



Estácio

Polo Tijuca

Aluno	Luiz Felipe Sarmiento Bonet
Matrícula	2023.09.34276-6
Curso	Desenvolvimento Full Stack
Disciplina	RPG0017 – Vamos Integrar Sistemas
Turma	9001
Período	2024 – 2º Semestre

Missão Prática: Mundo 3 - Nível 3

Título da Prática 1: Camadas de Persistência e Controle

Título da Prática 2: Interface Cadastral com *servlet* e JSPs

Título da Prática 3: Melhorando o *Design* da Interface

Objetivos:

- Implementar persistência com base em JPA;
- Implementar regras de negócio na plataforma JEE, através de EJBs;
- Implementar sistema cadastral *Web* com base em servlets e JSPs;
- Utilizar o framework Bootstrap para melhoria do design;
- Criar os elementos necessários para exibição e entrada de dados na plataforma Java *Web*.

Resultados de Execução – Prática 1

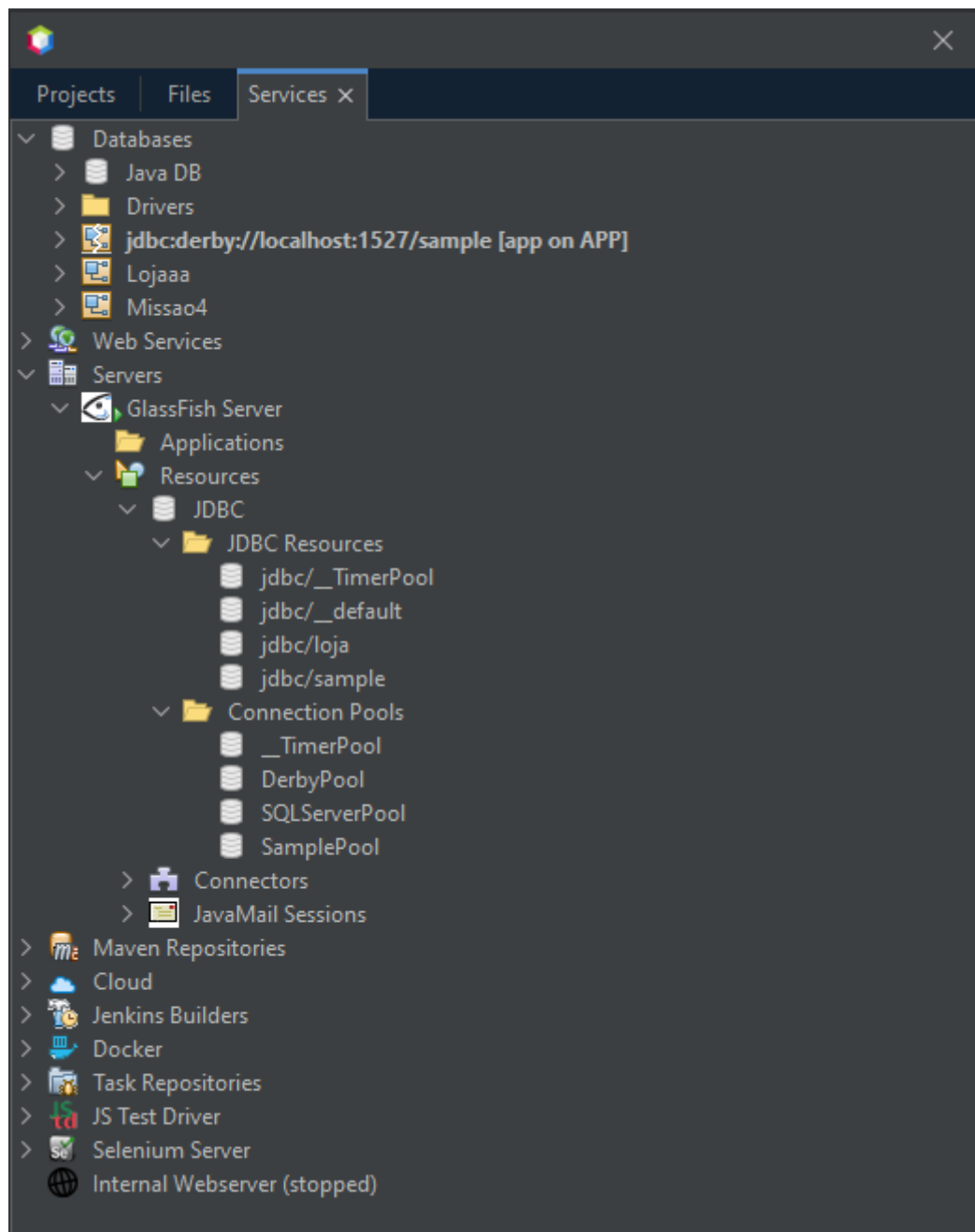


Figura 1. Aba de serviços do NetBeans mostrando as conexões com o banco de dados geradas no procedimento.

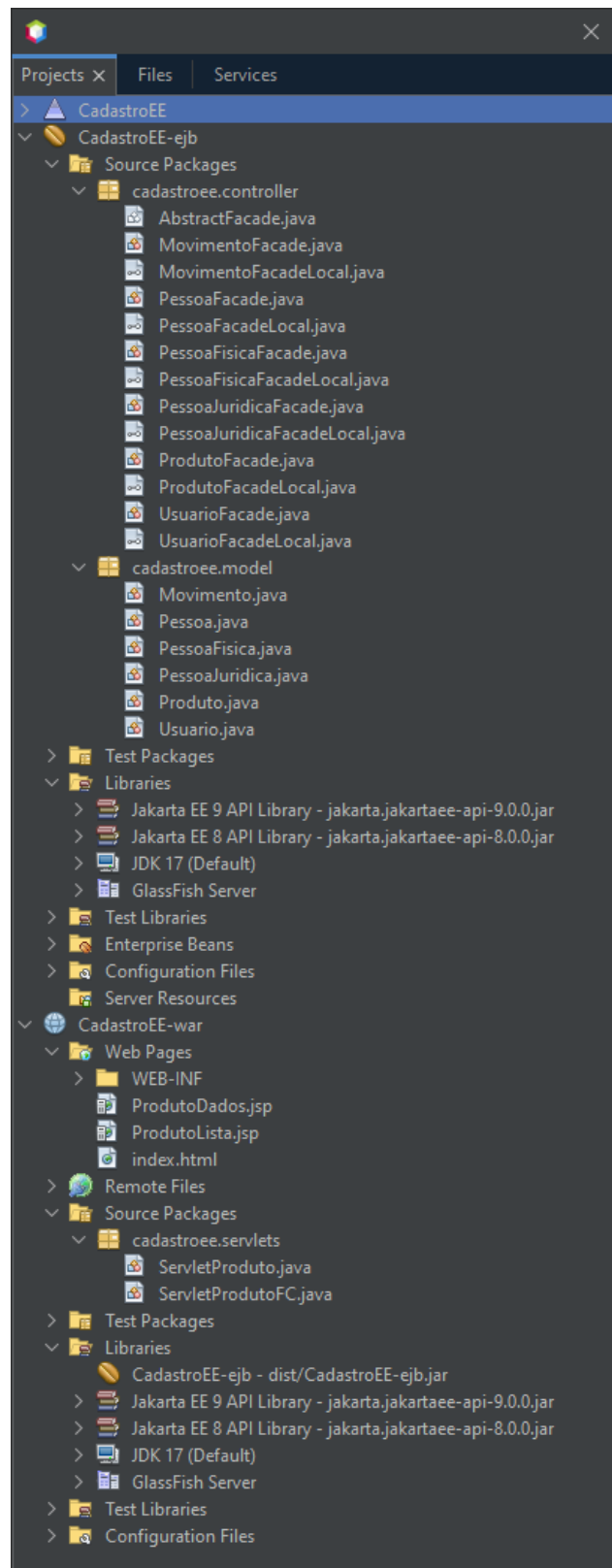


Figura 2. Aba de projetos mostrando muitos dos arquivos gerados no procedimento.

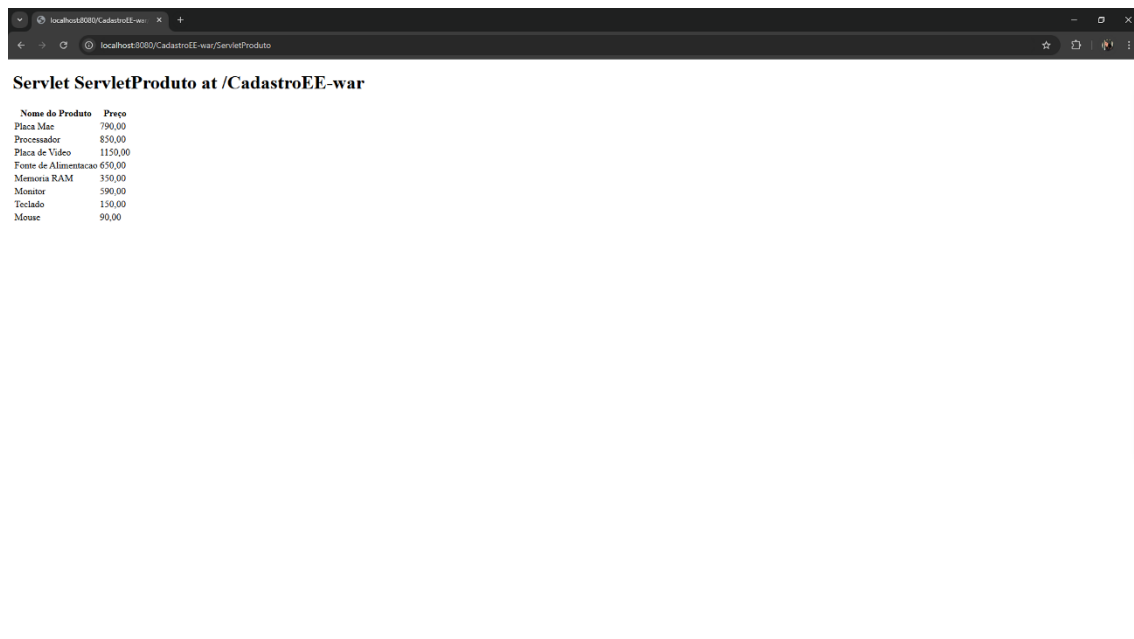
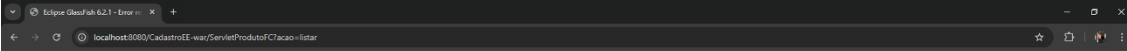


Figura 3. Página básica criada pelo *servlet* ServletProduto.

Resultados de Execução – Prática 2

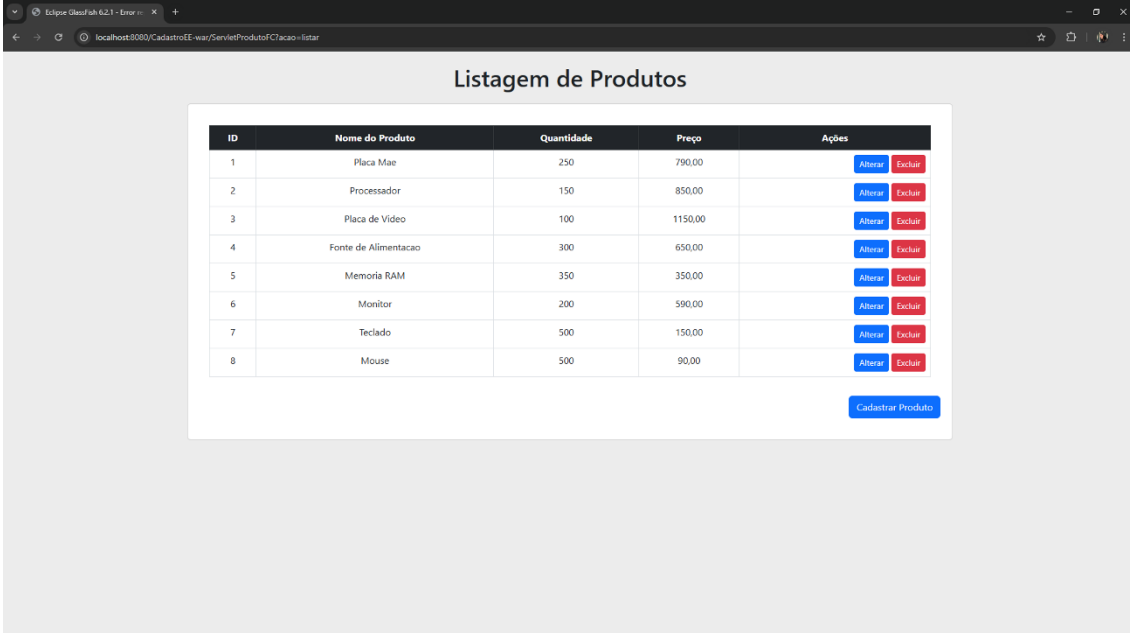


Listagem de Produtos

ID	Nome do Produto	Quantidade	Preço	Ações
1	Placa Mae	250	790,00	Alterar Excluir
2	Processador	150	850,00	Alterar Excluir
3	Placa de Video	100	1150,00	Alterar Excluir
4	Fonte de Alimentacao	300	650,00	Alterar Excluir
5	Memoria RAM	350	330,00	Alterar Excluir
6	Monitor	200	590,00	Alterar Excluir
7	Teclado	500	150,00	Alterar Excluir
8	Mouse	500	90,00	Alterar Excluir

Figura 3. Página simples criada pelo *serv/let* ServletProdutoFC ao acessar a opção de lista.

Resultados de Execução – Prática 3



The screenshot shows a web browser window with the URL `localhost:8080/CadastroEE-wa/ServletProdutoFC?acao=lista`. The page title is "Listagem de Produtos". It contains a table with the following data:

ID	Nome do Produto	Quantidade	Preço	Ações
1	Placa Mae	250	790,00	Alterar Excluir
2	Processador	150	850,00	Alterar Excluir
3	Placa de Video	100	1150,00	Alterar Excluir
4	Fonte de Alimentacao	300	650,00	Alterar Excluir
5	Memoria RAM	350	350,00	Alterar Excluir
6	Monitor	200	590,00	Alterar Excluir
7	Teclado	500	150,00	Alterar Excluir
8	Mouse	500	90,00	Alterar Excluir

At the bottom right of the table, there is a button labeled "Cadastrar Produto".

Figura 4. Página criada pelo *servlet* `ServletProdutoFC`, ao acessar a opção de lista, com formatação complexa feita através do framework Bootstrap.

Análise e Conclusão – Prática 1

A) Como é organizado um projeto corporativo no NetBeans?

Existem diferentes tipos de projetos corporativos na plataforma, que seguem um padrão comum. Projetos corporativos no NetBeans são organizados com uma estrutura modular e de maneira hierárquica, de modo que o projeto principal encapsula outros projetos, cada um com sua função. O programa também oferece ferramentas que ajudam o desenvolvedor, como gerenciamento de dependências, controle de versionamento, etc.

B) Qual o papel das tecnologias JPA e EJB na construção de um aplicativo para a plataforma Web no ambiente Java?

O *Java Persistence API* (JPA) é, como o nome indica, é uma API de persistência de dados que funciona, de certo modo, como interface entre a aplicação e o banco de dados. A ferramenta facilita também a interação com diferentes tipos de bancos de dados. Em contrapartida, os *Enterprise JavaBeans* (EJB) são componentes que encapsulam a lógica de negócios de um projeto em componentes individuais. Com isso, aumenta o controle, segurança e escalabilidade do projeto.

C) Como o NetBeans viabiliza a melhoria de produtividade ao lidar com as tecnologias JPA e EJB?

O NetBeans oferece suporte à essas tecnologias de maneira nativa. Além disso, ainda oferece ferramentas que expandem suas funcionalidades ou facilitam sua utilização, como geração automática de códigos, depuração, mapeamento de entidades, entre outras. Em conjunto, tudo isso facilita o trabalho do desenvolvedor e agiliza o processo de desenvolvimento.

D) O que são servlets, e como o NetBeans oferece suporte à construção desse tipo de componente em um projeto Web?

Servlets são componentes de classes Java que são capazes de estender e dinamizar o funcionamento de *web servers*. Estes componentes rodam no lado do servidor e são capazes de executar vários tipos distintos de tarefas. Além de oferecer assistentes para criação de novos *servlets*, o NetBeans também oferece ferramentas de produtividade como geração de algumas parte de código, depuração, realce de sintaxe, etc.

E) Como é feita a comunicação entre os *servlets* e os *Session Beans* do *pool* de EJBs?

É feita através do *Java Naming and Directory Interface* (JNDI), que permite que os *servlets* chamem métodos nos *Session Beans* em objetos locais. Para isso, o *servlets* precisa saber o endereço correto no JNDI. Também pode ser feita através de injeção de dependência que, de maneira similar ao JNDI, permite o acesso a métodos do *Session Bean* adequado.

Análise e Conclusão – Prática 2

A) Como funciona o padrão *Front Controller*, e como ele é implementado em um aplicativo Web Java, na arquitetura MVC?

O *Front Controller* centraliza tarefas como interpretar solicitações, atribuir tarefas para os modelos corretos e selecionar a visualização correta a ser usada como resposta, em vez de utilizar diversos controladores. No *Model View Controller* (MVC), o *Front Controller* fica responsável por receber todas as requisições do cliente e decidir qual controlador deve lidar com a solicitação.

B) Quais as diferenças e semelhanças entre *servlets* e JSPs?

Ambas são ferramentas que geram conteúdo HTML de maneira dinâmica através de comunicação com o servidor de aplicações. Entretanto, os *servlets* são classes Java que estendem a funcionalidade de servidores e geram código HTML a partir de instruções em Java, sendo ideais para criação da lógica de negócios. *JavaServer Pages* (JSP), em contrapartida, são arquivos de texto que contém HTML, mas que também processa comandos em Java, sendo mais útil para geração da interface visual.

C) Qual a diferença entre um redirecionamento simples e o uso do método *forward* a partir do *RequestDispatcher*? Para que servem parâmetros e atributos nos objetos *HttpRequest*?

O redirecionamento simples envia para o cliente uma resposta dizendo para fazer requisição de outra URL, enquanto, através do método *forward*, a solicitação passa por algum outro processo no lado do servidor. Parâmetros e atributos nos objetos *HttpRequest* são usados para transferir dados entre partes do servidor: usualmente, parâmetros são usados para enviar dados de entrada, como formulários ou solicitações de URL, enquanto atributos são usados para armazenar dados que podem ser usados em diferentes partes da aplicação.

Análise e Conclusão – Prática 3

A) Como o framework Bootstrap é utilizado?

O Bootstrap é um *framework* para *design* de interfaces visuais que oferece um grande conjunto de ferramentas pré-definidas, mas muito personalizáveis, que facilitam a construção de uma página *web*. Isso é feito através da adição de palavras-chave no atributo *Class* de cada elemento. Com isso, os elementos HTML ganham estilização rápida, sem a necessidade de escrever código em CSS. Idealmente, o Bootstrap deve ser usado juntamente com a estilização por CSS, visto que as palavras-chaves usadas pelo framework ainda são classes e podem ser usadas para controle fino da personalização no CSS também.

B) Por que o Bootstrap garante a independência estrutural do HTML?

Pois a inserção do Bootstrap no código HTML acontece como etapa posterior à construção do esqueleto HTML. As alterações visuais feitas pelo framework são construídas sobre este esqueleto, de maneira não destrutiva e que não exige que o HTML seja construído em função da estilização.

C) Qual a relação entre o Bootstrap e a responsividade da página?

O Bootstrap é amplamente utilizado para desenvolver interfaces visuais de todos os tipos e formatos, portanto, é essencial que o *framework* tenha a responsividade para se adaptar a todos esses diferentes cenários. Alguns exemplos do forte relacionamento do framework com responsividade são a ferramenta *grid* flexível, a alteração de visualização dependendo do formato/resolução de tela, etc.

Anexo – Códigos do Projeto

Arquivo – Produto.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
 this template
 */

package cadastroee.model;

import jakarta.persistence.Basic;
import jakarta.persistence.CascadeType;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.NamedQueries;
import jakarta.persistence.NamedQuery;
import jakarta.persistence.OneToOne;
import jakarta.persistence.Table;
import java.io.Serializable;
import java.util.Collection;
import javax.validation.constraints.NotNull;
import javax.validation.constraints.Size;
import jakarta.xml.bind.annotation.XmlRootElement;
import jakarta.xml.bind.annotation.XmlTransient;
```

```

/**
 *
 * @author fel-f
 */

@Entity
@Table(name = "Produto")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Produto.findAll", query = "SELECT p FROM Produto p"),
    @NamedQuery(name = "Produto.findByIdProduto", query = "SELECT p FROM Produto p WHERE p.idProduto = :idProduto"),
    @NamedQuery(name = "Produto.findByName", query = "SELECT p FROM Produto p WHERE p.nome = :nome"),
    @NamedQuery(name = "Produto.findByQuantidade", query = "SELECT p FROM Produto p WHERE p.quantidade = :quantidade"),
    @NamedQuery(name = "Produto.findByPrecoVenda", query = "SELECT p FROM Produto p WHERE p.precoVenda = :precoVenda"))
public class Produto implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @NotNull
    @Column(name = "idProduto")
    private Integer idProduto;

    @Basic(optional = false)
    @NotNull

```

```

@Size(min = 1, max = 255)
@Column(name = "Nome")
private String nome;

@Basic(optional = false)
@NotNull
@Column(name = "Quantidade")
private int quantidade;

// @Max(value=?) @Min(value=?)//if you know range of your decimal fields
consider using these annotations to enforce field validation

@Basic(optional = false)
@NotNull
@Column(name = "PrecoVenda")
private Float precoVenda;

@OneToMany(cascade = CascadeType.ALL, mappedBy = "fkProdutoid")
private Collection<Movimento> movimentoCollection;

public Produto() {
}

public Produto(Integer idProduto) {
    this.idProduto = idProduto;
}

public Produto(Integer idProduto, String nome, int quantidade, Float
precoVenda) {
    this.idProduto = idProduto;
    this.nome = nome;
    this.quantidade = quantidade;
    this.precoVenda = precoVenda;
}

```

```
public Integer getIdProduto() {  
    return idProduto;  
}  
  
public void setIdProduto(Integer idProduto) {  
    this.idProduto = idProduto;  
}  
  
public String getNome() {  
    return nome;  
}  
  
public void setNome(String nome) {  
    this.nome = nome;  
}  
  
public int getQuantidade() {  
    return quantidade;  
}  
  
public void setQuantidade(int quantidade) {  
    this.quantidade = quantidade;  
}  
  
public Float getPrecoVenda() {  
    return precoVenda;  
}
```

```

public void setPrecoVenda(Float precoVenda) {
    this.precoVenda = precoVenda;
}

@XmlTransient
public Collection<Movimento> getMovimentoCollection() {
    return movimentoCollection;
}

public void setMovimentoCollection(Collection<Movimento>
movimentoCollection) {
    this.movimentoCollection = movimentoCollection;
}

@Override
public int hashCode() {
    int hash = 0;
    hash += (idProduto != null ? idProduto.hashCode() : 0);
    return hash;
}

@Override
public boolean equals(Object object) {
    // TODO: Warning - this method won't work in the case the id fields are not
set
    if (!(object instanceof Produto)) {
        return false;
    }
    Produto other = (Produto) object;

```



```

        if ((this.idProduto == null && other.idProduto != null) || (this.idProduto !=
null && !this.idProduto.equals(other.idProduto))) {

            return false;

        }

        return true;

    }

    @Override
    public String toString() {

        return "cadastroee.model.Produto[ idProduto=" + idProduto + " ]";

    }

}

```

Arquivo – ServletProduto.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to
edit this template
 */

package cadastroee.servlets;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.List;
import java.text.DecimalFormat;
import jakarta.ejb.EJB;
import jakarta.servlet.ServletException;

```

```

import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import cadastroee.model.Produto;
import cadastroee.controller.ProdutoFacadeLocal;

/**
 *
 * @author fel-f
 */

@WebServlet(name = "ServletProduto", urlPatterns = {"/ServletProduto"})
public class ServletProduto extends HttpServlet {

    /**
     * Processes requests for both HTTP GET and
     * POST
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */

    @EJB
    private ProdutoFacadeLocal facade;

```

@Override

```
protected void doGet(HttpServletRequest request, HttpServletResponse  
response)
```

```
    throws ServletException, IOException {  
        response.setContentType("text/html;charset=UTF-8");  
        List<Produto> listaDeProdutos = facade.findAll();  
        String htmlResponse = buildHtmlResponse(listaDeProdutos);  
        try (PrintWriter out = response.getWriter()) {  
            out.println(htmlResponse);  
        }  
    }  
}
```

```
private String buildHtmlResponse(List<Produto> produtos) {  
    StringBuilder htmlBuilder = new StringBuilder();  
    DecimalFormat df = new DecimalFormat("#,##0.00");  
    htmlBuilder.append("<!DOCTYPE html>")  
        .append("<html>")  
        .append("<head>")  
        .append("<title>Lista de Produtos</title>")  
        .append("</head>")  
        .append("<body>")  
        .append("<h1>Lista de Produtos</h1>")  
        .append("<table >")  
        .append("<tr><th>Nome do Produto</th><th>Preço</th></tr>");  
    for (Produto produto : produtos) {  
        htmlBuilder.append("<tr><td>")  
            .append(produto.getNome())  
            .append("</td><td>")  
            .append(df.format(produto.getPrecoVenda()))
```

```

        .append("</td></tr>");
    }
    htmlBuilder.append("</table>")
        .append("</body>")
        .append("</html>");
    return htmlBuilder.toString();
}

@Override
public String getServletInfo() {
    return "Servlet para listagem de produtos";
}
}

```

Arquivo – ServletProdutoFC.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to
 edit this template
 */

package cadastroee.servlets;

import cadastroee.model.Produto;
import cadastroee.controller.ProdutoFacadeLocal;
import java.io.IOException;
import java.util.List;
import jakarta.ejb.EJB;
import jakarta.servlet.RequestDispatcher;

```

```

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

/**
 *
 * @author fel-f
 */

@WebServlet(name= "ServletProdutoFC", urlPatterns= {"/ServletProdutoFC"})
public class ServletProdutoFC extends HttpServlet {

    /**
     * Processes requests for both HTTP GET and
     * POST
     * methods.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */

    @EJB
    private ProdutoFacadeLocal facade;

    @Override

```

```

protected void doGet(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {

    String acao = request.getParameter("acao");
    String destino = "";

    switch (acao) {
        case "formIncluir" -> destino = "ProdutoDados.jsp";

        case "excluir" -> {
            int idDel = Integer.parseInt(request.getParameter("id"));
            facade.remove(facade.find(idDel));
            List<Produto> delProdutos = facade.findAll();
            request.setAttribute("produtos", delProdutos);
            destino = "ProdutoLista.jsp";
        }

        case "formAlterar" -> {
            int id = Integer.parseInt(request.getParameter("id"));
            Produto produto = facade.find(id);
            request.setAttribute("produto", produto);
            destino = "ProdutoDados.jsp";
        }

        default -> {
            List<Produto> produtos = facade.findAll();
            request.setAttribute("produtos", produtos);
            destino = "ProdutoLista.jsp";
        }
    }
}

```

```

    }
}

RequestDispatcher dispatcher = request.getRequestDispatcher(destino);
dispatcher.forward(request, response);

}

@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response)
    throws ServletException, IOException {

    String acao = request.getParameter("acao");
    acao = acao == null || acao.isEmpty() ? " " : acao;

    String destino = "ProdutoLista.jsp";

    switch (acao) {

        case "incluir" -> {
            String nome = request.getParameter("nome");
            int quantidade =
Integer.parseInt(request.getParameter("quantidade"));
            Float precoVenda =
Float.valueOf(request.getParameter("precoVenda"));

            Produto newProduto = new Produto();
            newProduto.setNome(nome);
            newProduto.setQuantidade(quantidade);

```

```

        newProduto.setPrecoVenda(precoVenda);

        facade.create(newProduto);

        List<Produto> newProdutos = facade.findAll();
        request.setAttribute("produtos", newProdutos);
    }

    case "alterar" -> {
        Produto alterarProduto =
facade.find(Integer.valueOf(request.getParameter("id")));

        String alterarNome = request.getParameter("nome");
        int alterarQuantidade =
Integer.parseInt(request.getParameter("quantidade"));
        Float alterarPrecoVenda =
Float.valueOf(request.getParameter("precoVenda"));

        alterarProduto.setNome(alterarNome);
        alterarProduto.setQuantidade(alterarQuantidade);
        alterarProduto.setPrecoVenda(alterarPrecoVenda);

        facade.edit(alterarProduto);
        List<Produto> alterarProdutos = facade.findAll();
        request.setAttribute("produtos", alterarProdutos);
    }

    default -> {
        List<Produto> produtos = facade.findAll();
        request.setAttribute("produtos", produtos);
    }

```



```

    }

}

    RequestDispatcher dispatcher = request.getRequestDispatcher(destino);
    dispatcher.forward(request, response);
}
}

```

Arquivo – ProdutoLista.jsp

```

<%--
    Document   : ProdutoLista
    Created on  : 2 de nov. de 2024, 18:14:01
    Author      : fel-f
--%>

<%@ page import="java.text.DecimalFormat" %>
<%@ page import="cadastroee.model.Produto" %>
<%@ page import="java.util.List" %>
<%@ page contentType="text/html" pageEncoding="UTF-8" %>

<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Lista de Produtos</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
4bw+/aepP/YC94hEpVNVgiZdgIC5+VKNBQNGCHeKRQN+PtmoHDEXuppvnD
JzQlu9" crossorigin="anonymous">

```

```

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js
" integrity="sha384-
HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrk
m" crossorigin="anonymous"></script>

<style>
  body {
    background-color:rgb(236, 236, 236)
  }
  h1 {
    margin: 20px
  }
  .card {
    padding: 20px;
  }
</style>
</head>

<body>
  <div class="container">
    <div class="header-section text-center">
      <h1>Listagem de Produtos</h1>
    </div>
    <div class="card">
      <div class="card-body">
        <table class="table table-bordered table-responsive">
          <tr class="table-dark text-center">
            <th>ID</th>
            <th>Nome do Produto</th>

```

```

        <th>Quantidade</th>

        <th>Preço</th>

        <th>Ações</th>
    </tr>

    <%
        DecimalFormat df = new DecimalFormat("#,##0.00");

        List<Produto> produtos = (List<Produto>)
request.getAttribute("produtos");

        if (produtos != null && !produtos.isEmpty()) {
            for (Produto produto : produtos) {
    %>
    <tr>

        <td class="text-center"><%=produto.getIdProduto()%></td>

        <td class="text-center"><%=produto.getNome()%></td>

        <td class="text-center"><%=produto.getQuantidade()%></td>

        <td class="text-center">R$
<%=df.format(produto.getPrecoVenda())%></td>

        <td class="text-end">

            <a class="btn btn-primary btn-sm"
href="ServletProdutoFC?acao=formAlterar&id=<%=produto.getIdProduto()%>">
Alterar</a>

            <a class="btn btn-danger btn-sm"
href="ServletProdutoFC?acao=excluir&id=<%=produto.getIdProduto()%>">Excl
uir</a>

        </td>

    </tr>

    <%
        }
    } else {
    %>

    <tr>

```

```

        <td>Nenhum produto foi encontrado.</td>

    </tr>

    <%
        }
    %>

</table>

<div class="text-end mb-3">

    <a class="btn btn-primary"
href="ServletProdutoFC?acao=formIncluir">Cadastrar Produto</a>

</div>

</div>

</div>

</div>

</body>

</html>

```

Arquivo – ProdutoDados.jsp

```

<%--
    Document   : ProdutoDados
    Created on : 2 de nov. de 2024, 18:14:39
    Author    : fel-f
--%>

<%@page import="java.text.DecimalFormat"%>
<%@page import="cadastroee.model.Produto"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

```

```

<head>

  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

  <title>Cadastro de Produtos</title>

  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
4bw+/aepP/YC94hEpVNVgiZdgIC5+VKNBQNGCHeKRQN+PtmoHDEXuppvnD
JzQlu9" crossorigin="anonymous">

  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js
" integrity="sha384-
HwwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrk
m" crossorigin="anonymous"></script>

<style>

  body {

    background-color:rgb(236, 236, 236)

  }

  h1 {

    margin: 20px

  }

  .card {

    padding: 20px;

  }

</style>

</head>

<body>

  <%

    DecimalFormat df = new DecimalFormat("#,##0.00");

    Produto produto = (Produto) request.getAttribute("produto");

    String acao = produto != null ? "alterar" : "incluir";

```

```

%>

<div>

    <div>

        <h1 class="header-section text-center"><%=acao == "alterar" ?
"Alteração" : "Cadastro"%> de Produtos</h1>

    </div>

    <div class="row">

        <div class="col-md-6 offset-md-3">

            <div>

                <a class="btn btn-secondary mb-3"
href="ServletProdutoFC?acao=listar">Voltar</a>

            </div>

            <div class="card">

                <div class="card-body">

                    <form class="form-container" action="ServletProdutoFC"
method="post">

                        <input type="hidden" name="acao" value="<%=acao%>">

                        <% if (produto != null) { %>

                            <input type="hidden" name="id"
value="<%=produto.getIdProduto()%>">

                            <% } %>

                            <div class="mb-3">

                                <label class="form-label" for="nome">Nome</label>

                                <input class="form-control" type="text" name="nome"
value="<%=produto != null ? produto.getNome() : ""%>" required>

                            </div>

                            <div class="mb-3">

                                <label class="form-label"
for="quantidade">Quantidade</label>

                                <input class="form-control" type="text" name="quantidade"
value="<%=produto != null ? produto.getQuantidade() : ""%>" required>

```

```

        </div>
        <div class="mb-3">
            <label class="form-label" for="precoVenda">Preço de
Venda</label>
            <input class="form-control" type="text" name="precoVenda"
value="<%=produto != null ? produto.getPrecoVenda() : ""%>" required>
        </div>
        <div class="mb-3">
            <input class="btn btn-primary" type="submit"
value="<%=acao == "incluir" ? "Cadastrar" : "Alterar"%> Produto">
        </div>
    </form>
</div>
</div>
</div>
</div>
</div>
</body>
</html>

```