UNIVERSIDADE ESTÁCIO DE SÁ

Campus Estácio - Castelo - Belo Horizonte – MG

Curso: Desenvolvimento Full Stack

Disciplina: Nível 4: Vamos Integrar Sistemas (RPG0017)

Turma: 9002

Semestre Letivo: 2025.1

Aluno: Ivan de Ávila Carvalho Fleury Mortimer

Repositório do Projeto: https://github.com/ivanmortimer/CadastroEE

Título da Prática: Missão Prática | Nível 4 | Mundo 3

Missão Prática | Nível 4 | Mundo 3

1. Objetivos da Prática

1. Implementar persistência com base em JPA.
2. Implementar regras de negócio na plataforma JEE, através de EJBs.
3. Implementar sistema cadastral Web com base em Servlets e JSPs.
4. Utilizar a biblioteca Bootstrap para melhoria do design. (A ser implementado no 3º procedimento.)
5. Tornar-se capacitado para lidar com contextos reais de aplicação Web com Java EE.

2. Códigos Relevantes e Estrutura da Aplicação

A seguir, apresentam-se trechos de código que representam partes essenciais da prática.

2.1 Servlet: ServletProdutoFC.java

Trecho representativo do método processRequest do Servlet:

```
request.setCharacterEncoding("UTF-8");
String acao = request.getParameter("acao");
```

2.2 Página de Listagem: ProdutoLista.jsp

Trecho responsável por exibir os dados do produto e links dinâmicos:

```
<td>${produto.nome}</td>
<td>${produto.quantidade}</td>
<td>${produto.precoVenda}</td>
```

2.3 Página de Cadastro: ProdutoDados.jsp

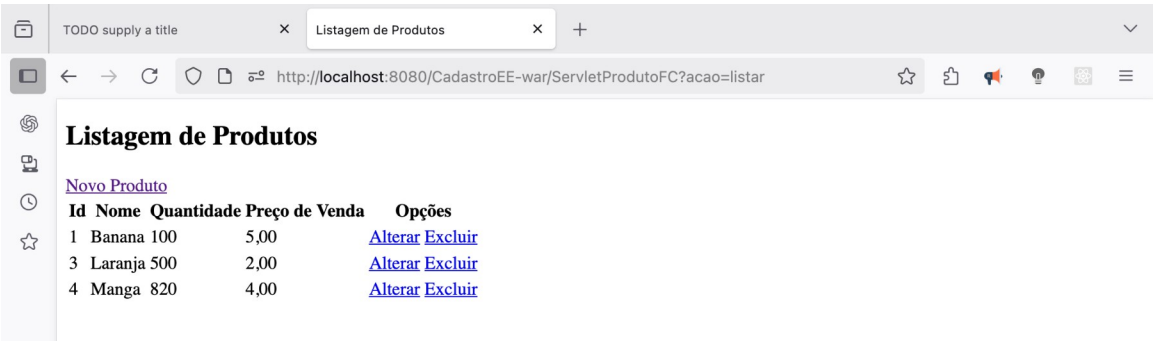
Trecho responsável por tratar a entrada de dados e enviar ao Servlet:

```
<form action="ServletProdutoFC" method="post">
  <input type="hidden" name="acao" value="${acao}"/>
```

3. Resultados da Execução

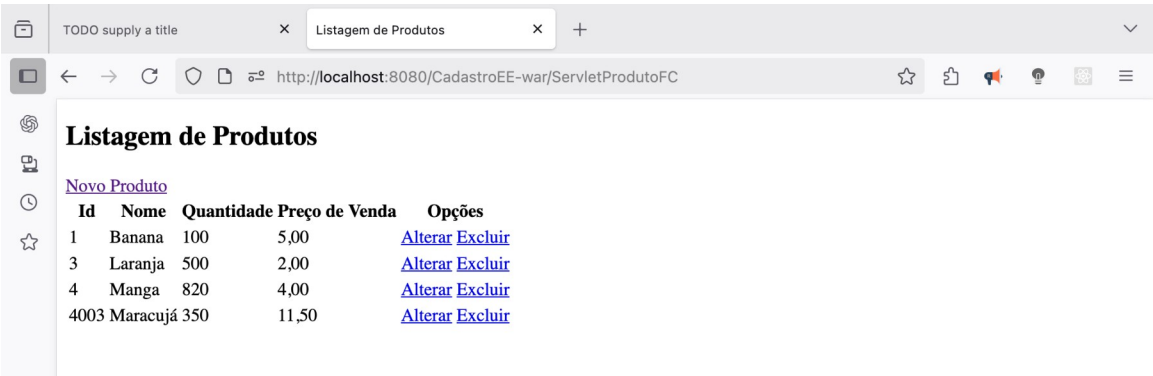
As imagens a seguir ilustram a execução da aplicação com as funcionalidades de listagem, inclusão, alteração e exclusão de produtos, conforme os testes realizados:

a) Listagem de Produtos



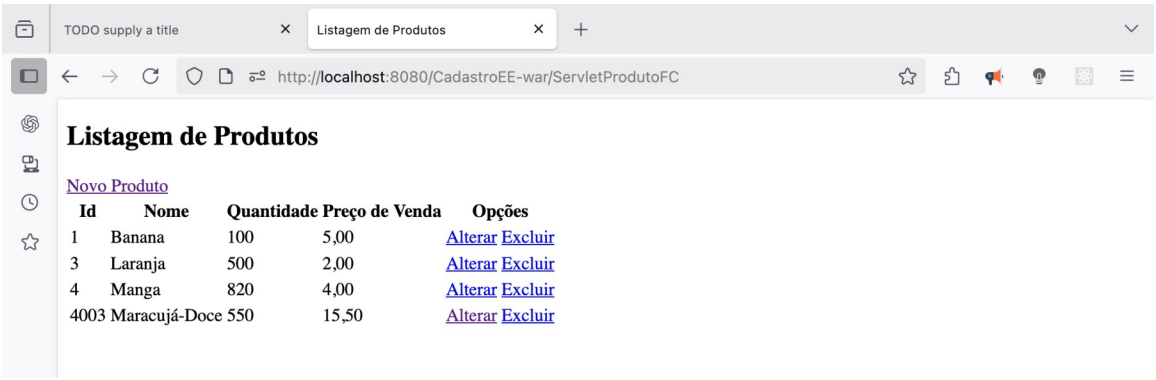
b) Inclusão de Produto

c) Listagem de Produtos com o Novo Produto Incluído

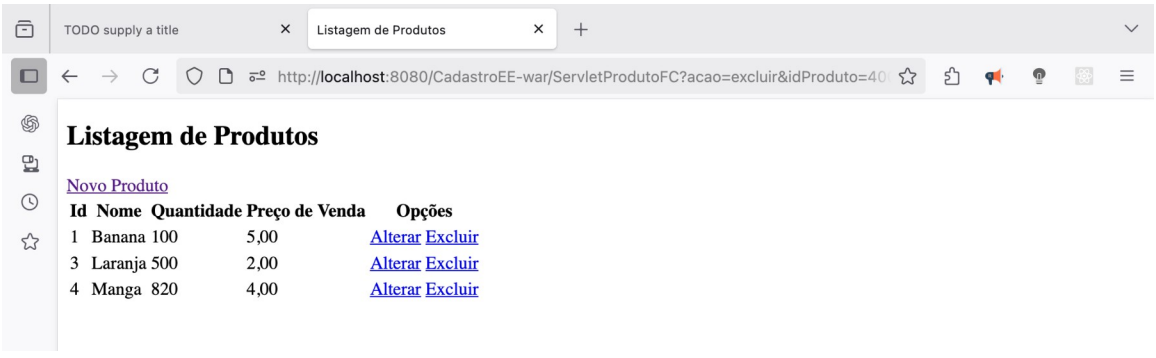


d) Alteração de Produto

e) Listagem de Produtos com o Produto Alterado



f) Exclusão de Produto e Consequente Listagem com Produto Excluído



4. Análise e Conclusão

a) O padrão Front Controller funciona como um ponto único de entrada para as requisições em uma aplicação. No contexto desta prática, ele foi implementado no ServletProdutoFC, que centraliza a lógica de controle das ações do sistema, como listar, incluir, alterar e excluir produtos.

b) Servlets são componentes Java responsáveis pelo processamento das requisições HTTP. Já JSPs são páginas com HTML que permitem a inserção de código Java. Ambos trabalham juntos para separar a lógica de negócio da apresentação.

c) O método forward do RequestDispatcher redireciona a requisição no servidor, mantendo os dados da requisição intactos. Diferentemente do sendRedirect, o forward é transparente ao cliente. Os parâmetros e atributos no objeto HttpServletRequest permitem passar dados entre componentes.

5. Códigos Completos

a) ProdutoDados.jsp:

```
<%--

    Document   : ProdutoDados

    Created on : 2 de jun. de 2025, 23:44:00

    Author     : Ivan

--%>


<%@page import="cadastroee.model.Produto"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<%

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

    String acao = (produto == null) ? "incluir" : "alterar";

%>


<!DOCTYPE html>
```

```
<html>

<head>

    <meta charset="UTF-8">

    <title><%= (produto == null) ? "Incluir Produto" : "Alterar Produto"
%></title>

    <style>

        label {

            display: inline-block;

            width: 120px;

        }

    </style>

</head>

<body>

    <h2><%= (produto == null) ? "Dados do Novo Produto" : "Dados do Produto"
%></h2>



    <% if (request.getAttribute("erro") != null) { %>

        <p style="color:red;"><%= request.getAttribute("erro") %></p>

    <% } %>



    <form action="ServletProdutoFC" method="post">

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

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

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

        <% } %>

        <div>

            <label for="nome">Nome:</label>

            <input type="text" id="nome" name="nome" required value="<%=
(produto != null) ? produto.getNome() : "" %>" />
```

```
        </div>

        <div>

            <label for="quantidade">Quantidade:</label>

            <input type="number" id="quantidade" name="quantidade" required
min="0" step="1" value="<%= (produto != null) ? produto.getQuantidade() : ""
%>" />

        </div>

        <div>

            <label for="precoVenda">Preço de Venda:</label>

            <input type="text" id="precoVenda" name="precoVenda" required
min="0" step="0.01" value="<%= (produto != null) ?
String.format(java.util.Locale.forLanguageTag("pt-BR"), "%.2f",
produto.getPrecoVenda()) : "" %>" pattern="\d+(\.|,)?\d*" title="Digite um
valor decimal não negativo com ponto ou vírgula" />

        </div>

        <input type="submit" value="<%= (produto == null) ? "Incluir Produto"
: "Alterar Produto" %>" />

    </form>

</body>

</html>
```

b) ProdutoLista.jsp:

```
<%--

    Document   : ProdutoDados

    Created on : 2 de jun. de 2025, 23:44:00

    Author     : Ivan

--%>



<%@page import="java.util.List"%>

<%@page import="cadastroee.model.Produto"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>
```

```
<!DOCTYPE html>

<html>

<head>

    <meta charset="UTF-8">

    <title>Listagem de Produtos</title>

</head>

<body>

    <h2>Listagem de Produtos</h2>


    <a href="ServletProdutoFC?acao=formIncluir">Novo Produto</a>


    <table>

        <thead>

            <tr>

                <th>Id</th>

                <th>Nome</th>

                <th>Quantidade</th>

                <th>Preço de Venda</th>

                <th>Opções</th>

            </tr>

        </thead>

        <tbody>

            <%

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

                if (listaProduto != null) {

                    for (Produto p : listaProduto) {

            %>
```

```
            <tr>
                <td><%= p.getIdProduto() %></td>
                <td><%= p.getNome() %></td>
                <td><%= p.getQuantidade() %></td>
                <td><%= String.format("%.2f", p.getPrecoVenda()) %></td>
                <td>
                    <a href="ServletProdutoFC?acao=formAlterar&idProduto=<%=
p.getIdProduto() %>">Alterar</a>
                    <a href="ServletProdutoFC?acao=excluir&idProduto=<%=
p.getIdProduto() %>">Excluir</a>
                </td>
            </tr>
            <%      }
                }
            %>
        </tbody>
    </table>
</body>
</html>
```

c) 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 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;

import java.io.IOException;


/**

 *

 * @author Ivan

 */

@WebServlet(name = "ServletProdutoFC", urlPatterns = {"/ServletProdutoFC"})

public class ServletProdutoFC extends HttpServlet {


    @EJB

    ProdutoFacadeLocal facade;


    /**

     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code>

     * 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
```

```
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
            throws ServletException, IOException {
        // Corrige problema de acentuação
        request.setCharacterEncoding("UTF-8");
        response.setContentType("text/html;charset=UTF-8");

        String acao = request.getParameter("acao");
        if (acao == null) {
            acao = "listar";
        }

        String destino;
        if (acao.equals("formIncluir") || acao.equals("formAlterar")) {
            destino = "ProdutoDados.jsp";
        } else {
            destino = "ProdutoLista.jsp";
        }

        try {
            switch (acao) {
                case "listar" -> {
                    request.setAttribute("listaProduto", facade.findAll());
                }

                case "formAlterar" -> {
                    Integer id =
Integer.valueOf(request.getParameter("idProduto"));
```

```
                    Produto produto = facade.find(id);

                    request.setAttribute("produto", produto);

                }



                case "excluir" -> {

                    Integer id =
Integer.valueOf(request.getParameter("idProduto"));

                    Produto produto = facade.find(id);

                    if (produto != null) {

                        facade.remove(produto);

                    }

                    request.setAttribute("listaProduto", facade.findAll());

                }



                case "alterar" -> {

                    String idStr = request.getParameter("idProduto");

                    String nome = request.getParameter("nome");

                    String quantidadeStr =
request.getParameter("quantidade");

                    String precoVendaStr =
request.getParameter("precoVenda");



                    if (nome == null || nome.isBlank() ||

                        quantidadeStr == null || quantidadeStr.isBlank() ||

                        precoVendaStr == null || precoVendaStr.isBlank()) {

                        request.setAttribute("erro", "Todos os campos devem
ser preenchidos.");

                        request.setAttribute("produto",
facade.find(Integer.valueOf(idStr)));

                        destino = "ProdutoDados.jsp";
```

```
                        break;

                    }


                    try {

                        Integer id = Integer.valueOf(idStr);

                        int quantidade = Integer.parseInt(quantidadeStr);

                        Float precoVenda =
Float.valueOf(precoVendaStr.replace(',', '.'));



                        Produto produto = facade.find(id);

                        if (produto != null) {

                            produto.setNome(nome);

                            produto.setQuantidade(quantidade);

                            produto.setPrecoVenda(precoVenda);

                            facade.edit(produto);

                        }



                        request.setAttribute("listaProduto",
facade.findAll());

                    } catch (NumberFormatException e) {

                        request.setAttribute("erro", "Quantidade e Preço
devem ser valores numéricos.");

                        request.setAttribute("produto",
facade.find(Integer.valueOf(idStr)));

                        destino = "ProdutoDados.jsp";

                    }

                }



                case "incluir" -> {

                    String nome = request.getParameter("nome");
```

```
                    String quantidadeStr =
request.getParameter("quantidade");

                    String precoVendaStr =
request.getParameter("precoVenda");



                    if (nome == null || nome.isBlank() ||

                        quantidadeStr == null || quantidadeStr.isBlank() ||

                        precoVendaStr == null || precoVendaStr.isBlank()) {

                        request.setAttribute("erro", "Todos os campos devem
ser preenchidos.");

                        destino = "ProdutoDados.jsp";

                        break;

                    }



                    try {

                        int quantidade = Integer.parseInt(quantidadeStr);

                        Float precoVenda =
Float.valueOf(precoVendaStr.replace(',', '.'));



                        Produto novoProduto = new Produto();

                        novoProduto.setNome(nome);

                        novoProduto.setQuantidade(quantidade);

                        novoProduto.setPrecoVenda(precoVenda);



                        facade.create(novoProduto);

                        request.setAttribute("listaProduto",
facade.findAll());

                    } catch (NumberFormatException e) {

                        request.setAttribute("erro", "Quantidade e Preço
devem ser valores numéricos.");

                        destino = "ProdutoDados.jsp";
```

```
                    }
                }
            }
        } catch (NumberFormatException ex) {
            // Logar o erro e encaminhar para página de erro (opcional)
            request.setAttribute("erro", "Erro interno: " + ex.getMessage());
            destino = "ProdutoDados.jsp";
        }

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

    // <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click
on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        processRequest(request, response);
    }
```

```
    /**
     * Handles the HTTP <code>POST</code> method.
     *
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     *
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "ServletProdutoFC - Front Controller para gerenciamento da
tabela 'Produto' do esquema 'dbo' do banco de dados 'loja' (SGBD SQL
Server).";
    }// </editor-fold>
}
```

d) 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.OneToMany;

import jakarta.persistence.Table;

import jakarta.validation.constraints.Min;

import jakarta.validation.constraints.NotNull;

import jakarta.validation.constraints.Size;

import jakarta.xml.bind.annotation.XmlRootElement;

import jakarta.xml.bind.annotation.XmlTransient;

import java.io.Serializable;

import java.util.Collection;



/**

 *
```

```
 * @author Ivan

 */

@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.findByNome", 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)

    @Column(name = "idProduto")

    private Integer idProduto;

    @Basic(optional = false)

    @NotNull

    @Size(min = 1, max = 255)

    @Column(name = "nome")

    private String nome;

    @Min(value=0)

    @Basic(optional = false)
```

```
    @NotNull
    @Column(name = "quantidade")
    private int quantidade;
    @Min(value=0)//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 = "idProduto")
    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 + " ]";
    }

}
```

e) AbstractFacade.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.controller;

import jakarta.persistence.EntityManager;
import java.util.List;

/**
 *
 * @author Ivan
 * @param <T>
 */
```

```
public abstract class AbstractFacade<T> {


    private Class<T> entityClass;


    public AbstractFacade(Class<T> entityClass) {

        this.entityClass = entityClass;

    }


    protected abstract EntityManager getEntityManager();


    public void create(T entity) {

        getEntityManager().persist(entity);

    }


    public void edit(T entity) {

        getEntityManager().merge(entity);

    }


    public void remove(T entity) {

        getEntityManager().remove(getEntityManager().merge(entity));

    }


    public T find(Object id) {

        return getEntityManager().find(entityClass, id);

    }


    public List<T> findAll() {

        jakarta.persistence.criteria.CriteriaQuery cq =
getEntityManager().getCriteriaBuilder().createQuery();
```

```
        cq.select(cq.from(entityClass));

        return getEntityManager().createQuery(cq).getResultList();

    }


    public List<T> findRange(int[] range) {

        jakarta.persistence.criteria.CriteriaQuery cq =
getEntityManager().getCriteriaBuilder().createQuery();

        cq.select(cq.from(entityClass));

        jakarta.persistence.Query q = getEntityManager().createQuery(cq);

        q.setMaxResults(range[1] - range[0] + 1);

        q.setFirstResult(range[0]);

        return q.getResultList();

    }


    public int count() {

        jakarta.persistence.criteria.CriteriaQuery cq =
getEntityManager().getCriteriaBuilder().createQuery();

        jakarta.persistence.criteria.Root<T> rt = cq.from(entityClass);

        cq.select(getEntityManager().getCriteriaBuilder().count(rt));

        jakarta.persistence.Query q = getEntityManager().createQuery(cq);

        return ((Long) q.getSingleResult()).intValue();

    }


}
```

f) ProdutoFacadeLocal.java:

```
/*

 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license

 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Interface.java to
```

```
edit this template
 */
package cadastroee.controller;


import cadastroee.model.Produto;
import jakarta.ejb.Local;
import java.util.List;


/**
 *
 * @author Ivan
 */
@Local
public interface ProdutoFacadeLocal {


    void create(Produto produto);


    void edit(Produto produto);


    void remove(Produto produto);


    Produto find(Object id);


    List<Produto> findAll();


    List<Produto> findRange(int[] range);


    int count();
```

```
}
```

g) ProdutoFacade.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.controller;



import cadastroee.model.Produto;

import jakarta.ejb.Stateless;

import jakarta.persistence.EntityManager;

import jakarta.persistence.PersistenceContext;



/**

 *

 * @author Ivan

 */

@Stateless

public class ProdutoFacade extends AbstractFacade<Produto> implements
ProdutoFacadeLocal {



    @PersistenceContext(unitName = "CadastroEE-ejbPU")

    private EntityManager em;



    @Override
```

```
    protected EntityManager getEntityManager() {
        return em;
    }

    public ProdutoFacade() {
        super(Produto.class);
    }

}
```