TRABALHO PRÁTICO | DGT2811
DESENVOLVIMENTO BACK-END CORPORATIVO COM JAVA E CLOUD

Felipe Eduardo Oliveira de Melo - 202411164421

Polo Votuporanga

Desenvolvimento Full Stack – 9001 3AA – 2025.4

Objetivo da Prática

Implementação de sistema cadastral com interface Web, baseado nas tecnologias de Servlets, JPA e JEE.

1º Procedimento | Camadas de Persistência e Controle

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.NotNull;
import jakarta.validation.constraints.Size;
import jakarta.xml.bind.annotation.XmlRootElement;
import jakarta.xml.bind.annotation.XmlTransient;
import java.io.Serializable;
```

```
import java.math.BigDecimal;
import java.util.Collection;

/**
 *
 * @author Felipe
 */
@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;
    @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 int 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, int precoVenda) {
        this.idProduto = idProduto;
        this.nome = nome;
        this.quantidade = quantidade;
        this.precoVenda = precoVenda;
    }
/… getter e setters
```

Persistence.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="1.0"
xmlns="http://java.sun.com/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/
xml/ns/persistence/persistence_1_0.xsd">
  <persistence-unit name="CadastroEE-ejbPU" transaction-type="JTA">
    <jta-data-source>jdbc/loja</jta-data-source>
    <exclude-unlisted-classes>false</exclude-unlisted-classes>
```

```
    <properties/>
  </persistence-unit>
</persistence>
```

```
package cadastroee.servlets;

import cadastroee.controller.ProdutoFacadeLocal;
import cadastroee.model.Produto;
import jakarta.ejb.EJB;
import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.util.List;

/**
 *
 * @author Felipe
 */
public class ServletProduto 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 {
      response.setContentType("text/html;charset=UTF-8");
      try (PrintWriter out = response.getWriter()) {
         /* TODO output your page here. You may use following sample code. */
         out.println("<!DOCTYPE html>");
         out.println("<html>");
         out.println("<head>");
         out.println("<title>Lista de Produtos</title>");
         out.println("</head>");
         out.println("<body>");
         out.println("<h1>Produtos cadastrados</h1>");
         out.println("<ul>");
         List<Produto> produtos = facade.findAll();
         for (Produto produto : produtos) {
            out.println("<li>" + produto.getNome() + " - R$ " + produto.getPrecoVenda() +
"</li>");
         }
         out.println("</ul>");
         out.println("</body>");
         out.println("</html>");
      }
   }
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="4.0" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/
javaee/web-app_4_0.xsd">
 <servlet>
  <servlet-name>ServletProduto</servlet-name>
  <servlet-class>cadastroee.servlets.ServletProduto</servlet-class>
 </servlet>
```

```
 <servlet>
  <servlet-name>ServletProdutoFC</servlet-name>
  <servlet-class>cadastroee.servlets.ServletProdutoFC
  </servlet-class>
 </servlet>
 <session-config>
  <session-timeout>30</session-timeout>
 </session-config>
</web-app>
```

a. Como é organizado um projeto corporativo no NetBeans?
   É organizado em módulos, Projeto EAR, EJB e WAR.

b. Qual o papel das tecnologias JPA e EJB na construção de um aplicativo para a plataforma Web no ambiente Java?
   Atuam na camada de persistencia e regras de negócio

c. Como o NetBeans viabiliza a melhoria de produtividade ao lidar com as tecnologias JPA e EJB?
   Atraves da geração automática de código.

d. O que são Servlets, e como o NetBeans oferece suporte à construção desse tipo de componentes em um projeto Web?
   São classes que ficam dentro de um container web, o NetBeans gera o Código automático como dito a cima.

e. Como é feita a comunicação entre os Serlvets e os Session Beans do pool de EJBs?
   Atraves de injeção de dependência.

2º Procedimento | Interface Cadastral com Servlet e JSPs

```
package cadastroee.servlets;

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

/**
 *
 * @author Felipe
 */
@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 {
       String acao = request.getParameter("acao");
       String destino = "ProdutoLista.jsp";

       if (acao == null) {
          acao = "listar";
       }

       switch (acao) {
          case "listar":
             List<Produto> produtos = facade.findAll();
             request.setAttribute("produtos", produtos);
             destino = "ProdutoLista.jsp";
             break;
          case "formIncluir":
             destino = "ProdutoDados.jsp";
             break;
```

```
case "formAlterar":
    String idStr = request.getParameter("id");
    int id = Integer.parseInt(idStr);
    Produto produto = facade.find(id);
    request.setAttribute("produto", produto);
    destino = "ProdutoDados.jsp";
    break;
case "excluir":
    int idExcluir = Integer.parseInt(request.getParameter("id"));
    Produto produtoExcluir = facade.find(idExcluir);
    if (produtoExcluir != null) {
        facade.remove(produtoExcluir);
    }
    List<Produto> listaAtualizada = facade.findAll();
    request.setAttribute("produtos", listaAtualizada);
    destino = "ProdutoLista.jsp";
    break;
case "alterar":
    int idAlterar = Integer.parseInt(request.getParameter("id"));
    Produto produtoAlterar = facade.find(idAlterar);

    String nome = request.getParameter("nome");
    String quantidadeStr = request.getParameter("quantidade");
    String precoVendaStr = request.getParameter("precoVenda");
    produtoAlterar.setNome(nome);
    produtoAlterar.setQuantidade(Integer.parseInt(quantidadeStr));
```

```
            produtoAlterar.setPrecoVenda(Integer.parseInt(precoVendaStr));
            facade.edit(produtoAlterar);
            List<Produto> listaAlterar = facade.findAll();
            request.setAttribute("produtos", listaAlterar);
            destino = "ProdutoLista.jsp";
            break;
        case "incluir":
            Produto produtoIncluir = new Produto();
            String nomeIncluir = request.getParameter("nome");
            int quantidade = Integer.parseInt(request.getParameter("quantidade"));
            int precoVenda = Integer.parseInt(request.getParameter("precoVenda"));
            produtoIncluir.setNome(nomeIncluir);
            produtoIncluir.setQuantidade(quantidade);
            produtoIncluir.setPrecoVenda(precoVenda);
            facade.create(produtoIncluir);
            List<Produto> listaIncluir = facade.findAll();
            request.setAttribute("produtos", listaIncluir);
            destino = "ProdutoLista.jsp";
            break;
        default:
            List<Produto> padrao = facade.findAll();
            request.setAttribute("produtos", padrao);
            destino = "ProdutoLista.jsp";
            break;
    }
    RequestDispatcher rd = request.getRequestDispatcher(destino);
```

```
        rd.forward(request, response);
    }
```

a. Como funciona o padrão Front Controller, e como ele é implementado em um aplicativo Web Java, na arquitetura MVC?
   Centraliza o tratamento de requisições em um manipulador somente, todas as requisições são interceptadas por um Servlet.

b. Quais as diferenças e semelhanças entre Servlets e JSPs?
   Ambas são executadas no lado servirdor. Servlets são classes Java que geram HTML, e JSPs são documentos baseados em HTML que permitem código Java.

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?
   A diferença é onde o redirecionamento acontece e se os objetos da requisição serão mantidos.

3º Procedimento | Melhorando o Design da Interface

```
        <%@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>JSP Page</title>
        <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
sRIl4kxILFvY47J16cr9ZwB07vP4J8+LH7qKQnuqkuIAvNWLzeN8tE5YBujZqJLB"
crossorigin="anonymous">
    </head>
    <body class="container">
        <h1>Dados do Produto</h1>
        <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/js/bootstrap.bundle.min.js"
```

```
integrity="sha384-
FKyoEForCGlyvwx9Hj09JcYn3nv7wiPVlz7YYwJrWVcXK/BmnVDxM+D2scQbITxI"
crossorigin="anonymous"></script>
    <form action="ServletProdutoFC" method="post" class="form">
      <%
        Produto produto = (Produto) request.getAttribute("produto");
        String acao = "incluir";
        if (produto != null && produto.getIdProduto() != null) {
          acao = "alterar";
        }
      %>

      <input type="hidden" name="acao" value="<%= acao %>">
      <% if (acao.equals("alterar")) { %>
        <input type="hidden" name="id" value="<%= produto.getIdProduto() %>"/>
      <% } %>
      <div class="mb-3">
        <label class="form-label">Nome:</label>
        <input  type="text" name="nome" value="<%= produto != null ?
produto.getNome() : "" %>" class="form-control"/>
      </div>
      <div class="mb-3">
        <label class="form-label">Quantidade:</label>
        <input type="number" name="quantidade" value="<%= produto != null ?
produto.getQuantidade(): "" %>" class="form-control"/>
      </div>
      <div class="mb-3">
        <label class="form-label">Preço de Venda:</label>
        <input type="number" name="precoVenda" value="<%= produto != null ?
produto.getPrecoVenda() : "" %>" class="form-control"/>
      </div>
      <button type="submit" class="btn btn-primary"><%= produto != null ? "Alterar
Produto" : "Adicionar Produto" %></button>
    </form>
```

```
  </body>
</html>


<%@page import="java.util.List"%>
<%@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>JSP Page</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
sRIl4kxILFvY47J16cr9ZwB07vP4J8+LH7qKQnuqkuIAvNWLzeN8tE5YBujZqJLB"
crossorigin="anonymous">
  </head>
  <body class="container">
    <h1>Listagem de Produtos</h1>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.8/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
FKyoEForCGlyvwx9Hj09JcYn3nv7wiPVlz7YYwJrWVcXK/BmnVDxM+D2scQbITxI"
crossorigin="anonymous"></script>
    <a href="ServletProdutoFC?acao=formIncluir" class="btn btn-primary m-2">Novo
Produto</a>
    <table class="table table-striped">
      <thead class="table-dark">
        <tr>
          <th scope="col">Id</th>
          <th scope="col">Nome</th>
          <th scope="col">Quantidade</th>
          <th scope="col">Preco</th>
          <th scope="col">Opções</th>
        </tr>
      </thead>
      <tbody>
```

```
            <% List<Produto> produtos =
(List<Produto>)request.getAttribute("produtos"); %>
            <% for (Produto produto : produtos) {%>
            <tr>
              <th scope="row"><%= produto.getIdProduto() %></th>
              <td><%= produto.getNome() %></td>
              <td><%= produto.getQuantidade() %></td>
              <td><%= produto.getPrecoVenda() %></td>
              <td>
                <a href="ServletProdutoFC?acao=formAlterar&id=<%=
produto.getIdProduto() %>" class="btn btn-primary btn-sm">Alterar</a>
                <a href="ServletProdutoFC?acao=excluir&id=<%= produto.getIdProduto()
%>" class="btn btn-danger btn-sm">Excluir</a>
              </td>
            </tr>
            <% } %>
          </tbody>
      </table>
    </body>
</html>
```

a. Como o framework Bootstrap é utilizado?
   É utilizado atraves de CDN.

b. Por que o Bootstrap garante a independência estrutural do HTML?
   Através de classes pré-definidas.

c. Qual a relação entre o Boostrap e a responsividade da página?
   Usa a abordagem mobile-first.

Conclusão

O objetivo central foi a contrução de uma aplicação corporative distribuida, utilizando JakartaEE. O projeto simulou um ambiente real de desenvolvimento, integrando um servidor de aplicação robusto (GlassFish), um banco de dados relacional corporativo (SQL Server) e uma arquitetura multicamadas.