Estácio - Unidade São pedro
RJ 140 Km 2, 512 loja 1, São Pedro da Aldeia - RJ, 28941-182

Desenvolvimento Full Stack
Classe: Missão Prática | Nível 4 | Mundo 3
3º Semestre
Marvin de Almeida Costa

Título da Prática: 3º Procedimento | Melhorando o Design da Interface

Objetivos da prática:

- 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 a biblioteca Bootstrap para melhoria do design.
- No final do exercício, o aluno terá criado todos os elementos necessários para
- exibição e entrada de dados na plataforma Java Web, tornando-se capacitado para
- lidar com contextos reais de aplicação.

Todos os códigos solicitados neste roteiro de aula:

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>

<servlet-mapping>
   <servlet-name>ServletProduto</servlet-name>
   <url-pattern>/ServletProduto</url-pattern>
</servlet-mapping>

<servlet-mapping>
   <servlet-name>ServletProdutoFC</servlet-name>
   <url-pattern>/ServletProdutoFC</url-pattern>
</servlet-mapping>

</web-app>
```

ProdutoDados.jsp

```
<%@ page contentType="text/html; charset=UTF-8" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<!DOCTYPE html>
```

```
<html lang="pt-BR">
<head>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet"
integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+A
LEwIH" crossorigin="anonymous">
   <meta charset="UTF-8">
   <title>Cadastro de Produto</title>
   <style>
   </style>
</head>
<body class="container">
   <h1>Dados do Produto</h1>

   <form action="${pageContext.request.contextPath}/ServletProdutoFC" method="post"
class="form">
      <input type="hidden" name="acao" value="${acao}">
      <input type="hidden" name="redirigir" value="true">
      <c:if test="${acao == 'alterar'}">
         <input type="hidden" name="id" value="${produto.idProduto}">
      </c:if>

      <div class="mb-3">
         <label for="nome" class="form-label">Nome:</label>
         <input type="text" id="nome" name="nome" value="${nomeproduto}"
class="form-control">
      </div>

      <div class="mb-3">
         <label for="quantidade" class="form-label">Quantidade:</label>
         <input type="number" id="quantidade" name="quantidade"
value="${produto.quantidade}" class="form-control">
      </div>

      <div class="mb-3">
         <label for="precoVenda" class="form-label">Preço de Venda:</label>
         <input type="number" id="precoVenda" name="precoVenda"
value="${produto.precoVenda}" class="form-control">
      </div>

      <c:if test="${acao == 'incluir'}">
         <input type="submit" value="Adicionar Produto" class="btn btn-primary">
      </c:if>
      <c:if test="${acao == 'alterar'}">
```

```
            <input type="submit" value="Alterar Produto" class="btn btn-primary">
        </c:if>
    </form>
</body>
</html>
```

ProdutoLista.jsp

```
<%@ page contentType="text/html; charset=UTF-8" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet"
integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+A
LEwIH" crossorigin="anonymous">
    <meta charset="UTF-8">
    <title>Listagem de Produtos</title>
    <style>

    </style>
</head>
<body class="container">
    <h1>Listagem de Produtos</h1>

    <a href="${pageContext.request.contextPath}/ServletProdutoFC?acao=incluir"
      class="btn  btn-primary m-2">Novo Produto</a>

      <table class="table table-striped">
       <tr class="table-dark">
            <th>#</th>
            <th>Nome</th>
            <th>Quantidade</th>
            <th>Preço de Venda</th>
            <th>Opções</th>
       </tr>

       <c:forEach var="produto" items="${produtos}">
            <tr>
                <td>${produto.idProduto}</td>
```

```
            <td>${produto.nome}</td>
            <td>${produto.quantidade}</td>
            <td>R$ ${produto.precoVenda}</td>
            <td>
               <a
href="${pageContext.request.contextPath}/ServletProdutoFC?acao=alterar&id=${produto.idPro
duto}"
                  class="btn btn-primary  btn-sm">Alterar</a>

               <a
href="${pageContext.request.contextPath}/ServletProdutoFC?acao=excluir&id=${produto.idPro
duto}"
                  onclick="return confirm('Tem certeza que deseja excluir esse produto?')"
class="btn btn-danger btn-sm"
                  >Excluir</a>
            </td>
         </tr>
      </c:forEach>
   </table>
</body>
</html>
```

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 java.io.IOException;
import jakarta.ejb.EJB;
import jakarta.servlet.RequestDispatcher;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import cadatroee.controller.ProdutoFacadeLocal;
```

```
import cadastroee.model.Produto;
import java.util.List;

public class ServletProdutoFC extends HttpServlet {

    @EJB
    private ProdutoFacadeLocal facade;

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String acao = request.getParameter("acao");
        String destino = "";

        switch (acao) {
            case "listar":
                destino = "ProdutoLista.jsp";
                break;
            case "incluir":
            case "alterar":
                destino = "ProdutoDados.jsp";
                break;
            default:
                // Caso não seja uma ação válida, redireciona para listar
                destino = "ProdutoLista.jsp";
        }

        try {
            switch (acao) {
                case "listar":
                    listarResultados(request);
                    break;
                case "incluir":
                    incluirProduto(request, acao);
                    if (request.getParameter("redirigir") != null) {
                        destino = "ServletProdutoFC?acao=listar";
                    }
                    break;
                case "excluir":
                    excluirProduto(request);
                    destino = "ServletProdutoFC?acao=listar";
                    break;
                case "alterar":
                    alterarProduto(request, acao);
                    if (request.getParameter("redirigir") != null) {
```

```
                destino = "ServletProdutoFC?acao=listar";
            }
            break;
        }
    } catch (Exception e) {
        request.setAttribute("mensagemErro", "Ocorreu um erro: " + e.getMessage());
    }

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

private void listarResultados(HttpServletRequest request) {
    List<Produto> produtos = facade.findAll();
    System.out.println("N produtos: " + produtos.size());
    request.setAttribute("produtos", produtos);
}

private void incluirProduto(HttpServletRequest request, String acao) {
    Produto produto = new Produto();
    request.setAttribute("acao", acao);
    produto.setNome(request.getParameter("nome"));
    produto.setQuantidade(Integer.parseInt(request.getParameter("quantidade")));
    produto.setPrecoVenda(Float.parseFloat(request.getParameter("precoVenda")));
    facade.create(produto);
}

private void alterarProduto(HttpServletRequest request, String acao) {
    int id = Integer.parseInt(request.getParameter("id"));
    request.setAttribute("acao", acao);
    Produto produto = facade.find(id);
    request.setAttribute("nomeproduto", produto.nome);
    request.setAttribute("produto", produto);
    produto.setNome(request.getParameter("nome"));
    produto.setQuantidade(Integer.parseInt(request.getParameter("quantidade")));
    produto.setPrecoVenda(Float.parseFloat(request.getParameter("precoVenda")));
    facade.edit(produto);
}

private void excluirProduto(HttpServletRequest request) {
    int id = Integer.parseInt(request.getParameter("id"));
    Produto produto = facade.find(id);
    facade.remove(produto);
    listarResultados(request);
```

```
    }

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

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

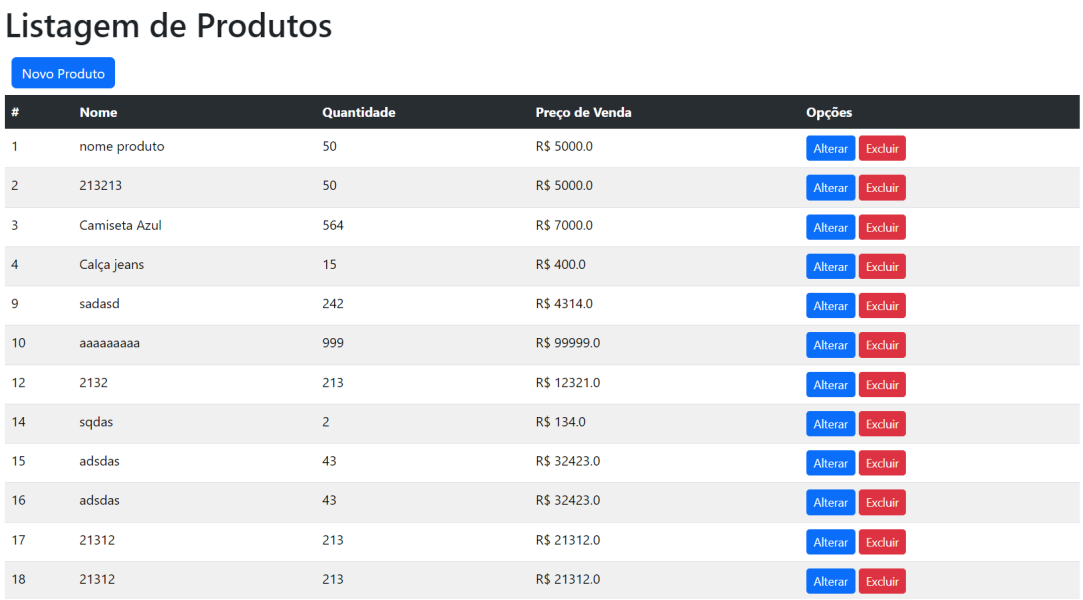
ServletProduto.java

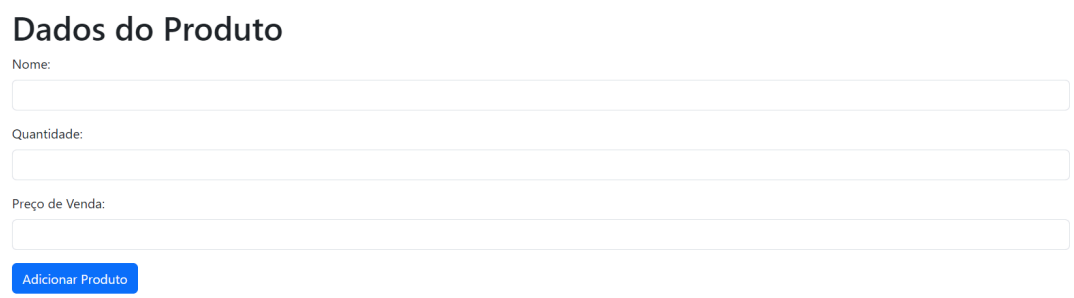
```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
license
 * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this
template
 */
package cadastroee.servlets;

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

/**
 *
 * @author marvin
 */
public class ServletProduto extends HttpServlet {

    @EJB
```

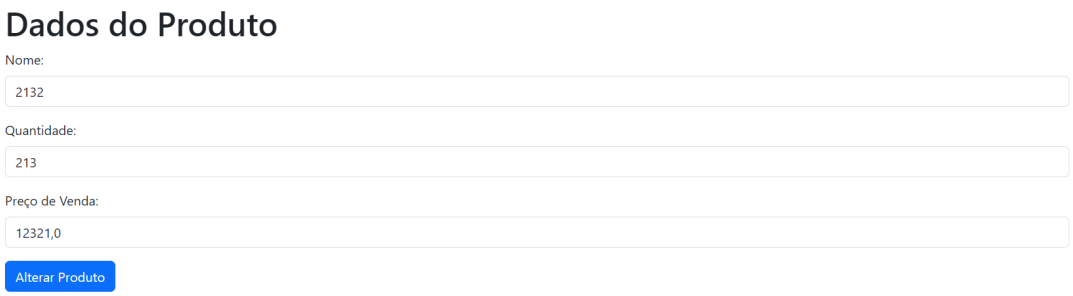
```
    ProdutoFacadeLocal facade;

    /**
     * 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()) {
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet ServletProduto</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Lista de Produtos</h1>");
            out.println("<ul>");

            // Utilizar o facade para recuperar os dados
            List<Produto> produtos = facade.findAll();

            // Apresentar os dados em uma lista HTML
            for (Produto produto : produtos) {
                out.println("<li>" + produto.getNome() + "</li>");
            }

            out.println("</ul>");
            out.println("</body>");
            out.println("</html>");
        }
    }

    // <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 "Short description";
    }// </editor-fold>

}
```

Os resultados da execução dos códigos também devem ser apresentados;







Análise e Conclusão:

Como o framework Bootstrap é utilizado?

O framework Bootstrap é uma das ferramentas mais populares e amplamente utilizadas para desenvolvimento web responsivo e de design moderno. Ele oferece um conjunto robusto de componentes HTML e CSS pré-construídos que facilitam a criação de interfaces de usuário consistentes e eficientes.

Por que o Bootstrap garante a independência estrutural do HTML?

O Bootstrap garante a independência estrutural do HTML de várias maneiras, proporcionando uma solução robusta e flexível para desenvolvimento web responsivo. Vamos explorar os principais motivos:

1. Estrutura semântica

Bootstrap utiliza classes semânticas que correspondem diretamente aos elementos HTML nativos. Isso significa que você pode usar tags como <header>, <nav>, <main>, <section>, <article>, <aside>, <footer>, entre outras, mantendo a estrutura lógica da página.

2. Flexibilidade na aplicação de classes

Bootstrap oferece uma variedade de classes que podem ser combinadas livremente. Isso permite que você aplique estilos específicos apenas onde necessário, mantendo a estrutura HTML puro em outros lugares.

3. Componentes personalizáveis

Bootstrap fornece componentes pré-construídos que podem ser facilmente integrados na sua estrutura HTML existente. Estes componentes são construídos usando classes Bootstrap, mas você pode personalizar seu próprio HTML dentro deles.

4. Responsividade nativa

Bootstrap usa classes responsivas que ajustam automaticamente o layout em diferentes tamanhos de tela. Isso permite que você mantenha uma estrutura HTML consistente em dispositivos móveis e desktops, sem necessidade de tags adicionais ou complexas estruturas condicionais.

Qual a relação entre o Boostrap e a responsividade da página?

O que é responsividade?

Responsividade refere-se à capacidade de um site ou aplicativo se adaptar automaticamente às diferentes tamanhos de tela dos dispositivos, garantindo uma experiência visual e funcional consistente em diversos dispositivos, desde desktops até smartphones.

Como Bootstrap ajuda na responsividade

Bootstrap é uma framework CSS popular que incorpora recursos de responsividade por padrão. Aqui estão algumas maneiras como Bootstrap contribui para a responsividade:

1. Grid System: Bootstrap implementa um sistema de grid flexível que permite criar layouts responsivos facilmente. A grid se ajusta automaticamente ao tamanho da tela disponível.
2. Classes de Responsividade: Bootstrap oferece classes como col-md, col-sm, col-lg, entre outras, que permitem definir o comportamento de elementos em diferentes breakpoints (pontos de quebra) da tela.
3. Responsividade em Componentes: Muitos componentes do Bootstrap são projetados para serem responsivos por padrão, como carrosséis, modais e formulários.
4. Media Queries: Embora ocultas, as media queries utilizadas pelo Bootstrap garantem que os estilos sejam aplicados corretamente em diferentes tamanhos de tela.

Benefícios da combinação Bootstrap e responsividade

1. Tempo de desenvolvimento reduzido: Bootstrap fornece componentes pré-construídos e classes úteis que aceleram o processo de criação de layouts responsivos.
2. Consistência visual: O uso de Bootstrap garante uma aparência consistente em diferentes dispositivos, melhorando a experiência do usuário.

3. Manutenção facilitada: Com uma estrutura baseada em classes, é mais fácil manter e atualizar o design da página.
4. Accessibilidade: Os recursos de acessibilidade incorporados no Bootstrap ajudam a garantir que seu site seja acessível em diversos dispositivos.