**UNIVERSIDADE PAULISTA**

**ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**LUCAS EVANGELISTA RA: D6156A0**

**NELSON JOSÉ BARNABÉ JUNIOR RA: D823DB-1**

**NILTON MUNIZ RA: D695169**

**TIAGO NUNES RA: D700570**

**SISTEMA PONTO DE VENDA – LOJA IMPERADOR**

**SOROCABA – SP**

**2019**

**LUCAS EVANGELISTA RA: D6156A0**

**NELSON JOSE BARNABE RA: D823DB-1**

**NILTON MUNIZ RA: D695169**

**TIAGO NUNES RA: D700570**

**SISTEMA PONTO DE VENDA – LOJA IMPERADOR**

Trabalho de conclusão de curso para obtenção do título de graduação em Análise e Desenvolvimento de sistemas apresentado à Universidade Paulista – UNIP.

Orientador: Todos os Professores

**SOROCABA – SP**

**2019**

**LUCAS EVANGELISTA RA: D6156A0**

**NELSON JOSE BARNABE RA: D823DB-1**

**NILTON MUNIZ RA: D695169**

**TIAGO NUNES RA: D700570**

**SISTEMA PONTO DE VENDA – LOJA IMPERADOR**

Trabalho de conclusão de curso para obtenção do título de graduação em Análise e Desenvolvimento de sistemas apresentado à Universidade Paulista – UNIP.

Aprovado em:

BANCA EXAMINADORA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Esp. Eduardo Martins Simi

Universidade Paulista – UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Dr. Irapuan Gloria Junior

Universidade Paulista – UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Esp. Richardson Kennedy Luz

Universidade Paulista UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Me. Ricardo Martins

Universidade Paulista UNIP

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_/\_\_\_

Prof. Esp. Reverdan Almeida Sparinger

Universidade Paulista UNIP

**RESUMO**

Os avanços tecnológicos estão transformando a realidade de muitas empresas, sendo um elemento essencial para obtenção de uma gestão eficiente, principalmente no ramo do comércio em que o fluxo de entrada e saída de produtos é intensa. O tema deste trabalho busca desenvolver um sistema direcionado ao ponto de venda de uma loja de materiais de construção, denominada Loja Imperador. Atualmente todos os procedimentos realizados pela loja são feitos de forma manual sem o auxílio de um software computacional que permita o gerenciamento dessas atividades, resultando na falta de segurança e organização. Todas as pesquisas e bases transcritas neste projeto foram com apoio e necessidades da loja, identificadas pelo levantamento de requisitos.Neste cenário o projeto busca automatizartodo o processo de entrada de produtos no estoque, as vendas e compras realizadas, além dos cadastros dos clientes, fornecedores, produtose fabricantes, possibilitando a gestão plena do ponto de venda da loja.

Palavras Chaves: Sistema, venda e produtos.

**ABSTRACT**

The technological advances are transforming the reality of many companies, that being an essential element for the attainment of an effective management, mainly on the field of commerce of which the inflow and outflow is intense. The theme of this paper seeks the development of a system aimed at the point of sale of a hardware store, named Loja Imperador. Currently, all the procedures performed by the store are carried manually, without the assistance of a computational software that allows the managing of these activities, resulting in the lack of safety and organization. All the research and points transcribed on this project were made with the support and need of the store, identified by the requirements survey. In this scenario, the project seeks to automatize all the process of entrance of products into the stock, the sales and purchases made, aswell as the client’s registrations, suppliers, products and manufacturers, making possible the full-fledged management of the store’s point of sale.

**Palavras Chaves: System, sale and products.**

**LISTA DE FIGURAS**

Figura 4.0 – Requisitos funcionais 13

Figura 4.1 – Requisitos não funcionais 16

Figura 4.2 – Fluxograma geral do sistema 17

Figura 4.3 – Cronograma do Projeto 18

Figura 4.4 – Diagrama de Classes de Manter Usuários 18

Figura 4.5 – Diagrama de Classes de Manter Unidades 19

Figura 4.6 – Diagrama de Classes de Manter Produtos 19

Figura 4.7 – Diagrama de Classes de Manter Pessoa 20

Figura 4.8 – Diagrama de Classes de Manter Fabricantes 20

Figura 4.9 – Diagrama de Classes de Manter Cidades 21

Figura 4.10 – Diagrama de Classes de Manter Bairros 21

Figura 4.11 – Diagrama de Classes de Manter Ponto de Compra 21

Figura 4.12 – Diagrama de Classes de Manter Ponto de Venda 22

Figura 4.13 – Diagrama de Classes do Login 22

Figura 4.14 – Diagrama de Pacotes do Projeto 22

Figura 4.15 – Diagrama de Sequencia de Manter Usuários 23

Figura 4.16– Diagrama de Sequencia de Manter Unidades 24

Figura 4.17 – Diagrama de Sequencia de Manter Produtos 24

Figura 4.18 – Diagrama de Sequencia de Manter Pessoa 25

Figura 4.19 – Diagrama de Sequencia de Manter Fabricantes 25

Figura 4.20 – Diagrama de Sequencia de Manter Cidades 26

Figura 4.21– Diagrama de Sequencia de Manter Bairros 26

Figura 4.22 – Diagrama de Sequencia de Manter Pedido 27

Figura 4.23 – Diagrama de Sequencia de Manter PDV 28

Figura 4.24 – Diagrama de Sequencia do Login 29

Figura 4.25 – Diagrama de Atividade do Projeto 30

Figura 4.26 – Diagrama de Estados de Manter Usuários 30

Figura 4.27 – Diagrama de Estados de Manter Pedido 30

Figura 4.28 – Diagrama de Casos de Uso de Manter Usuários 31

Figura 4.29 – Diagrama de Casos de Uso de Manter Unidades 31

Figura 4.30 – Diagrama de Casos de Uso de Manter Produtos 32

Figura 4.31– Diagrama de Casos de Uso de Manter Pessoa 33

Figura 4.32 – Diagrama de Casos de Uso de Manter Fabricantes 33

Figura 4.33 – Diagrama de Casos de Uso de Manter Cidades 34

Figura 4.34 – Diagrama de Casos de Uso de Manter Bairros 34

Figura 4.35 – Diagrama de Casos de Uso de Manter Ordem de Compra 35

Figura 4.36 – Diagrama de Casos de Uso de Manter PDV 35

Figura 4.37 – Diagrama de Casos de Uso do Login 35

Figura 4.38 – Modelo Conceitual do Banco de Dados 36

Figura 4.39 – Modelo Lógico do Banco de Dados 37

Figura 5.0 – Tela do Login do Projeto 38

Figura 5.0 – Tela de Inicio 38

Figura 5.0 – Tela do Login do Projeto 38

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.0 – Tela do Login do Projeto 39

Figura 5.11 – Tela do Login do Projeto 40

Figura 5.0 – Tela do Login do Projeto 40

Figura 5.0 – Tela do Login do Projeto 40

Figura 5.0 – Tela do Login do Projeto 40

**SUMÁRIO**

[1.0 INTRODUÇÃO 1](#_Toc25860425)

[1.1 JUSTIFICATIVA 1](#_Toc25860426)

[1.2 OBJETIVOS 1](#_Toc25860427)

[**1.2.1 Objetivo Geral** 1](#_Toc25860428)

[**1.2.2 Objetivo específico** 1](#_Toc25860429)

[2.0 REVISAO DE LITERATURA 2](#_Toc25860430)

[2.1 SOFTWARE 2](#_Toc25860431)

[**2.1.1** **Qualidade de Software** 2](#_Toc25860432)

[2.2 CONCEITOS DA ORIENTAÇÃO A OBJETOS 2](#_Toc25860433)

[**2.2.1** **Análise de Sistemas Orientada a Objeto** 3](#_Toc25860434)

[**2.2.2** **Programação Orientada a Objeto** 3](#_Toc25860435)

[2.2.2.1 Objeto 4](#_Toc25860436)

[2.2.2.2 Classes 4](#_Toc25860437)

[2.2.2.3 Herança 4](#_Toc25860438)

[2.2.2.4 Polimorfismo 4](#_Toc25860439)

[2.2.2.5 Encapsulamento 5](#_Toc25860440)

[2.3 PADRÕES DE ARQUITETURA DE SOFTWARE 5](#_Toc25860441)

[**2.3.1 MVC** 5](#_Toc25860442)

[2.3.1.1 Modelo 5](#_Toc25860443)

[2.3.1.1 Visão 6](#_Toc25860444)

[2.3.1.1 Controlador 6](#_Toc25860445)

[2.4 GESTÃO DE PROJETOS 6](#_Toc25860446)

[2.5 GESTÃO DA QUALIDADE DE SOFTWARE 7](#_Toc25860447)

[3.0 FERRAMENTAS DE DESENVOLVIMENTO 8](#_Toc25860448)

[3.1 VISUAL STUDIO 8](#_Toc25860449)

[3.2 SQL SERVER 8](#_Toc25860450)

[3.3 DIAGRAMA DE CLASSES 9](#_Toc25860451)

[3.4 DIAGRAMA DE PACOTES 9](#_Toc25860452)

[3.5 DIAGRAMA DE SEQUÊNCIA 9](#_Toc25860453)

[3.6 DIAGRAMA DE ATIVIDADES 10](#_Toc25860454)

[3.7 DIAGRAMA DE ESTADOS 10](#_Toc25860455)

[3.8 DIAGRAMA DE CASOS DE USO 10](#_Toc25860456)

[3.9MODELO CONCEITUAL DO BANCO DE DADOS 10](#_Toc25860457)

[3.10 MODELO LÓGICO DO BANCO DE DADOS 11](#_Toc25860458)

[3.11 MODELO FÍSICO DO BANCO DE DADOS 11](#_Toc25860459)

[4.0 DESENVOLVIMENTO 12](#_Toc25860460)

[4.1 IDENTIFICAÇÃO DA EMPRESA 12](#_Toc25860461)

[4.2 SITUAÇÃO PROBLEMA 12](#_Toc25860462)

[4.3 REQUISITOS FUNCIONAIS 13](#_Toc25860463)

[4.4 REQUISITOS NÃO FUNCIONAIS 15](#_Toc25860464)

[4.5 FLUXOGRAMA GERAL 16](#_Toc25860465)

[4.6 PLANEJAMENTO (CRONOGRAMA) 16](#_Toc25860466)

[4.7 DIAGRAMA DE CLASSES 17](#_Toc25860467)

[4.8 DIAGRAMA DE PACOTES 22](#_Toc25860468)

[4.9 DIAGRAMA DE SEQUÊNCIA 23](#_Toc25860469)

[4.10 DIAGRAMA DE ATIVIDADES 29](#_Toc25860470)

[4.11 DIAGRAMA DE ESTADOS 29](#_Toc25860471)

[4.12 DIAGRAMA DE CASOS DE USO 30](#_Toc25860472)

[4.13 MODELO CONCEITUAL DO BANCO DE DADOS 35](#_Toc25860473)

[4.14 MODELO LÓGICO DO BANCO DE DADOS 36](#_Toc25860474)

[4.16 TESTES DE SOFTWARE 37](#_Toc25860475)

[5.0 MÉTODOS 38](#_Toc25860476)

[5.1 FUNCIONAMENTO DO SISTEMA E APRESENTAÇÃO DAS TELAS 38](#_Toc25860477)

[5.2 FUNCIONAMENTO DO SISTEMA 40](#_Toc25860478)

[6.0 CONCLUSÃO 41](#_Toc25860479)

[REFERÊNCIAS BIBLIOGRÁFICAS 42](#_Toc25860480)

[ANEXOS 45](#_Toc25860481)

# 1.0 INTRODUÇÃO

## 1.1 JUSTIFICATIVA

Com o constante desenvolvimento e expansão das tecnologias, os sistemas computacionais estão sendo cada vez mais necessários, seja em industrias, comércios e em até em pequenos empreendimentos.

Nesse contexto, surge a proposta de desenvolvimento para Loja mperador, onde todo o processo hoje é feito manualmente, com poucos recursos tecnológicos.

O sistema busca facilitar o atendimento ao cliente e gerir o ponto de venda, realizando o controle de estoque e vendas realizadas. Hoje, todo esse processo é feito manualmente, com o sistema, vamos implementar a praticidade e a rapidez para um atendimento e gestão mais significativo.

## 1.2 OBJETIVOS

## **1.2.1 Objetivo Geral**

O objetivo deste projeto é contemplar a necessidade da Loja Imperador e através dos conceitos das disciplinas da graduação de análise e desenvolvimento de sistema desenvolver um sistema com uma base sólida para futuras implementações, buscando melhoraria continua dos processos realizados.

## **1.2.2 Objetivo específico**

Destina-se:

* Desenvolver o projeto utilizando-se das metodologias e boaspraticas de programação e gestão de projetos lecionados durante o semestre letivo.
* Apresentar uma solução ao cenário proposto para o desenvolvimento de um sistema PDV.
* Elaborar toda a documentação de software exemplificando os processos que ocorrem em toda a operação do sistema.
* Suprir as necessidades da empresa utilizando os conceitos de Análise Orientada a Objetos com o levantamento de requisitos, elaboração de diagramas e fluxogramas para melhor entendimento.

# 2.0 REVISAO DE LITERATURA

Nesta seção é apresentado todo conteúdo teórico para a compreensão e desenvolvimento do trabalho, abordando alguns conceitos que contemplam as bases para a concepção do sistema proposto.

## 2.1 SOFTWARE

## 

O mundo moderno não poderia existir sem o software. Infraestruturas e serviços nacionais são controlados por sistemas computacionais, e a maioria dos produtos elétricos inclui um computador e um software que o controla. A manufatura e a distribuição industriais são totalmente informatizadas, assim como o sistema financeiro (SOMMERVILLE,2011).

O software será cada vez mais o principal articulador da inteligência humana aplicada aos computadores e a uma infinidade de aparelhos que utilizam microprocessadores (SILVEIRA, 2004).

Existem diversas plataformas no mundo em que são desenvolvidos variados tipos de software, sendo eles aplicativos para smartphones que viraram febre neste século e sistemas para empresas e indústrias. É de grande importância sistemas para controlar e gerenciar uma linha de produção, fazendo o controle de grandes máquinas e substituindo ainda mais a mão de obra humana.

## **2.1.1** **Qualidade de Software**

Qualidade de software é a conformidade com requisitos funcionais e de desempenho explicitamente declarados, padrões de desenvolvimento explicitamente documentados e características implícitas, que são esperadas em todo software desenvolvido profissionalmente (PRESSMAN, 2002).

## 2.2 CONCEITOS DA ORIENTAÇÃO A OBJETOS

A proposta de orientação a objetos é representar da forma mais fiel possível as situações do mundo real nos sistemas computacionais. No caso, o mundo pode ser considerado como um todo, o qual é composto por vários objetos, e estes possuem interação entre si. Da mesma forma a orientação a objetos não considera os sistemas computacionais como uma coleção estruturada de processos, mas sim, como uma coleção de objetos que possuem interação uns com os outros (CORREIA, 2006).

## **2.2.1** **Análise de Sistemas Orientada a Objeto**

A análise de sistema orientada a objeto é extremamente importante para o analista saber quais são as verdadeiras necessidades do cliente, saber os princípios da utilização do sistema, fazer o levantamento de requisitos e fazer os diagramas e fluxogramas para maior compreensão de como será desenvolvido o sistema.

Após todo este trabalho, é dada a continuação para o projetista, que, entretanto observa como o projeto pode ser representado por um conjunto de objetos que interagem e que gerenciam seu estado e operações. Apresentam vários modelos que podem ser usados para descrever o sistema, como por exemplo, os diagramas de classe. Ambos terão de se comunicar pois após a analise, o projetista não poderá fazer futuras alterações nos diagramas, a não ser que o cliente solicite a mudança de algo interno do sistema.

O conceito de orientação a objetos surgiu com o intuito de minimizar os problemas encontrados até então na criação de softwares complexos, projetados por meio de decomposição funcional e sub-rotinas.

ParaRumbaugh,OrientaçãoaObjetoséumanovamaneiradepensarosproblemas utilizandomodelosorganizadosapartirdeconceitosdomundoreal.

## **2.2.2** **Programação Orientada a Objeto**

Uma vez que os objetos possuem comportamentos em determinadas situações, para que os objetos computacionais possam realizar tais comportamentos corretamente, é preciso que sejam utilizadas estruturas de dados que os simulem, facilitando a programação pelo uso de uma metodologia unificada para análise e programação. Os objetos que foram identificados e modelados pelo analista podem ser implementados da mesma maneira que foram projetados, não sendo necessário traduzir as informações dos diagramas de análise para estrutura de dados e de programação, da forma que ocorre na análise estruturada (CORREIA, 2006).

Para entender um pouco mais sobre programação orientada o objeto, é necessário conhecer alguns conceitos descritos nos itens a seguir.

## 2.2.2.1 Objeto

Um objeto pode ser entendido como uma “coisa” física, por exemplo, um carro, uma casa ou uma moto. Pode-se definir objeto como uma equação ou uma conta bancária, neste caso puramente mental, pois não existiria uma “coisa” física que possa impressionar nossos sentidos para que estes o percebam como um objeto físico (CORREIA, 2006).

## 2.2.2.2 Classes

Juntamente com objetos, as classes são um dos pilares fundamentais da orientação a objetos. Classes são basicamente uma coleção de objetos similares que podem ser definidos através de um conjunto de atributos e operações (métodos) de natureza similar (PUGA, RISSETTI, 2004).

## 2.2.2.3 Herança

Herança é um mecanismo que permite que características comuns a diversas classes sejam facilitadoras em uma classe base, ou superclasse. A partir de uma classe base, outras classes podem ser especificadas. Cada classe derivada ou subclasse apresenta as características (estrutura e métodos) da classe base e acrescenta a elas o que for definido de particularidade para ela (RICARTE, 2001).

## 2.2.2.4 Polimorfismo

Segundo (MOTA, 2003), em Orientação a Objetos polimorfismo é o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação (assinatura), mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse.

## 2.2.2.5 Encapsulamento

Encapsulamento é agrupar em um determinado lugar códigos que por muitas vezes apareciam em conjunto dentro de um programa, atribuindo um nome a esse grupo. Assim surgiu o conceito de sub-rotina, onde sempre que fosse necessário, poder-se-ia requisitar a execução do grupo utilizando o nome atribuído (GOETTEN; WINCK, 2006).

## 2.3 PADRÕES DE ARQUITETURA DE SOFTWARE

Padrões de arquitetura nada mais são do que uma forma de compartilhar experiências de desenvolvimento a partir de outros desenvolvedores para disseminar o reuso de sistemas de software. O início da utilização de padrões foi marcado pela publicação de um livro por Gamma, em 1995. O objetivo do livro era fornecer padrões de projeto para linguagens orientadas a objeto, que haviam apenas recentemente começado a serem utilizadas no mercado (SOMMERVILLE, 2011).

Existem diversos padrões de arquitetura de software, neste projeto foi utilizado o MVC *(Model-view-controller)* pela sua facilidade e organização.

## **2.3.1 MVC**

A fundamentação do MVC está em separar o que está sendo apresentado para o usuário através da GUI (Graphical User Interface) e a manipulação dos dados do sistema em si. Esta separação é feita através de três componentes com interações diretas entre eles. Os componentes são modelo (model), visão (view) e controlador (controller) (SOMMERVILLE, 2011).

## 2.3.1.1 Modelo

A camada de modelo é a responsável por encapsular os dados e as funcionalidades do software. O componente age como um gerente, interceptando todas as operações associadas à manipulação destes dados. Também é chamada esta camada de modelo de domínio; normalmente o modelo não tem noção nenhuma das interações do usuário com a interface. (FOWLER, 2006).

## 2.3.1.1 Visão

O componente de visão é responsável por exibir as informações e decidir como estas informações serão exibidas ao usuário (SOMMERVILLE, 2011).

## 2.3.1.1 Controlador

O componente do controlador é responsável por gerenciar a interação do usuário com a interface, como por exemplo: clicar em teclas, mouse ou interagir com os elementos visíveis e repassar as informações para a visão e o modelo para que os dados sejam persistidos (SOMMERVILLE, 2011).

## 2.4 GESTÃO DE PROJETOS

A gestão de projetos é uma das áreas mais importante na tomadas de decisões, ela compreende métodos e ferramentas que organizam as tarefas, identificam sua sequência de execução e dependências existentes, apoia a alocação de recursos e tempo, além de permitir o rastreamento da execução das atividades e medição do progresso relativo ao que foi definido no plano de projeto.

De acordo com o PMBOK (2004), os processos de gerenciamento da qualidade do projeto incluem todas as atividades da organização executora que determinam as responsabilidades, os objetivos e as políticas de qualidade, de modo que o projeto atenda às necessidades que motivaram sua realização.

Eles implementam o sistema de gerenciamento da qualidade através da política, dos procedimentos e dos processos de planejamento da qualidade, garantia da qualidade e controle da qualidade, com atividades de melhoria contínua dos processos conduzidas do início ao fim, conforme adequado. (PMI, 2004, p.179)

## 2.5 GESTÃO DA QUALIDADE DE SOFTWARE

Um dos principais desafios atuais enfrentados pelos grupos de desenvolvedores de sistemas é definir a qualidade do software. A qualidade pode ser definida como “Conformidade a requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo o software profissionalmente desenvolvido.”(PRESMANN, 2007).

De acordo com a ISO/IEC 9126, um software de qualidade precisa conter a funcionalidade, ou seja, se foi desenvolvido de acordo com o que foi definida no levantamento de requisitos, a confiabilidade, se o software continua no nível de desempenho das condições que foram estabelecidas, a usabilidade, facilidade de compreensão do software (fácil de usar), eficiência, se o tempo de execução do software é compatível com os requisitos do sistema, a manutenibilidade, se o software tem a capacidade de ser modificado e a portabilidade, se o software pode ser transferido de um lugar para o outro.

# 3.0 FERRAMENTAS DE DESENVOLVIMENTO

## 3.1 VISUAL STUDIO

O Visual Studio é uma ótima ferramenta para desenvolvimento de sistemas, segundo José Carlos Macoratti, umcódigo bem arrumado e organizado se torna mais legível e mostra o nível de cuidado e atenção que o desenvolvedor tem com o seu projeto, o Visual Studio oferece diversas opções para organizar e formatar o seu código de forma a torná-lo mais legível.

A WPF (Windows Presentantion Fundation)  é uma tecnologia que se propõe ser uma alternativa ao modelo Windows Form, e, em se tratando de recursos a WPF vai além de acordo com José Carlos Macoratti. Este projeto também possui um nível de organização maior e contém mais recursos que o próprio Windows Form.

Com o padrão de desenvolvimento singleton, é possível impedir que outras classes façam a instancia da classe que contém o mesmo. De acordo com José Carlos Macoratti, este padrão é efetivo por limitar o número máximo de instâncias de uma classe a somente uma única instância. Neste caso se mais que um objeto necessitar usar uma instância da classe Singleton estes objetos compartilham a mesma instância da classe Singleton.

## 3.2 SQL SERVER

Baseada nas linguagens de Álgebra e Cálculo Relacional, e inicialmente denominada SEQUEL (Structured English QUEry Language), SQL hoje é a linguagem padrão para Sistemas Gerenciadores de Bancos de Dados Relacionais (SGBDR), sendo mais intelegível do que suas linguagens maternas – consideradas

técnicas demais para o usuário (ELMASRI, NAVATHE, 2011).

A Structured Query Language (SQL) ou Linguagem de Consulta Estruturadafoi criada pela IBM Research, no início da década de 1970, para o protótipo de umsistema de banco de dados chamado System R (DATE, 2004).

## 3.3 DIAGRAMA DE CLASSES

Um diagrama de classes é um modelo fundamental de uma especificação orientada a objetos. Produz a descrição mais próxima da estrutura do código de um programa, ou seja, mostra o conjunto de classes com seus atributos e métodos e os relacionamentos entre classes. Classes e relacionamentos constituem os elementos sintáticos básicos do diagrama de classes (SILVA, 2009). O diagrama de classes é fundamental na interpretação dos métodos e atributos contidos no sistema.

## 3.4 DIAGRAMA DE PACOTES

O pacote é um elemento sintático voltado a conter elementos sintáticos de uma especificação orientada a objetos. Esse elemento foi definido na primeira versão de UML para ser usado nos diagramas então existentes, como diagrama de classes, por exemplo. Na segunda versão da linguagem, foi introduzido um novo diagrama, o diagrama de pacotes, voltado a conter exclusivamente pacotes e relacionamentos entre pacotes (SILVA, 2009). O diagrama de pacotes é utilizado também para fácil organização dos diagramas de classe, deste modo fica exposta facilmente a visualização dos objetos e onde estão conectados.

## 3.5 DIAGRAMA DE SEQUÊNCIA

O foco do diagrama de sequência está na identificação de interações entre os objetos com o tempo. O maior benefício do diagrama é que ele ajuda a identificar as mensagens trocadas entre os objetos (Pender, 2004).

No diagrama de seqüência a troca de mensagens exige um transmissor e um receptor, um receptor precisa ter uma interface para poder receber uma mensagem. Logo, se uma mensagem tiver que ser enviada de um objeto para outro, o receptor terá de definir uma interface em conformidade com a mensagem (Pender, 2004).

O Beneficio deste diagrama é o relacionamento que ele possui com o algoritmo do software e na troca de informações de cada processo.

## 3.6 DIAGRAMA DE ATIVIDADES

O diagrama de atividades representa a execução das ações e as transições que são acionadas pela conclusão de outras ações ou atividades. Uma atividade pode ser descrita como um conjunto de ações e um conjunto de atividades. A diferença básica entre os dois conceitos que descrevem comportamento é que a ação é atômica, admitindo particionamento, o que não se aplica a atividade, que pode ser detalhada em atividades e ações (SILVA, 2009).

## 3.7 DIAGRAMA DE ESTADOS

O diagrama de máquina de estados é um diagrama comportamental semelhante ao diagrama de atividades. Porém uma máquina de estados representa o comportamento de um objeto através de uma seqüência de estados pelos quais o objeto passa durante o tempo de vida, em resposta a eventos (GONÇALVES, T. J. ENYO; CORTÉS, MARIELA INÉS, 2015)

## 3.8 DIAGRAMA DE CASOS DE USO

O Diagrama de Casos de Uso é necessário da parte da analise, para melhor entendimento da esquematização do projeto. De acordo com Eduardo Bezerra, a realização de um caso de uso é responsabilidade de um conjunto de objetos que devem colaborar para produzir o resultado daquele caso de uso. Com base nisso, o modelador aplica a técnica de análise dos casos de uso para identificar as classes necessárias à produção do comportamento que está documentado na descrição do caso de uso.

## 3.9MODELO CONCEITUAL DO BANCO DE DADOS

O modelo conceitual do banco de dados é uma descrição de banco de dados de forma independente de implementação num sistema de gerenciamento. Registra que dados podem aparecer no banco, mas não registra COMO estes dados estão armazenados no SGBD.

## 3.10 MODELO LÓGICO DO BANCO DE DADOS

O modelo lógico do banco de dados compreende uma descrição das estruturas que serão armazenadas no banco e que resulta numa representação gráfica dos dados de uma maneira lógica, inclusive nomeando os componentes e ações que exercem uns sobre os outros.

## 3.11 MODELO FÍSICO DO BANCO DE DADOS

O modelo físico é uma descrição de um banco de dados no nível de abstração visto pelo usuário do SGBD. Assim, esse modelo depende do SGBD que está sendo usado. Aqui são detalhados os componentes da estrutura física do banco, como tabelas, campos, tipos de valores, índices, etc.

# 4.0 DESENVOLVIMENTO

Neste projeto foi desenvolvido um sistema PDV utilizando a plataforma de desenvolvimento Visual Studio dentro de um padrão de projeto tipo WPF, inspirado no padrão de projeto de software M.V.C *(Model, View, Controller)*, o mesmo foi dividido em 3 projetos internos, *Model, View e Controller*, ambos tem sua própria função dentro do sistema. Todas as informações digitadas no *View* sãoenviadas para o *Model*, o mesmo envia para a *Controller* e ele envia para objeto de acesso a dadose ela se comunica com obanco de dados, fazendo assim as validações necessárias, para retornar para a *View*. No mesmo foram desenvolvidas telas de cadastro e consulta para usuários, pessoa, fabricante, produtos, unidades, cidades, bairros, vendas, compras e estoque.

## 4.1 IDENTIFICAÇÃO DA EMPRESA

A empresa atua no ramo de materiais de construção a mais de 10 anos no mercado, sempre trabalhando com produtos de qualidade e prestando serviços para seus clientes da região de Votorantim e Sorocaba. Atualmente a empresa trabalha com todo o acabamento interno e externo de obras, como: telhas, madeiras, churrasqueiras, granitos e acessórios para churrasco.

* **Razão Social**: Muniz & Muniz Telhas, Mármores e Granitos LTDA - ME
* **Nome Fantasia**: Loja Imperador
* **Endereço**: Av. Gisele Constantino, 222 – Parque Bela Vista
* **Telefone(s)**: (15) 3243-8735 / (15) 3247-1231
* **Cidade**: Votorantim
* **Proprietário**: Nilton Augusto Muniz

## 4.2 SITUAÇÃO PROBLEMA

Atualmente a empresa Loja Imperador não possui um sistema que faça a parte de gerenciamento de cadastro e consultas de clientes, fornecedores e produtos, tudo isso é feito através de uma planilha que na qual a mesma faz também todo o controle de estoque.

Em função disto, foi desenvolvido um sistema PDV para que esse problema seja solucionadoda melhor forma possível, assim garantindo segurança e organização das informações.

## 4.3 REQUISITOS FUNCIONAIS

Os requisitos funcionais são aqueles que descrevem o comportamento do sistema, suas ações para cada entrada, ou seja, é aquele que descreve as funcionalidades, as quais se esperam para que sistema forneça. Eles dependem do tipo de software que está sendo desenvolvido, do conhecimento passado pelos usuários sobre o negocio em si e do que se deve fazer o software que se espera desenvolver.

A tabela da imagem da figura 4.0 descreve todos os requisitos funcionais do sistema.

Figura 4.0 – Requisitos funcionais

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requisito** | **Descrição** | **Caso de Uso** |
| RF001 | Cadastro de Usuário | O sistema poderá realizar o cadastro de usuários. | Cadastrar Usuário |
| RF002 | Consulta de usuários | O sistema poderá realizar consultas de usuários. | Consultar Usuários |
| RF003 | Consulta de Usuários por ID | O sistema poderá realizar consultas de usuários por ID. | Consultar Usuário Por ID |
| RF006 | Atualização de Usuários | O sistema poderá realizar atualizações de usuários cadastrados. | Atualizar Usuário |
| RF007 | Exclusão de Usuários | O sistema poderá fazer a exclusão de usuários. | Excluir Usuário |
| RF012 | Cadastrar Bairros | Poderá fazer um cadastro de bairro relacionado ás entregas. | Cadastrar Bairros |
| RF013 | Consultar Bairros Por Cidades | Poderá realizar consultas de bairros através das cidades. | Consultar Bairros Por Cidades |
| RF014 | Consultar Bairro | Poderá realizar consultas somente pelo bairro. | Consultar Bairro |
| RF015 | Consultar Bairro por ID | Poderá realizar consultas de bairros através do ID. | Consultar Bairro por ID |
| RF016 | Atualizar Bairro | Poderá fazer atualização dos bairros. | Atualizar Bairro |
| RF017 | Excluir Bairro | Poderá fazer a exclusão dos bairros. | Excluir Bairro |
| RF018 | Cadastrar Cidades | Poderá realizar cadastros de cidades. | Cadastrar Cidade |
| RF019 | Atualizar Cidade | Poderá realizar atualizações das cidades. | Atualizar Cidade |
| RF020 | Consultar Cidade | Poderá realizar consultas das cidades. | Consultar Cidade |
| RF021 | Consultar Cidades Por Estado | Poderá realizar consultas das cidades por estado. | Consultar Cidades Por Estado |
| RF022 | Consultar Cidade Por ID | Poderá realizar consultas das cidades pelo ID. | Consultar Cidade Por ID |
| RF023 | Excluir Cidade | Poderá fazer a exclusão das cidades. | Excluir Cidade |
| RF024 | Cadastrar Pessoa | Poderá realizar cadastros de pessoas. | Cadastrar Pessoa |
| RF025 | Excluir Pessoa | Poderá fazer exclusão de pessoas. | Excluir Pessoa |
| RF026 | Salvar Pessoa | Poderá salvar informações de pessoas. | Salvar Pessoa |
| RF027 | Consultar Pessoa | Poderá realizar consultas de pessoas. | Consultar Pessoa |
| RF028 | Atualizar Pessoa | Poderá realizar atualizações de informações de pessoas | Atualizar Pessoa |
| RF029 | Consultar Pessoa por ID | Poderá realizar consultas de pessoas por ID. | Consultar Pessoa por ID |
| RF030 | Cadastro de Produto | Poderá realizar cadastros de produtos. | Cadastrar Produto |
| RF031 | Consulta de Produtos | Poderá realizar consultas de produtos. | Consultar Produtos |
| RF032 | Consulta de Produto Pelo ID | Poderá realizar consultas de produtos pelo ID. | Consultar Produto Por ID |
| RF033 | Atualizaçãode Produto | Poderá realizar Atualizações dos produtos. | Atualizar Produto |
| RF034 | Exclusão de Produto | Poderá fazer a exclusão de produtos. | Excluir Produto |
| RF035 | Cadastro de Fabricante | Poderá realizar cadastro de fabricantes. | CadastrarFabricante |
| RF036 | Consulta de Fabricante | Poderá realizar consultas de fabricantes | Consultar Fabricante |
| RF037 | Consultade Fabricante Por ID | Poderá realizar consultas de fabricantes por ID | Consultar Fabricante Por ID |
| RF038 | Atualização de Fabricante | Poderá realizar atualizações em fabricantes | Atualizar Fabricante |
| RF039 | Exclusão de fabricante | Poderá fazer a exclusão de fabricantes | ExcluirFabricante |
| RF040 | Cadastro de Unidade | Poderá realizar cadastro de unidades | Cadastrar Unidade |
| RF041 | Consulta de Unidades | Poderá realizar consultas de unidades | Consultar Unidades |
| RF042 | Consulta de Unidade Por ID | Poderá realizar consultas de unidades por ID | Consultar Unidade Por ID |
| RF043 | Exclusão de Unidades | Poderá fazer a exclusão de unidades | Excluir Unidades |
| RF044 | Consulta Pedido de Venda | Poderá fazer a consulta de Pedidos de Venda | Consultar Pedido |
| RF045 | Consulta de Pedido de Venda por ID | Poderá fazer a consulta de Pedidos de Venda pelo ID | Consultar Pedido de Venda por ID |
| RF046 | Cadastra Pedido de Venda | Poderá fazer o cadastro dos Pedidos de Venda | Cadastrar Pedido de Venda |
| RF047 | Adiciona Quantidades dos produtos | Poderá acrescentar produtos no pedido | Adicionar Quantidade Produto Venda |
| RF048 | Remove Quantidades dos produtos | Poderá fazer a remoção de quantidade dos produtos | Remover Quantidade Produto Venda |
| RF049 | Calcula Valor Total dos pedidos | Poderá calcular o valor total do Pedido | Calcular Valor Total do Pedido |
| RF050 | Cadastra produtos no Pedido da Venda | Poderá fazer o cadastro de produtos no pedido da venda | Cadastrar Produto Pedido Venda |
| RF051 | Consulta de produtos por ID na Ordem de compra | Poderá consultar os produtos por ID na ordem de compra | Consultar produtos por ID na ordem de compra |
| RF052 | Consulta Todas as ordens de compra | Poderá consultar todas as ordens de compra cadastradas | Consultar Todas as Ordens de compra |
| RF053 | Cadastra ordem de compra | Poderá fazer cadastro de ordens de compra | Cadastrar Ordem de Compra |
| RF054 | Adiciona Quantia aos produtos | Poderá fazer adição de produtos na ordem compra | Adicionar Quantidade Produto Ordem de Compra |
| RF055 | Remove Quantia de produto na ordem de compra | Poderá fazer remoção dos produtos na ordem de compra | Remover Quantidade Produto Ordem de Compra |
| RF056 | Calcula Valor Total da ordem de compra | Poderá fazer o calculo total da ordem de compra | Calcular Valor Total da Ordem de Compra |

Fonte: Autoria Própria, 2019

## 4.4 REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais não estão ligados diretamente com as funções fornecidas pelo sistema. Em geral se preocupam com padrões de qualidade como confiabilidade, desempenho, segurança, usabilidade, portabilidade, legibilidade, qualidade, manutenção, entre outros. São muito importantes, pois definem se o sistema será eficiente para a tarefa que se propõe a fazer.

A tabela da imagem 4.1 descreve todos os requisitos não funcionais do sistema.

Figura 4.1 – Requisitos não funcionais

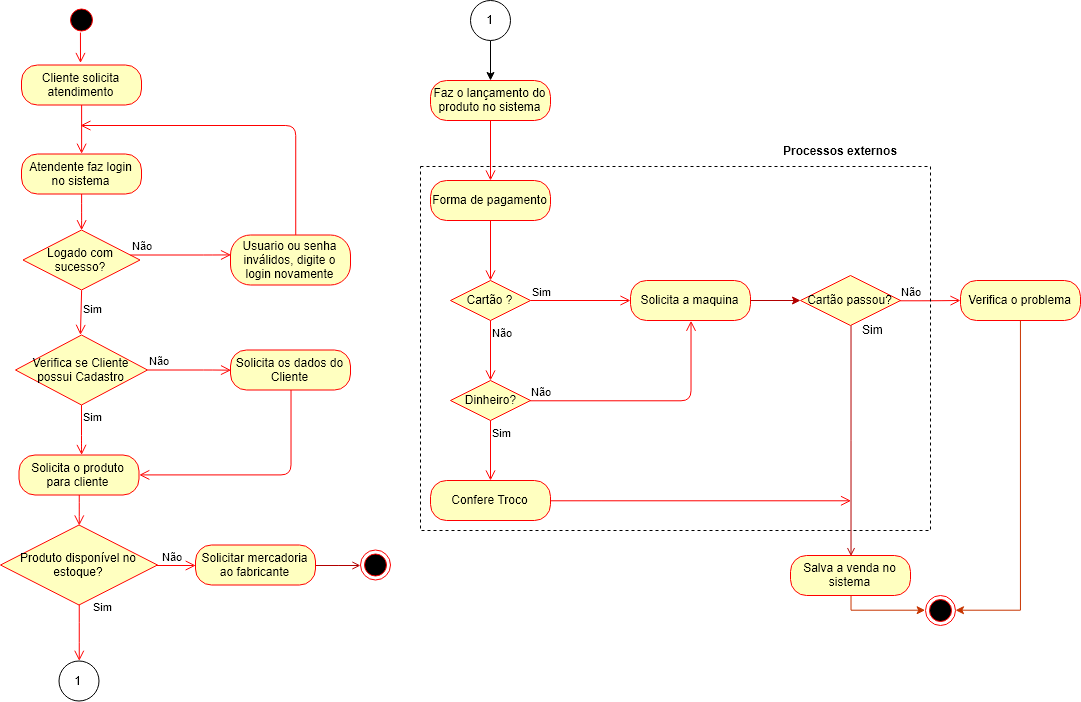
|  |  |
| --- | --- |
| **ID** | **Descrição** |
| RNF001 | Plataforma Windows 7 ou superior; |
| RNF002 | Sistema com linguagem C#; |
| RNF003 | Comunicação com banco de dados SQL Server; |
| RNF006 | O sistema deverá suportar carga máxima de 15 usuários simultâneos com degradação de desempenho de, no máximo 10% em qualquer operação |
| RNF007 | Disponibilidade de pelo menos 99,95% do tempo em dias de semana entre 08:00 ao meio-dia e pelo menos 99,1% entre meio-dia e 18:00 |
| RNF008 | Não haver falhas em comunicação com o banco durante a gravação |
| RNF009 | Variáveis de entrada com valores default |
| RNF011 | Um novo usuário deverá ser capaz de utilizar o sistema após não mais de 30 minutos de orientação |
| RNF012 | Máquinas com no mínimo 1024MB de memória |
| RNF013 | Dicas de operação do sistema |

Fonte: Autoria Própria, 2019

## 4.5 FLUXOGRAMA GERAL

A figura 4.2 exibe o Fluxograma Geral do procesos de venda de um produto.

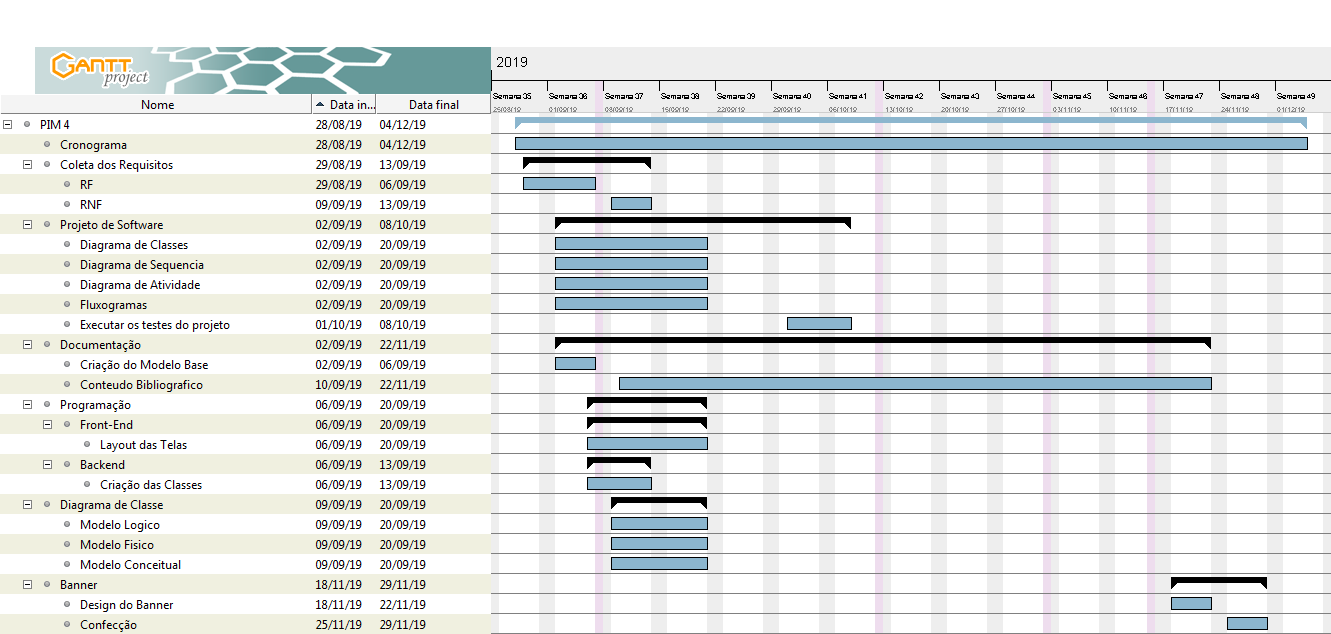
Figura 4.2 – Fluxograma geral do sistema



Fonte: Autoria Própria, 2019

## 4.6 PLANEJAMENTO (CRONOGRAMA)

A figura 4.3 exibe o Cronograma do Projeto.

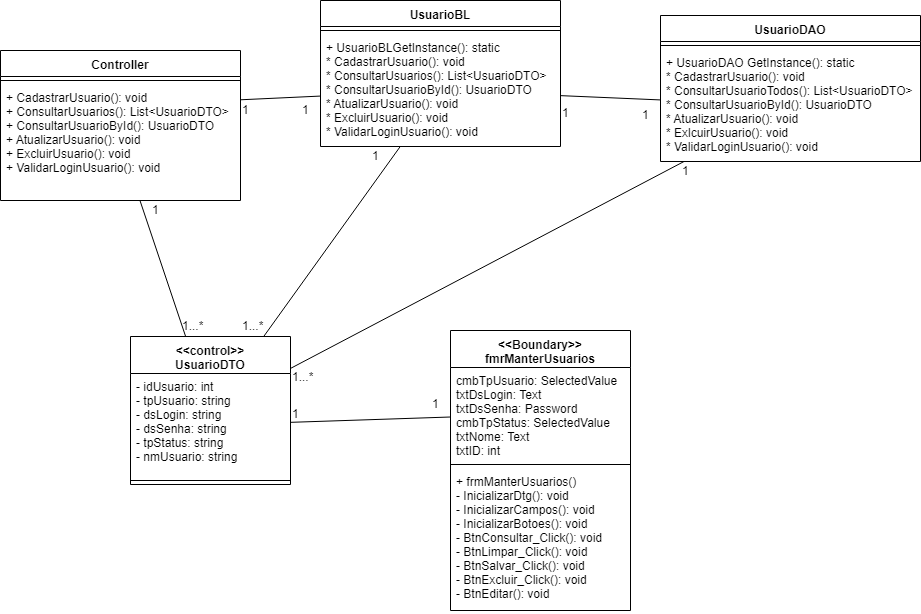
Figura 4.3 – Cronograma do Projeto

Fonte: Autoria Própria, 2019

## 4.7 DIAGRAMA DE CLASSES

Neste projeto foi elaborado um diagrama para cada classe conforme a figura 4.4 que representa a classe de Manter Usuários.

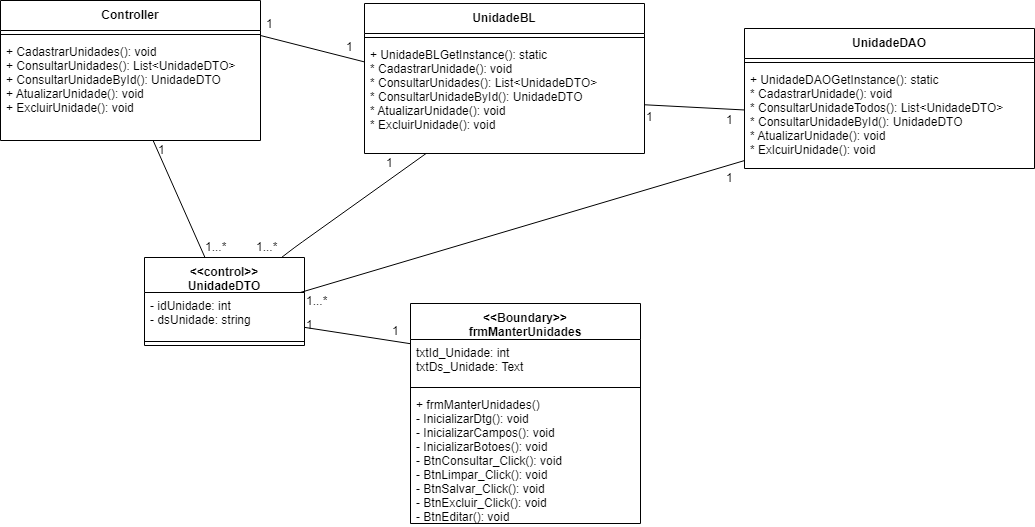
Figura 4.4 – Diagrama de Classes de Manter Usuários



Fonte: Autoria Própria, 2019.

A figura 4.5 exibe o Diagrama de Classes de Manter Unidades.

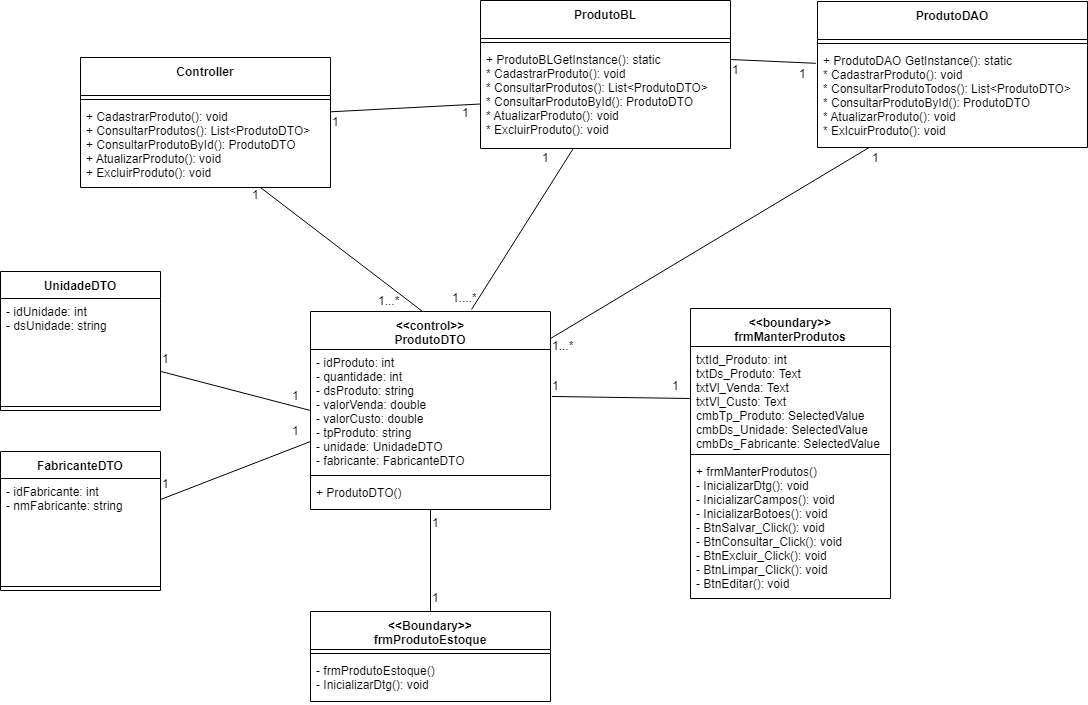
Figura 4.5 – Diagrama de Classes de Manter Unidades



Fonte: Autoria Própria, 2019

A figura 4.6 exibe o Diagrama de Classes de Manter Produtos.

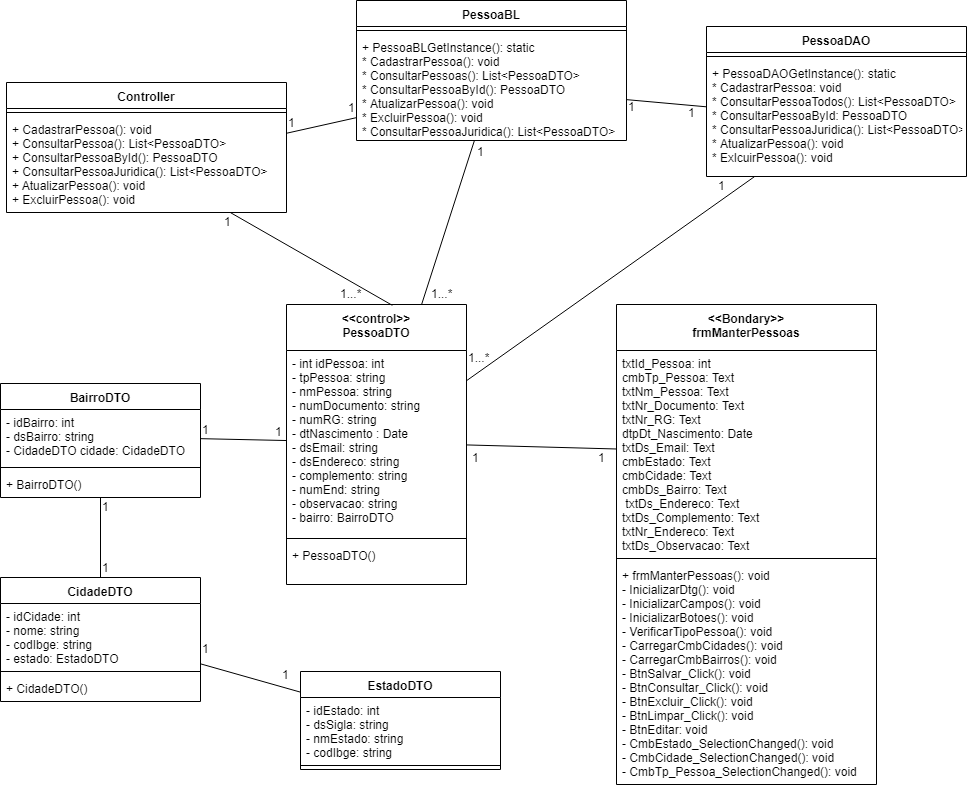
Figura 4.6 – Diagrama de Classes de Manter Produtos



Fonte: Autoria Própria, 2019

A figura 4.7 exibe o Diagrama de Classes de Manter Pessoa.

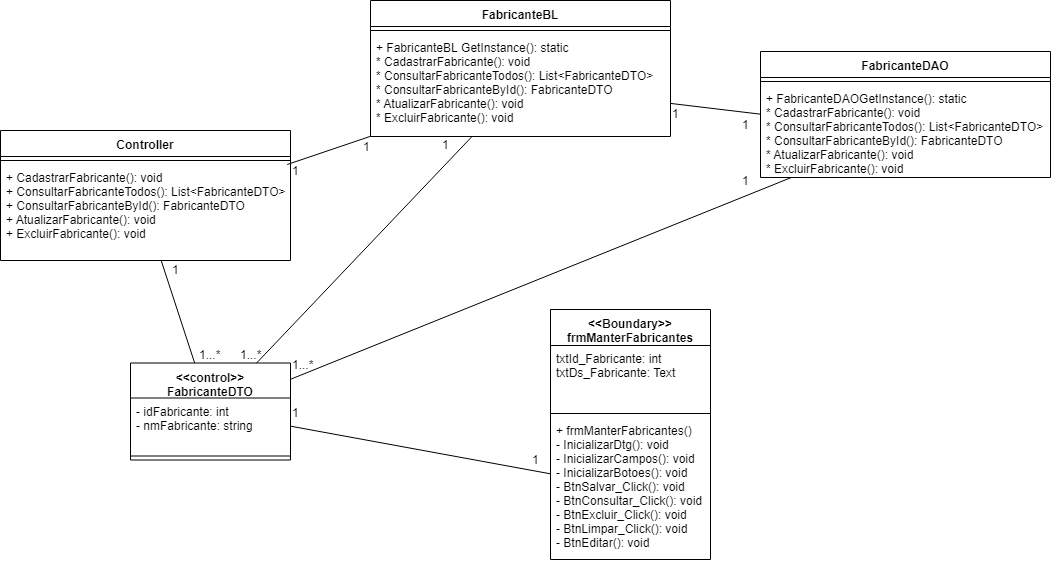
Figura 4.7 – Diagrama de Classes de Manter Pessoa



Fonte: Autoria Própria, 2019

A figura 4.8 exibe o Diagrama de Classes de Manter Fabricantes.

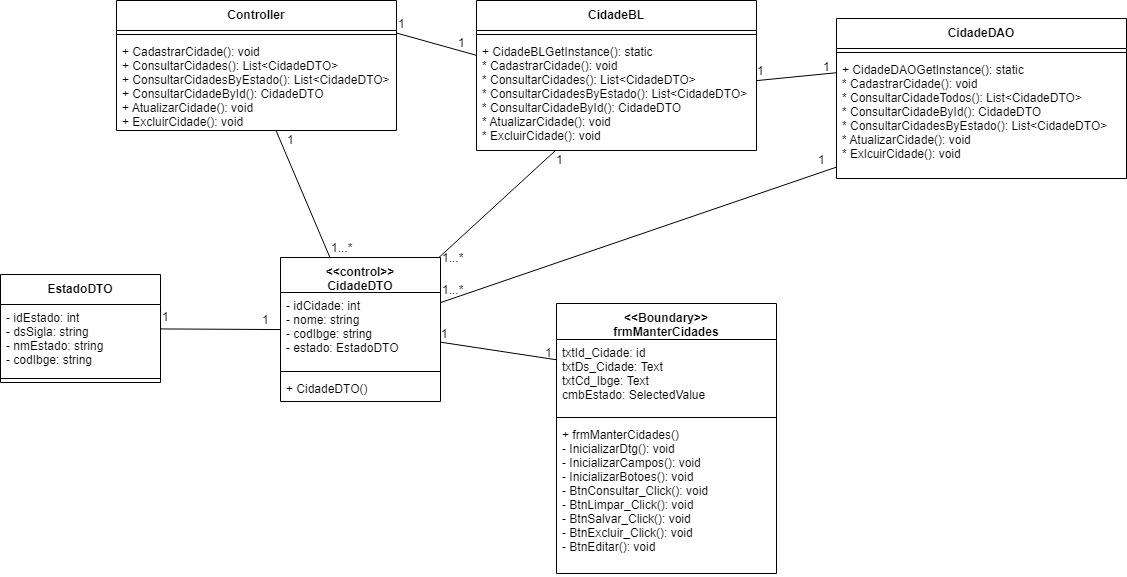
Figura 4.8 – Diagrama de Classes de Manter Fabricantes



Fonte: Autoria Própria, 2019

A figura 4.9 exibe o Diagrama de Classes de Manter Cidades.

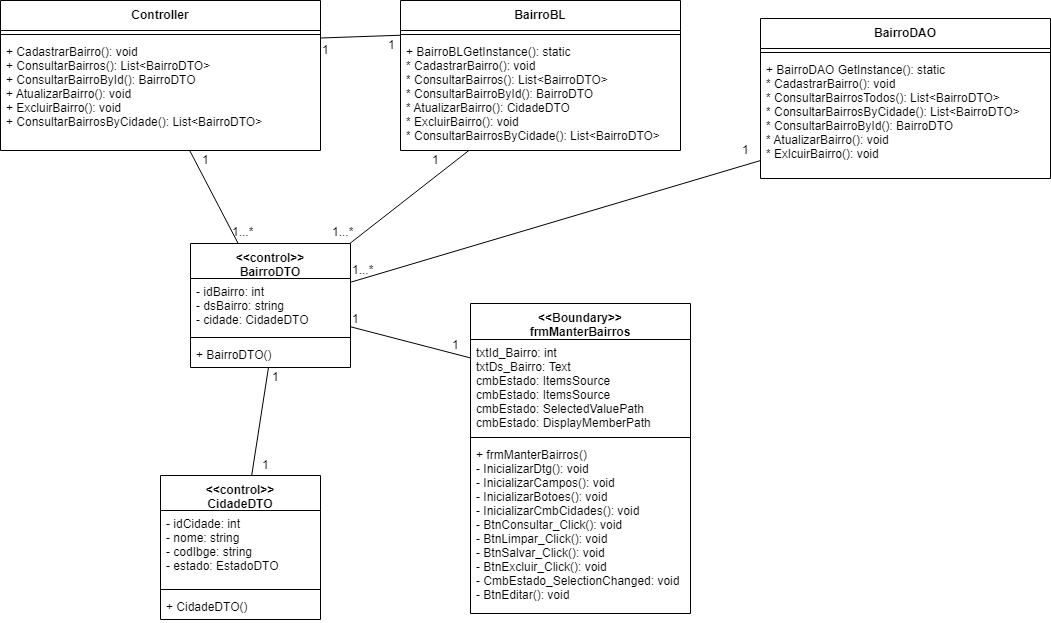
Figura 4.9 – Diagrama de Classes de Manter Cidades



Fonte: Autoria Própria, 2019

A figura 4.10 exibe o Diagrama de Classes de Manter Bairros.

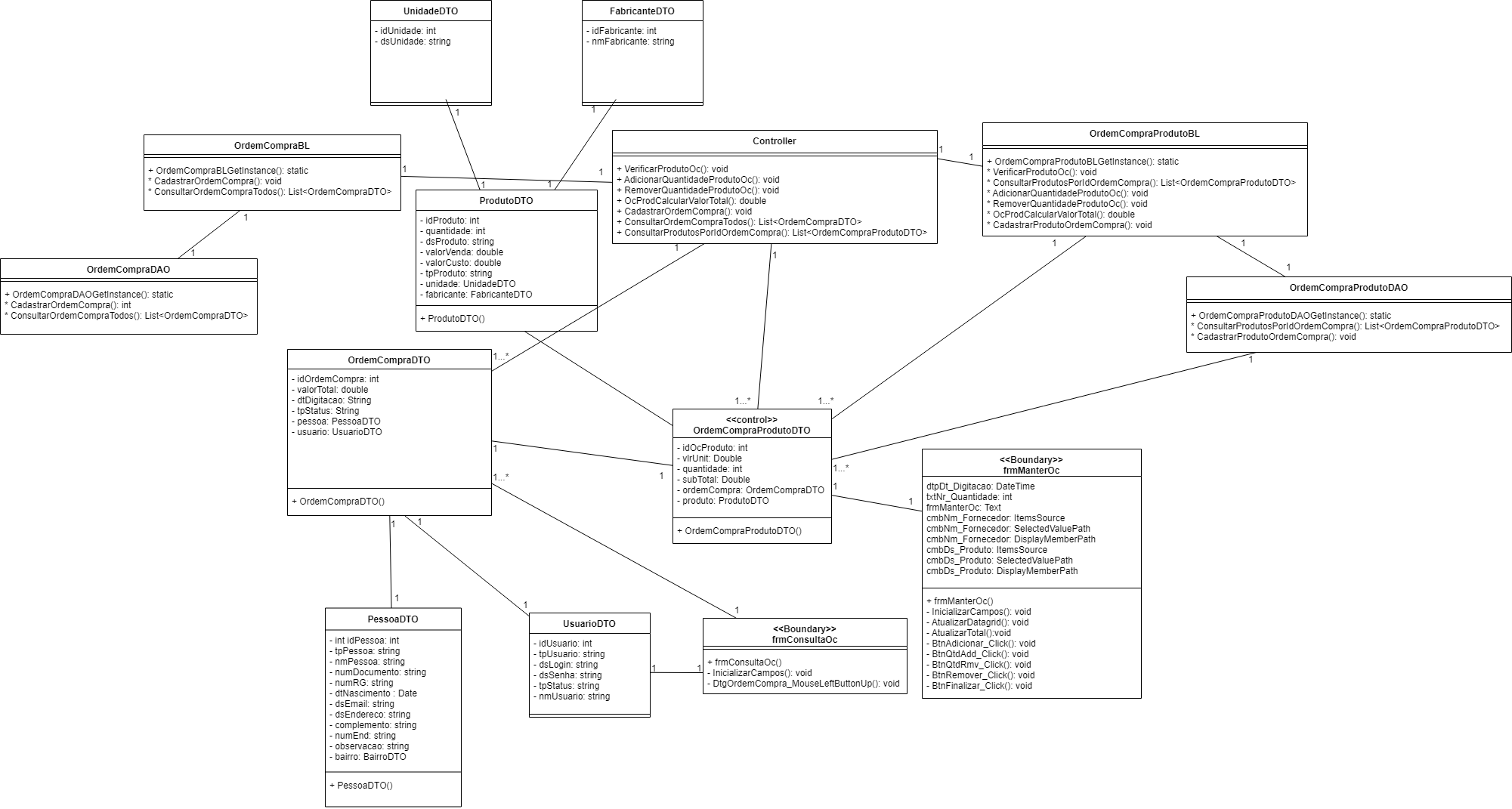
Figura 4.10 – Diagrama de Classes de Manter Bairros



Fonte: Autoria Própria, 2019

A figura 4.11 exibe o Diagrama de Classes de Manter Ponto de Compra.

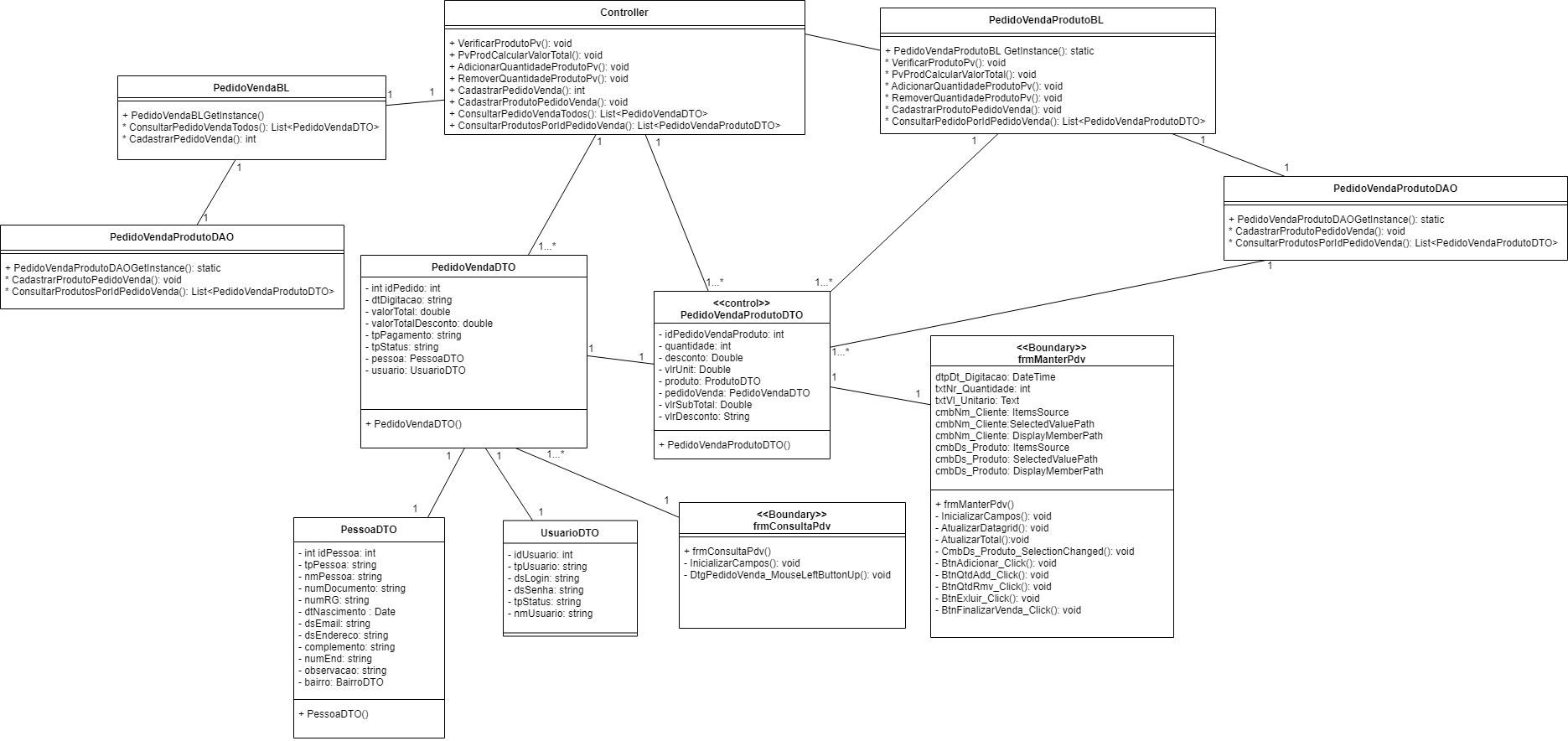
Figura 4.11 – Diagrama de Classes de Manter Ponto de Compra



Fonte: Autoria Própria, 2019

A figura 4.12 exibe o Diagrama de Classes de Manter Ponto de Venda.

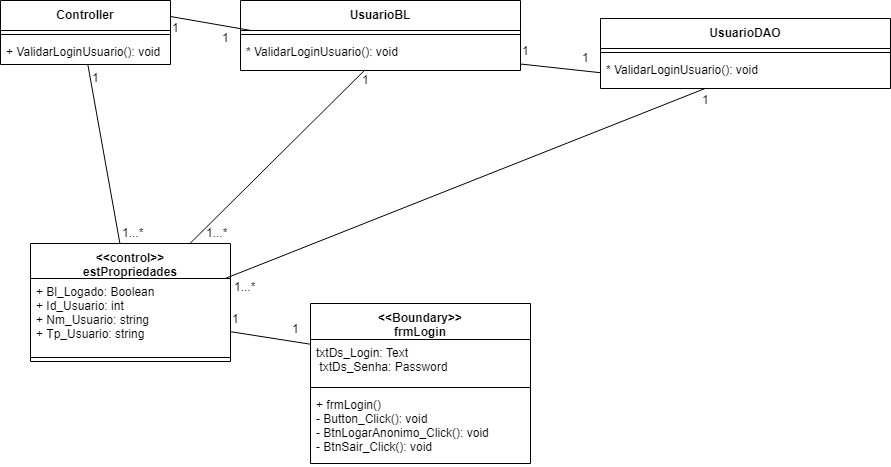
Figura 4.12 – Diagrama de Classes de Manter Ponto de Venda



Fonte: Autoria Própria, 2019

A figura 4.13 exibe o Diagrama de Classes do Login.

Figura 4.13 – Diagrama de Classes do Login

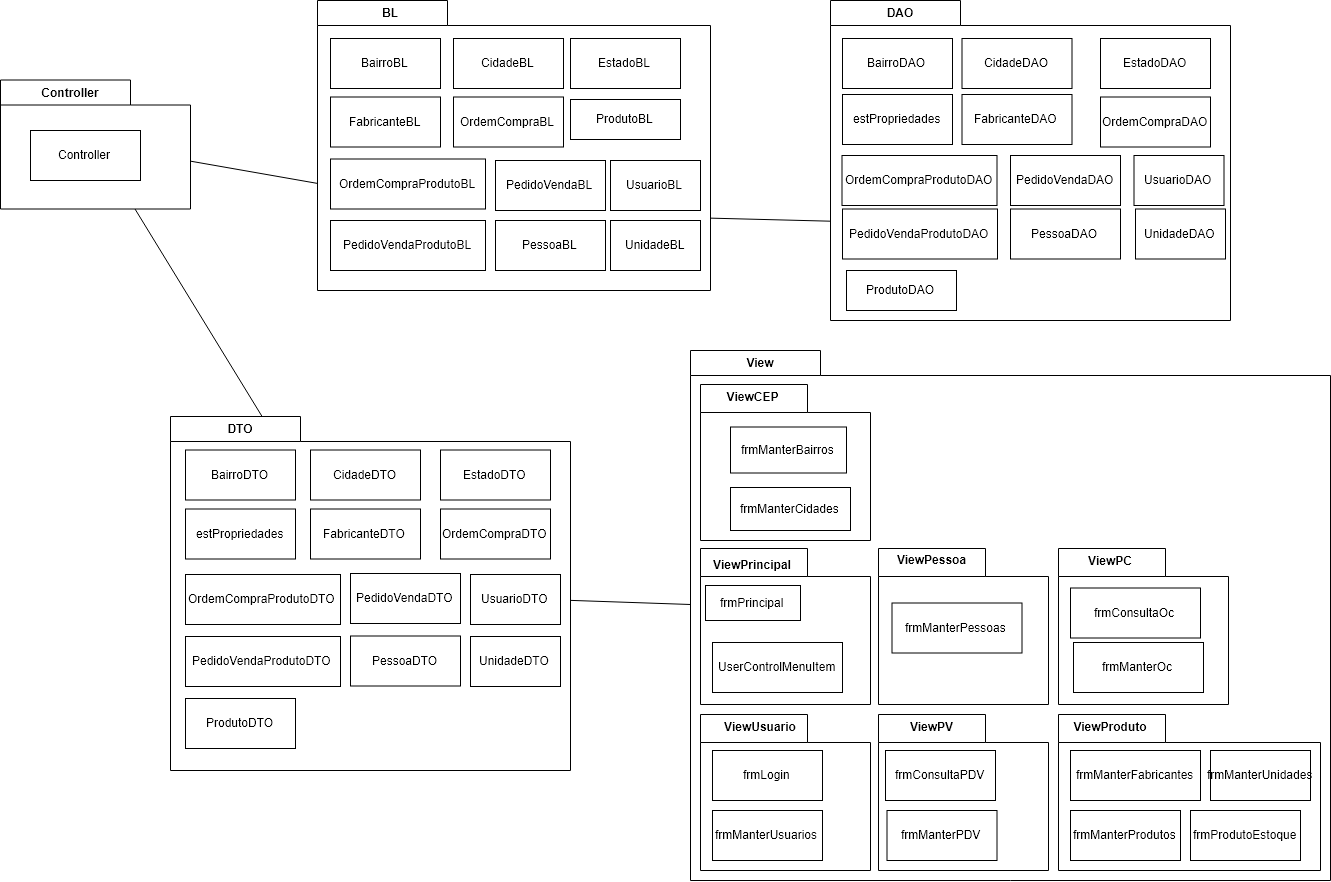


Fonte: Autoria Própria, 2019

## 4.8 DIAGRAMA DE PACOTES

Para o projeto desenvolvemos um diagrama de pacotes conforme a figura 4.14.

Figura 4.14 – Diagrama de Pacotes do Projeto

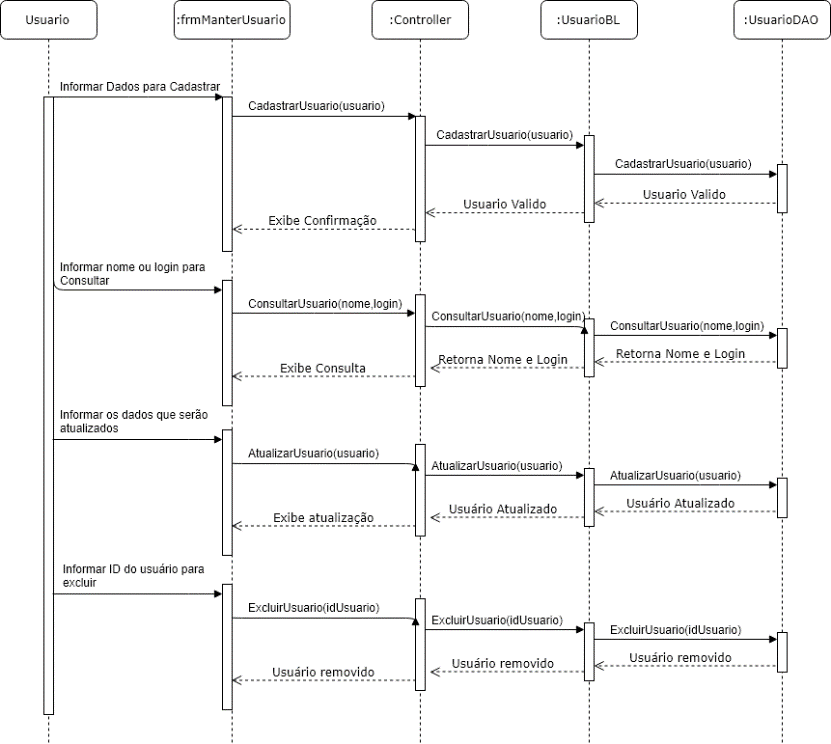


Fonte: Autoria Própria, 2019

## 4.9 DIAGRAMA DE SEQUÊNCIA

Para o projeto desenvolvemos um diagrama de sequência para exemplificar cada etapa do projeto conforme a figura 4.15.

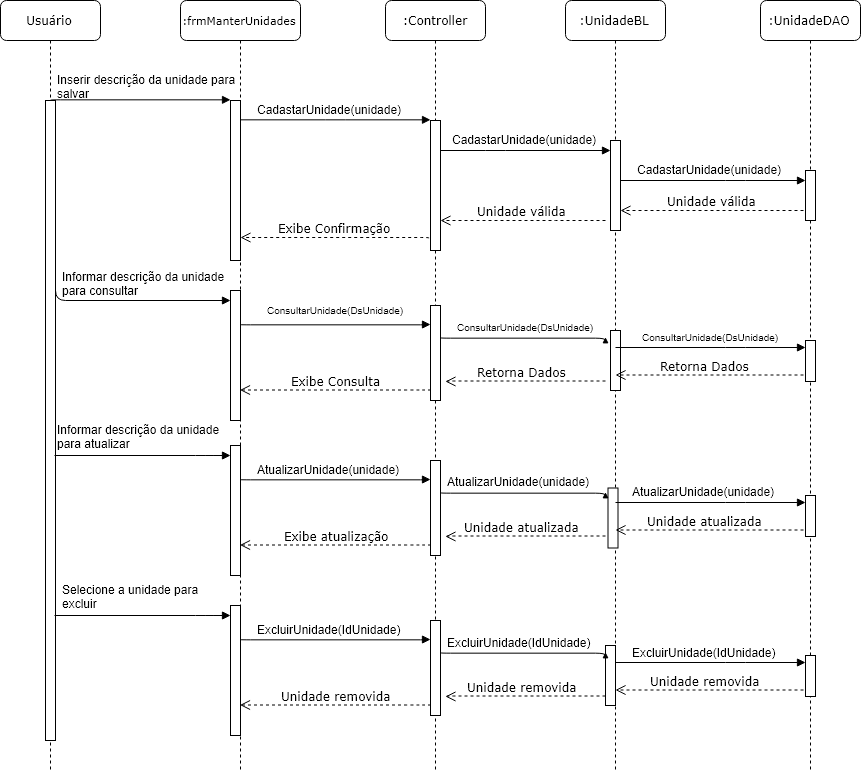
Figura 4.15 – Diagrama de Sequencia de Manter Usuários



Fonte: Autoria Própria, 2019

A figura 4.16 exibe o Diagrama de Sequencia de Manter Unidades.

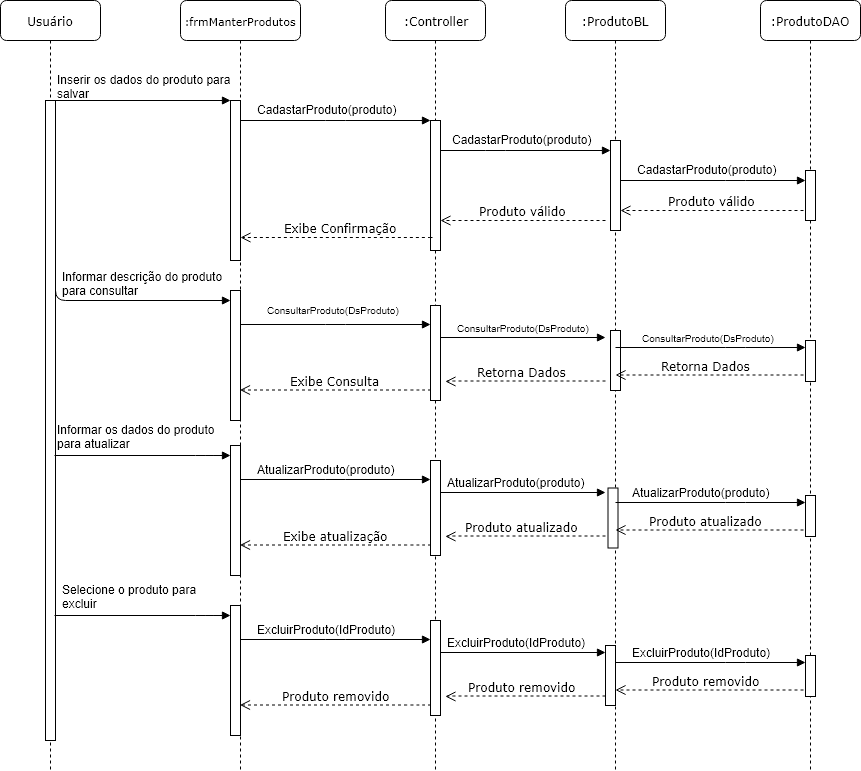
Figura 4.16– Diagrama de Sequencia de Manter Unidades



Fonte: Autoria Própria, 2019

A figura 4.17exibe o Diagrama de Sequencia de Manter Produtos.

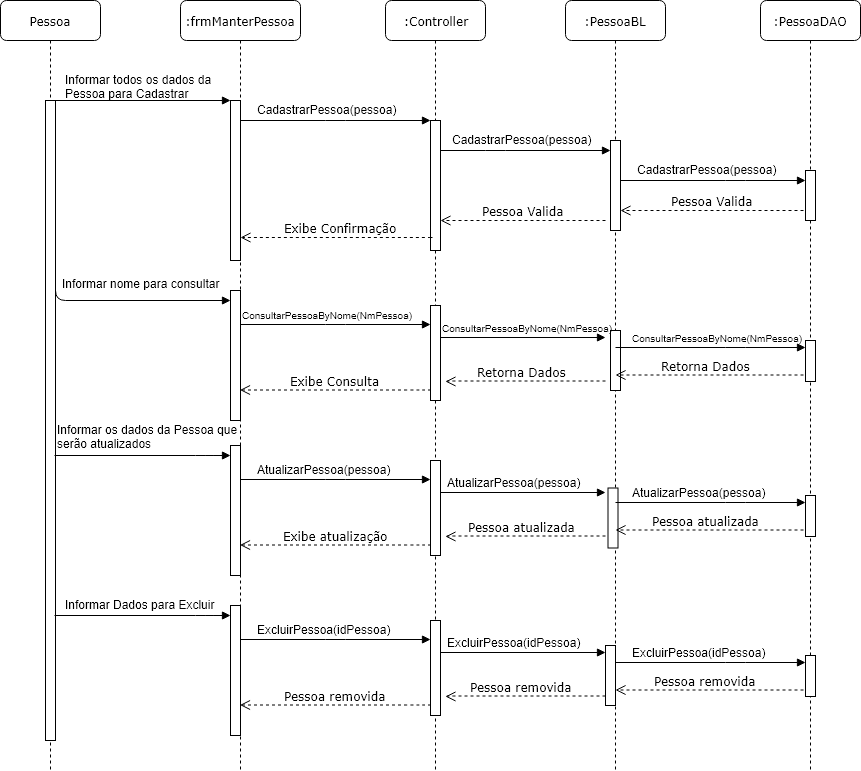
Figura 4.17 – Diagrama de Sequencia de Manter Produtos



Fonte: Autoria Própria, 2019

A figura 4.18exibe o Diagrama de Sequencia de Manter Pessoa.

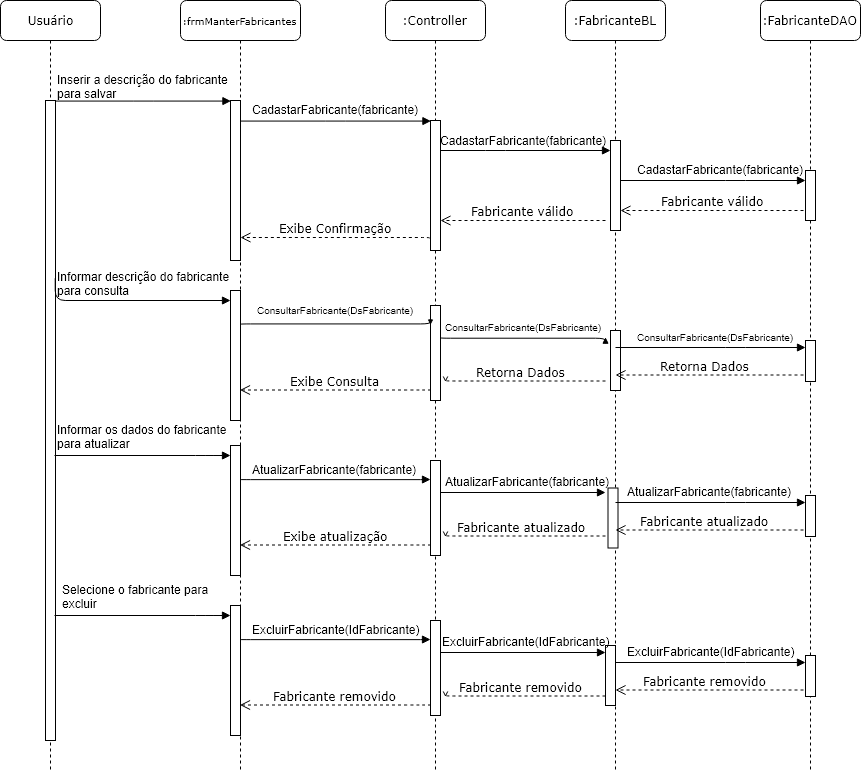
Figura 4.18 – Diagrama de Sequencia de Manter Pessoa



Fonte: Autoria Própria, 2019

A figura 4.19 exibe o Diagrama de Sequencia de Manter Fabricantes.

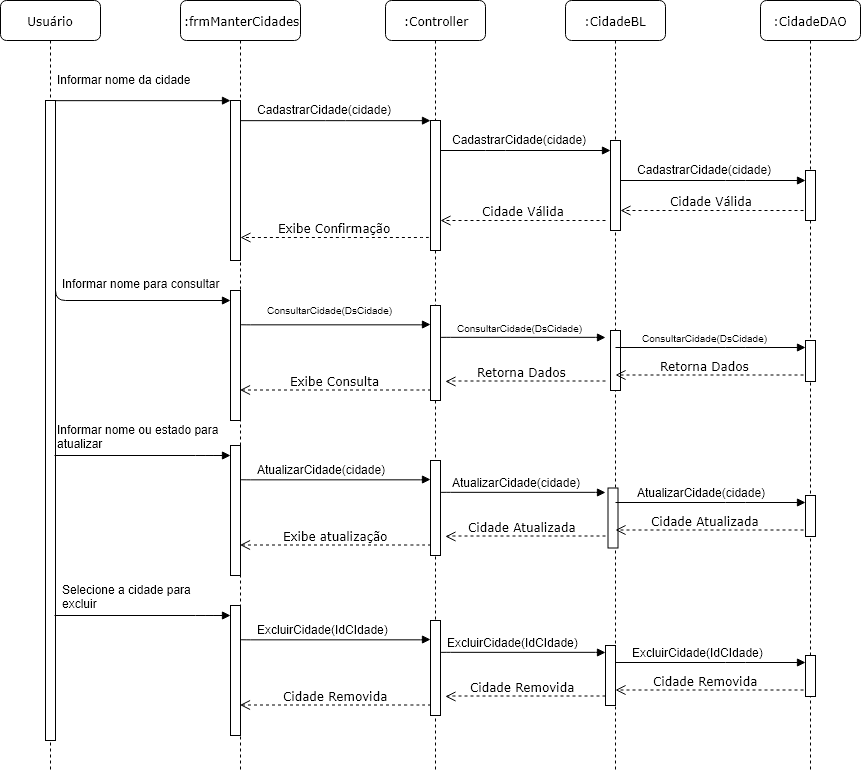
Figura 4.19 – Diagrama de Sequencia de Manter Fabricantes



Fonte: Autoria Própria, 2019

A figura 4.20 exibe o Diagrama de Sequencia de Manter Cidades.

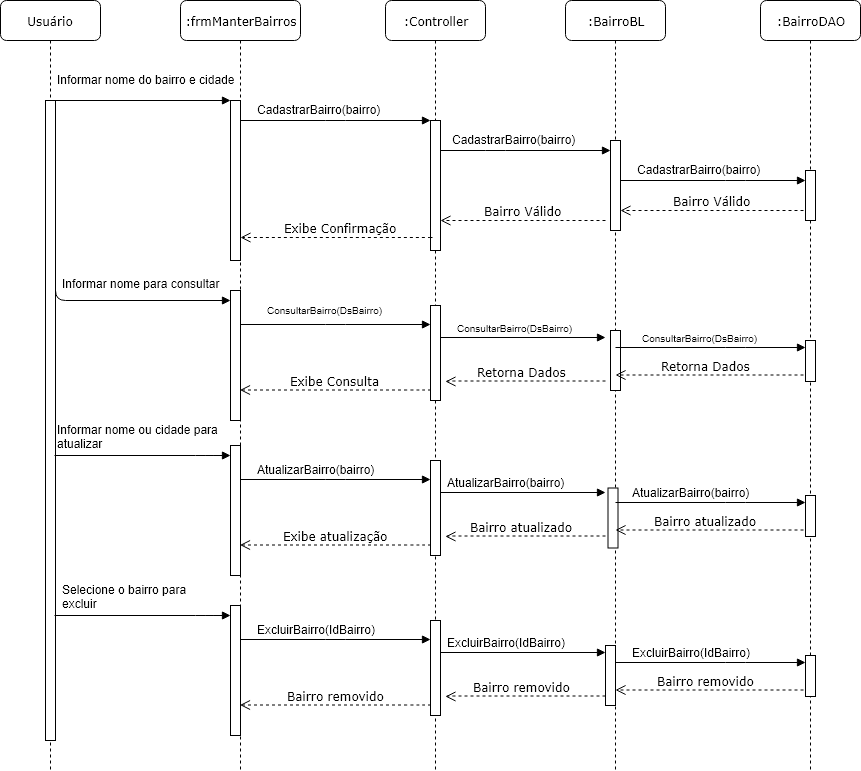
Figura 4.20 – Diagrama de Sequencia de Manter Cidades



Fonte: Autoria Própria, 2019

A figura 4.21 exibe o Diagrama de Sequencia de Manter Bairros.

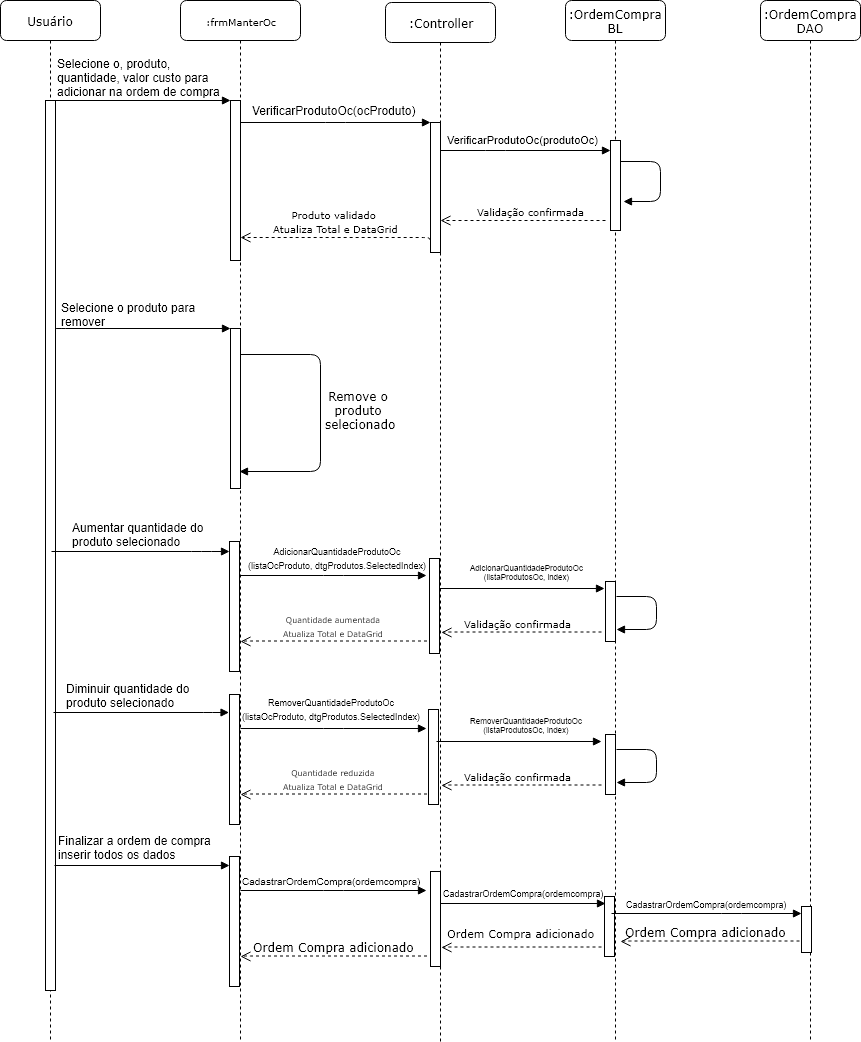
Figura 4.21– Diagrama de Sequencia de Manter Bairros



Fonte: Autoria Própria, 2019

A figura 4.22 exibe o Diagrama de Sequencia de Manter Ordem de Compra.

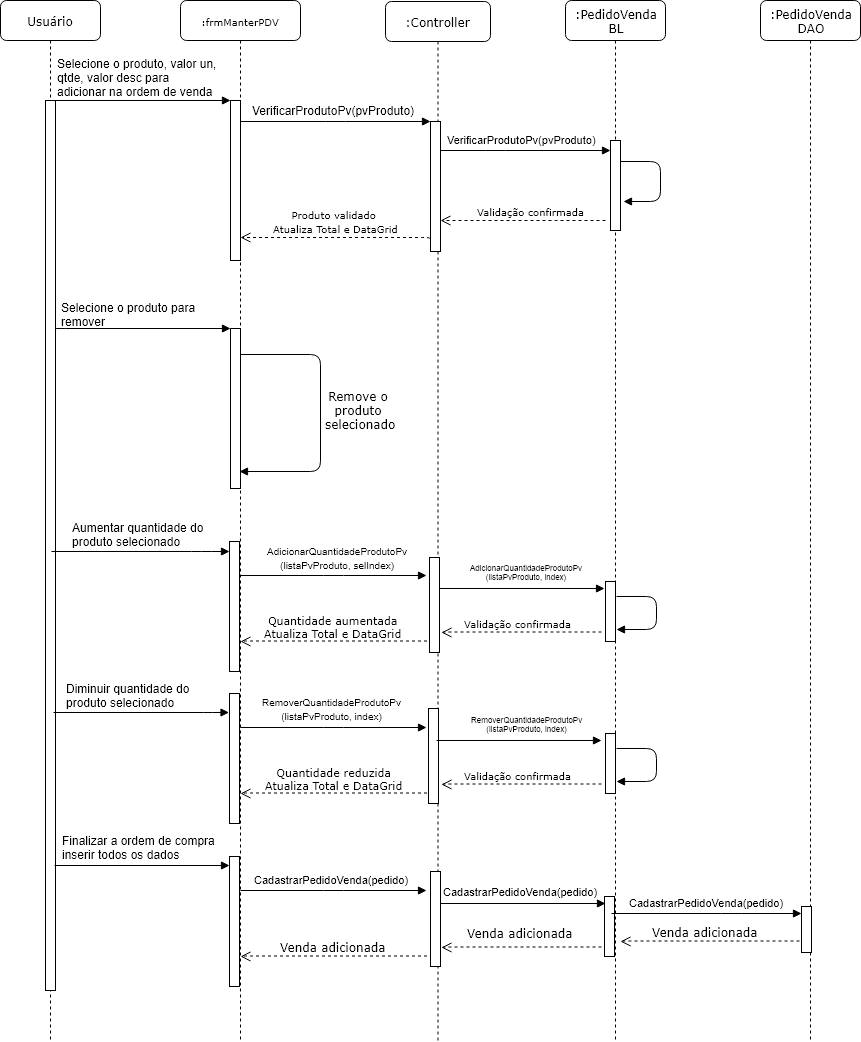
Figura 4.22 – Diagrama de Sequencia de Manter Pedido



Fonte: Autoria Própria, 2019

A figura 4.23 exibe o Diagrama de Sequencia de Manter PDV.

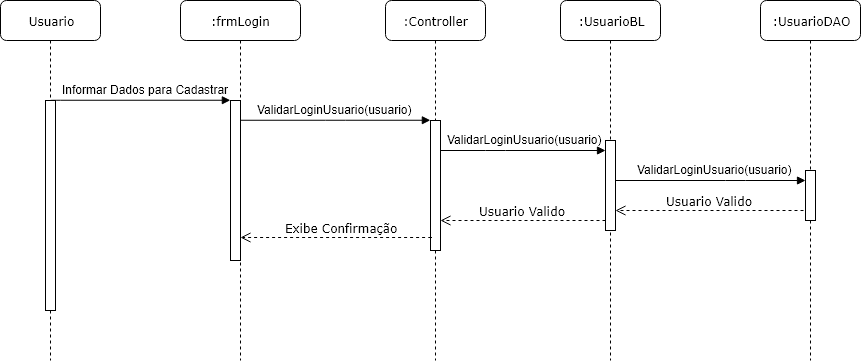
Figura 4.23 – Diagrama de Sequencia de Manter PDV



Fonte: Autoria Própria, 2019

A figura 4.24 exibe o Diagrama de Sequencia do Login.

Figura 4.24 – Diagrama de Sequencia do Login

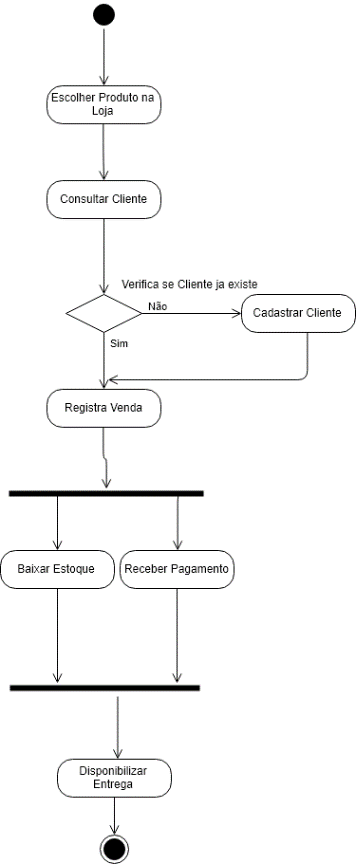


Fonte: Autoria Própria, 2019

## 4.10 DIAGRAMA DE ATIVIDADES

Nesta etapa foi desenvolvido um diagrama para exemplificar as atividades do projeto conforme figura 4.25.

Figura 4.25 – Diagrama de Atividade do Projeto

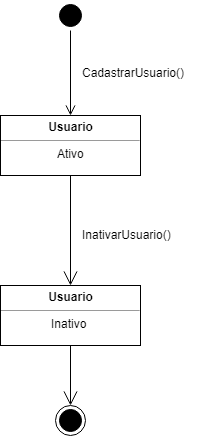


Fonte: Autoria Própria, 2019

## 4.11 DIAGRAMA DE ESTADOS

Para exemplificar os estados do projeto foi elaborado dois diagrama de estados conforme figura 4.26 e figura 4.27.

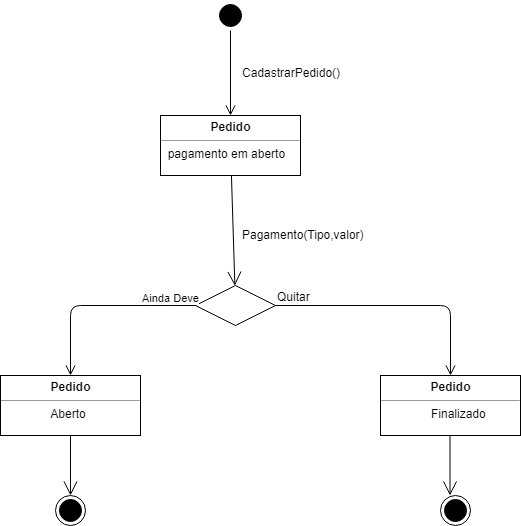
Figura 4.26 – Diagrama de Estados de Manter Usuários



Fonte: Autoria Própria, 2019

A figura 4.27 exibe o Diagrama de Estados de Manter Pedido.

Figura 4.27 – Diagrama de Estados de Manter Pedido

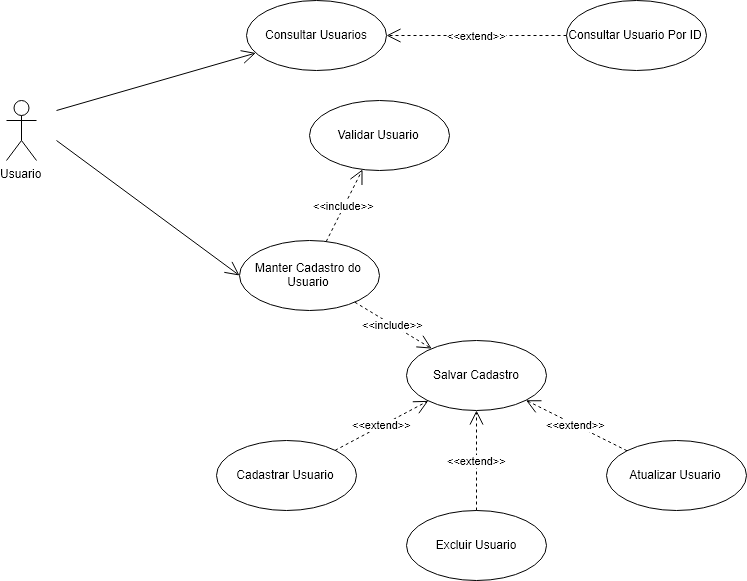


Fonte: Autoria Própria, 2019

## 4.12 DIAGRAMA DE CASOS DE USO

Para cada interação do usuário com o sistema foi elaborado um diagrama conforme figura 4.28 que representa da tela de Manter Usuários.

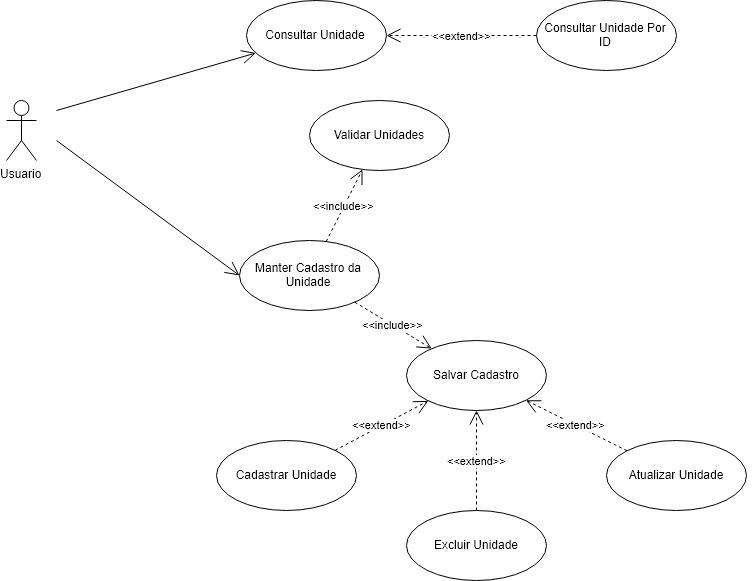
Figura 4.28 – Diagrama de Casos de Uso de Manter Usuários



Fonte: Autoria Própria, 2019

A figura 4.29 exibe o Diagrama de Casos de Uso de Manter Unidades.

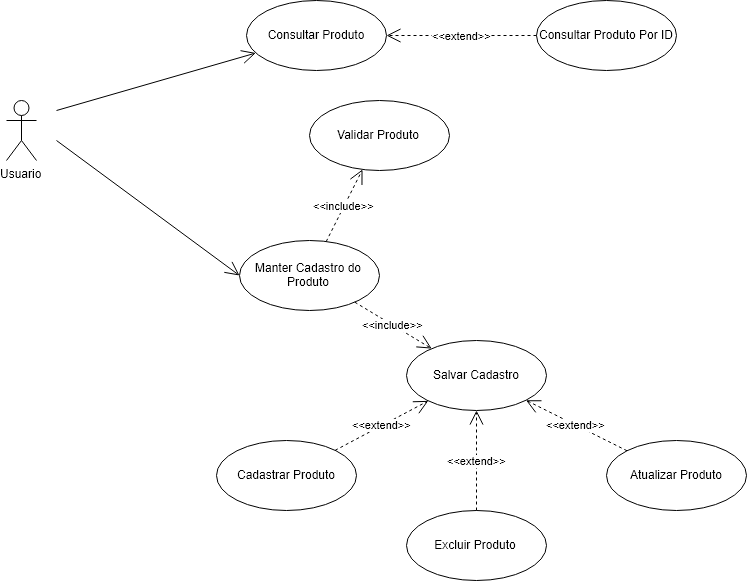
Figura 4.29– Diagrama de Casos de Uso de Manter Unidades



Fonte: Autoria Própria, 2019

A figura 4.30 exibe o Diagrama de Casos de Uso de Manter Produtos.

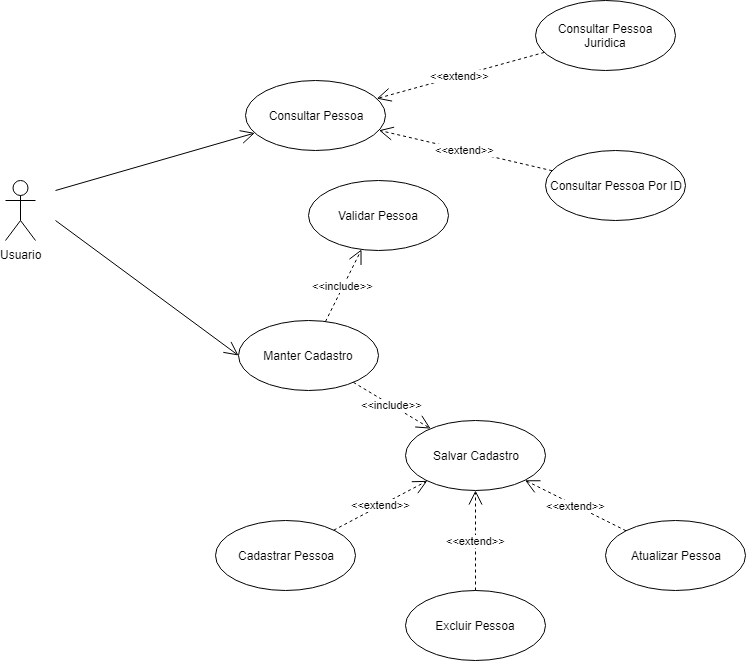
Figura 4.30 – Diagrama de Casos de Uso de Manter Produtos



Fonte: Autoria Própria, 2019

A figura 4.31 exibe o Diagrama de Casos de Uso de Manter Pessoa.

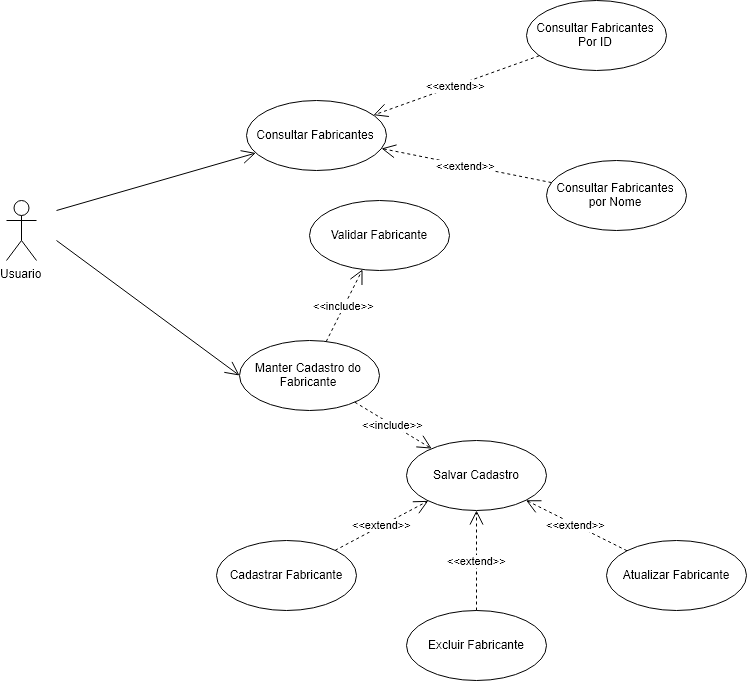
Figura 4.31– Diagrama de Casos de Uso de Manter Pessoa



Fonte: Autoria Própria, 2019

A figura 4.32 exibe o Diagrama de Casos de Uso de Manter Fabricantes.

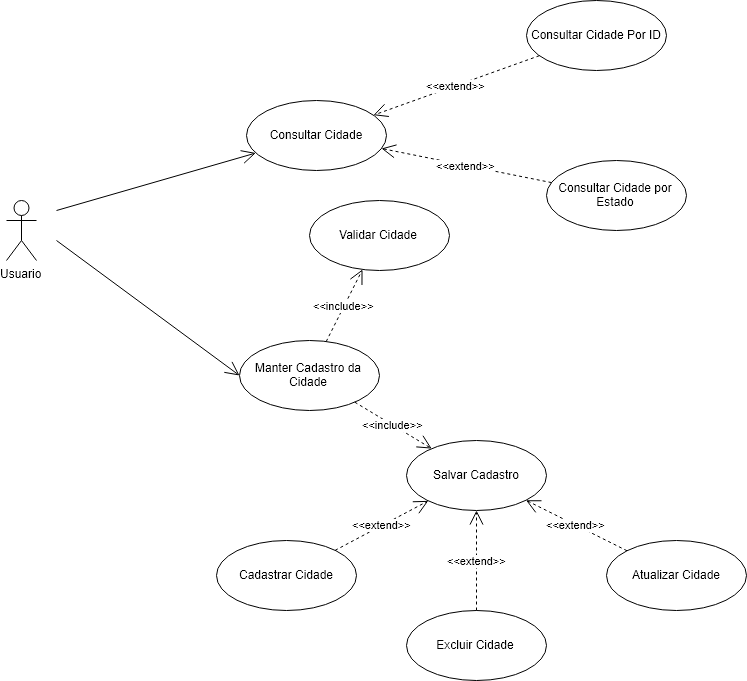
Figura 4.32 – Diagrama de Casos de Uso de Manter Fabricantes



Fonte: Autoria Própria, 2019

A figura 4.33 exibe o Diagrama de Casos de Uso de Manter Cidades.

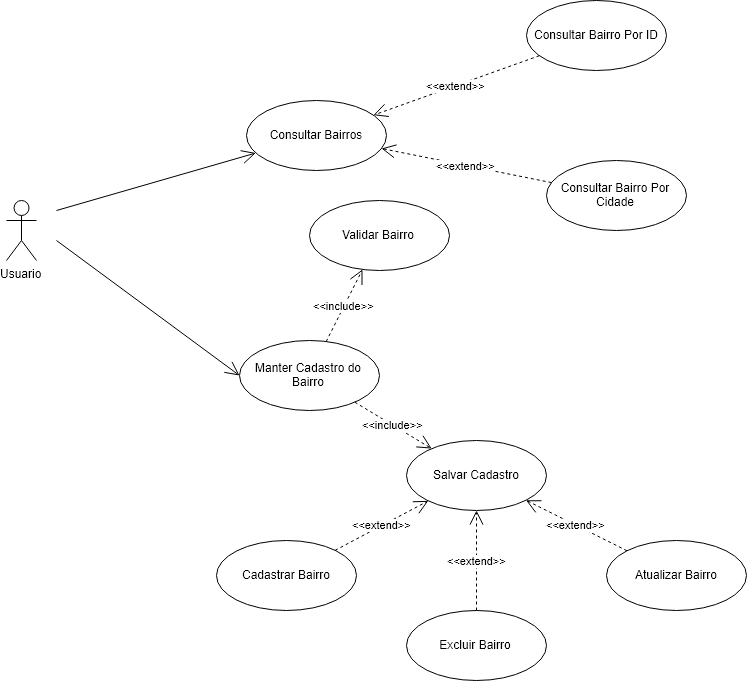
Figura 4.33 – Diagrama de Casos de Uso de Manter Cidades



Fonte: Autoria Própria, 2019

A figura 4.34 exibe o Diagrama de Casos de Uso de Manter Bairros.

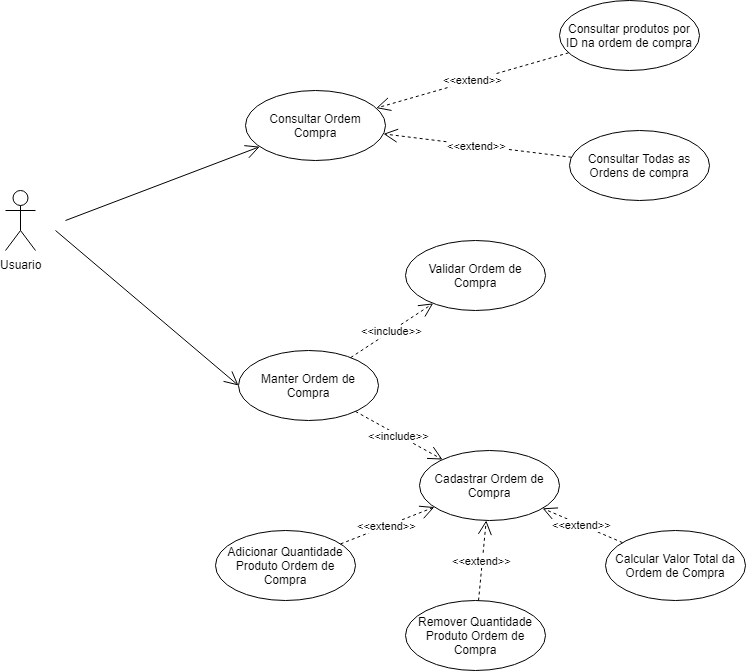
Figura 4.34 – Diagrama de Casos de Uso de Manter Bairros



Fonte: Autoria Própria, 2019

A figura 4.35 exibe o Diagrama de Casos de Uso de Manter Ordem de Compra.

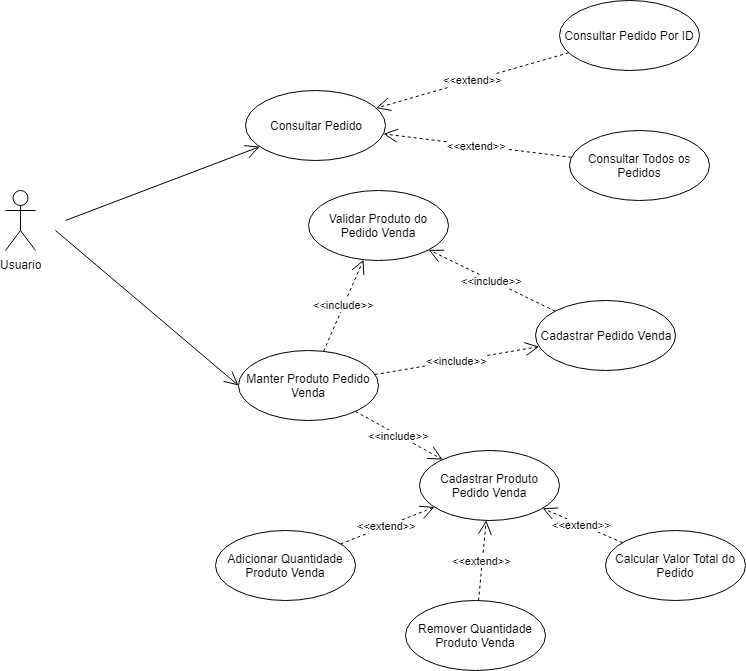
Figura 4.35 – Diagrama de Casos de Uso de Manter Ordem de Compra



Fonte: Autoria Própria, 2019

A figura 4.36 exibe o Diagrama de Casos de Uso de Manter PDV.

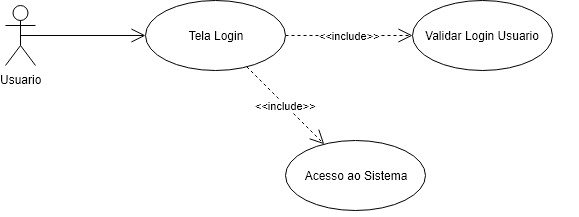
Figura 4.36 – Diagrama de Casos de Uso de Manter PDV



Fonte: Autoria Própria, 2019

A figura 4.37 exibe o Diagrama de Casos de Uso do Login.

Figura 4.37 – Diagrama de Casos de Uso do Login

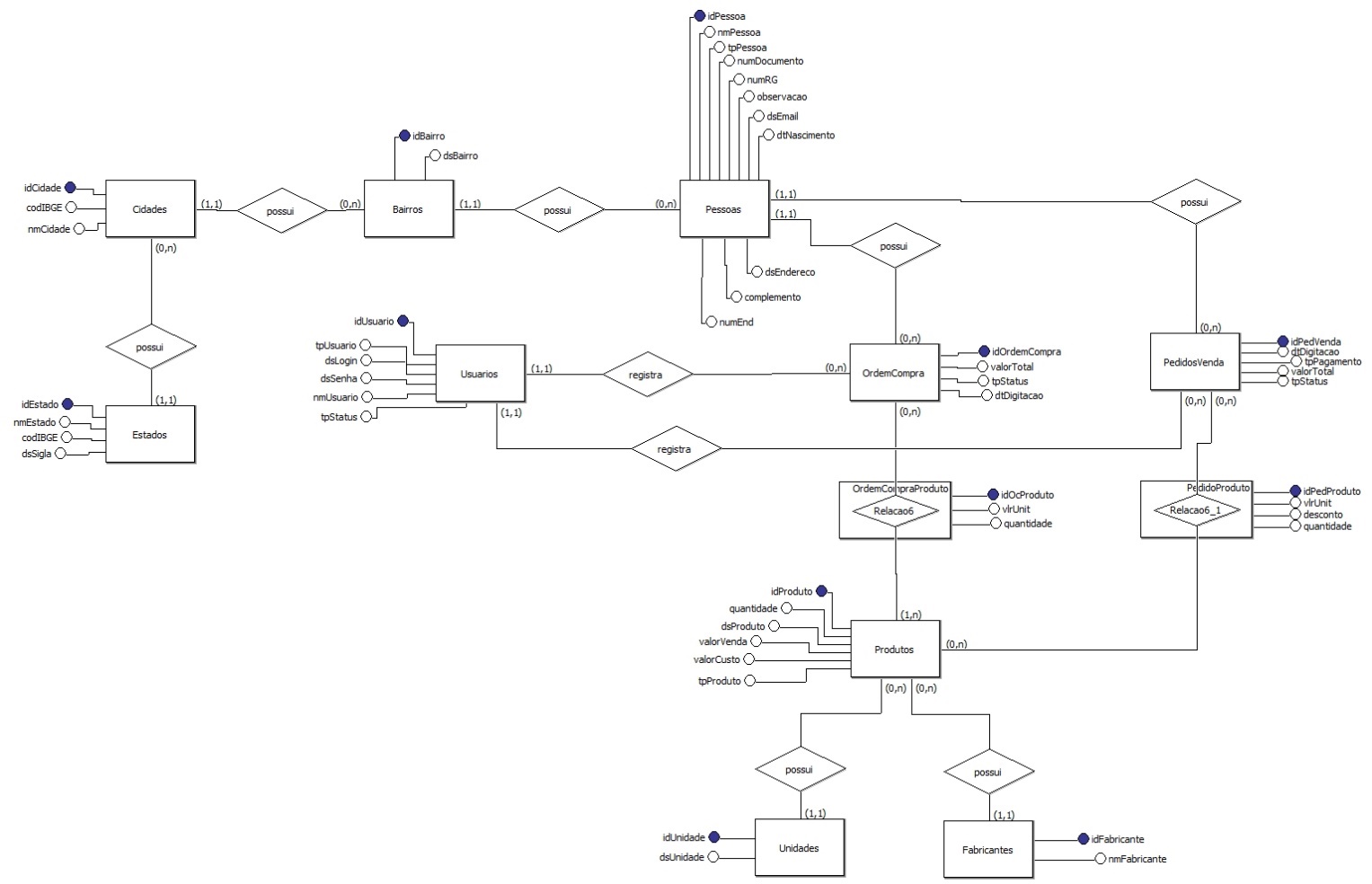


Fonte: Autoria Própria, 2019

## 4.13 MODELO CONCEITUAL DO BANCO DE DADOS

Para a solução do projeto foi elaborado o modelo conceitual conforme a figura 4.38.

Figura 4.38 – Modelo Conceitual do Banco de Dados

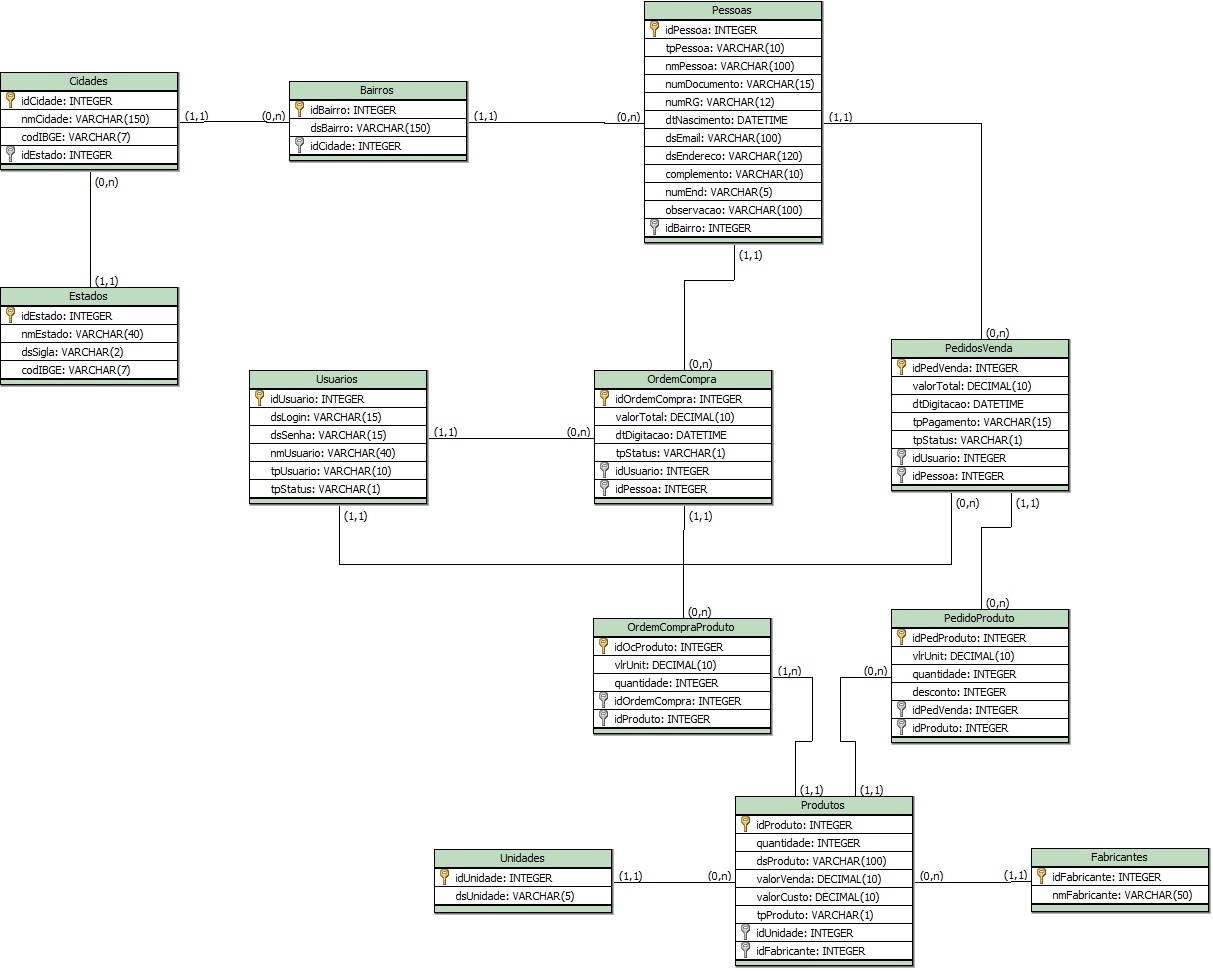


Fonte: Autoria Própria, 2019

## 4.14 MODELO LÓGICO DO BANCO DE DADOS

Com o modelo conceitual elaborado permitiu a criação do modelo lógico conforme a figura 4.39.

Figura 4.39 – Modelo Lógico do Banco de Dados



Fonte: Autoria Própria, 2019

## 4.16 TESTES DE SOFTWARE

A figura 4.40 exibe os testes realizados e os resultados alcançados.

Figura 4.40 – Teste de software

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **REQUISITOS** | **DESCRIÇÃO** | **CENARIO** |
| T001 | Login no Sistema | Usuário faz login do sistema | **ATOR:** Usuário  **DESCRIÇÃO:** Usuário terá um login de acesso ao sistema. **ERRADO:** Senha incorreta para usuário não cadastrado.  **FINAL:** Usuário Logado; |
| T002 | Cadastro de Usuários | Usuário poderá fazer cadastros de outro usuário dentro do sistema. | **ATOR:** Usuário  **DESCRIÇÃO:** Usuário criará um cadastro de outros usuários.  **ERRADO:** Usuário já cadastrado.  **FINAL:** Usuário Cadastrado. |
| T003 | Cadastro de Pessoas | Usuário poderá fazer um cadastro de pessoas dentro do sistema. | **ATOR:** Usuário. **DESCRIÇÃO:** Usuário criará um cadastro de pessoas.  **ERRADO:** Cadastro já existe.  **FINAL:** Pessoa Cadastrada. |
| T004 | Cadastro de Produtos | Usuário poderá fazer um cadastro de produtos. | **ATOR:**Usuário **DESCRIÇÃO:**Usuário cadastrará produtos no sistema. **ERRADO:** Produto já existente.  **FINAL:** Produto cadastrado. |
| T005 | Emissão de Pedidos | Usuário poderá fazer emissão de pedidos no sistema | **ATOR:** Usuário  **DESCRIÇÃO:**Usuário poderá fazer o pedido no sistema  **ERRADO:** Pedido de Venda já existe, quantidade não pode ser menor que 0, produto indisponível em estoque.  **FINAL:** Pedido Finalizado. |

Fonte: Autoria Própria, 2019

# 5.0 MÉTODOS

## 5.1 FUNCIONAMENTO DO SISTEMA E APRESENTAÇÃO DAS TELAS

Nesta sessão, explicaremos o funcionamento do sistema com suas respectivas telas desenvolvidas pelo grupo, com as pecularidades de cada etapa do processo.

O primeiro passo para acessar o sistema é realizar a validação de acesso, para isto é necessário inserir o usuário e senha conforme a figura 5.0.

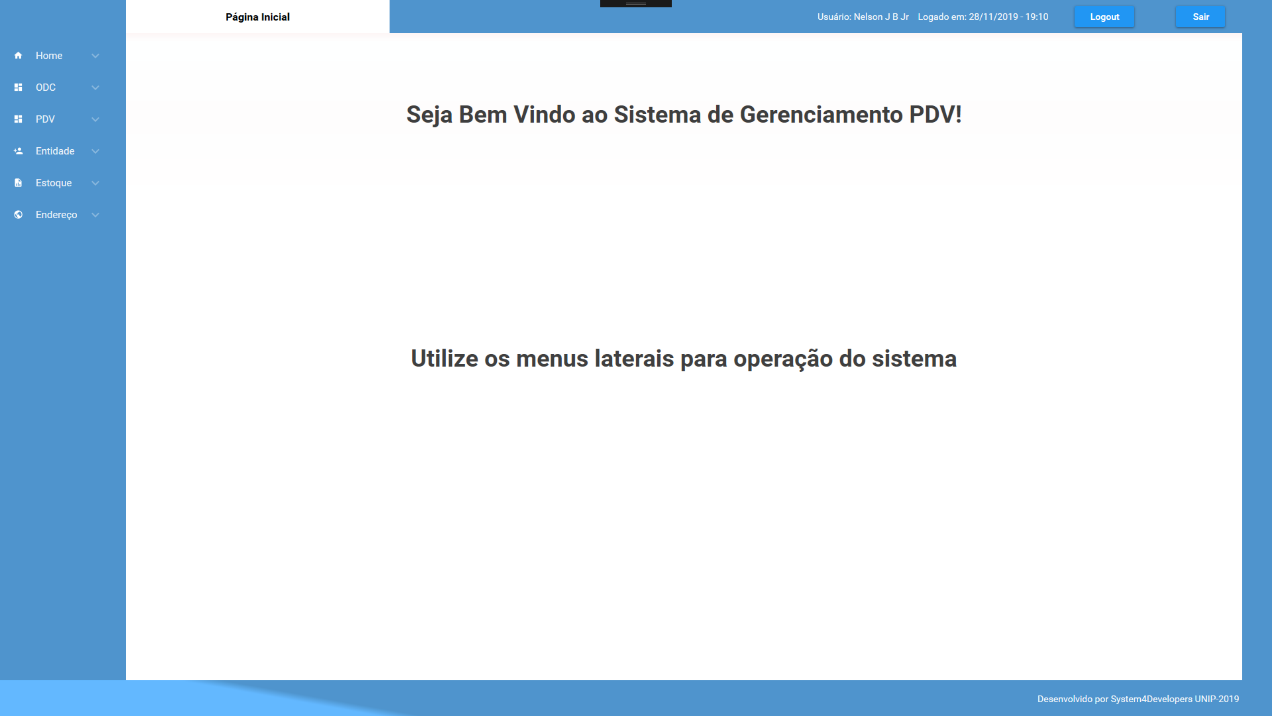
Figura 5.0 – Tela do Login do Projeto



Fonte: Autoria Própria, 2019

Após a validação do usuário, é redirecionado para a tela inicial do sistema conforme figura 5.1. Na barra lateral temos as opções de ODC (Ordem de compra), PDV (Ponto de Venda), Entidade, Estoque e Endereço.

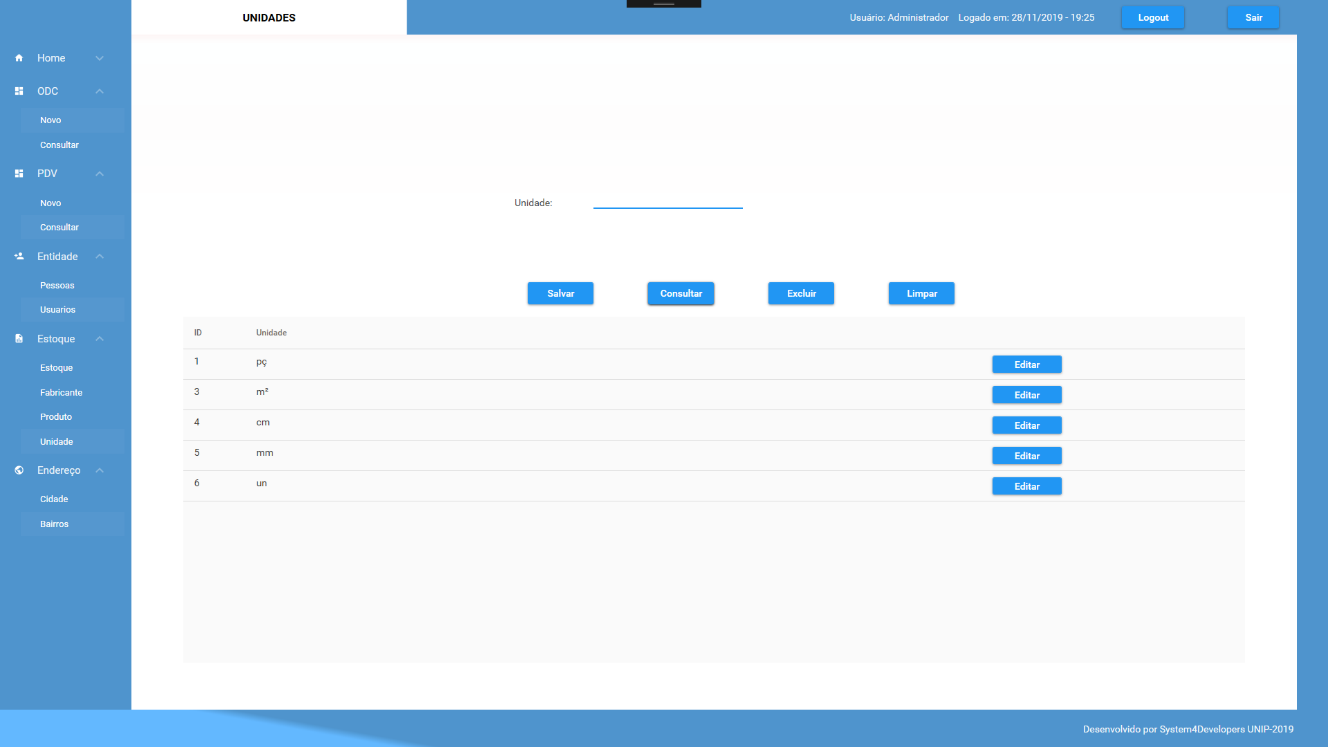
Figura 5.1 – Tela de Inicio



Fonte: Autoria Própria, 2019

Na opção unidade em estoque, podemos visualizar, cadastrar, editar e excluir os tipos de unidades de um produto, como por exemplos metros, centímetro, peço, unidade conforme figura 5.2.

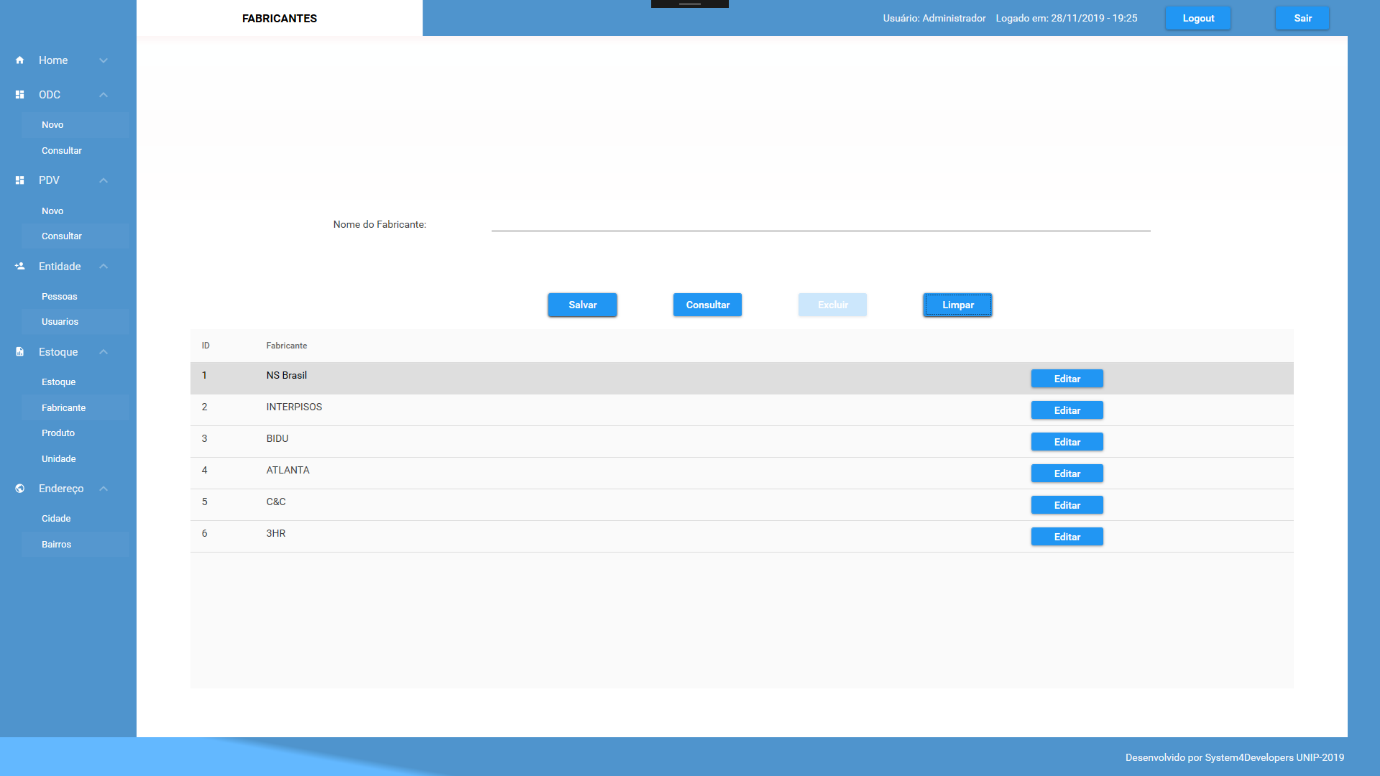
Figura 5.2 – Tela de Unidade



Fonte: Autoria Própria, 2019

Na opção fabricantes em estoque, podemos visualizar, cadastrar, editar e excluir os fabricantes dos produtos conforme a figura 5.3.

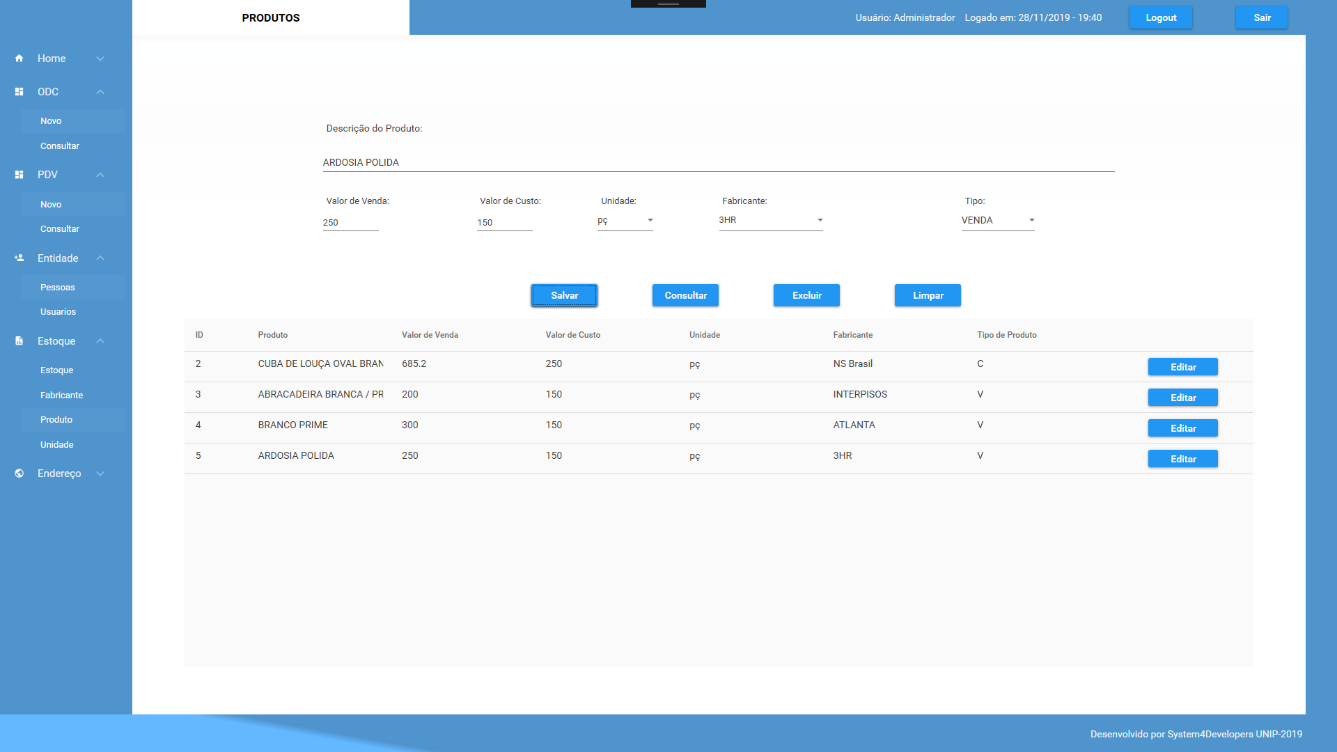
Figura 5.3 – Tela de Fabricantes



Fonte: Autoria Própria, 2019

Na opção produto em estoque, podemos visualizar, cadastar, editar e excluir as características dos produtos. Para um produto é possível registrar valor de venda, valor de custo, unidade, fabricante e tipo conforme a figura 5.4.

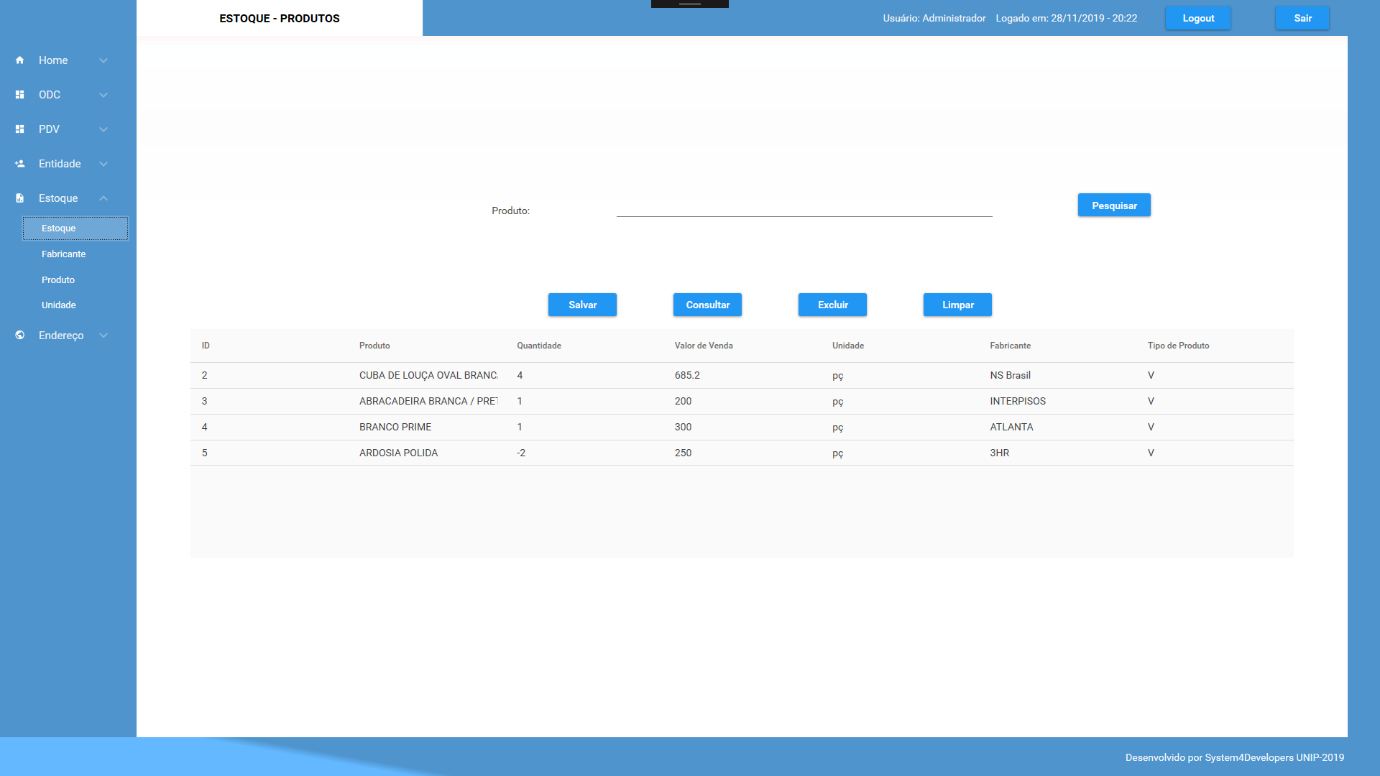
Figura 5.4 – Tela de Produtos



Fonte: Autoria Própria, 2019

Em estoque é possível visualizar e pesquisar os produtos disponíveis em estoque para a venda, além de exibir a quantidade de produtos, valor de venda, unidade e fabricante conforme figura 5.5.

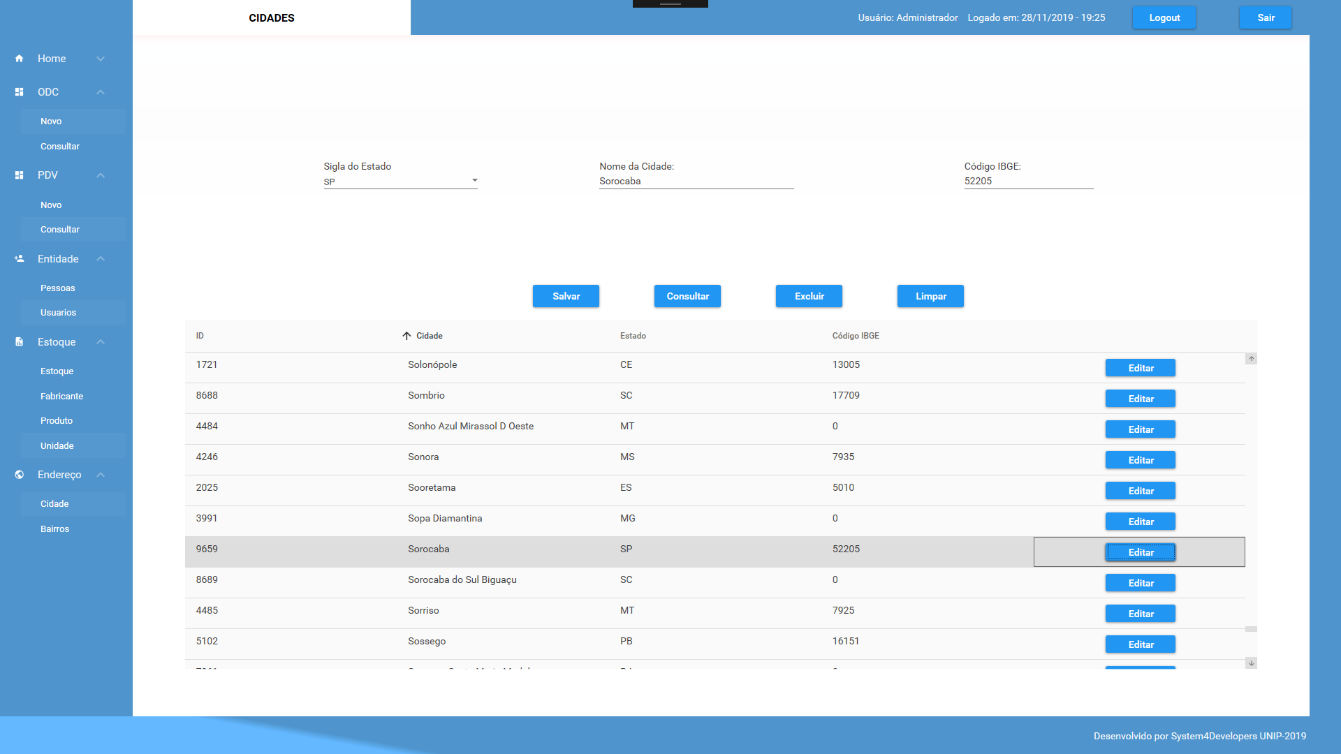
Figura 5.5 – Tela de Estoque



Fonte: Autoria Própria, 2019

Na opção cidade em endereço, conseguimos visualizar, cadastrar, editar e excluir cidades conforme a figura 5.6.

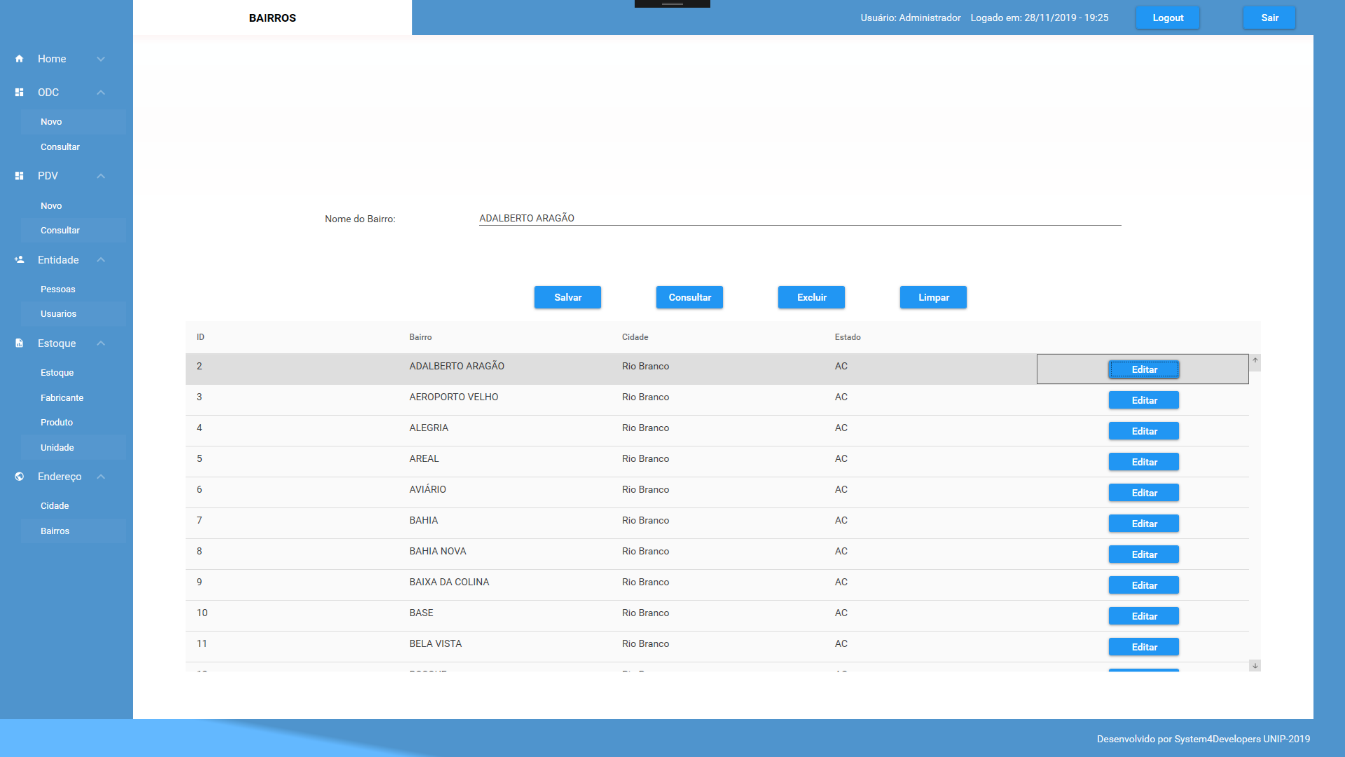
Figura 5.6 – Tela de Cidades



Fonte: Autoria Própria, 2019

Em endereço na opção bairros, é possível visualizar, inserir, editar e excluir os bairros conforme a figura 5.7.

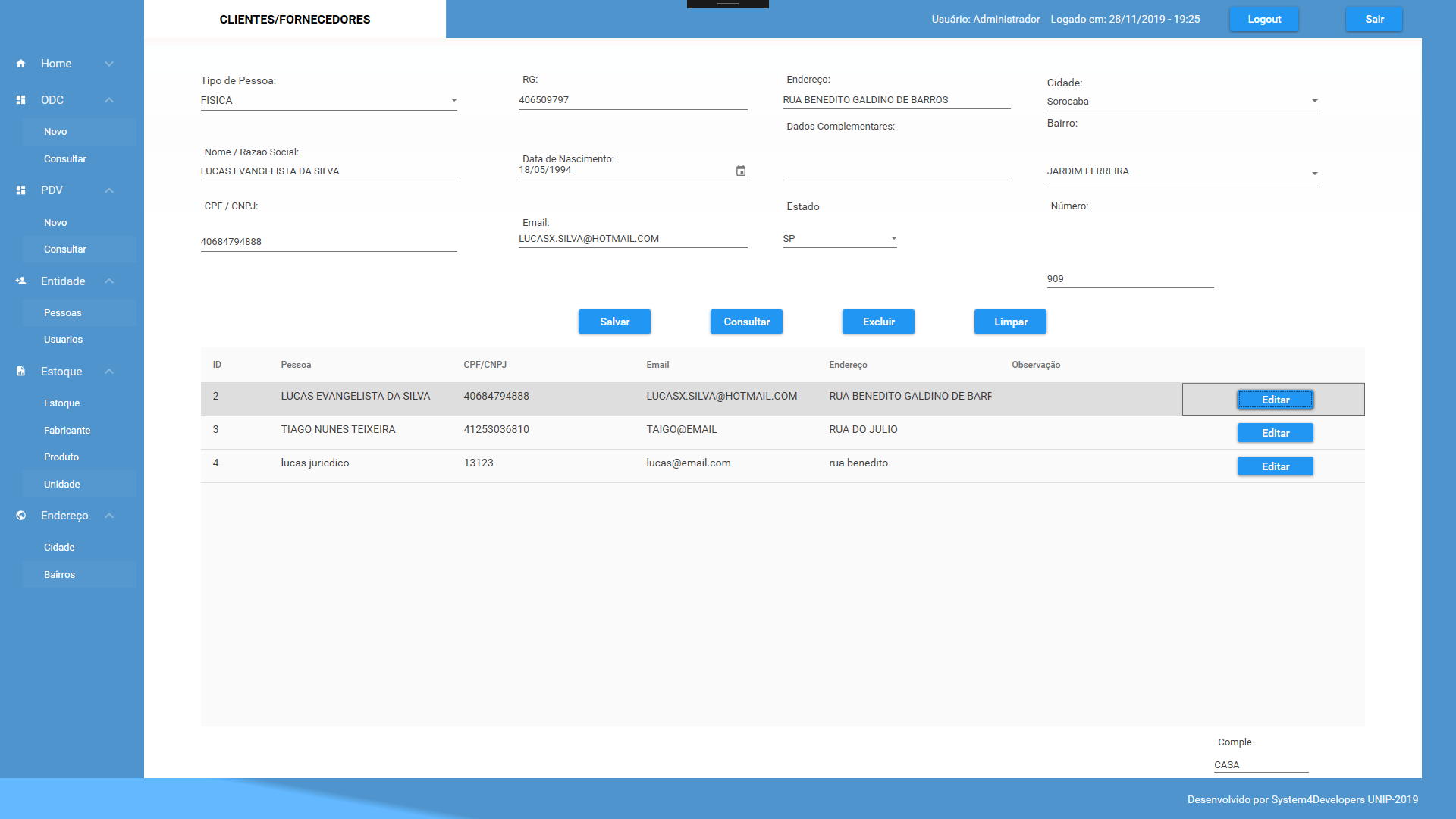
Figura 5.7 – Tela de Bairro



Fonte: Autoria Própria, 2019

Em Entidade na opção Pessoas é possível visualizar, cadastrar, editar e excluir os clientes e fornecedores conforme a figura 5.8.

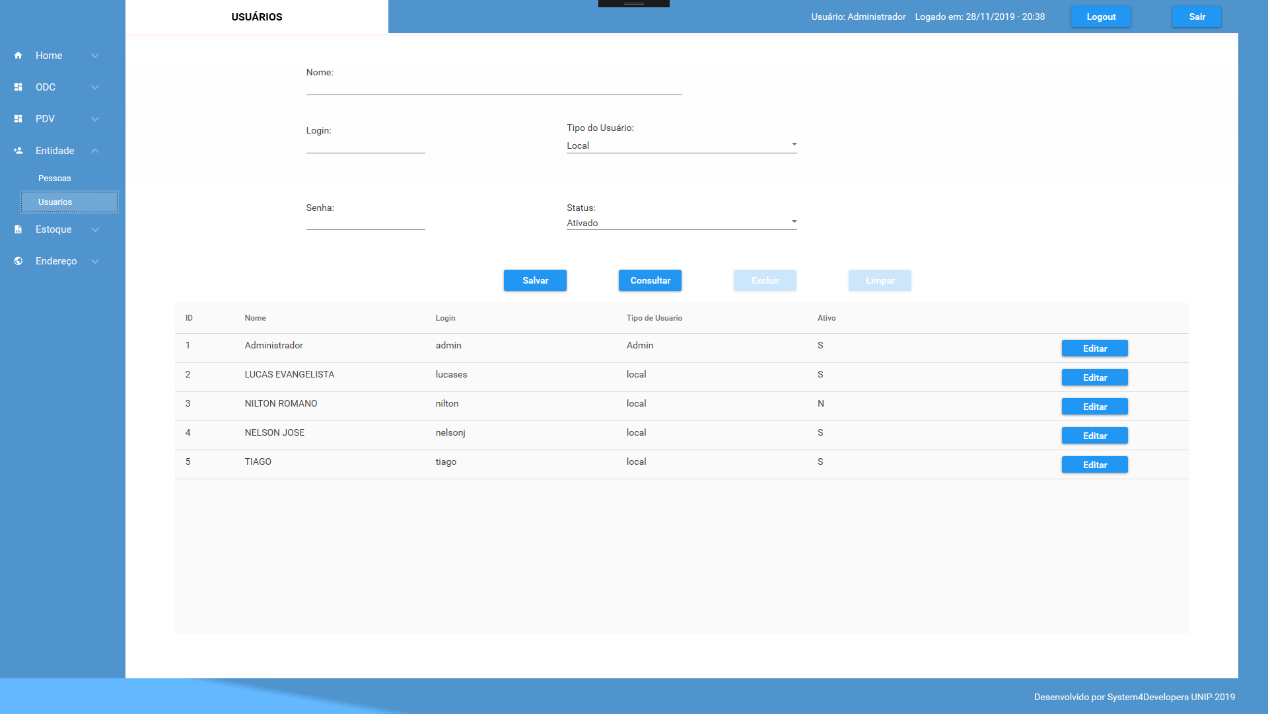
Figura 5.8 – Tela Pessoas



Fonte: Autoria Própria, 2019

Em Entidade na opção Usuários é possível visualizar, cadastrar, editar e excluir os clientes e fornecedores conforme a figura 5.8.

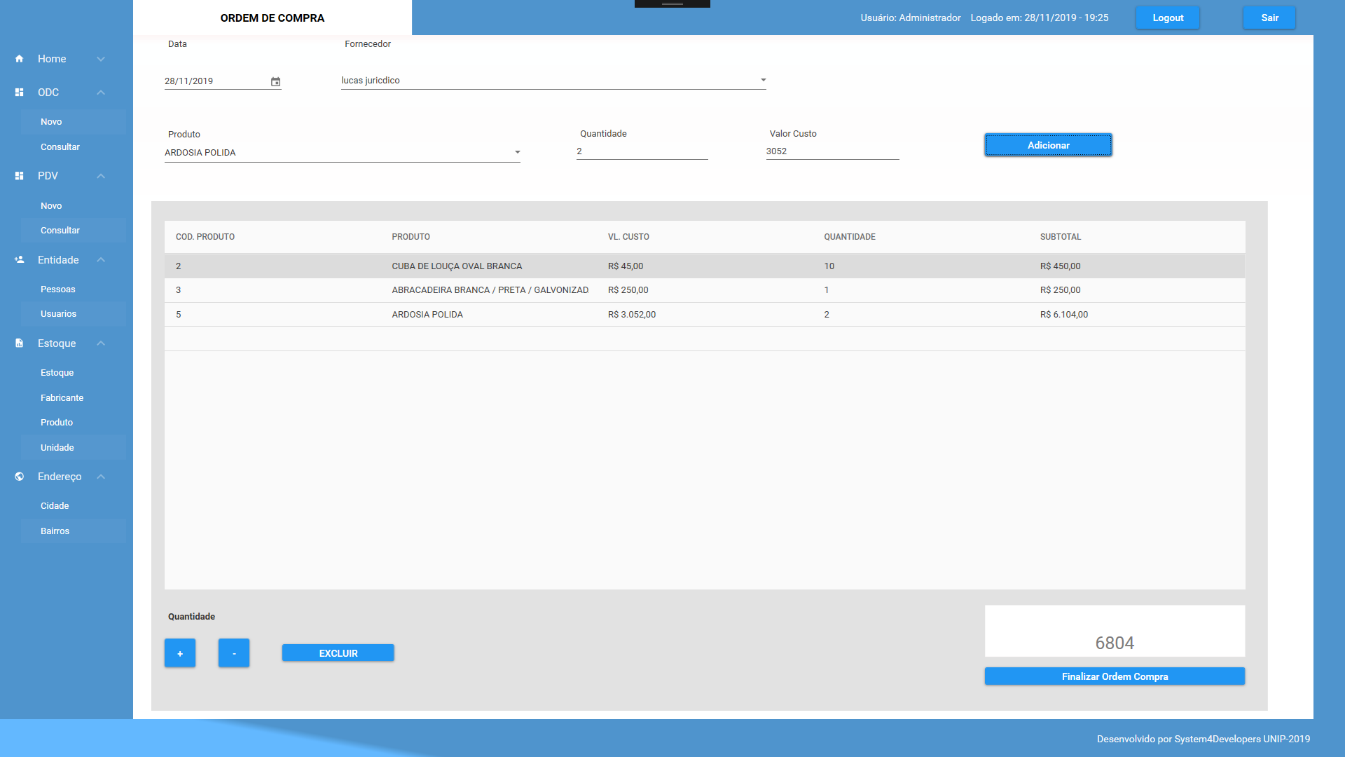
Figura 5.9 – Tela Usuários



Fonte: Autoria Própria, 2019

Em ODC (Ordem de Compra) na opção Novo, é possível visualizar, cadastrar, editar e excluir os produtos que entraram no estoque por meio da uma compra, com opções de fornecedor, nome do produto, quantidade e valor de custo conforme figura 5.10. A ordem de compra é vínculada a um ou vários produtos.

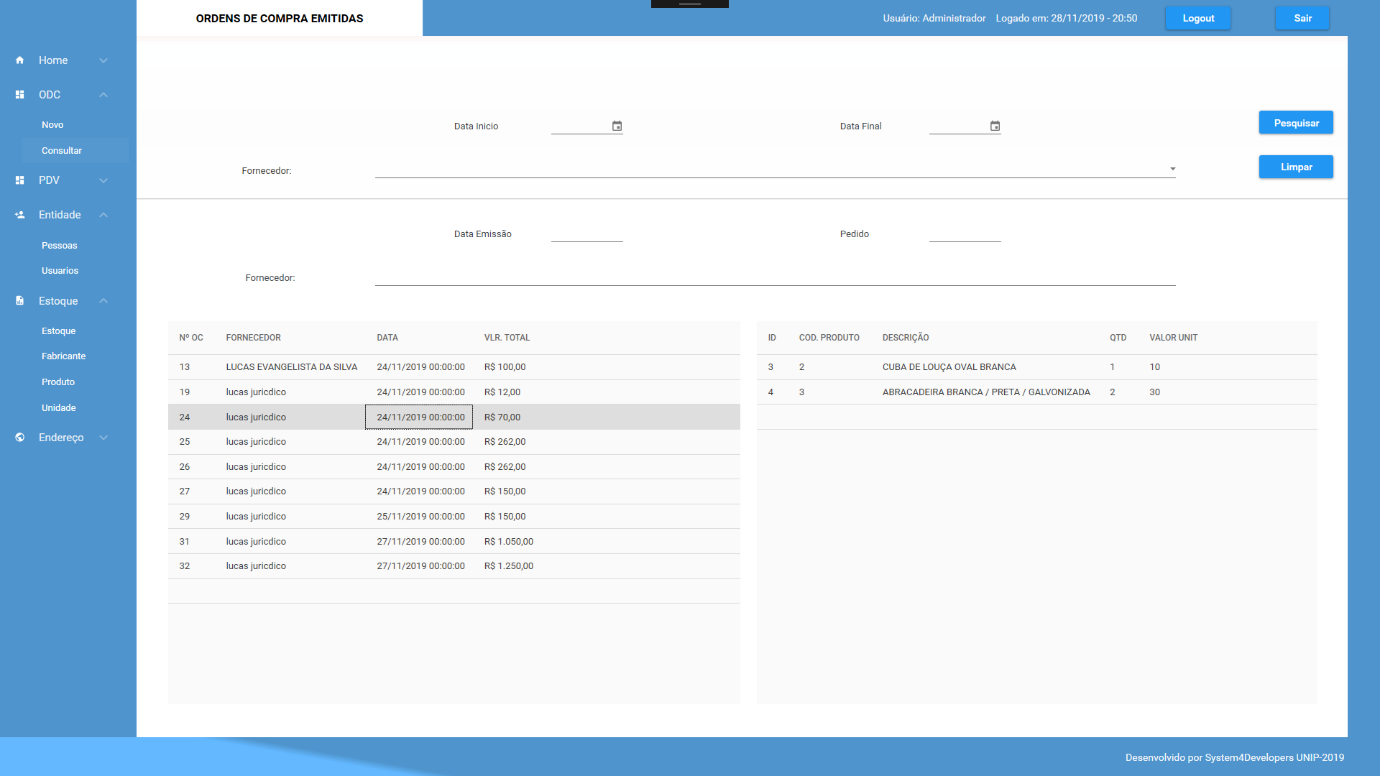
Figura 5.10 – Tela ODC Novo



Fonte: Autoria Própria, 2019

Em ODC (Ordem de Compra) na opção Consultar, é possível visualizar e pesquisar conforme figura 5.11. Na pesquisa é possível realizar um filtro por data inicio, data final e fornecedor. No painel direto exibe os itens/produtos vinculados a ordem de compra que deu entrada no estoque.

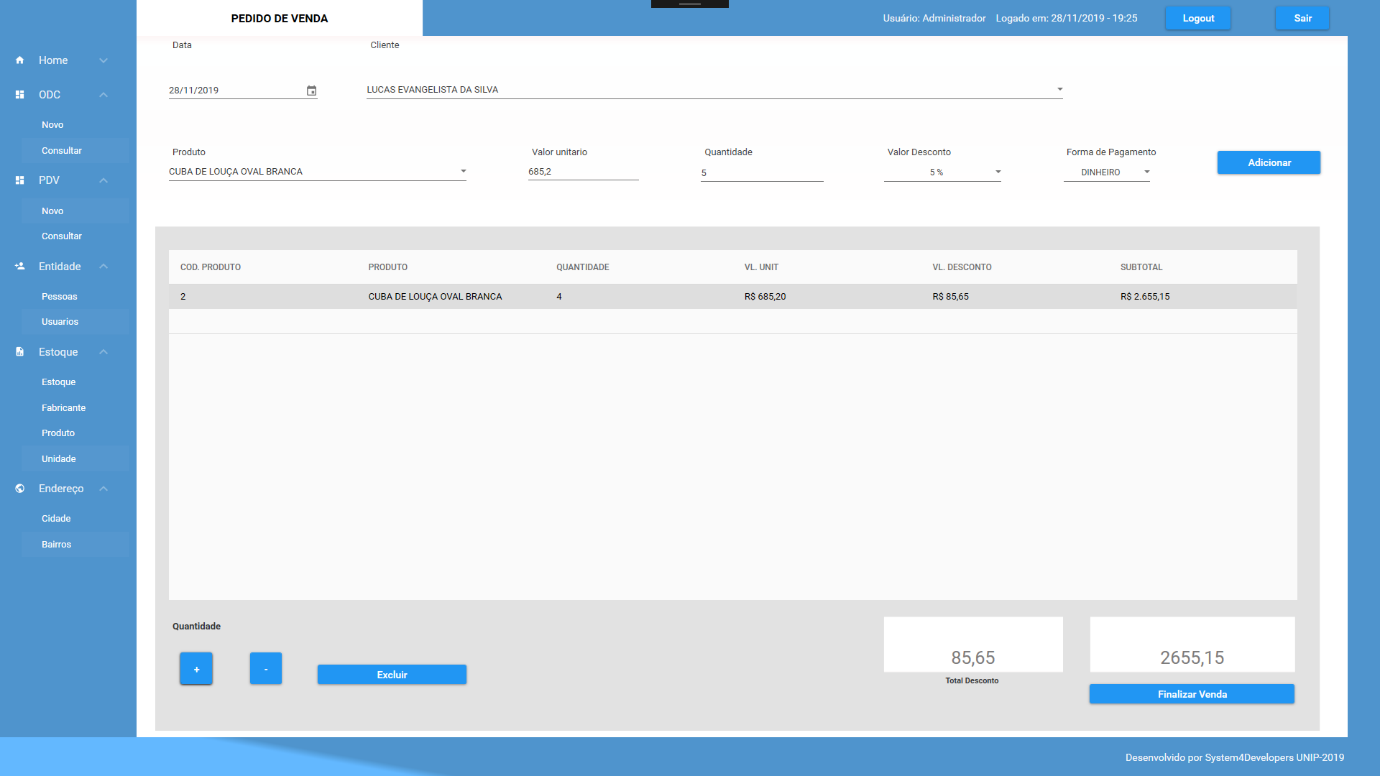
Figura 5.11 – Tela ODC Consultar



Fonte: Autoria Própria, 2019

Na opção Novo do PDV (Ponto de Venda), é possível visualizar, cadastrar, editar e excluir uma venda. Nesta etapa tem a possibilidade de inserir os produtos com o valor unitário, quantidade, valor percentual de desconto sobre o item quando houver, após inserir os produtos da venda, selecionamos a data, cliente e forma de pagamento conforme figura 5.12.

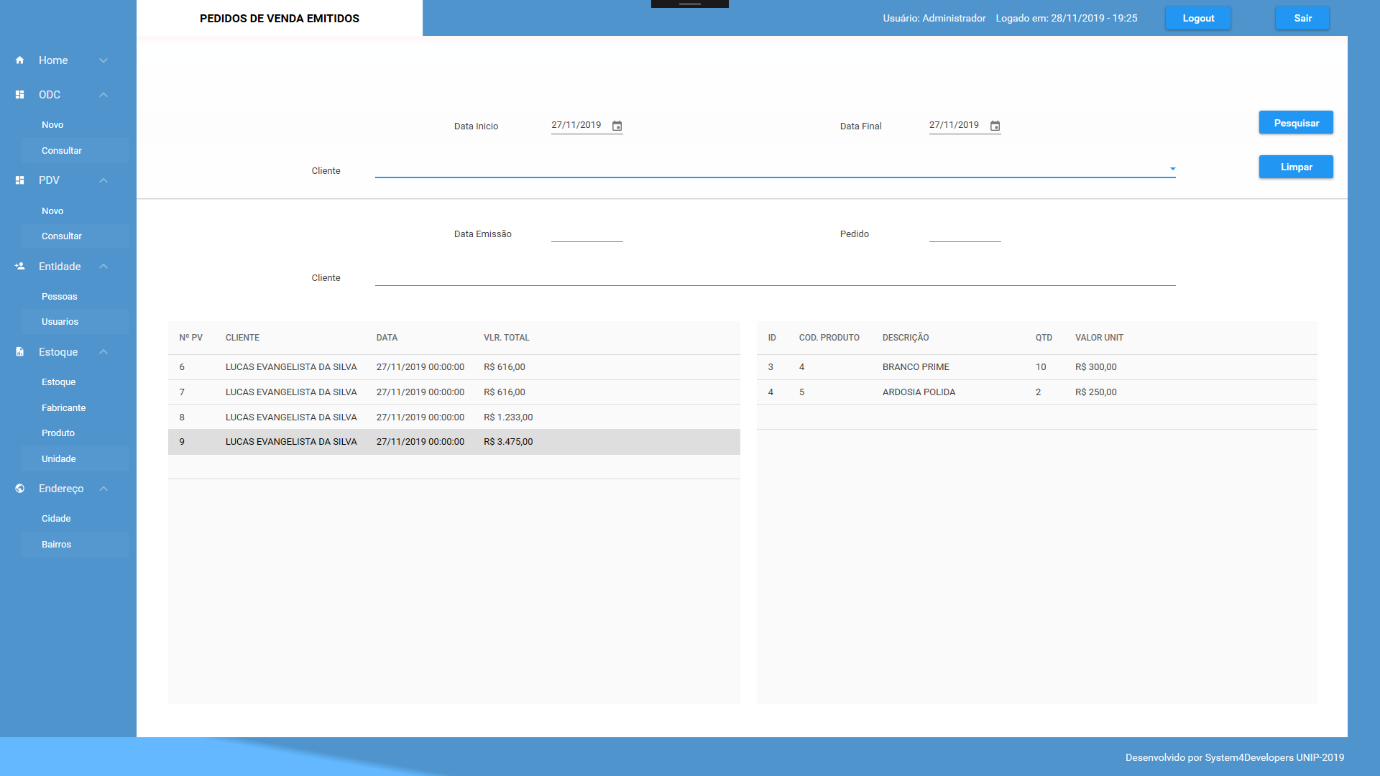
Figura 5.12 – Tela PDV Novo



Fonte: Autoria Própria, 2019

Na opção Consultar do PDV (Ponto de Venda), é possível visualizar e pesquisar as vendas realizadas por meio das opções data inicio, fim e nome do cliente. No painel lateral direito conseguimos visualizar os produtos, quantidade e valor da venda selecionada conforme figura 5.14.

Figura 5.14 – Tela PDV Consultar



Fonte: Autoria Própria, 2019

## 5.2 MELHORIAS FUTURAS

Nesta sessão, explicaremos as possíveis melhorias a serem feitas no sistema com suas respectivas necessidades do cliente.

# 6.0 CONCLUSÃO

A Loja Imperador encontra-se com processos precários em relação ao fluxo de informações, em sua maior parte sem nenhum tipo de controle, totalmente manual. Um exemplo disto é que sem o sistema seria impossível contabilizar as entradas e saídas de produtos em um ano. Com base nisso, foi possível desenvolver um projeto de um sistema de ponto de venda.

Para o desenvolvimento deste projeto foram necessários conhecimentos em Gerenciamento de Projeto de Softwares, Programação Orientada a Objetos II, Projetos de sistemas Orientado a Objetos, Empreendedorismo e Gestão de Qualidade.

Com o projeto finalizado, pode-se concluir que o objetivo deste trabalho foi alcançado com sucesso. No trabalho foi proposto desenvolver e elaborar um projeto de um sistema PDV para solucionar os problemas de gerenciamentos de vendas da Loja Imperador, atendendo suas principais demanadas e aperfeiçoando o processo de controle.

# REFERÊNCIAS BIBLIOGRÁFICAS

SANTINI, Rafael de Melo Lima. **Uma linguagem de domínio específico externa para implementação de testes de unidade e de integração em softwares orientados a objetos. Instruções aos Autores de Contribuições para o SIBGRAPI**,2017. Disponível em: <iago://goo.gl/rc2nTH >. Acesso em 2 de outubro de 2018.

PRESSMAN, Roger S. **Engenharia de software**. 6. i. Rio de Janeiro, RJ: McGraw-Hill, 2006.

RUMBAUGH, James ii. **Modelagem e projetos baseados em objetos**. 2.ed.Rio de Janeiro: Campus, 1994.

PRESSMAN, Roger S. **Engenharia de Software**. 6 ed. Rio de Janeiro: McGraw-Hill, 2002.

CORREIA, Carlos Henrique; TAFNER, Malcon Anderson. **Análise Orientada a Objetos**. Florianópolis: Visual Books, 2006.

PMBOK. **Um Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos**.3.ed. Four Campus Boulevard, Newtown Square, EUA: Project Management Institute, 2004.

MACORATTI, J.C. Padrões de projetos. 2010. **O modelo MVC – Model View Controller**. Disponível em:<iag://www.macoratti.net/vbn\_mvc.htm>. Acesso em 16 de Outubro de 2019.

MACORATTI, J.C. Padrões de projetos. 2018. **Visual Studio – Dica de produtividade – Organizando o seu código.** Disponível em: <iag://www.macoratti.net/18/11/vs17\_dica1.htm>. Acesso em 16 de Outubro de 2019.

MACORATTI, J.C. Padrões de projetos. **.NET - O padrão Singleton.** Disponivel em:<iag://www.macoratti.net/net\_psgt.htm>. Acesso em 16 de Outubro de 2019.

MACORATTI, J.C. Padrões de projetos. **WPF – Revisando conceitos bem básicos. Disponível em: <**iag://www.macoratti.net/11/07/wpf\_conc1.htm>. Acesso em 16 de Outubro de 2019

Associação Brasileira de Normas Técnicas (ABNT). “NBR ISSO/IEC 9126-3:**Engenharia de Software – Qualidade de Produto**”, Parte 1: Modelo de Qualidade, 2003.

ELMASRI, Ramez; NAVATHE, ShamkantB.**SISTEMAS DE BANCO DE DADOS**.6. i. São Paulo: Addison Wesley, 2011.

Bezerra, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML.**2.ed.São Paulo:Editora Campus/Elsevier, 2006.

DATE, C. J..**INTRODUÇÃO A SISTEMAS DE BANCOS DE DADOS**. 8. i. Rio deJaneiro: Elsevier, 2003.

GILLENSON, Mark L. **Fundamentos de Sistemas de Gerência de Banco de Dados**. LTC, 2006.

SILVA, R. P. e. **Como modelar como UML 2**. Visual Books, 2009.

PENDER, Tom. **UML, a Bíblia**, Tradução Daniel Vieira – Rio de Janeiro: Editora Campus, 2004.

RICARTE, Ivan Luiz Marques. **Programação Orientada a Objetos: Uma 61 Abordagem com Java**. UNICAMP, 2001. 118 p. Disponível em <http://www.loopncd.hpg.com.br/Apostilas/poojava.pdf>. Acesso em: 04 Jun. 2009.

GONÇALVES, T. J. ENYO; CORTÉS, MARIELA INÉS. **Computação Análise e Projeto de Sistemas.** 3.ed. Fortaleza – Ceará: Editora EdUECE, 2015.

PUGA, Sandra; RISSETI, Gerson. **Lógica de Programação e Estruturas de Dados.** 1.ed. São Paulo: Pearson, 2004.

GOETTEN Junior, WINCK Diogo. AspectJ – **Programação Orientada a Aspectos com Java**. São Paulo – SP: Novatec Editora, 2006.

# ANEXOS A - TERMO DE ABERTURA DO PROJETO

|  |  |
| --- | --- |
| **1 – Nome do Projeto** | **2 – Código** |
| Loja de Material de Construção | COD45B6 |
| **3 – Líder do Projeto** | **3. 1 – Área de Lotação** |
| Lucas Evangelista da Silva | Regra Negócio e Banco de Dados |
| **3. 2 – E-mail** | **3. 3 – Telefone** |
| [lucasx.silva@hotmail.com](mailto:lucasx.silva@hotmail.com) | 15 988012181 |
| **4 – Gestores do Projeto** | **4. 1 – Área de Lotação** |
| Lucas Evangelista da Silva  Nelson José Barnabé Junior  Nilton Augusto Muniz Junior  Tiago Nunes Teixeira | Líder e Projetista  Back-End e Doc.  Analista e Doc.  Front-End e Doc. |
| **4. 2 – E-mail** | **4. 3 – Telefone** |
| [lucasx.silva@hotmail.com](mailto:lucasx.silva@hotmail.com)  [njunior2593@gmail.com](mailto:njunior2593@gmail.com)  [Nilton\_juninho1@hotmail.com](mailto:Nilton_juninho1@hotmail.com)  [tiago.nunees@yahoo.com.br](mailto:tiago.nunees@yahoo.com.br) | 15 988012181  15 981183646  15 991592292  15 997901006 |
|  | |
| **5. Objetivo do Documento** | |
| Este documento tem como objetivo autorizar formalmente o início de um projeto e contém informações necessárias para o entendimento do projeto, fornecendo uma visão macro do serviço a ser desenvolvido. | |

|  |  |  |  |
| --- | --- | --- | --- |
| **6 – Histórico de Mudança** | | | |
| **Versão** | **Data** | **Descrição** | **Autor** |
| 1.0  \*4 Entregas | 23/08/2019 | Tema de abertura do Projeto | Lucas;Nelson;Tiago;  Nilton |

|  |
| --- |
| **7 – Objetivo do Projeto** |
| Criar um sistema PDV para registro de vendas e controle da saída e entrada de produtos de uma loja de material para construção. |

|  |
| --- |
| **8 – Justificativa** |
| Promover a melhoria do processo de venda e o gerenciamento de produtos e valores relacionados. |

|  |
| --- |
| **9 – Escopo** |
| 1. **Pesquisar:** c#, Camada MVC, Projeto do tipo WPF. 2. **Desenvolver:** Tela PDV para desktop para consulta do estoque via padrão de projeto WPF. 3. **Implementar:** Projeto WPF em um sistema PDV. |

|  |
| --- |
| **10 – Não-Escopo** |
| Desenvolvimento em Java para Mobile. |

|  |  |  |
| --- | --- | --- |
| **11 – Parte Interessada** | **Representante** | **Relacionamento com o Projeto** |
| Loja de material de construção Imperador – Votorantim | Nilton | Parte Interessada |

|  |  |
| --- | --- |
| **12 – Equipe Básica** | **Papel Desempenhado** |
| Lucas Evangelista da Silva | Líder do Grupo e Projetista |
| Nelson Jose Barnabé Junior | Back-End e Documentação |
| Nilton Augusto Muniz Junior | Analista e Documentação |
| Tiago Nunes Teixeira | Front-End e Documentação |
| **13 – Orçamento Previsto** | **14 – Prazo Previsto** |
| R$ 25.000,00 | 07/12/2019 |

|  |  |
| --- | --- |
| **14 – Premissas (Suposições dadas como certaspara oprojeto)** | |
| **1.** | Criação de layouts interativos para realização da venda e consulta dos produtos disponíveis. |
| **2.** | Desenvolver um estoque com categorias separadas por fornecedor e tabela de preços. |
| **3.** | Desenvolvimento da tela de Login. |
| **4.** | Desenvolvimento de Cadastro e Consulta de cliente, usuários e fornecedores. |
| **5.** | Gerenciamento de entrada e saída de produtos |

|  |  |  |
| --- | --- | --- |
| Aprovação | | |
| **Responsável** | **Data** | **Assinatura** |
|  |  |  |
|  |  |  |

*Fica acordado o Projeto de Serviço, favor imprimir 2 vias da Abertura do Projeto e assinar ambas, para cada Responsável possui uma (01) via.*

|  |
| --- |
| **15 – Cronograma** |
| 23/08/2019 - Foi feito a Abertura de Projeto  28/08/2019 - Inicio do Projeto  29/08/2019 - Levantamento de Requisitos  02/09/2019 - Desenvolvimento dos Diagramas de Casos de Uso e de Classe  03/09/2019 – Desenvolvimento dos Diagramas de Sequencia, Estados e Atividade  06/09/2019 - Inicio da Programação e das Telas  07/09/2019 – Desenvolvimento das Classes do Projeto  15/09/2019 - Inicio ao Desenvolvimento da Documentação  17/09/2019 - Desenvolvimento Conceitual, Lógico e Físico do Banco  10/10/2019 – Reunindo todas as atividades desenvolvidas  15/11/2019 - Execução dos Testes do Projeto  20/11/2019 - Desenvolvimento da Designer do Banner  07/12/2019 – Apresentação do Projeto |

|  |
| --- |
| **16 – Atividades Desenvolvidas** |
| - Desenvolvimento da Tela PDV;  - Desenvolvimento do Modelo Conceitual, lógico e físico do banco;  - Desenvolvimento de Diagramas e Fluxogramas;  - Desenvolvimento da Documentação;  - Desenvolvimento da Lógica de Programação; |

# ANEXOS B – MODELO FÍSICO

CREATE DATABASE pimads4

GO

USE pimads4

GO

CREATE TABLE Bairros

(

idBairro INT PRIMARY KEY IDENTITY(1,1),

dsBairro VARCHAR(150) NOT NULL,

fk\_idCidade\_Cidades INT

)

CREATE TABLE Pessoas

(

idPessoa INT PRIMARY KEY IDENTITY(1,1),

tpPessoa VARCHAR(1) NOT NULL,

nmPessoa VARCHAR(100) NOT NULL,

numDocumento VARCHAR(15) NOT NULL,

numRG VARCHAR(12),

dtNascimento DATETIME,

dsEmail VARCHAR(100),

dsEndereco VARCHAR(120),

complemento VARCHAR(10),

numEnd VARCHAR(5),

observacao VARCHAR(100),

fk\_idBairro\_Bairros INT,

)

CREATE TABLE Estados

(

idEstado INT PRIMARY KEY IDENTITY(1,1),

nmEstado VARCHAR(40) NOT NULL,

dsSigla VARCHAR(2) NOT NULL,

codIBGE VARCHAR(7)

)

CREATE TABLE Usuarios

(

idUsuario INT PRIMARY KEY IDENTITY(1,1),

dsLogin VARCHAR(15) NOT NULL,

dsSenha VARCHAR(15) NOT NULL,

nmUsuario VARCHAR(40) NOT NULL,

tpUsuario VARCHAR(10) NOT NULL,

tpStatus VARCHAR(1) NOT NULL

)

CREATE TABLE OrdemCompra

(

idOrdemCompra INT PRIMARY KEY IDENTITY(1,1),

valorTotal FLOAT,

dtDigitacao DATETIME NOT NULL,

tpStatus VARCHAR(1) NOT NULL,

fk\_idUsuario\_Usuarios INT,

fk\_idPessoa\_Pessoas INT

)

CREATE TABLE OrdemCompraProduto

(

idOcProduto INT PRIMARY KEY IDENTITY(1,1),

vlrUnit FLOAT NOT NULL,

quantidade INT NOT NULL,

fk\_idOrdemCompra\_OrdemCompra INT,

fk\_idProduto\_Produtos INT

)

CREATE TABLE PedidoVenda

(

idPedidoVenda INT PRIMARY KEY IDENTITY(1,1),

valorTotal FLOAT,

dtDigitacao DATETIME NOT NULL,

tpPagamento VARCHAR(15) NOT NULL,

tpStatus VARCHAR(1) NOT NULL,

fk\_idPessoa\_Pessoas INT,

fk\_idUsuario\_Usuarios INT

)

CREATE TABLE Cidades

(

idCidade INT PRIMARY KEY IDENTITY(1,1),

nmCidade VARCHAR(150) NOT NULL,

codIBGE VARCHAR(7),

fk\_idEstado\_Estados INT

)

CREATE TABLE PedidoVendaProduto

(

idPedidoVendaProduto INT PRIMARY KEY IDENTITY(1,1),

vlrUnit FLOAT NOT NULL,

quantidade INT NOT NULL,

desconto INT,

fk\_idPedidoVenda\_PedidoVenda INT,

fk\_idProduto\_Produtos INT,

)

CREATE TABLE Produtos

(

idProduto INT PRIMARY KEY IDENTITY(1,1),

quantidade INT,

dsProduto VARCHAR(100) NOT NULL,

valorVenda FLOAT,

valorCusto FLOAT,

tpProduto VARCHAR(1) NOT NULL,

fk\_idUnidade\_Unidades INT,

fk\_idFabricante\_Fabricantes INT

)

CREATE TABLE Fabricantes

(

idFabricante INT PRIMARY KEY IDENTITY(1,1),

nmFabricante VARCHAR(50) NOT NULL

)

CREATE TABLE Unidades

(

idUnidade INT PRIMARY KEY IDENTITY(1,1),

dsUnidade VARCHAR(5) NOT NULL

)

ALTER TABLE Bairros ADD FOREIGN KEY(fk\_idCidade\_Cidades) REFERENCES Cidades(idCidade)

ALTER TABLE Pessoas ADD FOREIGN KEY(fk\_idBairro\_Bairros) REFERENCES Bairros (idBairro)

ALTER TABLE OrdemCompra ADD FOREIGN KEY(fk\_idUsuario\_Usuarios) REFERENCES Usuarios (idUsuario)

ALTER TABLE OrdemCompra ADD FOREIGN KEY(fk\_idPessoa\_Pessoas) REFERENCES Pessoas (idPessoa)

ALTER TABLE OrdemCompraProduto ADD FOREIGN KEY(fk\_idOrdemCompra\_OrdemCompra) REFERENCES OrdemCompra (idOrdemCompra)

ALTER TABLE OrdemCompraProduto ADD FOREIGN KEY(fk\_idProduto\_Produtos) REFERENCES Produtos (idProduto)

ALTER TABLE PedidoVenda ADD FOREIGN KEY (fk\_idPessoa\_Pessoas) REFERENCES Pessoas (idPessoa)

ALTER TABLE PedidoVenda ADD FOREIGN KEY (fk\_idUsuario\_Usuarios) REFERENCES Usuarios(idUsuario)

ALTER TABLE Cidades ADD FOREIGN KEY(fk\_idEstado\_Estados) REFERENCES Estados (idEstado)

ALTER TABLE PedidoVendaProduto ADD FOREIGN KEY(fk\_idPedidoVenda\_PedidoVenda) REFERENCES PedidoVenda (idPedidoVenda)

ALTER TABLE PedidoVendaProduto ADD FOREIGN KEY(fk\_idProduto\_Produtos) REFERENCES Produtos (idProduto)

ALTER TABLE Produtos ADD FOREIGN KEY(fk\_idUnidade\_Unidades) REFERENCES Unidades(idUnidade)

ALTER TABLE Produtos ADD FOREIGN KEY(fk\_idFabricante\_Fabricantes) REFERENCES Fabricantes(idFabricante)

**PROCEDURES ManterBairros**

**sp\_AtualizarBairro**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarBairro

@idBairro int,

@dsBairro varchar(150),

@fk\_idCidade\_Cidades int

AS

BEGIN

UPDATE Bairros

SET dsBairro = @dsBairro

,fk\_idCidade\_Cidades = @fk\_idCidade\_Cidades

WHERE idBairro = @idBairro

END

**sp\_CadastrarBairro**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarBairro

@dsBairro varchar(150),

@fk\_idCidade\_Cidades INT

AS

BEGIN

INSERT INTO Bairros

(

dsBairro

,fk\_idCidade\_Cidades

)

VALUES

(

@dsBairro

,@fk\_idCidade\_Cidades

)

END

**sp\_ConsultarBairroById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarBairroById

@idBairro int

AS

BEGIN

SELECT \* FROM Bairro s

JOIN Cidades on Bairros.fk\_idCidade\_Cidades = Cidades.idCidade

WHERE idBairro = @idBairro

END

**sp\_ExcluirBairro**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirBairro

(

@idBairro int

)

AS BEGIN

DELETE FROM Bairros

WHERE idBairro=@idBairro

END

**PROCEDURES ManterCidades**

**sp\_AtualizarCidade**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarCidade

@idCidade int,

@nmCidade varchar(150),

@codIBGE varchar(7),

@fk\_idEstado\_Estados int

AS

BEGIN

UPDATE Cidades

SET nmCidade = @nmCidade,

codIBGE=@codIBGE,

fk\_idEstado\_Estados = @fk\_idEstado\_Estados

WHERE idCidade = @idCidade

END

**sp\_CadastrarCidade**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarCidade

@nmCidade varchar(150),

@codIBGE varchar(7),

@fk\_idEstado\_Estados int

AS

BEGIN

INSERT INTO Cidades

(

nmCidade

,codIBGE

,fk\_idEstado\_Estados

)

VALUES

(

@nmCidade

,@codIBGE

,@fk\_idEstado\_Estados

)

END

**sp\_ConsultarCidadeById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarCidadeById

@idCidade int

AS

BEGIN

SELECT \* FROM Cidades

WHERE idCidade = @idCidade

END

**sp\_ExcluirCidade**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirCidade

(

@idCidade int

)

AS BEGIN

DELETE FROM Cidades

WHERE idCidade=@idCidade

END

**PROCEDURES ManterFabricantes**

**sp\_AtualizarFabricante**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarFabricante

@idFabricante int,

@nmFabricante varchar(50)

AS

BEGIN

UPDATE fabricantes

SET nmFabricante = @nmFabricante

WHERE idFabricante = @idFabricante

END

**sp\_CadastrarFabricante**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarFabricante

@nmFabricante varchar(50)

AS

BEGIN

INSERT INTO fabricantes (nmFabricante)

VALUES(@nmFabricante)

END

**sp\_ConsultarFabricanteById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarFabricanteById

@idFabricante int

AS

BEGIN

SELECT \* FROM fabricantes

WHERE idFabricante = @idFabricante

END

**sp\_ExcluirFabricante**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirFabricante

(

@idFabricante int

)

AS BEGIN

DELETE FROM Fabricantes

WHERE idFabricante=@idFabricante

END

**PROCEDURES ManterOrdemCompra**

**sp\_AtualizarOrdemCompra**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarOrdemCompra

@idOrdemCompra INT

,@dtDigitacao DATETIME

,@tpStatus VARCHAR(1)

,@fk\_idUsuario\_Usuarios INT

,@fk\_idPessoa\_Pessoas INT

AS

BEGIN

UPDATE OrdemCompra

SET dtDigitacao = @dtDigitacao

,tpStatus = @tpStatus

,fk\_idUsuario\_Usuarios=@fk\_idUsuario\_Usuarios

,fk\_idPessoa\_Pessoas=@fk\_idPessoa\_Pessoas

WHERE idOrdemCompra = @idOrdemCompra

END

**sp\_CadastrarOrdemCompra**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarOrdemCompra

@dtDigitacao DATETIME

,@tpStatus VARCHAR(1)

,@fk\_idUsuario\_Usuarios INT

,@fk\_idPessoa\_Pessoas INT

AS

BEGIN

INSERT INTO OrdemCompra

(

dtDigitacao

,valorTotal

,tpStatus

,fk\_idUsuario\_Usuarios

,fk\_idPessoa\_Pessoas

)

VALUES

(

@dtDigitacao

,0

,@tpStatus

,@fk\_idUsuario\_Usuarios

,@fk\_idPessoa\_Pessoas

)

END

**sp\_ConsultarOrdemCompraById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarOrdemCompraById

@idOrdemCompra int

AS

BEGIN

SELECT \* FROM OrdemCompra

WHERE idOrdemCompra = @idOrdemCompra

END

**sp\_ExcluirOrdemCompra**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirOrdemCompra

(

@idOrdemCompra int

)

AS BEGIN

DELETE FROM OrdemCompra

WHERE idOrdemCompra=@idOrdemCompra

END

**PROCEDURES ManterOrdemCompraProduto**

**sp\_AtualizarOrdemCompraProduto**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarOrdemCompraProduto

@idOcProduto INT

,@vlrUnit FLOAT

,@quantidade INT

,@fk\_idProduto\_Produtos INT

AS

BEGIN

UPDATE OrdemCompraProduto

SET vlrUnit = @vlrUnit

,quantidade = @quantidade

,fk\_idProduto\_Produtos=@fk\_idProduto\_Produtos

WHERE idOcProduto = @idOcProduto

END

**sp\_CadastrarOrdemCompraProduto**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarOrdemCompraProduto

@vlrUnit FLOAT

,@quantidade INT

,@fk\_idOrdemCompra\_OrdemCompra INT

,@fk\_idProduto\_Produtos INT

AS

BEGIN

INSERT INTO OrdemCompraProduto

(

vlrUnit

,quantidade

,fk\_idOrdemCompra\_OrdemCompra

,fk\_idProduto\_Produtos

)

VALUES

(

@vlrUnit

,@quantidade

,@fk\_idOrdemCompra\_OrdemCompra

,@fk\_idProduto\_Produtos

)

END

**sp\_ConsultarOrdemCompraProdutoById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarOrdemCompraProdutoById

@idOcProduto INT

AS

BEGIN

SELECT \* FROM OrdemCompraProduto

WHERE idOcProduto = @idOcProduto

END

**sp\_ExcluirOrdemCompraProduto**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirOrdemCompraProduto

(

@idOcProduto INt

)

AS BEGIN

DELETE FROM OrdemCompraProduto

WHERE idOcProduto=@idOcProduto

END

**PROCEDURES ManterPedidoVenda**

**sp\_AtualizarPedidoVenda**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarPedidoVenda

@idPedidoVenda INT

,@dtDigitacao DATETIME

,@tpPagamento VARCHAR(15)

,@tpStatus VARCHAR(1)

,@fk\_idPessoa\_Pessoas INT

,@fk\_idUsuario\_Usuarios INT

AS

BEGIN

UPDATE PedidoVenda

SET dtDigitacao = @dtDigitacao

,tpPagamento = @tpPagamento

,tpStatus = @tpStatus

,fk\_idPessoa\_Pessoas=@fk\_idPessoa\_Pessoas

,fk\_idUsuario\_Usuarios=@fk\_idUsuario\_Usuarios

WHERE @idPedidoVenda = @idPedidoVenda

END

**sp\_CadastrarPedidoVenda**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarPedidoVenda

@dtDigitacao DATETIME

,@tpPagamento VARCHAR(15)

,@tpStatus VARCHAR(1)

,@fk\_idPessoa\_Pessoas INT

,@fk\_idUsuario\_Usuarios INT

AS

BEGIN

INSERT INTO PedidoVenda

(

valorTotal

,dtDigitacao

,tpPagamento

,tpStatus

,fk\_idPessoa\_Pessoas

,fk\_idUsuario\_Usuarios

)

VALUES

(

0

,@dtDigitacao

,@tpPagamento

,@tpStatus

,@fk\_idPessoa\_Pessoas

,@fk\_idUsuario\_Usuarios

)

END

**sp\_ConsultarPedidoVendaById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarPedidoVendaById

@idPedidoVenda int

AS

BEGIN

SELECT \* FROM PedidoVenda

WHERE idPedidoVenda = @idPedidoVenda

END

**sp\_ExcluirPedidoVenda**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirPedidoVenda

(

@idPedidoVenda int

)

AS BEGIN

DELETE FROM PedidoVenda

WHERE idPedidoVenda=@idPedidoVenda

END

**PROCEDURES ManterPedidoVendaProduto**

**sp\_AtualizarPedidoVendaProduto**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarPedidoVendaProduto

@idPedidoVendaProduto INT

,@vlrUnit FLOAT

,@quantidade INT

,@desconto INT

,@fk\_idProduto\_Produtos INT

AS

BEGIN

UPDATE PedidoVendaProduto

SET vlrUnit = @vlrUnit

,quantidade = @quantidade

,desconto = @desconto

,fk\_idProduto\_Produtos=@fk\_idProduto\_Produtos

WHERE idPedidoVendaProduto = @idPedidoVendaProduto

END

**sp\_CadastrarPedidoVendaProduto**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarPedidoVendaProduto

@vlrUnit FLOAT

,@quantidade INT

,@desconto INT

,@fk\_idPedidoVenda\_PedidoVenda INT

,@fk\_idProduto\_Produtos INT

AS

BEGIN

INSERT INTO PedidoVendaProduto

(

vlrUnit

,quantidade

,desconto

,fk\_idPedidoVenda\_PedidoVenda

,fk\_idProduto\_Produtos

)

VALUES

(

@vlrUnit

,@quantidade

,@desconto

,@fk\_idPedidoVenda\_PedidoVenda

,@fk\_idProduto\_Produtos

)

END

**sp\_ConsultarPedidoVendaProdutoById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarPedidoVendaProdutoById

@idPedidoVendaProduto INT

AS

BEGIN

SELECT \* FROM PedidoVendaProduto

WHERE idPedidoVendaProduto = @idPedidoVendaProduto

END

**sp\_ExcluirPedidoVendaProduto**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirPedidoVendaProduto

(

@idPedidoVendaProduto INT

)

AS BEGIN

DELETE FROM PedidoVendaProduto

WHERE idPedidoVendaProduto=@idPedidoVendaProduto

END

**PROCEDURES ManterPessoas**

**sp\_AtualizarPessoa**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarPessoa

@idPessoa int,

@tpPessoa VARCHAR(1),

@nmPessoa VARCHAR(100),

@numDocumento VARCHAR(15),

@numRG VARCHAR(12),

@dtNascimento DATETIME,

@dsEmail VARCHAR(100),

@dsEndereco VARCHAR(120),

@complemento VARCHAR(10),

@numEnd VARCHAR(5),

@observacao VARCHAR(100),

@fk\_idBairro\_Bairros INT

AS

BEGIN

UPDATE Pessoas

SET tpPessoa = @tpPessoa

,nmPessoa=@nmPessoa

,numDocumento=@numDocumento

,numRG=@numRG

,dtNascimento=@dtNascimento

,dsEmail=@dsEmail

,dsEndereco=@dsEndereco

,complemento=@complemento

,numEnd=@numEnd

,observacao=@observacao

,fk\_idBairro\_Bairros=@fk\_idBairro\_Bairros

WHERE idPessoa = @idPessoa

END

**sp\_CadastrarPessoa**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarPessoa

@tpPessoa VARCHAR(10)

,@nmPessoa VARCHAR(100)

,@numDocumento VARCHAR(15)

,@numRG VARCHAR(12)

,@dtNascimento DATETIME

,@dsEmail VARCHAR(100)

,@dsEndereco VARCHAR(120)

,@complemento VARCHAR(10)

,@numEnd VARCHAR(5)

,@observacao VARCHAR(100)

,@fk\_idBairro\_Bairros INT

AS

BEGIN

INSERT INTO Pessoas

(

tpPessoa

,nmPessoa

,numDocumento

,numRG

,dtNascimento

,dsEmail

,dsEndereco

,complemento

,numEnd

,observacao

,fk\_idBairro\_Bairros

)

VALUES

(

@tpPessoa

,@nmPessoa

,@numDocumento

,@numRG

,@dtNascimento

,@dsEmail

,@dsEndereco

,@complemento

,@numEnd

,@observacao

,@fk\_idBairro\_Bairros

)

END

**sp\_ConsultarPessoaById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarPessoaById

@idPessoa int

AS

BEGIN

SELECT \* FROM Pessoas

JOIN Bairros on Bairros.idBairro = Pessoas.fk\_idBairro\_Bairros

JOIN Cidades on Cidades.idCidade = Bairros.fk\_idCidade\_Cidades

WHERE idPessoa = @idPessoa

END

**sp\_ExcluirPessoa**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirPessoa

(

@idPessoa int

)

AS BEGIN

DELETE FROM Pessoas

WHERE idPessoa=@idPessoa

END

**PROCEDURES ManterProdutos**

**sp\_AtualizarProduto**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarProduto

@idProduto int

,@dsProduto varchar(100)

,@valorVenda float

,@valorCusto float

,@tpProduto varchar(1)

,@fk\_idUnidade\_Unidades int

,@fk\_idFabricante\_Fabricantes int

AS

BEGIN

UPDATE Produtos

SET dsProduto = @dsProduto

,valorVenda = @valorVenda

,valorCusto=@valorCusto

,tpProduto=@tpProduto

,fk\_idUnidade\_Unidades=@fk\_idUnidade\_Unidades

,fk\_idFabricante\_Fabricantes=@fk\_idFabricante\_Fabricantes

WHERE idProduto = @idProduto

END

**sp\_CadastrarProduto**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarProduto

@dsProduto VARCHAR(100)

,@valorVenda FLOAT

,@valorCusto FLOAT

,@tpProduto VARCHAR(1)

,@fk\_idUnidade\_Unidades INT

,@fk\_idFabricante\_Fabricantes INT

AS

BEGIN

INSERT INTO Produtos

(

quantidade

,dsProduto

,valorVenda

,valorCusto

,tpProduto

,fk\_idUnidade\_Unidades

,fk\_idFabricante\_Fabricantes

)

VALUES

(

0

,@dsProduto

,@valorVenda

,@valorCusto

,@tpProduto

,@fk\_idUnidade\_Unidades

,@fk\_idFabricante\_Fabricantes

)

END

**sp\_ConsultarProdutoById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarProdutoById

@idProduto int

AS

BEGIN

SELECT \* FROM Produtos

WHERE idProduto = @idProduto

END

**sp\_ExcluirProduto**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirProduto

(

@idProduto int

)

AS BEGIN

DELETE FROM Produtos

WHERE idProduto=@idProduto

END

**PROCEDURES ManterUnidades**

**sp\_AtualizarUnidade**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarUnidade

@idUnidade int,

@dsUnidade varchar(5)

AS

BEGIN

UPDATE Unidades

SET dsUnidade = @dsUnidade

WHERE idUnidade = @idUnidade

END

**sp\_CadastrarUnidade**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarUnidade

@dsUnidade varchar(5)

AS

BEGIN

INSERT INTO Unidades (dsUnidade)

VALUES(@dsUnidade)

END

**sp\_ConsultarUnidadeById**

use pimads4

go

CREATE PROCEDURE sp\_ConsultarUnidadeById

@idUnidade int

AS

BEGIN

SELECT \* FROM Unidades

WHERE idUnidade = @idUnidade

END

**sp\_ExcluirUnidade**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirUnidade

(

@idUnidade int

)

AS BEGIN

DELETE FROM Unidades

WHERE idUnidade=@idUnidade

END

**PROCEDURES ManterUsuarios**

**sp\_AtualizarUsuario**

use pimads4

go

CREATE PROCEDURE sp\_AtualizarUsuario

@idUsuario int

,@dsLogin varchar(15)

,@dsSenha varchar(15)

,@tpStatus varchar(1)

,@nmUsuario varchar(40)

,@tpUsuario varchar(10)

AS

BEGIN

UPDATE Usuarios

SET dsLogin = @dsLogin

,dsSenha =@dsSenha

,tpStatus =@tpStatus

,nmUsuario =@nmUsuario

,tpUsuario = @tpUsuario

WHERE idUsuario = @idUsuario

END

**sp\_CadastrarUsuario**

use pimads4

go

CREATE PROCEDURE sp\_CadastrarUsuario

@dsLogin varchar(15)

,@dsSenha varchar(15)

,@tpStatus varchar(1)

,@nmUsuario varchar(40)

,@tpUsuario varchar(10)

AS

BEGIN

INSERT INTO Usuarios

(

dsLogin

,dsSenha

,tpStatus

,nmUsuario

,tpUsuario

)

VALUES

(

@dsLogin

,@dsSenha

,@tpStatus

,@nmUsuario

,@tpUsuario

)

END

**sp\_ConsultarUsuarioById**

USE [pimads4]

GO

CREATE PROCEDURE sp\_ConsultarUsuarioById

@idUsuario int

AS

BEGIN

SELECT \* FROM Usuarios

WHERE idUsuario = @idUsuario

END

**sp\_ExcluirUsuario**

USE pimads4

GO

CREATE PROCEDURE sp\_ExcluirUsuario

(

@idUsuario int

)

AS BEGIN

DELETE FROM Usuarios

WHERE idUsuario=@idUsuario

END