

Procedimento 3 - Melhorando o Design da Interface

Objetivos

O objetivo deste procedimento foi implementar e aprimorar um sistema de cadastro de produtos, permitindo as operações de **inclusão, alteração, listagem e exclusão** de dados de produtos de forma prática, eficiente e visualmente agradável.

Foram aplicadas boas práticas de desenvolvimento utilizando **design patterns** como **Facade** e estilização por meio de interfaces com o **Bootstrap**, de maneiras a proporcionar uma melhor experiência visual ao usuário.

Tecnologias Utilizadas

  Jakarta EE

 Servlets

 EJB (Enterprise JavaBeans)

 JPA (Jakarta Persistence API)

 SQL Server

 JDBC Driver para SQL Server (configurado no GlassFish)

 NetBeans

 GlassFish

 Git


 Java

 JSP

 HTML

 Ant (sistema de build)

 JDK 17

 Google Chrome

 Bootstrap



Estrutura do Projeto

O projeto foi dividido nas seguintes camadas:

- modelo/Produto.java → Classe entidade representando a tabela produto, anotada com JPA para mapeamento objeto-relacional.
- modelo/ProdutoDAO.java → Classe de acesso a dados usando JPA.
- Interface ProdutoFacadeLocal encapsulando o acesso aos dados com EJB.
- controle/ServletProdutoFC.java → *Servlet* controlador responsável por gerenciar as requisições CRUD para inclusão, listagem, alteração e exclusão e encaminhar para as páginas JSP.
- ProdutoDados.jsp → Formulário para inclusão e edição de produtos.
- ProdutoLista.jsp → Página para listagem e gerenciamento de produtos com opções de alteração e exclusão.



Funcionalidades

- Cadastro de novos produtos com nome, quantidade e preço.
- Listagem de todos os produtos em formato de tabela.
- Edição dos dados de um produto existente.
- Exclusão de produtos com atualização automática da lista.
- Links dinâmicos para ações no ServletProdutoFC.
- Encaminhamento de requisições por RequestDispatcher.



Atividades Realizadas

- ✓ Criação do ProdutoFacade encapsulando a lógica de validação e interação com o Facade.
- ✓ Desenvolvimento do ServletProdutoFC responsável por receber requisições HTTP e delegar ações ao ProdutoFacade.
- ✓ Implementação e estilização das páginas JSP:
 - ProdutoLista.jsp: aplicação das classes do Bootstrap para estilização da tabela e botões.
 - ProdutoDados.jsp: encapsulamento de campos em div com classes Bootstrap e ajustes para responsividade.

- ✅ Validações nos campos garantindo que não sejam aceitos valores nulos ou inválidos.
- ✅ Formatação correta dos campos, como a exibição do preço de venda com duas casas decimais.
- ✅ Configuração do ambiente incluindo GlassFish, SQL Server, JDBC e integração com o NetBeans.

Desafios Enfrentados

- ❗ Compatibilidade de versões: ajustar a versão do Bootstrap para garantir compatibilidade com Jakarta EE 10 e o ambiente atual.
- ❗ Conversão de dados: tratamento adequado de formatos numéricos, especialmente na exibição de preços com duas casas decimais.
- ❗ Integração frontend-backend: garantir que os valores enviados e recebidos entre JSP, Servlet e banco de dados estivessem corretos.
- ❗ Validação server-side e client-side: implementar validações robustas tanto no frontend quanto no backend, prevenindo falhas e inconsistências.

Códigos

Os códigos foram desenvolvidos com a IDE NetBeans e se encontram em repositório no GitHub onde podem ser acessados pelo link: https://github.com/CarlosCatao/Mundo_3_Nivel_4_Missao_Pratica/tree/main/Procedimento-3/CadastroEE.

Testes

O processo de testes é essencial para garantir a robustez do sistema e identificar pontos de melhoria tanto na interface quanto na lógica de negócios. Através dessas iterações, o sistema evolui para uma versão estável e funcional.

✅ **Testes Realizados**

1. **Teste de Inclusão**

- Foram testados cenários com campos válidos e inválidos. Verificou-se a adição correta do produto na lista e a persistência dos dados no banco.

2. Teste de Alteração

- Verificou-se se os dados do produto eram carregados corretamente ao clicar em "Alterar", e se, após a edição, os dados eram atualizados corretamente na listagem.

3. Teste de Exclusão

- Testou-se a exclusão de produtos com e sem confirmação. Após confirmar a exclusão, foi verificado se o produto desaparecia da lista.

4. Teste de Validação

- Foram realizados testes para envio do formulário com campos em branco, valores negativos e preço com formato inválido.

5. Teste de Navegação

- Todos os fluxos de navegação foram testados: incluir, alterar, excluir, voltar à lista. A consistência do estado da aplicação foi verificada após cada ação.

Resultados

Os resultados da execução dos códigos se encontram ilustrados no arquivo *Resultados.pdf* que se encontra em repositório no GitHub onde podem ser acessados pelo

link:

https://github.com/CarlosCatao/Mundo_3_Nivel_4_Missao_Pratica/blob/main/Procedimento-3/RESULTADOS.pdf.

Conclusão

A realização deste procedimento permitiu consolidar conhecimentos essenciais em desenvolvimento de sistemas web com Java e Jakarta EE, além de reforçar a importância do uso de frameworks de estilização como o Bootstrap para proporcionar interfaces modernas e responsivas. A aplicação do padrão Facade trouxe maior organização e manutenibilidade ao sistema, centralizando regras de negócio.

Apesar dos desafios técnicos encontrados, especialmente relacionados à compatibilidade e formatação de dados, as soluções implementadas demonstraram eficácia e alinhamento com boas práticas de desenvolvimento. O sistema resultante é funcional, organizado e com uma interface intuitiva para o usuário final.

? Questionamentos

▶ Como o framework Bootstrap é utilizado?

O **Bootstrap** é um framework front-end baseado em **HTML**, **CSS** e **JavaScript** que facilita a criação de interfaces modernas, responsivas e compatíveis com diversos navegadores e dispositivos.

Para usar o **CDN do Bootstrap**, basta incluir as tags de *link* e *script* no seu arquivo HTML. O **Bootstrap** oferece os arquivos necessários para **CSS** e **JavaScript** diretamente através de um link de **CDN (Content Delivery Network)**, o que facilita a implementação sem precisar baixar nada no seu servidor.

▶ Por que o Bootstrap garante a independência estrutural do HTML?

O **Bootstrap** separa a estrutura (**HTML**) da aparência e comportamento, promovendo independência estrutural, porque:

- HTML mantém apenas a marcação sem estilo personalizado embutido.
- Toda a estilização é feita por meio de classes CSS padronizadas.
- Esta padronização permite que se mude o layout visual sem reescrever o HTML, apenas alterando ou combinando as classes.
- Reutilização de código
- Padronização visual
- Facilidade de manutenção

▶ Qual a relação entre o Bootstrap e a responsividade da página?

A responsividade é uma característica que permite a uma página web se adaptar automaticamente a diferentes tamanhos de tela (desktop, tablet, celular).

A responsividade no Bootstrap é possível graças a um conjunto de grade flexível (grid system), classes utilitárias e componentes interativos, tudo projetado para se ajustar automaticamente às diferentes resoluções de tela.

Desta forma, por meio de várias ferramentas e classes integradas, o Bootstrap habilita a criação de layouts responsivos, sem precisar escrever media queries manuais, o que economiza tempo e reduz erros.