POLO CHAPADA - MANAUS - AM/UNIVERSIDADE ESTÁCIO DE SÁ

Missão Prática | Nível 4 | Mundo 3

Curso: Desenvolvimento Full Stack
Disciplina Nível 4: RPG0017 - Vamos integrar sistemas
Número da Turma: 2024.2
Semestre Letivo: Mundo-3
Aluno: Gilvan Júnior Nascimento Gonçalves
Matrícula: 202304560188
URL GIT: https://github.com/Kakarotox10/Mundo3-MissaoPratica-Nivel4.git

1º Título da Prática: Vamos integrar sistemas

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

2º Objetivo 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.
5. 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.

1º Procedimento | Camadas de Persistência e Controle

1. Configurar a conexão com SQL Server via NetBeans...

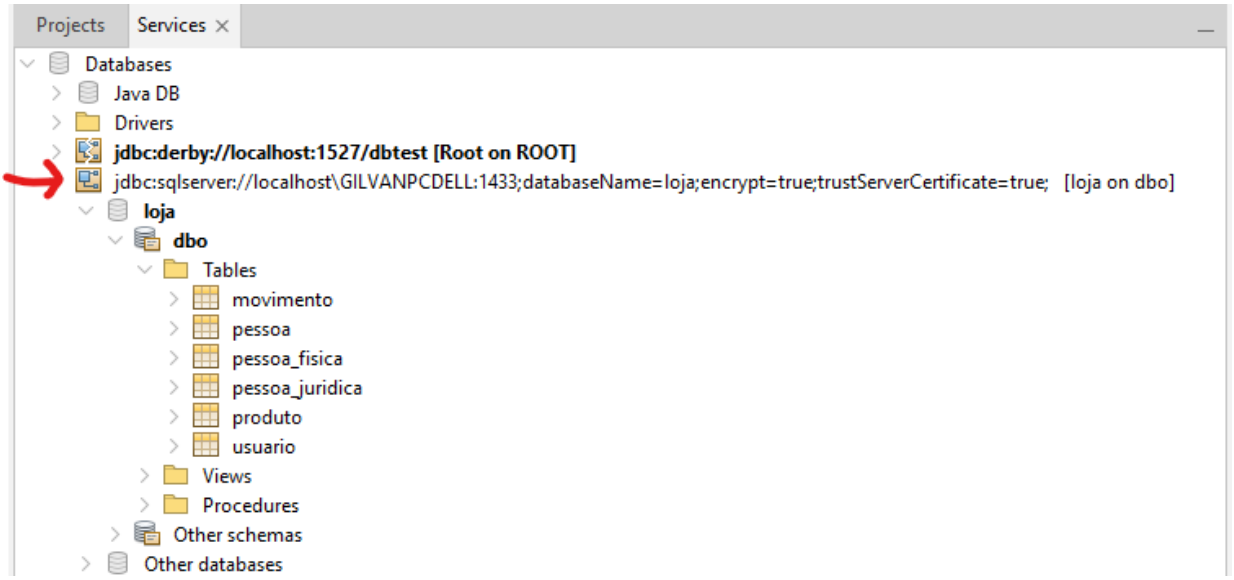


FIGURA1 – conexão com o banco de dados Sql Server.

... e o pool de conexões no GlassFish Server 6.2.1.

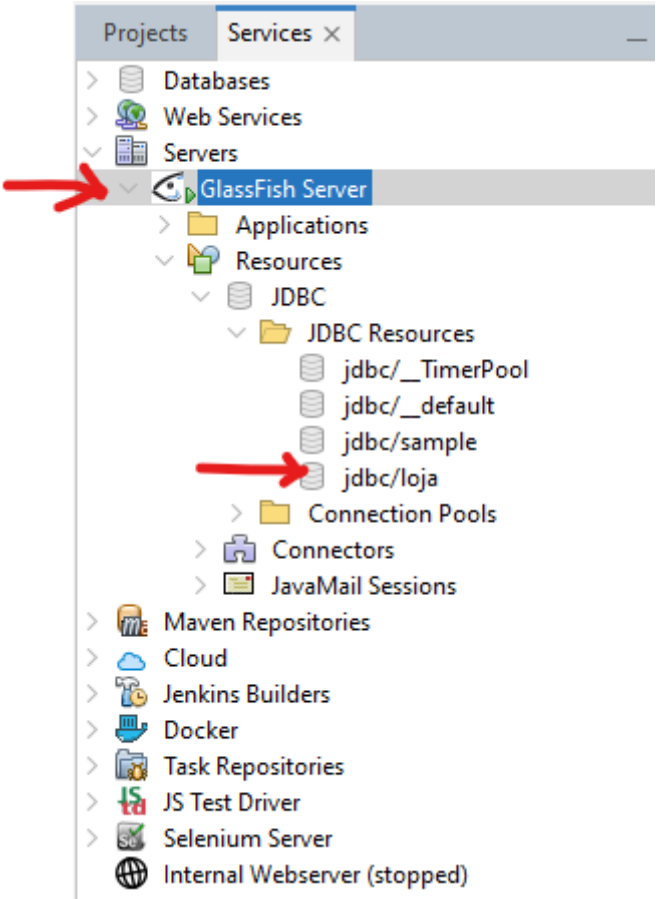


FIGURA2 – Serviços configurados no NetBeans: JDBC, GlassFish, jdbc/Loja, SQLServerPool.

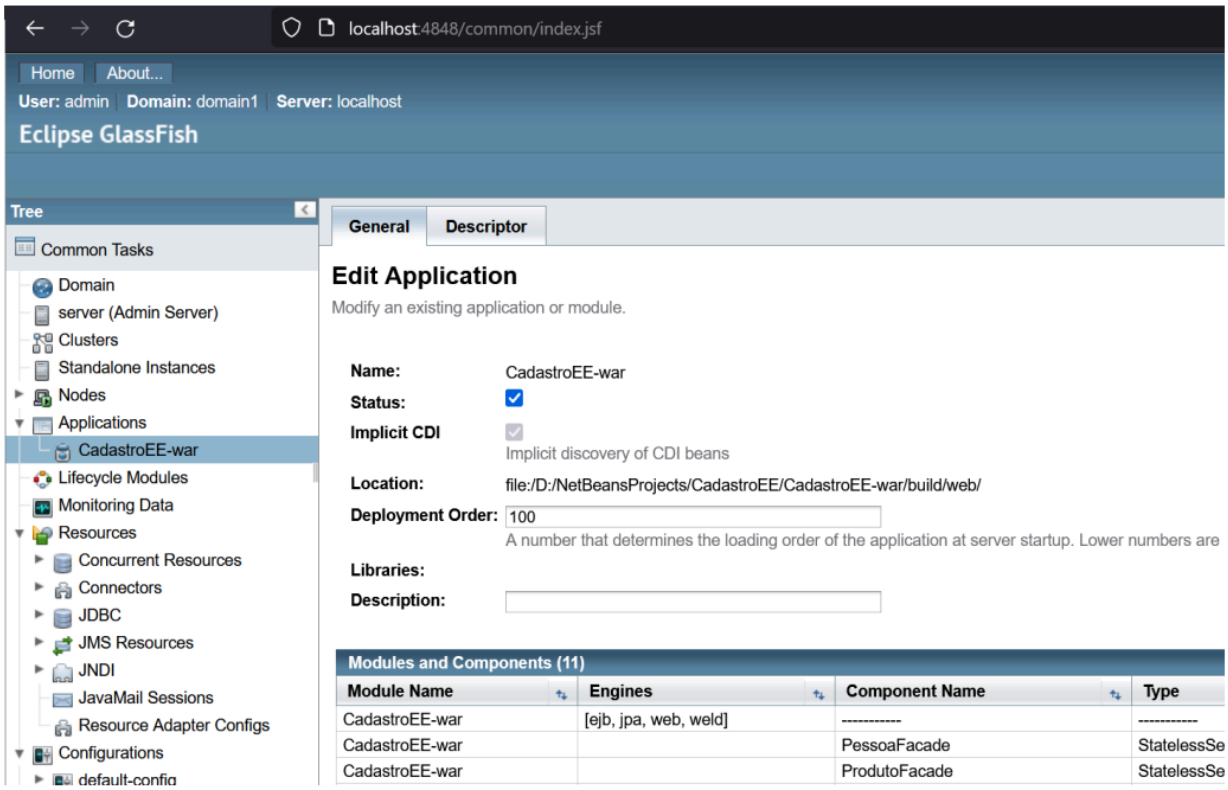


Figura3: aplicação CadastroEE-war em execução no GlassFish.

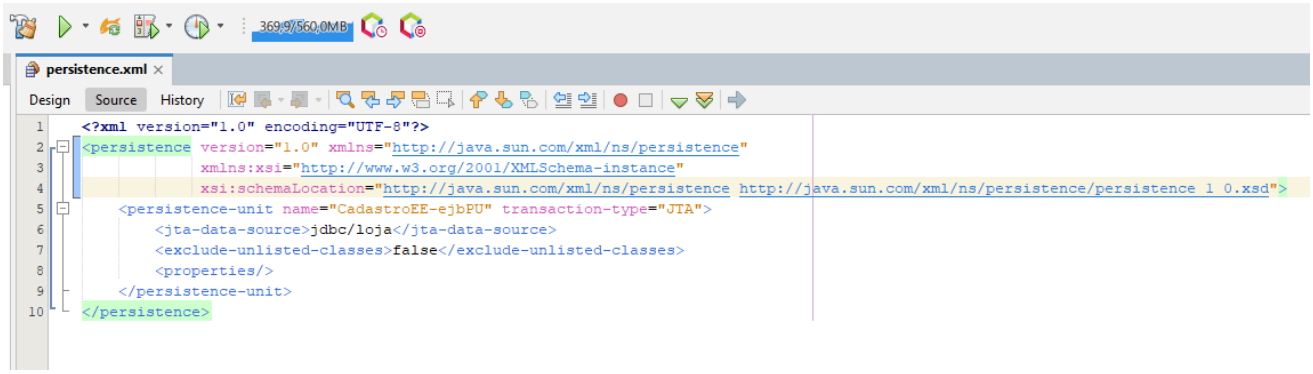


Figura4: Arquivo de configuração persistence.xml

Análise e Conclusão do 1º Procedimento:

a. Como é organizado um projeto corporativo no NetBeans?

A organização de um projeto corporativo no NetBeans, assim como em qualquer outra IDE, é fundamental para garantir a manutenibilidade, escalabilidade e colaboração entre os desenvolvedores. A estrutura ideal pode variar dependendo do tipo de aplicação, da equipe e das convenções da empresa, mas algumas práticas comuns e recomendadas incluem:

Estrutura de Pacotes

- **Pacotes por Camada:** Agrupe as classes de acordo com as camadas da aplicação (modelo, controle, visão). Isso facilita a compreensão do fluxo de dados e a localização de código.
- **Pacotes por Módulo:** Se o projeto for grande, divida-o em módulos menores, cada um com seus próprios pacotes. Isso torna o projeto mais modular e facilita a manutenção.
- **Pacotes por Funcionalidade:** Organize as classes relacionadas a uma determinada funcionalidade em um mesmo pacote.

Nomenclatura de Classes e Pacotes

- **Convenções:** Adote convenções de nomenclatura claras e consistentes (camelCase, PascalCase) para classes, métodos e variáveis.
- **Significado:** Os nomes devem refletir a função da classe ou do pacote.
- **Evite Abreviações:** Use nomes completos para facilitar a compreensão.

Uso de Bibliotecas Externas

- **Gerenciamento de Dependências:** Utilize ferramentas como Maven ou Gradle para gerenciar as dependências do projeto.
- **Versões:** Mantenha as bibliotecas atualizadas para garantir a segurança e aproveitar novas funcionalidades.

Controle de Versão

- **Git:** Utilize o Git para controlar as alterações no código fonte e facilitar a colaboração entre os desenvolvedores.
- **Branches:** Crie branches para diferentes funcionalidades ou versões do software.
- **Commits:** Faça commits frequentes com mensagens claras e concisas.

Testes Unitários

- **Cobertura de Código:** Escreva testes unitários para garantir a qualidade do código e detectar erros precocemente.
- **Frameworks:** Utilize frameworks de testes como JUnit ou TestNG.

Documentação

- **Javadoc:** Gere documentação automaticamente a partir dos comentários no código.
- **Wiki:** Utilize uma wiki para documentar a arquitetura do sistema, as decisões de design e outros aspectos importantes do projeto.

Exemplo da Criação de um novo PROJETO:

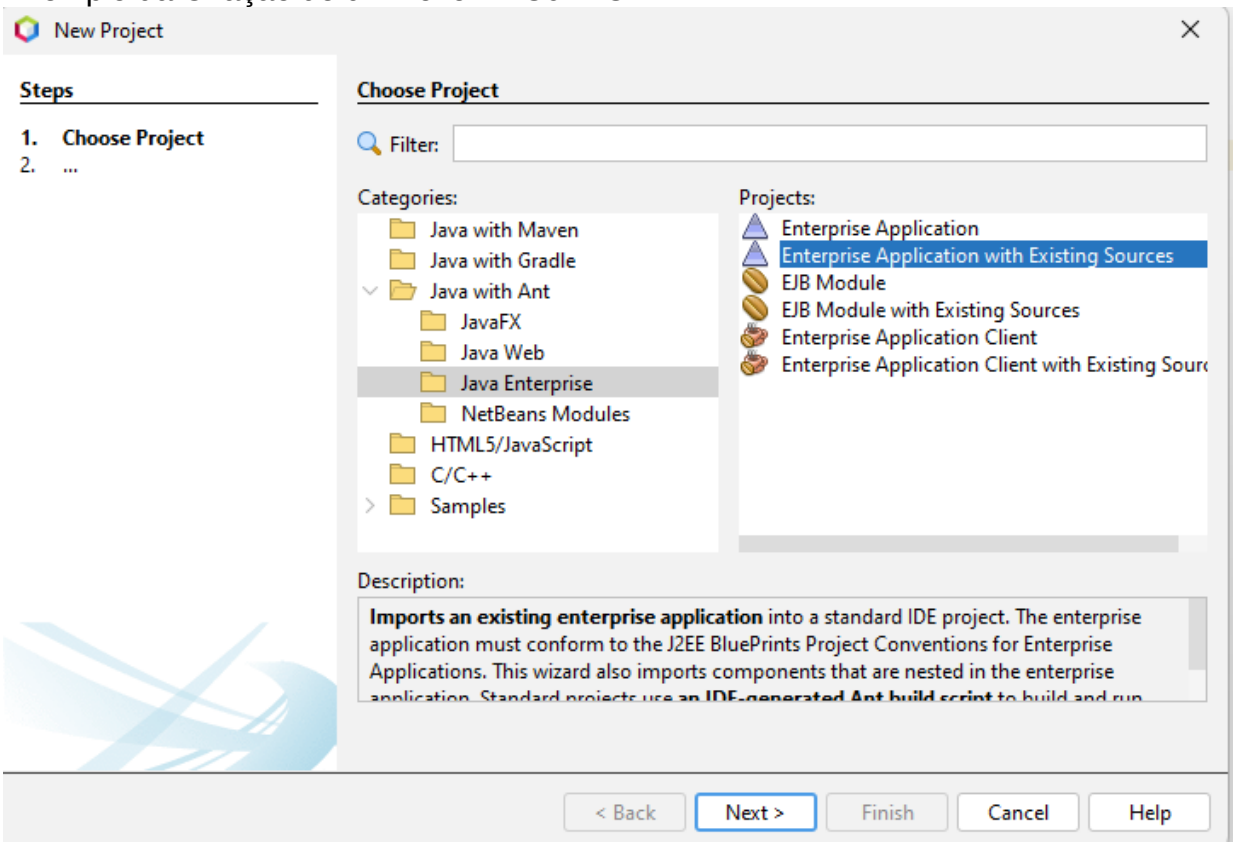


Figura6. Criação de um novo projeto corporativo no NetBeans.

Em resumo, a organização de um projeto corporativo no NetBeans envolve a escolha de uma estrutura de pacotes lógica, a adoção de convenções de nomenclatura, o uso de ferramentas de gerenciamento de dependências e controle de versão, a escrita de testes unitários e a documentação adequada. Ao seguir essas práticas, você contribuirá para a criação de um projeto mais sustentável e colaborativo.

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

As tecnologias **JPA (Java Persistence API)** e **EJB (Enterprise JavaBeans)** desempenham um papel crucial na construção de aplicações web robustas e escaláveis na plataforma Java. Elas oferecem um conjunto de ferramentas e padrões que simplificam o desenvolvimento e a gestão de componentes empresariais.

JPA (Java Persistence API)

- **Persistência de Dados:** A JPA é a especificação padrão para mapear objetos Java em um banco de dados relacional. Ela abstrai os detalhes de acesso ao banco, permitindo que os desenvolvedores se concentrem na lógica de negócio.
- **ORM (Object-Relational Mapping):** A JPA utiliza o ORM para mapear classes Java em tabelas de um banco de dados. Isso facilita a manipulação de dados, como consultas, inserções, atualizações e exclusões.
- **Annotations:** Através de annotations, você define como as classes e propriedades dos objetos serão mapeados para o banco de dados.

- **JPQL (Java Persistence Query Language):** A JPQL é uma linguagem de consulta orientada a objetos que permite realizar consultas complexas sobre os dados persistidos.

Benefícios da JPA:

- **Produtividade:** Aumenta a produtividade dos desenvolvedores, pois elimina a necessidade de escrever manualmente o código SQL.
- **Portabilidade:** Facilita a migração para outros bancos de dados, pois a JPA abstrai as diferenças entre eles.
- **Objeto-Orientado:** Permite trabalhar com objetos Java de forma natural, sem se preocupar com os detalhes de baixo nível do banco de dados.

EJB (Enterprise JavaBeans)

- **Componentes Empresariais:** Os EJBs são componentes reutilizáveis que encapsulam a lógica de negócio de uma aplicação.
- **Tipos de EJB:** Existem diferentes tipos de EJB, como session beans (com estado e sem estado), message-driven beans e entity beans (substituídos pela JPA).
- **Transações:** Os EJBs oferecem suporte a transações, garantindo a consistência dos dados em um sistema distribuído.
- **Segurança:** Os EJBs podem ser protegidos com mecanismos de segurança baseados em papéis e outros recursos.

Benefícios dos EJB:

- **Modularidade:** Permite dividir a aplicação em componentes menores, facilitando a manutenção e o teste.
- **Escalabilidade:** Os EJBs podem ser distribuídos em vários servidores, aumentando a capacidade de processamento da aplicação.
- **Gerenciamento de Contêiner:** O contêiner EJB gerencia o ciclo de vida dos beans, as transações, a segurança e outros aspectos.

JPA e EJB juntos

A combinação de JPA e EJB oferece uma plataforma poderosa para o desenvolvimento de aplicações web empresariais. A JPA cuida da persistência dos dados, enquanto os EJBs encapsulam a lógica de negócio e oferecem serviços como transações e segurança.

Em resumo:

- **JPA** simplifica o acesso a dados e o mapeamento objeto-relacional.
- **EJB** oferece um modelo de componentes para a construção de aplicações empresariais.
- **Juntos** proporcionam uma plataforma robusta e escalável para o desenvolvimento de aplicações web Java.

Ao entender o papel da JPA e EJB, estaremos mais bem preparado para desenvolver aplicações web Java de alta qualidade e escalabilidade.

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

O NetBeans, como uma IDE (Integrated Development Environment) poderosa e popular para Java, oferece uma série de recursos que facilitam significativamente o desenvolvimento de aplicações utilizando JPA e EJB. Essa integração permite que os desenvolvedores sejam mais produtivos ao automatizar tarefas repetitivas, fornece ferramentas de visualização e depuração e oferecer suporte a diversas funcionalidades dessas tecnologias.

Principais recursos do NetBeans que otimizam a produtividade com JPA e EJB:

- **Editor Inteligente:**
    - **Autocompletar:** Sugere automaticamente nomes de classes, métodos, atributos e palavras-chave relacionadas à JPA e EJB, reduzindo a quantidade de código digitado manualmente.
    - **Refatoração:** Permite realizar mudanças estruturais no código de forma segura e eficiente, como renomear classes, métodos e atributos, extrair métodos e alterar assinaturas de métodos.
    - **Verificação de erros:** Identifica erros de sintaxe e semântica em tempo real, permitindo corrigir problemas antes que eles se tornem mais sérios.
- **Geração de código:**
    - **Entidades JPA:** Gera automaticamente classes de entidade a partir de um banco de dados existente ou de um modelo de dados.
    - **EJBs:** Cria templates para diferentes tipos de EJBs, como session beans e message-driven beans, com as configurações padrão.
    - **Queries JPQL:** Auxilia na criação de consultas JPQL, oferecendo sugestões de palavras-chave e entidades.
- **Visualização de dados:**
    - **Diagrama de entidades:** Permite visualizar o relacionamento entre as entidades JPA em um diagrama gráfico, facilitando a compreensão do modelo de dados.
    - **Navegação de código:** Permite navegar facilmente entre as classes, métodos e atributos relacionados à JPA e EJB, utilizando a estrutura do projeto e as referências cruzadas.
- **Depuração:**
    - **Pontos de interrupção:** Permite pausar a execução do código em pontos específicos e inspecionar o estado das variáveis, facilitando a identificação de erros.

    - **Visualização de objetos:** Permite visualizar o conteúdo de objetos Java, incluindo objetos gerenciados pela JPA.
- **Integração com servidores de aplicação:**
    - **Depuração remota:** Permite depurar aplicações Java EE em um servidor de aplicação remoto.
    - **Gerenciamento de projetos:** Facilita a criação, configuração e implantação de projetos Java EE em servidores de aplicação como o GlassFish e o WildFly.
- **Suporte a frameworks:**
    - **Hibernate:** Oferece suporte nativo para o Hibernate, um dos frameworks JPA mais populares.
    - **Outros frameworks:** Suporta outros frameworks JPA e EJB, como EclipseLink.

Benefícios da utilização do NetBeans com JPA e EJB:

- **Aumento da produtividade:** Automatiza tarefas repetitivas e reduz o tempo de desenvolvimento.
- **Melhoria da qualidade do código:** Auxilia na criação de código mais limpo, consistente e livre de erros.
- **Facilidade de aprendizado:** Torna mais fácil aprender e utilizar as tecnologias JPA e EJB.
- **Integração completa:** Oferece uma experiência de desenvolvimento unificada para todas as etapas do ciclo de vida de uma aplicação Java EE.

Em resumo, o NetBeans oferece um conjunto completo de ferramentas que facilitam o desenvolvimento de aplicações com JPA e EJB, tornando os desenvolvedores mais produtivos e permitindo que se concentrem na lógica de negócio.

d. **O que são Servlets, e como o NetBeans oferece suporte à construção desse tipo de componentes em um projeto Web?**

Servlets são programas Java que rodam em um servidor web e respondem a solicitações HTTP. Eles formam a espinha dorsal de muitas aplicações web Java, sendo responsáveis por processar requisições, gerar conteúdo dinâmico e interagir com outros componentes da aplicação.

Como o NetBeans facilita a criação de Servlets:

O NetBeans, como uma IDE voltada para Java, oferece um ambiente de desenvolvimento completo e intuitivo para a criação e gerenciamento de servlets. Algumas das principais funcionalidades que o NetBeans disponibiliza para trabalhar com servlets incluem:

- **Criação de Projetos Web:** O NetBeans permite criar projetos web com facilidade, configurando automaticamente o ambiente necessário para o desenvolvimento de servlets, incluindo a integração com servidores de aplicação como Tomcat.
- **Geração de Templates:** Ao criar um novo servlet, o NetBeans oferece templates pré-configurados, agilizando o processo de desenvolvimento e garantindo uma estrutura básica para o código.
- **Editor Inteligente:** O editor do NetBeans oferece recursos como autocompletar, refatoração e verificação de erros, tornando o processo de codificação mais eficiente e preciso.
- **Integração com o Servidor de Aplicação:** O NetBeans permite configurar e iniciar o servidor de aplicação diretamente da IDE, facilitando a depuração e o teste dos servlets.
- **Suporte a JSP:** Os servlets geralmente trabalham em conjunto com JSP (JavaServer Pages) para gerar a interface do usuário. O NetBeans oferece suporte completo para JSP, incluindo edição, depuração e compilação.
- **Gerenciamento de Bibliotecas:** O NetBeans facilita a inclusão de bibliotecas externas necessárias para o projeto, como frameworks de persistência ou utilitários.
- **Depuração:** O NetBeans permite depurar servlets passo a passo, inspecionar variáveis e identificar erros de forma eficiente.
- **Perfil de Desempenho:** O NetBeans oferece ferramentas para analisar o desempenho da aplicação, ajudando a identificar gargalos e otimizar o código.

Em resumo, o NetBeans oferece um ambiente de desenvolvimento completo e intuitivo para a criação de servlets, simplificando tarefas como configuração, codificação, depuração e teste. Ao utilizar o NetBeans, os desenvolvedores podem se concentrar na lógica de negócio da aplicação, deixando que a IDE cuide dos detalhes técnicos.

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

A comunicação entre Servlets e Session Beans é um aspecto fundamental nas aplicações Java EE. Essa interação permite que a camada de apresentação (Servlets) se conecte à lógica de negócio encapsulada nos Session Beans.

Mecanismo Principal: JNDI

O mecanismo mais comum para essa comunicação é o **JNDI (Java Naming and Directory Interface)**. Ele funciona como uma espécie de lista telefônica para os componentes da aplicação, permitindo que os Servlets procurem e obtenham referências para os Session Beans que precisam utilizar.
Passo a Passo da Comunicação:

1. **Configuração do Deployment Descriptor:** No arquivo web.xml do seu projeto web, você configura uma conexão com o container EJB. Isso inclui informações como o nome JNDI do contexto inicial e a URL do servidor.

2. **Obtenção da Referência do Session Bean:** No servlet, você utiliza o método InitialContext.lookup() para obter uma referência ao Session Bean. Essa referência é obtida através do nome JNDI configurado no deployment descriptor.
3. **Invocar Métodos do Session Bean:** Uma vez com a referência, você pode invocar os métodos do Session Bean como se estivesse chamando um método de qualquer outro objeto Java.

Pontos Importantes:

- **Interface Remota:** O Session Bean deve ter uma interface remota que define os métodos que podem ser acessados remotamente.
- **Contexto Inicial:** O contexto inicial é o ponto de partida para a pesquisa de objetos no JNDI.
- **Narrowing:** O método PortableRemoteObject.narrow() é utilizado para converter a referência obtida do JNDI na interface remota do Session Bean.
- **Transações:** As transações são gerenciadas pelo container EJB, garantindo a consistência dos dados.

Benefícios da Comunicação via JNDI:

- **Desacoplamento:** Os Servlets não precisam conhecer os detalhes de implementação dos Session Beans.
- **Flexibilidade:** Permite alterar a implementação dos Session Beans sem afetar os Servlets.
- **Gerenciamento Centralizado:** A configuração das referências JNDI é centralizada no deployment descriptor.
- **Suporte a Transações:** As transações são gerenciadas pelo container EJB, garantindo a consistência dos dados.

Em resumo:

O JNDI proporciona um mecanismo robusto e flexível para a comunicação entre Servlets e Session Beans. Essa abordagem permite que as aplicações Java EE sejam mais modulares, escaláveis e fáceis de manter.

2º Procedimento | Interface Cadastral com Servlet e JSPs

Resultados da execução dos códigos

O 2º procedimento é continuação do 1º procedimento, com a diferença que é criado um novo servlet denominado **ServletProdutoFC** (fig.7), que utiliza o padrão FC (Front Controller), com a capacidade de exibir a lista de produtos, cadastrar, alterar e excluir produtos, armazenados em banco de dados, através de parâmetros de "ação" na URL ("incluir", "alterar", "excluir").

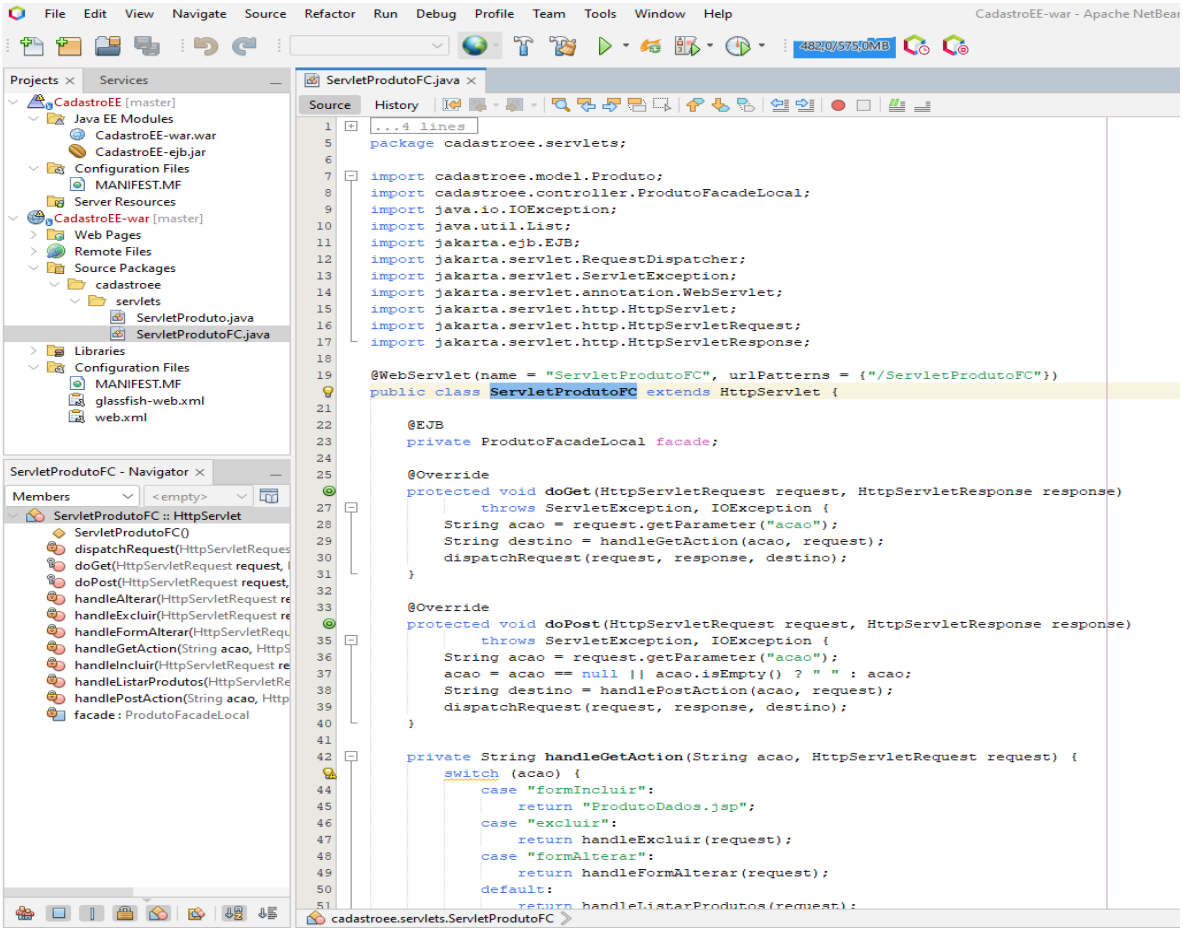
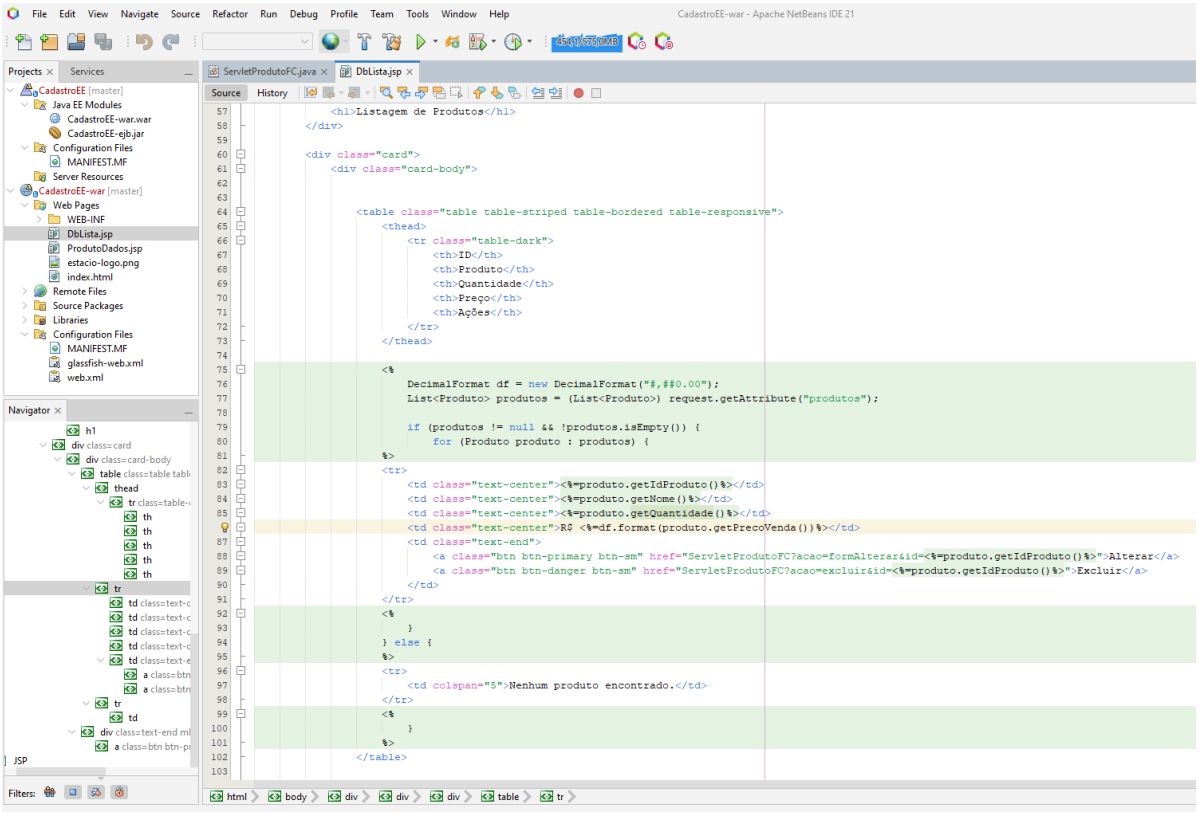


Figura7. Trecho do código do ServletProdutoFC.

As interfaces gráficas JSP (Java Server Page) (fig.8), conforme solicitado neste 2º procedimento, utilizam basicamente HTML, sem uso de bibliotecas CSS, como mostrado na fig.9. Essas interfaces JSP são templates escritos em HTML salvos em arquivos de extensão *.jsp, com uso adequado e posicionado de tags <% %>, onde os dados da aplicação são corretamente inseridos e exibidos.



```
<h1>Listagem de Produtos</h1>
</div>

<div class="card">
    <div class="card-body">


        <table class="table table-striped table-bordered table-responsive">
            <thead>
                <tr class="table-dark">
                    <th>ID</th>
                    <th>Produto</th>
                    <th>Quantidade</th>
                    <th>Preço</th>
                    <th>Ações</th>
                </tr>
            </thead>

            <%
                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()%>">Excluir</a>
                </td>
            </tr>
            <%
                }
            } else {
            %>
            <tr>
                <td colspan="5">Nenhum produto encontrado.</td>
            </tr>
            <%
                }
            %>
        </table>
```

Figura 8. Trecho do código JSP responsável pela exibição da lista de produtos.

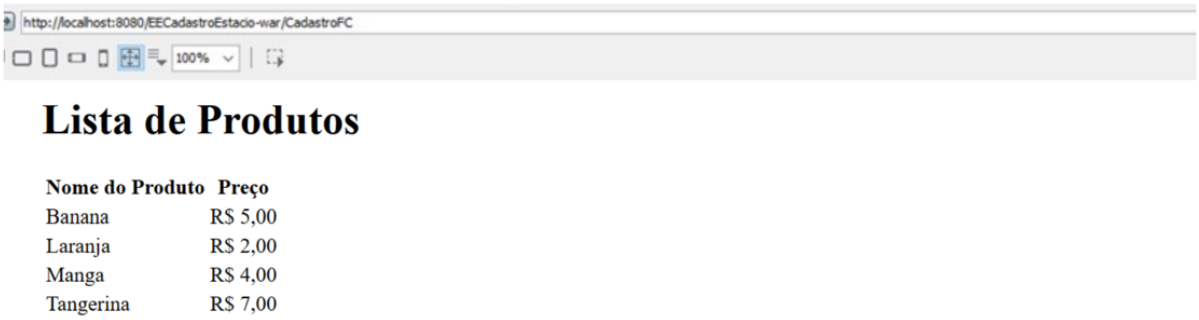


Figura9. ServletProdutoFC em execução, sem uso de bibliotecas CSS, apenas CSS básico

Análise e Conclusão do 2º Procedimento:

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

O padrão **Front Controller** é um padrão de projeto que centraliza o tratamento de todas as requisições HTTP em um único ponto de entrada. Isso simplifica o fluxo de controle da aplicação, facilita a manutenção e promove a reutilização de código.

Como Funciona:

1. **Ponto de Entrada Único:** Todas as requisições HTTP são direcionadas para um único servlet (o Front Controller).
2. **Roteamento:** O Front Controller analisa a URL da requisição e outros parâmetros para determinar qual ação deve ser executada.
3. **Despacho:** O Front Controller delega a execução da ação para outro componente, geralmente um controlador específico para aquela ação.
4. **Retorno:** O controlador processa a requisição e retorna uma resposta, que pode ser uma página HTML, um redirecionamento ou outro tipo de conteúdo.

Componentes Principais:

- **Front Controller:** O servlet que recebe todas as requisições.
- **Request Handler:** Um componente que mapeia as requisições para os controladores específicos.
- **Controlador:** Um componente que processa a lógica de negócio da requisição e prepara os dados para a view.
- **View:** A interface do usuário que apresenta os dados ao usuário.
- **Model:** Representa os dados da aplicação.

Benefícios do Padrão Front Controller:

- **Reutilização de código:** A lógica comum a todas as requisições pode ser centralizada no Front Controller.
- **Facilidade de manutenção:** As alterações no fluxo da aplicação podem ser feitas em um único ponto.
- **Segurança:** O Front Controller pode ser usado para implementar mecanismos de autenticação e autorização.
- **Melhoria da performance:** Ao centralizar o processamento, o Front Controller pode otimizar o desempenho da aplicação.

Frameworks que Utilizam o Padrão Front Controller:

- **Struts:** Um dos primeiros frameworks Java EE a utilizar o padrão Front Controller de forma extensiva.
- **Spring MVC:** Um framework muito popular que oferece uma implementação flexível e configurável do padrão Front Controller.
- **JavaServer Faces (JSF):** Utiliza um Front Controller implícito para gerenciar o ciclo de vida das requisições.

Conclusão:

O padrão Front Controller é um padrão de projeto fundamental para o desenvolvimento de aplicações web Java. Ele proporciona uma arquitetura organizada e escalável, facilitando a manutenção e a evolução da aplicação ao longo do tempo. Ao entender

os princípios do Front Controller, você estará mais bem preparado para construir aplicações web robustas e eficientes.

b) Quais as diferenças e semelhanças entre Servlets e JSPs?

Servlets e JSPs são tecnologias fundamentais para o desenvolvimento de aplicações web em Java, mas possuem características e finalidades distintas. Abaixo segue as principais diferenças e semelhanças entre elas:

Semelhanças:

- **Baseada em Java:** Tanto Servlets quanto JSPs são tecnologias baseadas em Java, o que permite a reutilização de código e a integração com outras ferramentas e frameworks Java.
- **Execução no Servidor:** Ambos são executados no servidor web, gerando conteúdo dinâmico para o cliente.
- **Parte da Plataforma Java EE:** Tanto Servlets quanto JSPs fazem parte da plataforma Java EE, que oferece um conjunto de APIs e ferramentas para o desenvolvimento de aplicações empresariais.
- **Objetivo Comum:** O objetivo principal de ambos é gerar conteúdo dinâmico para o cliente em resposta a requisições HTTP.

Diferenças:

- **Natureza:**
  - **Servlets:** São classes Java que implementam a interface HttpServlet. São mais adequados para a lógica de negócios e a manipulação de dados.
  - **JSPs:** São páginas HTML que contêm tags especiais para a inserção de código Java. São mais adequados para a apresentação de dados e a criação de interfaces de usuário.
- **Foco:**
  - **Servlets:** Focam na lógica de processamento, como a manipulação de dados, a comunicação com bancos de dados e a geração de respostas.
  - **JSPs:** Focam na apresentação dos dados, utilizando tags e expressões para gerar o conteúdo HTML.
- **Sintaxe:**
  - **Servlets:** Utilizam a sintaxe da linguagem Java.
  - **JSPs:** Utilizam uma mistura de HTML e tags específicas para a inserção de código Java.
- **Ciclo de Vida:**
  - **Servlets:** Possuem um ciclo de vida mais complexo, envolvendo inicialização, serviço e destruição.
  - **JSPs:** São compiladas em Servlets na primeira requisição, simplificando o desenvolvimento.

- **Uso:**
    - **Servlets:** São mais adequados para tarefas complexas, como a validação de dados, a comunicação com outros sistemas e a geração de relatórios.
    - **JSPs:** São mais adequados para a criação de interfaces de usuário dinâmicas e a apresentação de dados.

Quando Usar Cada Um?

- **Servlets:**
    - Lógica de negócios complexa
    - Manipulação de dados
    - Geração de conteúdo dinâmico
    - Filtros e listeners
- **JSPs:**
    - Criação de interfaces de usuário
    - Apresentação de dados
    - Mistura de HTML e Java
    - Templates

Trabalhando Juntos

Na prática, Servlets e JSPs são frequentemente utilizados em conjunto. Os Servlets são responsáveis por processar as requisições e preparar os dados, enquanto os JSPs são responsáveis por gerar a interface do usuário.

Exemplo:

Um Servlet pode ser responsável por buscar dados de um banco de dados e armazená-los em um objeto JavaBean. Em seguida, o Servlet encaminha a requisição para um JSP, que exibe os dados em uma tabela HTML.

Em resumo:

Servlets e JSPs são ferramentas poderosas para o desenvolvimento de aplicações web em Java. A escolha entre um ou outro depende da tarefa a ser realizada. Ao entender as diferenças e semelhanças entre eles, você poderá tomar decisões mais informadas sobre a arquitetura da sua aplicação.

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?

Redirecionamento Simples vs. Forward

Redirecionamento Simples:

- **Nova requisição:** O servidor envia uma resposta HTTP 302 (Found) ao cliente, indicando que o recurso solicitado foi movido para um novo local. O navegador faz uma nova requisição para a URL indicada no cabeçalho Location da resposta.
- **Duas requisições:** Envolve duas requisições HTTP distintas.
- **URL visível:** A nova URL fica visível na barra de endereço do navegador.
- **Uso:** Ideal para redirecionar o usuário para uma página completamente diferente, como após um login bem-sucedido ou após a conclusão de um processo.

Forward:

- **Mesma requisição:** O servidor encaminha a requisição para outro recurso dentro da mesma aplicação, sem que o cliente seja notificado.
- **Uma única requisição:** Envolve apenas uma requisição HTTP.
- **URL não visível:** A URL original permanece na barra de endereço do navegador.
- **Uso:** Ideal para encaminhar a requisição para outro recurso dentro da mesma aplicação, como para exibir diferentes partes de uma mesma página, dependendo da ação do usuário.

Parâmetros e Atributos em HttpRequest

Parâmetros:

- **Dados enviados pelo cliente:** São informações enviadas pelo cliente para o servidor, geralmente através de formulários HTML ou na URL.
- **Nome/valor:** Cada parâmetro possui um nome e um valor associado.
- **Acesso:** Os parâmetros podem ser acessados através do método getParameter() do objeto HttpServletRequest.
- **Uso:** Para passar dados entre páginas diferentes, como os dados de um formulário.

Atributos:

- **Dados internos da aplicação:** São informações armazenadas no objeto HttpServletRequest para serem utilizadas dentro da mesma requisição.
- **Nome/valor:** Assim como os parâmetros, possuem um nome e um valor.
- **Acesso:** Os atributos podem ser acessados e definidos através dos métodos setAttribute() e getAttribute() do objeto HttpServletRequest.

- **Uso:** Para compartilhar dados entre diferentes partes da mesma requisição, como passar informações entre um Servlet e um JSP.

Quando Usar Cada Um?

- **Parâmetros:** Para passar dados entre diferentes requisições ou entre o cliente e o servidor.
- **Atributos:** Para compartilhar dados dentro da mesma requisição, como passar informações entre Servlets e JSPs.

Em resumo:

- **Redirecionamento:** Cria uma nova requisição e é ideal para mudar de página completamente.
- **Forward:** Encaminha a requisição para outro recurso dentro da mesma aplicação e é ideal para compartilhar dados entre diferentes partes da mesma requisição.
- **Parâmetros:** São utilizados para passar dados entre diferentes requisições.
- **Atributos:** São utilizados para compartilhar dados dentro da mesma requisição.

Escolhendo a opção correta:

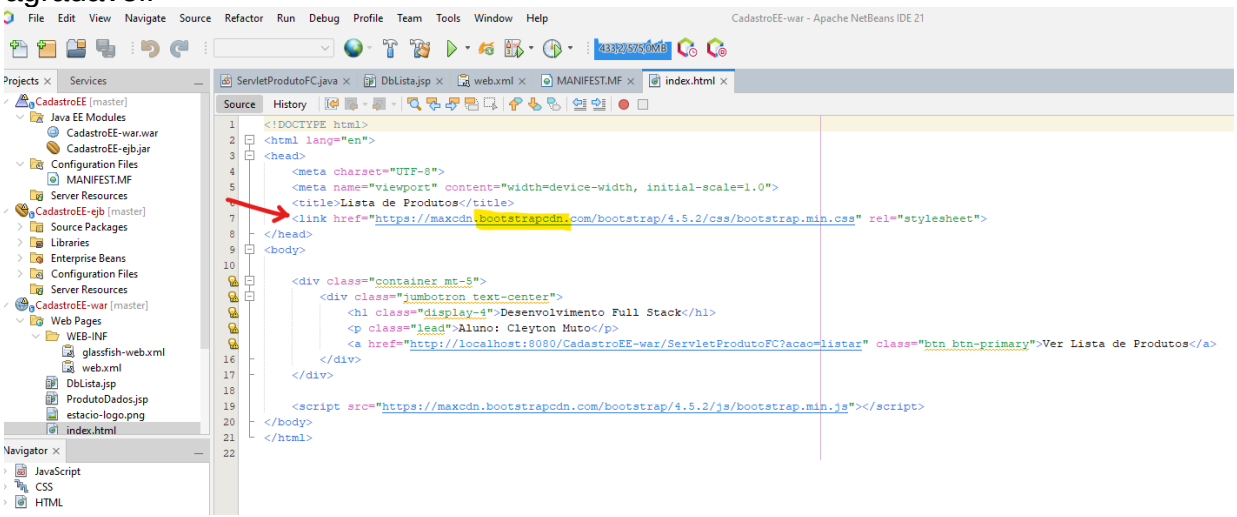
A escolha entre redirecionamento e forward depende da sua necessidade específica. Se precisarmos mudar de página completamente, use o redirecionamento. Se precisarmos compartilhar dados entre diferentes partes da mesma requisição, use o forward e os atributos.

Ao entender as diferenças entre esses conceitos, poderemos construir aplicações web mais eficientes e organizadas.

3º Procedimento | Melhorando o Design da Interface

1. Resultados da execução dos códigos

0 3º procedimento utiliza a biblioteca do framework Bootstrap, referenciada no arquivo index.html com mostra a fig.10, o que torna a interface gráfica muito mais bonita e agradável.



Figrua10. Mostra a tag <link href=””> referenciando o framework bootstrap para ser utilizado na aplicação.

Figura11. Mostra a mesma listagem de produtos do procedimento anterior, mas com uso de Bootstrap, conforme solicitado no enunciado da Missão Prática.

Figura12. Tela de Cadastro de Produto.

Figura13. ServletProdutoFC em execução, com uso do framework CSS Bootstrap.

d) Como o framework Bootstrap é utilizado?

O Bootstrap é um framework front-end extremamente popular que oferece um conjunto de classes CSS e componentes JavaScript pré-construídos, com o objetivo de agilizar e facilitar o desenvolvimento de sites e aplicações web responsivos.

Principais Utilidades:

- **Layout responsivo:** O Bootstrap fornece um sistema de grid responsivo que se adapta automaticamente a diferentes tamanhos de tela, desde smartphones até desktops, garantindo uma experiência de usuário consistente em diversos dispositivos.
- **Componentes pré-construídos:** Oferece uma ampla variedade de componentes prontos para uso, como botões, formulários, modais, carrosséis, etc., com estilos e funcionalidades pré-definidas, economizando tempo e esforço no desenvolvimento.

- **Estilos CSS:** Possui um conjunto de estilos CSS bem estruturado e personalizável, que permite criar interfaces visuais atraentes e consistentes com facilidade.
- **JavaScript:** Inclui plugins JavaScript para adicionar interatividade aos componentes, como modais, tooltips e carrosséis.

Vantagens do Bootstrap:

- **Agilidade:** Acelera o desenvolvimento, pois muitos elementos já estão prontos para uso.
- **Consistência:** Garante um visual consistente em diferentes projetos.
- **Responsividade:** Se adapta a diferentes dispositivos automaticamente.
- **Comunidade:** Possui uma grande comunidade e ampla documentação.

Em resumo:

O Bootstrap é uma ferramenta poderosa para desenvolvedores web que desejam criar sites responsivos e modernos de forma rápida e eficiente. Ao utilizar as classes e componentes pré-construídos, você pode se concentrar na lógica da sua aplicação, em vez de se preocupar com os detalhes de estilização e layout.

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

O Bootstrap garante a independência estrutural do HTML ao separar a estrutura do conteúdo da sua apresentação visual.

Entendendo a Afirmação:

- **Estrutura do Conteúdo**: Refere-se à organização lógica do seu HTML, como as tags , <header>, <section>, <footer>, etc., que definem as diferentes partes de uma página.
- **Apresentação Visual**: Refere-se a como os elementos da página se apresentam visualmente, como cores, fontes, tamanhos, espaçamentos, etc.

Como o Bootstrap Concretiza Isso:

1. **Classes CSS:**
    - **Semântica:** As classes CSS do Bootstrap são projetadas para ter um significado semântico, ou seja, elas descrevem o propósito de um elemento na página, e não apenas sua aparência. Por exemplo, a classe btn indica que um elemento é um botão, e a classe container indica um container para outros elementos.
    - **Reusabilidade:** Essas classes podem ser aplicadas a qualquer elemento HTML, permitindo que você altere a aparência de um elemento simplesmente adicionando ou removendo classes.
    - **Personalização:** Você pode personalizar as classes CSS do Bootstrap para criar um estilo único para sua aplicação, sem precisar reescrever todo o CSS.
2. **Grid System:**
    - **Flexibilidade:** O sistema de grid do Bootstrap permite criar layouts responsivos e adaptáveis a diferentes tamanhos de tela, separando a estrutura da página da sua apresentação visual. Você pode definir colunas e linhas de forma flexível, e as classes CSS do grid se encarregam de ajustar o layout automaticamente.

3. **Componentes Pré-construídos:**
    - **Abstração:** Os componentes pré-construídos do Bootstrap, como botões, formulários e modais, encapsulam a estrutura HTML e as classes CSS necessárias para criar esses elementos, permitindo que você se concentre na lógica da sua aplicação, em vez de se preocupar com os detalhes de implementação.

Vantagens da Independência Estrutural:

- **Manutenção:** Ao separar a estrutura do conteúdo da sua apresentação visual, fica mais fácil fazer alterações no design da sua aplicação sem afetar a estrutura do HTML.
- **Reusabilidade:** As classes CSS do Bootstrap podem ser reutilizadas em diferentes projetos, economizando tempo e esforço.
- **Acessibilidade:** O Bootstrap segue as melhores práticas de acessibilidade, garantindo que seus sites sejam acessíveis a todos os usuários, independentemente de suas habilidades ou dispositivos.
- **Colaboração:** Desenvolvedores com diferentes níveis de experiência podem trabalhar juntos em um projeto, pois a estrutura do HTML é clara e intuitiva.

Em resumo:

O Bootstrap permite que criemos sites e aplicações web bem estruturados, responsivos e visivelmente atraentes, sem se preocupar com os detalhes de implementação. Ao separar a estrutura do conteúdo da sua apresentação visual, o Bootstrap facilita a manutenção, a colaboração e a criação de experiências de usuário consistentes.

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

O **Bootstrap** é um framework front-end extremamente popular que oferece um conjunto de ferramentas e recursos para criar sites e aplicações web responsivos. A **responsividade** de um site, por sua vez, garante que ele se adapte automaticamente a diferentes tamanhos de tela, desde smartphones até desktops, proporcionando uma experiência de usuário consistente em qualquer dispositivo.

Como o Bootstrap Garante a Responsividade:

1. **Grid System Flexível:**
    - **Base do Layout:** O Bootstrap utiliza um sistema de grid baseado em flexbox, que permite criar layouts flexíveis e adaptáveis.
    - **Colunas Responsivas:** As colunas do grid se ajustam automaticamente de acordo com o tamanho da tela, permitindo que você crie layouts que se reorganizam de forma inteligente.
    - **Breakpoints:** O Bootstrap define pontos de quebra (breakpoints) para ajustar o layout em diferentes tamanhos de tela, como smartphones, tablets e desktops.

2. **Classes CSS Responsivas:**

    - **Modificadores:** As classes CSS do Bootstrap incluem modificadores que permitem alterar o comportamento de um elemento em diferentes tamanhos de tela. Por exemplo, a classe col-md-6 fará com que um elemento ocupe 6 colunas em telas médias e maiores.

    - **Utilitários:** O Bootstrap oferece uma variedade de classes utilitárias para ajustar o espaçamento, margens, padding e outras propriedades CSS de forma responsiva.

3. **Componentes Pré-construídos Responsivos:**

    - **Adaptação Automática:** Os componentes pré-construídos do Bootstrap, como botões, formulários e modais, são projetados para se adaptar automaticamente a diferentes tamanhos de tela, garantindo uma aparência consistente em todos os dispositivos.

Benefícios da Responsividade com Bootstrap:

- **Experiência do Usuário:** Garante que o seu site seja fácil de usar em qualquer dispositivo.
- **SEO:** Melhora o posicionamento do seu site nos resultados de busca, pois os mecanismos de busca valorizam sites responsivos.
- **Manutenção:** Facilita a manutenção do seu site, pois você não precisa criar versões diferentes para cada dispositivo.
- **Desenvolvimento Rápido:** Agiliza o desenvolvimento, pois você pode criar layouts complexos com poucas linhas de código.

Resumo:

O Bootstrap oferece um conjunto de ferramentas poderosas para criar sites responsivos de forma rápida e fácil. Ao utilizar o sistema de grid, as classes CSS responsivas e os componentes pré-construídos, você pode garantir que o seu site se adapte perfeitamente a qualquer dispositivo, proporcionando uma experiência de usuário otimizada.