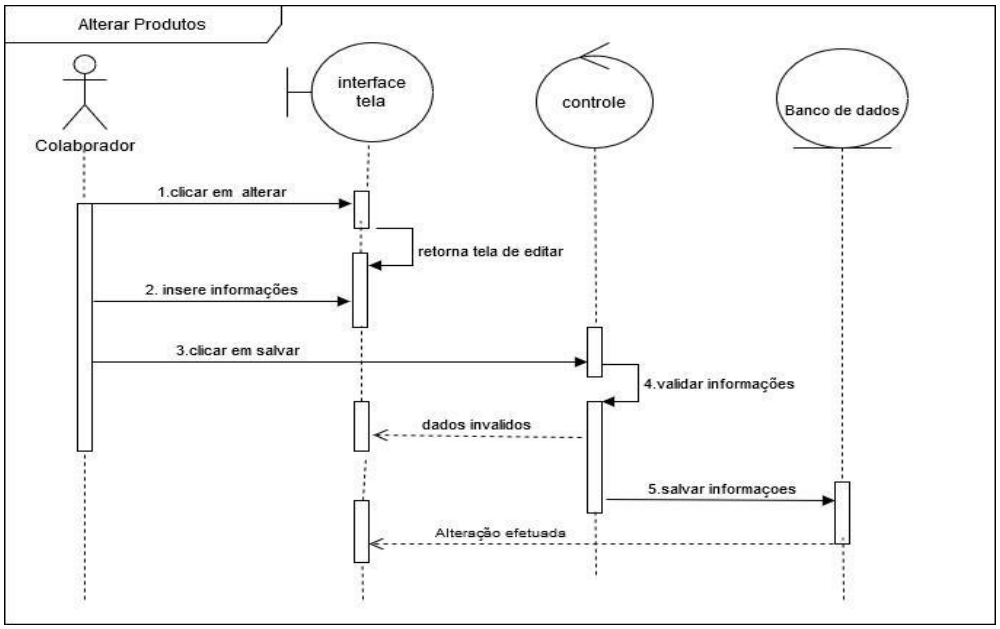
Diagrama de sequência na UML é tipo uma interação, como irá agir a ordem de um grupo de objetos. A figura 11 exibe o diagrama de sequência do aplicativo para a função “cadastrar”. Já na figura 12 está ilustrado o diagrama da sequência do alterar produto. E na figura 13 há o diagrama de sequência de emitir relatório.

Figura 11: Diagrama de sequência do cadastrar



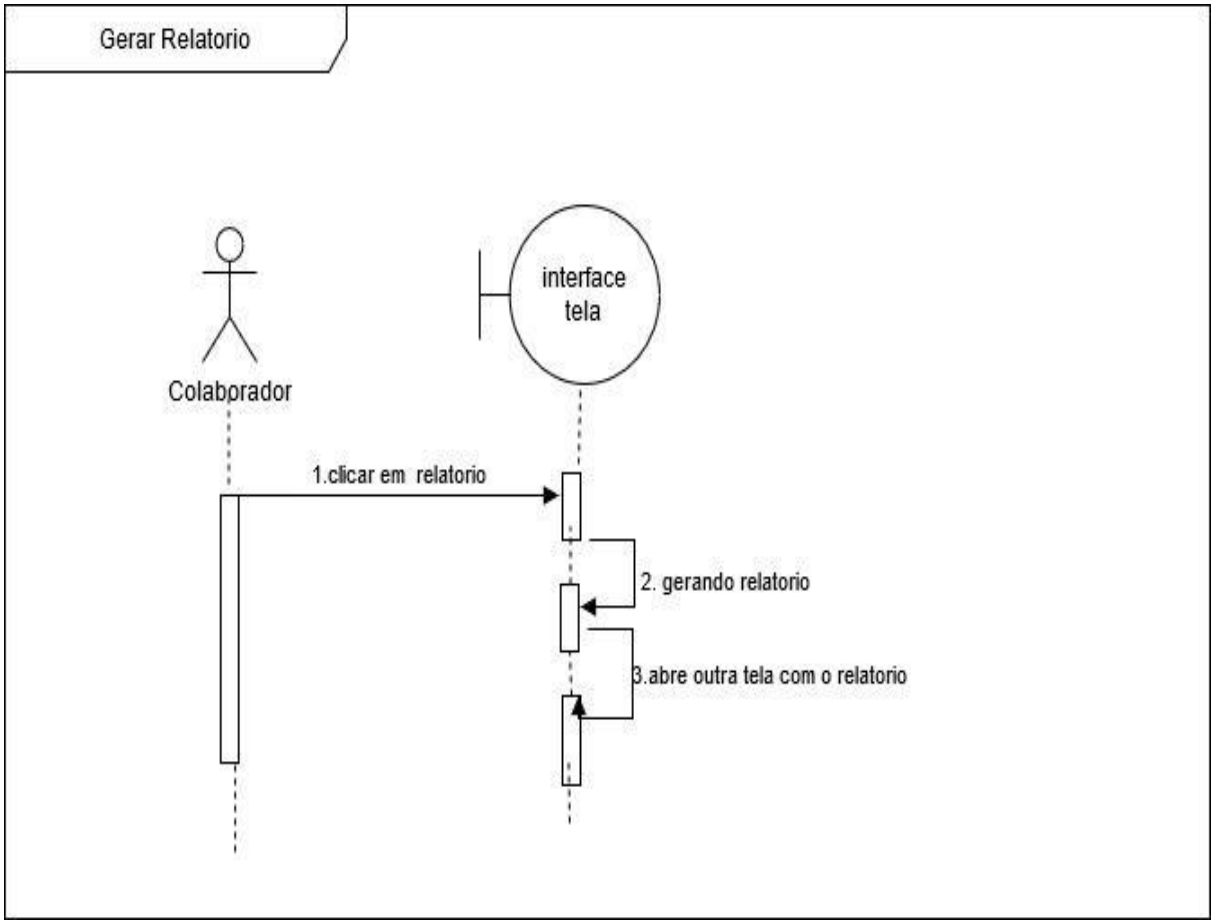
Fonte: Própria (2020)

Figura 12: Diagrama de sequência do alterar produto



Fonte: própria (2020)

Figura 13: Diagrama de sequência do emitir relatório



Fonte: Própria (2020)

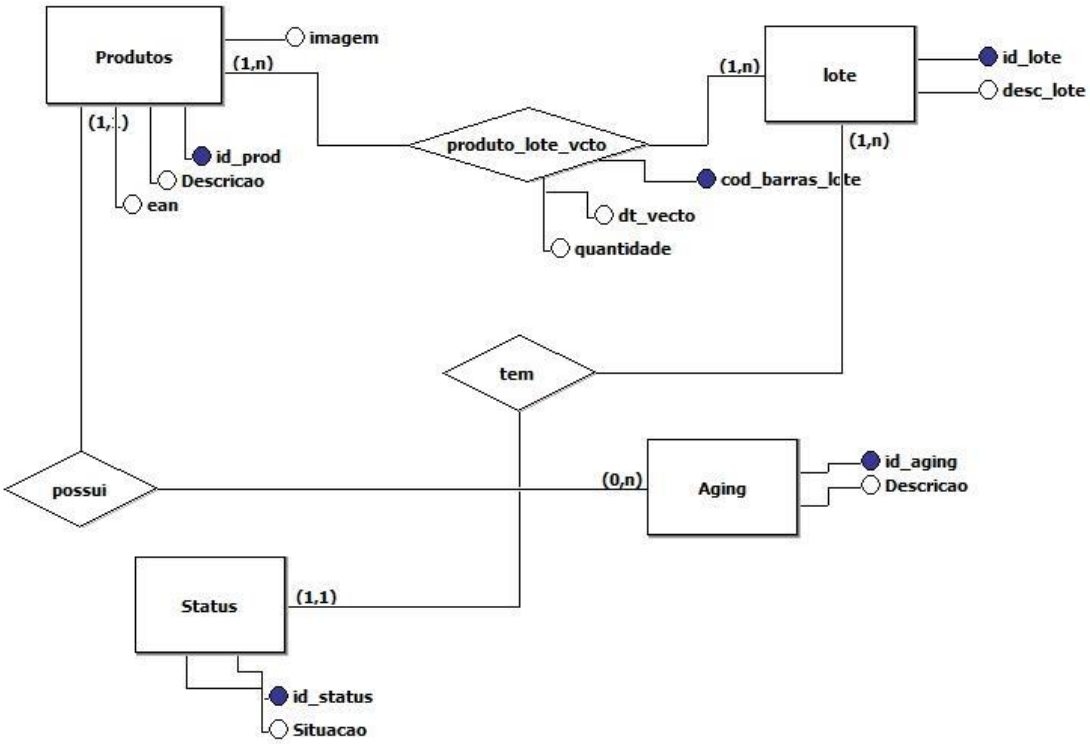
2.3 BANCO DE DADOS

Um banco de dados ou coleção de dados, no contexto, um dado é um fato que deve ser persistido (armazenado), ele tem significado implícito nele. Tem estrutura lógica que tem significado aos dados.

2.3.1 Modelo Entidade Relacionamento

Modelo conceitual utilizado na engenharia de software, representa visualmente os objetos ou (entidades) que são envolvidos num todo com seus atributos (características) e como que vão se relacionar entre si. Na figura 14 pode se ver a ilustração do molde do modelo.

Figura 14: Molde do modelo.



Fonte: Própria (2020).

2.3.2 Modelo Lógico[2]

Uma descrição das estruturas armazenadas no banco e que vai resultar na representação visual dos dados, de uma maneira lógica.

2.3.3 Modelo Físico[3]

Uma descrição de um banco de dados visto no nível de abstração, visto pelo usuário do SGBD, são detalhados os componentes, como tabelas, campos e tipos de valores, índices.

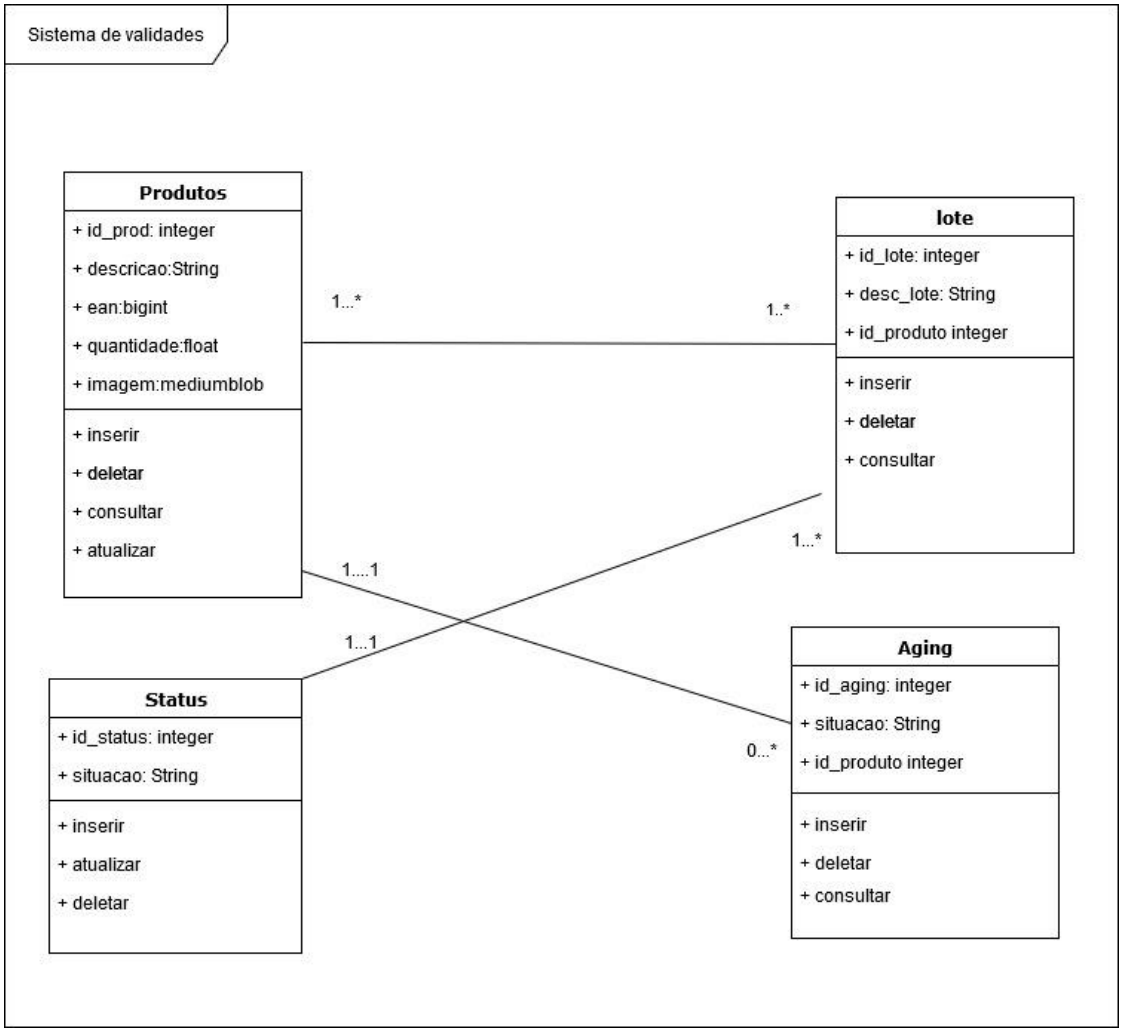
2.4 DIAGRAMA DE CLASSES

O diagrama de classes basicamente usado para documentar arquitetura de software, diagramas de classes são um tipo de estrutura, onde descrevem o

[2] Modelo lógico do sistema desenvolvido está disponível no apêndice.
[3] Modelo físico do sistema desenvolvido está disponível no apêndice.

que deve estar presente no sistema a ser modelado. Sendo assim na figura 15 há a ilustração do diagrama de classes do sistema.

Figura 15: Diagrama de classes do sistema



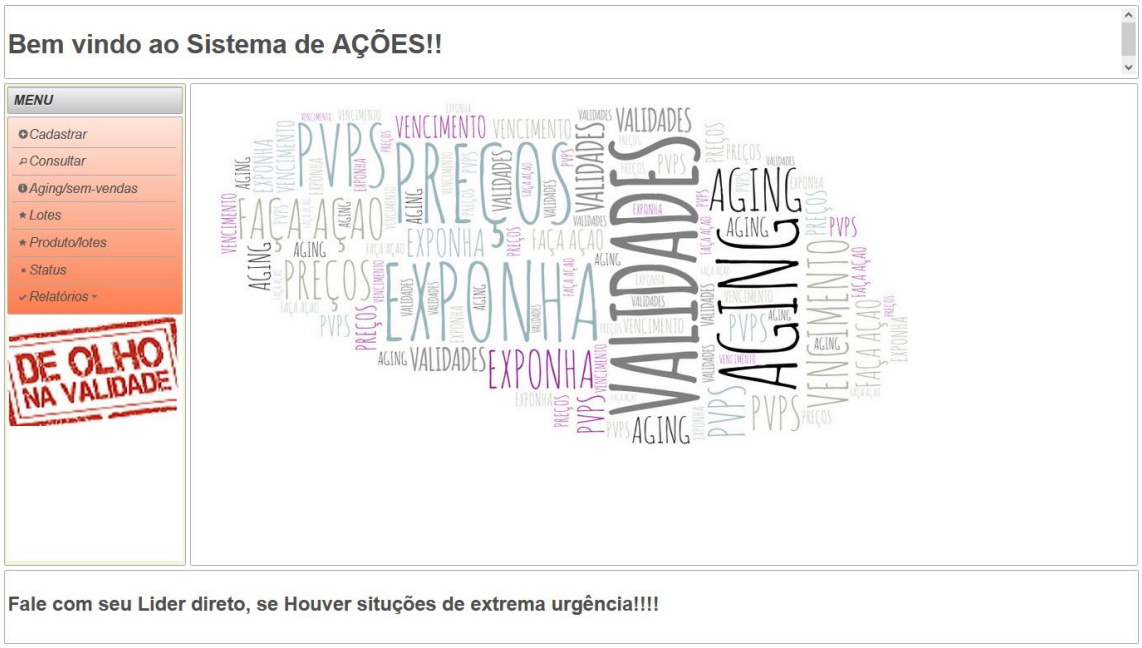
Fonte: Própria (2020).

2.5 TELAS DO SISTEMA

2.5.1 Tela De Home Do Sistema.

Como é possível ver na figura 16 é a tela home, ela é a tela que inicia a aplicação, pode se escolher as opções do menu lateral.

Figura 16: Tela home

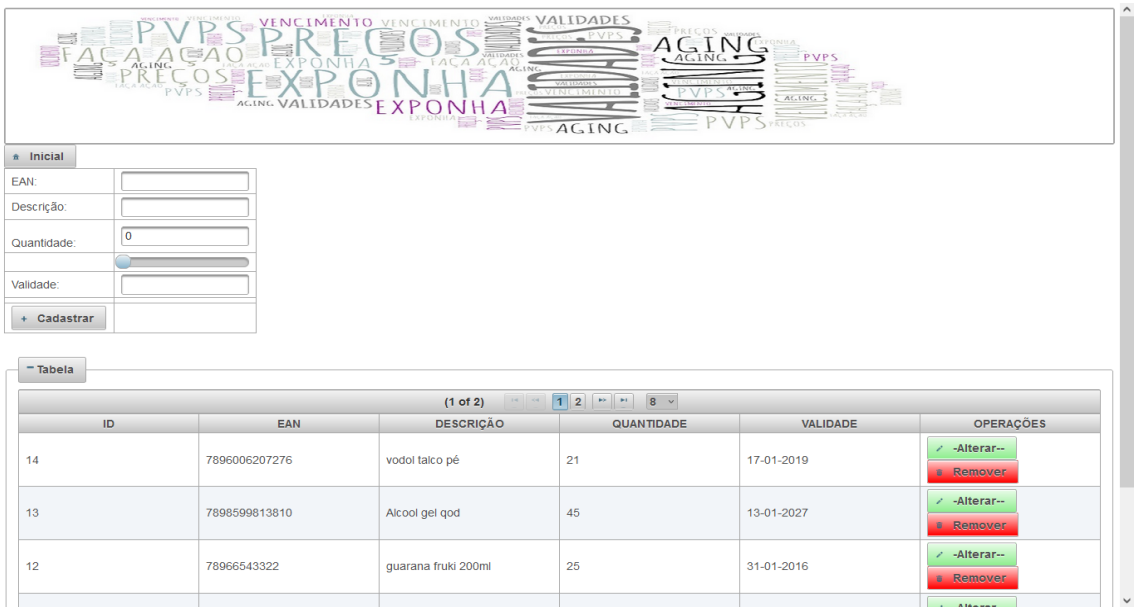


Fonte: Própria (2020).

2.5.2 Tela De Cadastro.

Na figura 17 podemos ver a tela de cadastro, onde se faz o lançamento do cadastro dos produtos a serem manipulados durante seu tempo de validade.

Figura 17: Tela de cadastro

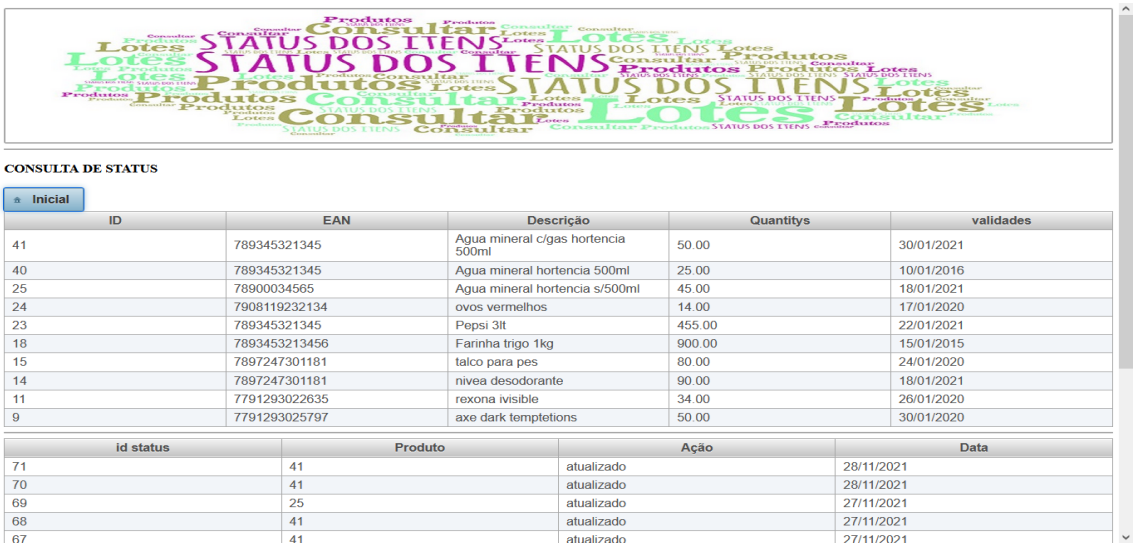


Fonte: Própria (2020).

2.5.3 Tela De Status.

Na figura 18 há a tela de status, nela, irá mostrar uma breve descrição que ocorreu, quando foi modificado o item, dizendo se foi inserido, excluído ou alterado.

Figura 18: Tela de status



Fonte: Própria (2020)

2.5.4 Tela De Aging.

Nesta figura 19, iremos encontrar uma verificação dos itens que não estão vendendo já algum tempo, onde se chama de aging/ sem vendas.

Figura 19: Tela de aging

Fonte: Própria (2020)

2.5.5 Tela Do Relatório Obtido.

E por fim demostramos a figura 20, onde pode se observar a tela de relatório. É nesta tela que veremos o relatório dos itens cadastrados, tornando sua visualização melhor de se trabalhar, para dar destino aos itens perto da validade.

Figura 20: Tela do relatório gerado

id_pro	EAN	DESCRIÇAO	QUANTIDAD	VALIDADES
9	779129302579	axe dark temptetions	50.00	30/01/20
11	779129302263	rexona ivisible	34.00	26/01/20
14	789724730118	nivea desodorante	90.00	18/01/21
15	789724730118	talco para pes	80.00	24/01/20
18	789345321345	Farinha trigo 1kg	900.00	15/01/15
23	789345321345	Pepsi 3lt	455.00	22/01/21
24	790811923213	ovos vermelhos	14.00	17/01/20
25	78900034565	Agua mineral hortencia s/500ml	45.00	18/01/21
40	789345321345	Agua mineral hortencia 500ml	25.00	10/01/16
41	789345321345	Agua mineral c/gas hortencia	50.00	30/01/21

Fonte: Própria (2020)

3 RESULTADOS E DISCUSSÕES

3.1 Questionário De Avaliação E Testes Dos Sistemas

O sistema pode ter acesso em qualquer navegador da web para computador, e irá interagir em qualquer sistema operacional que venha o usuário a usar.

O sistema deverá ser implementado em qualquer máquina ou servidor que suporte os programas: Netbeans, Glassfish, Mysql, jasperreports.

A seguir há uma avaliação realizada com as notas, sendo 1 insatisfeito a 5 de satisfeito, essa avaliação foi realizada pelo cliente usuário que testou o

sistema, este que podemos visualizar na figura 21. E na figura 22 os computadores o qual o programa foi usado.

Tabela 4: Questionário de avaliação

Questionário Ergonômico	1	2	3	4	5
Design é atraente?			x		
O sistema é de fácil entendimento?				x	
O sistema vai facilitar a lista de validades?				x	
Emissão de relatório satisfaz?				x	
Recurso da máquina para utilizar.					x
O sistema deixa claro o item exibido?			x		
Interação dentro do sistema.				x	
Está satisfeito com o sistema?			x		

Fonte: Própria (2020)

Figura 21: Cliente que testou o sistema

Fonte: Própria (2020)

Figura 22:Computadores para instalação do programa.

Fonte: Própria (2020)

3.1.1 Resultados Do Sistema

A aplicação se demonstrou vantajosa em partes, pois se gera relatório de fácil entendimento, proporcionando uma rapidez em buscar o destino dos itens que estão para vencer.

Assim podendo qualquer funcionário ou terceiros, estar lançando os itens que estão perto de vencer no sistema.

4 CONCLUSÃO

Conclui-se que o sistema para controlar e visualizar validades, demonstrou ser eficiente e necessário na rotina vivenciada e observada no ambiente de trabalho em que ele foi testado.

Pois o mesmo agregou a instituição com a qualidade de identificar as validades dos produtos, facilitou o lançamento de informações com maior agilidade.

O que nos possibilita dizer que o sistema não só contribui para a visualização da validade dos produtos, mas também fornece maior rendimento de tempo. Isto porque, o que por muitas vezes é um processo manual, através do sistema se torna em um processo automatizado.

Com essas qualidades e contribuições é cabível dizer que o sistema, não precisa ser necessariamente limitado à supermercados, com algumas modificações, pode-se também ser usado em outros segmentos futuros.

Contudo existem vários itens que possam vir a melhorar o projeto, pois se levar em consideração a avaliação do cliente que o testou: melhorar o design para se tornar mais atraente, e também deixar mais claro o item exibido. Alterações essas que podem tornar o sistema mais atrativo, intuitivo e dinâmico, como:

- Imagens
- Categorias
- Leituras com Qr code

REFERÊNCIAS

DIAS, Marco Aurélio P. **Administração de materiais**: uma abordagem Logística. 5. ed. São Paulo: Atlas, 2010.

GUEDES, Gilleanes T. A. UML 2: **uma abordagem prática** / Gilleanes T. A. Guedes. -- São Paulo: Novatec Editora, 2009.

(Balbino, A. **Como Controlar Validade de Produtos**) cafecomprevencao.com.br/como-controlar-a-validade-dos-produtos> acesso em: 03 abr. 2020.

Ramez Elmasri, Shamkant B. Navathe. **Sistemas de Banco de Dados**. 6ª Ed. São. Paulo: Pearson, 2011.

O QUE É NET BEANS? **Oficinadanet**, 2008. Disponível em:< https://www.oficinadanet.com.br/artigo/1061/o_que_e_o_netbeans> acesso em: 07 abr. de 2020.

PEREIRA, Ana Paula. O que é wireframe? **tecmundo,** 2008. Disponível em:<https://www.tecmundo.com.br/programacao/976-o-que-e-wireframe-.htm>acesso em:10abr de 2020.

wireframe-diagrams.net (formerly *draw.io*) is free online diagram software. You can use it as a flowchart maker, network diagram software, to create UML online.

Disponivel <https://app.diagrams.net/>acesso em 14 de abr. de 2020.

O QUE É JASPER REPORTS? disponível em < https://community.jaspersoft.com/> acesso em 8abr de 2020.

Padrão de código de barras **Gs1br,**2016.

Disponível em <https://blog.gs1br.org/6-principais-duvidas-sobre-o-codigo-de-barras-ean/>acesso em:06 mai. 2020.

Códigos de barras EAN-8 **Cognex,** Disponível em <https://www.cognex.com/pt-br/resources/symbologies/1-d-linear-barcodes/ean-8-barcodes> acesso em 06 mai. 2020.

O que é diagrama de classe UML? Disponível em https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-classe-uml#section_0> acesso em 11 jun. 2020.

Modelo Entidade Relacionamento (MER) Disponível em <https://www.devmedia.com.br/modelo-entidade-relacionamento-mer-e-diagrama-entidade-relacionamento-der/14332> acesso em 11 jun. 2020.

Disponível em < https://balbino.info/como-controlar-a-validade-dos-produtos/> acesso em: 03 abr. 2020.

[1] Disponível em <http://www.maxview.com.br/serie_kpis_aging_estoque/> acesso em: 01 mai. 2020.

Disponível em https://www.javaavancado.com/ebooks/JSF_PrimeFaces_Hibernate.pdf> acesso em 15abr 2020.

APÊNDICE

APÊNDICE A - MODELO LÓGICO

Produtos (id_prod, Descricao, ean, imagem,)

Produto_lote_vcto (cod_barras_lote, dt_vcto, quantidade, id_prod, id_lote)

Id_lote referência tabela lote;

Id_prod referência tabela Produtos;

Lote (id_lote, desc_lote)

Aging (id_aging, Descricao)

Status (id_status, situcao)

APÊNDICE B - MODELO FÍSICO

```
CREATE TABLE `aging` (
          `id_aging` int(11) NOT NULL,
    ‘id_produto’ int(11) NOT NULL,
  `descricao` varchar(30) COLLATE utf8_bin NOT NULL
FOREIGN KEY (`id_produto`) REFERENCES `produtos`  (id_produto`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

CREATE TABLE `lote` (
  `id_lote` int(11) NOT NULL,
    ‘id_produto’ int(11) NOT NULL,
    ‘id_status’ int(11) NOT NULL,
  `desc_lote` varchar(10) COLLATE utf8_bin NOT NULL

FOREIGN KEY (`id_produto`) REFERENCES `produtos` (`id_produto`)

FOREIGN KEY(`id_status`) REFERENCES `status`(id_status`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

CREATE TABLE `produtos` (
  `id_produto` int(11) NOT NULL,
  `ean` bigint(15) NOT NULL,
  `descricao` varchar(30) COLLATE utf8_bin NOT NULL,
  `quantidade` float NOT NULL,
  `validade` date NOT NULL,
  `imagens` mediumblob DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;

CREATE TABLE `produto_lote_vcto` (
  `id_prod_lote_vcto` int(11) NOT NULL,
  `id_produto` int(11) NOT NULL,
  `id_lote` int(11) NOT NULL,
  `cod_barras_novo` int(15) NOT NULL,
  `data_vcto` date NOT NULL,
  `quantidade` float NOT NULL
 FOREIGN KEY (`id_produto` ) REFERENCES `produtos`        (`íd_produto`)
FOREIGN KEY (´id_lote`) REFERENCES `lote` (id_lote`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;



CREATE TABLE `status` (

          `id_status` int(11) NOT NULL,

    ‘situacao’ varchar(30) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin;
```

```
ALTER TABLE `aging`
  ADD PRIMARY KEY (`id_aging`);
ALTER TABLE `lote`
  ADD PRIMARY KEY (`id_lote`);
ALTER TABLE `produtos`
  ADD PRIMARY KEY (`id_produto`);
ALTER TABLE `produto_lote_vcto`
  ADD PRIMARY KEY (`id_prod_lote_vcto`);
ALTER TABLE `status`
  ADD PRIMARY KEY (`id_status`);


ALTER TABLE `status`
  MODIFY `id_status` int(11) NOT NULL  AUTO_INCREMENT;
ALTER TABLE `aging`
  MODIFY `id_aging` int(11) NOT NULL  AUTO_INCREMENT;
ALTER TABLE `lote`
  MODIFY `id_lote` int(11) NOT NULL AUTO_INCREMENT,
        AUTO_INCREMENT;
ALTER TABLE `produtos`
  MODIFY `id_produto` int(11) NOT NULL AUTO_INCREMENT,
        AUTO_INCREMENT;
```

APÊNDICE C- CODIGO FONTE

Código Fonte da tela cadastro produtos.

```
<?xml version='1.0' encoding='UTF-8' ?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html    xmlns="http://www.w3c.org/1999/xhtml"

        xmlns:h="http://xmlns.jcp.org/jsf/html"

        xmlns:f="http://java.sun.com/jsf/core"

        xmlns:p="http://primefaces.org/ui">

   <h:outputStylesheet name="primeicons/primeicons.css" library="primefaces" />

   <h:head>

      <title>Cadastro de Produtos</title>

   </h:head>

   <p:layoutUnit position="center"  resizable="true" closable="true" collapsible="true">

      <img src="imagens/art1.jpeg" alt=""/>

   </p:layoutUnit>

   <h:body>

      <h:form>

         <p:growl id="growl" showDetail="true" />

         <p:commandButton value="Inicial"  icon="ui-icon-home"
action="dashboard.xhtml">

         </p:commandButton>

      </h:form>

      <h:form enctype="multipart/form-data">

         <!-- painel dos input -->

         <p:panelGrid columns="2" styleClass="ui-noborder" id="grid">

            EAN:<p:inputText id="ean" value ="#{produtosBean.produto.ean}"
required="true" label="CodBarras" maxlength="15"/>

            Descrição:<p:inputText id="descricao" value
```

```
="#{produtosBean.produto.descricao}" required="true" label="Descriçao"/> <br/>

            Quantidade: <p:inputText id="qtd" value
="#{produtosBean.produto.quantidade}" required="true" label="Quantidade"/>  <br/>
<p:slider for="qtd"/>

            Validade:<p:calendar id="validade" value
="#{produtosBean.produto.validade}"  effect="fold"  locale="pt" navigator="true"
required="true" label="Vencimento" />

            <p:growl id="messages" showDetail="true" />

            <p:commandButton value="Cadastrar" ajax="true" icon="ui-icon-plus"
styleClass="ui-ribbon-bigbutton"  action="#{produtosBean.adicionarProdutos()}"
update="@form"  />

          </p:panelGrid>

      </h:form>

      <br>

      </br>

        <p:fieldset legend="Tabela" toggleable="true"  toggleSpeed="500">
<p:dataTable value="#{produtosBean.listarProdutoFull()}" var="dados"

                  rows="5" paginator="true" id="dados"

                  paginatorTemplate="{CurrentPageReport}  {FirstPageLink}
{PreviousPageLink} {PageLinks} {NextPageLink} {LastPageLink}
{RowsPerPageDropdown}"

                  rowsPerPageTemplate="8,12,18" >

            <p:column>

               <f:facet name="header">ID</f:facet>

               <h:outputText value="#{dados[0]}" />

            </p:column>

            <p:column>

               <f:facet name="header">EAN</f:facet>

               <h:outputText value="#{dados[2]}" />

            </p:column>

            <p:column>

               <f:facet name="header">DESCRIÇÃO</f:facet>
```

```
                <h:outputText value="#{dados[1]}" />
            </p:column>
            <p:column>
                <f:facet name="header">QUANTIDADE</f:facet>
                <h:outputText value="#{dados[3]}" />
            </p:column>
            <p:column>
                <f:facet name="header">VALIDADE</f:facet>
                <h:outputText value="#{dados[4]}"> <f:convertDateTime pattern="dd-MM-
yyyy"/>
                </h:outputText>
            </p:column>
            <!--   <p:column>
                <f:facet name="header">IMAGEM</f:facet>
                  <p:graphicImage value="#{dados[5]}" />
             </p:column>
            <p:column>
                <f:facet name="header">IMAGEM</f:facet>
                <p:graphicImage
value="imagens/#{produtosBean.produto.setImagens(dados[5])}"/>
            </p:column>-->
            <p:column>
                <f:facet name="header">OPERAÇÕES</f:facet>
                <p:growl id="message" showDetail="true" />
                <p:commandButton style="background-color: lightgreen" icon="ui-icon-
pencil" onclick="PF('dados').show();" value="-Alterar--"
action="#{produtosBean.carregaProduto(dados[0])}"  update="dados" />
                <p:commandButton style="background-color: red" icon="ui-icon-trash"
value="Remover" action="#{produtosBean.removerProduto(dados[0])}"
update="dados">
                    <p:confirm header="Confirmação de Exclusão!" message="Você tem
```

```
certeza?"  icon="ui-icon-trash" />

                </p:commandButton>

                <p:confirmDialog global="true" showEffect="scale" hideEffect="slide">

                    <p:commandButton value="Sim" type="button" styleClass="ui-
confirmdialog-yes" icon="ui-icon-check" />

                    <p:commandButton value="Nao" type="button" styleClass="ui-
confirmdialog-no" icon="ui-icon-close" />

                </p:confirmDialog>

            </p:column>

        </p:dataTable>

    </p:fieldset>

    <p:dialog widgetVar="dados" modal="true" showEffect="scale" header="Atualizar
Dados" resizable="false">

        <h:form>

            <p:messages id="messages" showDetail="true"  closable="true" />

            <p:panelGrid columns="2" styleClass="ui-noborder" id="grid">

                Ean:<p:inputText id="ean" value ="#{produtosBean.produto.ean}"/>

                Descrição<p:inputText id="descricao" value
="#{produtosBean.produto.descricao}"/>

                Quantidade:<p:inputText id="qtd" value
="#{produtosBean.produto.quantidade}"/><br/> <p:slider for="qtd"/>

                Validade<p:calendar id="validade" value
="#{produtosBean.produto.validade}"  pattern="dd-mm-yyyy"  effect="fold"  locale="pt"
navigator="true" onfocus="validade"/>

               <p:message for="messages" display="tooltip" />

                <p:commandButton value="Voltar" action ="#{produtosBean.voltaTela()}"
update="@form" icon="ui-icon-home"/>

                <p:commandButton value="Atualizar"
action="#{produtosBean.atualizarProduto()}"  update="@form" icon="ui-icon-refresh"
oncomplete="PF('dados').show();" />

            </p:panelGrid>

        </h:form>

    </p:dialog>
```

```
<p:blockUI block="dados" trigger="dados">
    LOADING<br />
    <p:graphicImage value="preloader.gif"/>
</p:blockUI>
<script type='text/javascript'>$(function () {
        $(window).scroll(function () {
            if ($(this).scrollTop() !== 0) {
                $("#rb-top")
                        .fadeIn();
            } else {
                $("#rb-top").fadeOut();
            }
        });
        $("#rb-top").click(function () {
            $("datatable,html").animate({scrollTop: 0}, 800);
            return false;
        });
    });
</script>
<style type="text/css">
    a.toTop {
        background-color: #fb8094;
        border: 0;
        border-radius: 53px;
        bottom: 10px;
        color: #ffffff;
        cursor: pointer;
        display: none;
        font-size: 14px;
```

```
        padding: 10px 12px;
        position: fixed;
        text-decoration:none;
        right: 8px;
        text-align: center;
        width: auto;
        z-index:500;
    }
    a.toTop:hover {background-color: #eda2af;}
</style>
<!-- <style type="text/css">
     body .ui-panelgrid .ui-panelgrid-cell:last-child {
         background-color: #ffffff;
     }
     .custom-scrolltop {
         width: 2rem !important;
         height: 2rem !important;
         border-radius: 4px !important;
         background-color: var(primary-color) !important;
     }
   .custom-scrolltop:hover {
         background-color: var(primary-color) !important;
     }
     .custom-scrolltop .ui-scrolltop-icon {
         font-size: 1rem !important;
         color: var(primary-color-text) !important;
     }
     body{
         background-color: activecaption;
```

```
        }
        .ui-datatable tbody tr {
            background-color: appworkspace;
        }
        .ui-dialog{
            background-color: appworkspace;
        }
        .ui-dialog-titlebar{
            background-color: brown;
        }
        .ui-dialog-title-dialog{
            background-color: brown;
        }
        .ui-panelgrid td, .ui-panelgrid tr
        {
            border-style: none !important;
            background-color: appworkspace;
        }
        .ui-fieldset{
            background-color: dodgerblue;
        }
     </style> -->
    <a href=""></a>
    <a id='rb-top' class="toTop" style='display: none; position: fixed; bottom: 90px;
right:1%; cursor:pointer;'>
        <i class="fa fa-chevron-up"></i>
        <span style="font-size:20px;"> ▲ </span>
    </a>
  </h:body>
```

```
</html>
```

Código Fonte Controllers

```
package Controller;
import java.io.Serializable;
import java.util.List;
import java.util.Objects;
import javax.inject.Named;
import javax.enterprise.context.SessionScoped;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.model.SelectItem;
import Model.Produtos;
import Model.ProdutosDao;
import javax.annotation.PostConstruct;
//import javax.annotation.PostConstruct
@Named(value = "produtosBean")
@SessionScoped
public class ProdutosBean implements Serializable {
    private Produtos produto = new Produtos();
    private ProdutosDao produtosDao = new ProdutosDao();
    private List<Produtos> listaProdutos;
    private List<Object> objetosGrid;
    private List<SelectItem> produtos;
    /*public String voltaTela() {
        this.produtos = (List<SelectItem>) new Produtos();
        return "cadastro";
```

```
}*/
public String voltaTela() {
    this.produto = new Produto();
    return "cadastro";
}
public List<SelectItem> getProdutos() {
    return produtos;
}
public void setProdutos(List<SelectItem> produtos) {
    this.produtos = produtos;
}
public List<Produtos> getListaProdutos() {
    produto = new Produtos();
    return listaProdutos;
}
public void setListaProdutos(List<Produtos> listaProdutos) {
    this.listaProdutos = listaProdutos;
}
public String atualizarProduto() {
    produtosDao.update(produto);
    addMessage("Sucesso, Dados atualizados", "");
    produto = new Produtos();
    return "cadastro";
}
public String carregaProduto(int id) {
    this.produto = produtosDao.read(id);
    return "editar_cadastros";
}
public List listarProduto() {
```

```
    return listaProdutos = produtosDao.getList();
}
public List<Object> listarProdutoFull() {
    return objetosGrid = produtosDao.getListProdutos();
}
public String adicionarProdutos() {
    if (produtosDao.create(produto) == true) {
        addMessage("Sucesso, dados inseridos", "Dados Salvos");
        produto = new Produtos();
        return "cadastro";
    } else {
        addMessage("Erro,ao inserir os dados", "Erro");
        return "cadastro";
    }
}
public String removerProduto(int id) {
    System.out.println(id);
    this.produto = produtosDao.read(id);
    produtosDao.delete(produto);
    return "cadastro";
}
public Produtos getProduto() {
    return produto;
}
public void setProduto(Produtos produtos) {
    this.produto = produto;
}
public ProdutosBean() {
}
```

```
    @PostConstruct
    public void init() {
        produto = new Produtos();
    }
    public void addMessage(String summary, String detail) {
        FacesMessage message = new
FacesMessage(FacesMessage.SEVERITY_INFO, summary, detail);
        FacesContext.getCurrentInstance().addMessage(null, message);
    }
    @Override
    public int hashCode() {
        int hash = 5;
        hash = 97 * hash + Objects.hashCode(this.produto);
        return hash;
    }
    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null) {
            return false;
        }
        if (getClass() != obj.getClass()) {
            return false;
        }
        final ProdutosBean other = (ProdutosBean) obj;
        if (!Objects.equals(this.produto, other.produto)) {
            return false;
```

```
        }
        return true;
    }
    private static class Produto extends Produtos {
        public Produto() {
        }
    }
}
```

Código Fonte Referente à Model

```
package Model;
// Generated 25/06/2020 20:33:40 by Hibernate Tools 4.3.1
import java.util.Date;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import static javax.persistence.GenerationType.IDENTITY;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
/** * Produtos generated by hbm2java */

@Entity
@Table(name="produtos"
    ,catalog="validades"
)
public class Produtos  implements java.io.Serializable {
```

```
    private Integer idProduto;
    private String ean;
    private String descricao;
    private long quantidade;
    private Date validade;
    private byte[] imagens;
   public Produtos() {
   }
   public Produtos(String ean, String descricao, long quantidade, Date validade, byte[]
imagens) {
      this.ean = ean;
      this.descricao = descricao;
      this.quantidade = quantidade;
      this.validade = validade;
      this.imagens = imagens;
   }
    @Id @GeneratedValue(strategy=IDENTITY)
   @Column(name="id_produto", unique=true, nullable=false)
   public Integer getIdProduto() {
       return this.idProduto;
   }
   public void setIdProduto(Integer idProduto) {
       this.idProduto = idProduto;
   }
   @Column(name="ean", nullable=false, precision=22, scale=0)
   public String getEan() {
       return this.ean;
   }
   public void setEan(String ean) {
```

```
    this.ean = ean;
}
@Column(name="descricao", nullable=false, length=44)
public String getDescricao() {
    return this.descricao;
}
public void setDescricao(String descricao) {
    this.descricao = descricao;
}
@Column(name="quantidade", nullable=false, precision=18, scale=0)
public long getQuantidade() {
    return this.quantidade;
}
public void setQuantidade(long quantidade) {
    this.quantidade = quantidade;
}
//@Temporal(TemporalType.TIMESTAMP)
@Column(name="validade", nullable=false, length=10)
public Date getValidade() {
    return this.validade;
}
public void setValidade(Date validade) {
    this.validade = validade;
}



@Column(name="imagens")
public byte[] getImagens() {
    return this.imagens;
```

```
    }
    public void setImagens(byte[] imagens) {
        this.imagens = imagens;
    }
```

Código Fonte Referente a Controller de lotes/produtos/vencimento

```
package Controller;
import java.io.Serializable;
import java.util.List;
import java.util.Objects;
import javax.inject.Named;
import javax.enterprise.context.SessionScoped;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.model.SelectItem;
import Model.ProdutoLoteVcto;
import Model.ProdutoLoteVctoDao;
import Model.Produtos;
import javax.annotation.PostConstruct;
@Named(value = "ProdutoLoteVctoBean")
@SessionScoped
public class ProdutoLoteVctoBean implements Serializable {
    private ProdutoLoteVcto produtoLoteVcto = new ProdutoLoteVcto();
    private ProdutoLoteVctoDao produtoLoteVctoDao = new ProdutoLoteVctoDao();
    private List<Object> objetosGrid;
    private List<SelectItem> ProdutoLoteVctos;
    private List<ProdutoLoteVcto> ListaProdutoLoteVcto;
```

```
private ProdutoLoteVcto ProdutoLoteVcto;
public ProdutoLoteVcto getProdutoLoteVcto() {
    return produtoLoteVcto;
}
public void setProdutoLoteVcto(ProdutoLoteVcto produtoLoteVcto) {
    this.produtoLoteVcto = produtoLoteVcto;
}
public ProdutoLoteVctoDao getProdutoLoteVctoDao() {
    return produtoLoteVctoDao;
}
public void setProdutoLoteVctoDao(ProdutoLoteVctoDao produtoLoteVctoDao) {
    this.produtoLoteVctoDao = produtoLoteVctoDao;
}
public String carregaProdutoLoteVcto(int id) {
    this.produtoLoteVcto = produtoLoteVctoDao.read(id);
    return "editar_prodlotevcto";
}
public List<Object> listarProdutoLoteVctoFull() {
    return objetosGrid = produtoLoteVctoDao.getListProdutoLoteVctos();
}
public String atualizarProdutoLoteVcto() {
    produtoLoteVcto.update(produtoLoteVcto);
    addMessage("Sucesso, Dados atualizados", "");
    produtoLoteVcto = new ProdutoLoteVcto();
    return "cadastroprodvcto";
}
public String voltaTela() {
    this.produtoLoteVcto = new ProdutoLoteVcto();
    return "cadastroprodvcto";
```

```
    }
    public String adicionar() {
        if (produtoLoteVctoDao.create(produtoLoteVcto) == true) {
            addMessage("Sucesso, dados inseridos", "Dados Salvos");
            String codbarranovo
                    = String.format("%04d", produtoLoteVcto.getIdProdLoteVcto())
                    + String.format("%05d", produtoLoteVcto.getIdProduto())
                    + String.format("%05d", produtoLoteVcto.getIdLote());
            codbarranovo = codbarranovo.trim();//tira os espacos em branco
            produtoLoteVcto.setCodBarrasNovo(codbarranovo);
            produtoLoteVctoDao.update(produtoLoteVcto);
            produtoLoteVcto = new ProdutoLoteVcto();
            return "cadastroProdloteVcto";
        } else {
            addMessage("Dados não inseridos", "Erro");
            return "cadastroProdLoteVcto";
        }
    }
    public String removerProdutoLoteVcto(int id) {
        System.out.println(id);
        this.produtoLoteVcto = produtoLoteVctoDao.read(id);
        produtoLoteVctoDao.delete(produtoLoteVcto);
        return "cadastroProdLoteVcto";
    }
    public ProdutoLoteVctoBean() {
    }
    public void addMessage(String summary, String detail) {
        FacesMessage message = new
FacesMessage(FacesMessage.SEVERITY_INFO, summary, detail);
```

```
        FacesContext.getCurrentInstance().addMessage(null, message);
    }
    @Override
    public int hashCode() {
        int hash = 5;
        hash = 97 * hash + Objects.hashCode(this.produtoLoteVcto);
        return hash;
    }
    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        if (obj == null) {
            return false;
        }
        if (getClass() != obj.getClass()) {
            return false;
        }
        final ProdutoLoteVctoBean other = (ProdutoLoteVctoBean) obj;
        if (!Objects.equals(this.produtoLoteVcto, other.produtoLoteVcto)) {
            return false;
        }
        return true;
    }
}
```