UNIVERSIDADE ESTÁCIO DE SÁ

Missão Prática | Nível 4 | Mundo 3

Curso: Desenvolvimento FullStack

Disciplina Nível 4: RPG0017 - Vamos Integrar Sistemas

Polo: Arautos, Campinas, SP

Semestre Letivo: Mundo-3

Aluno: FRANCISCO SÉRGIO

ALVES DE ARAÚJO

Matrícula: 202308029169

URL GITHUB:

https://github.com/SergioAlves1327/trabalho_n-vel_04.git

RELATÓRIO DA MISSÃO PRÁTICA

Objetivos da prática

Este projeto tem como propósito principal desenvolver um sistema cadastral com interface web, usando tecnologias como Servlets, JPA e JEE. A prática visa proporcionar uma visão ampla do desenvolvimento de aplicativos web na plataforma Java, desde a camada de persistência até a interface de usuário. Mais específicamente, pretende-se implementar a persistência de dados utilizando Java Persistence API (JPA), garantindo a integridade e a segurança das informações no banco de dados. Além disso, será necessário desenvolver regras de negócio na plataforma Java Enterprise Edition (JEE), por meio de Enterprise JavaBeans (EJBs), para realizar operações de manipulação dos dados de forma eficiente e organizada.

A criação de um sistema cadastral web utilizando Servlets e JavaServer Pages (JSPs) é outro objetivo importante deste projeto, proporcionando uma interface amigável para o usuário interagir com o sistema. Por fim, será utilizado o framework Bootstrap para aprimorar o design e a usabilidade da interface do sistema, garantindo uma experiência acessível e agradável ao usuário final.

Resumindo, os objetivos específicos do projeto tratam de:

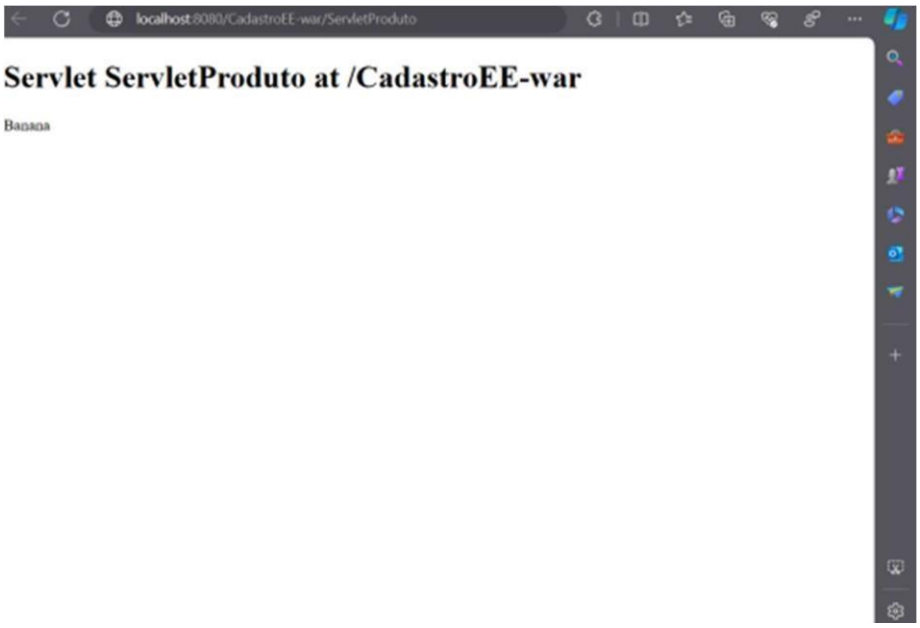
1. **Persistência de Dados com JPA:** Implementar a camada de persistência utilizando a Java Persistence API (JPA) para garantir a integridade e segurança dos dados no banco de dados.
2. **Regras de Negócio com EJB:** Desenvolver regras de negócio na plataforma JEE usando Enterprise JavaBeans (EJBs), assegurando a eficiência e organização na manipulação de dados.
3. **Sistema Cadastral com Servlets e JSPs:** Criar um sistema cadastral que ofereça ao usuário uma interface amigável de interação.
4. **Design e Usabilidade com Bootstrap:** Utilizar o framework Bootstrap para aprimorar o design e a usabilidade da interface, garantindo uma experiência agradável ao usuário final.

E os Procedimentos Realizados

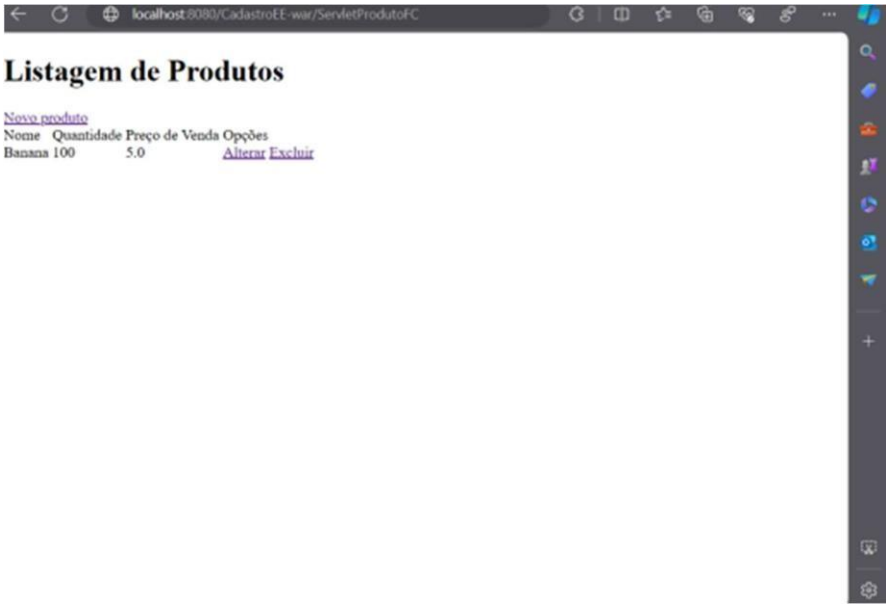
1. **Camadas de Persistência e Controle:** Implementação da persistência de dados e controle de acesso.
2. **Interface Cadastral com Servlets e JSPs:** Desenvolvimento da interface de cadastro.
3. **Melhoria no Design da Interface:** Aperfeiçoamento do visual da interface para maior acessibilidade e facilidade de uso.

Resultados obtidos

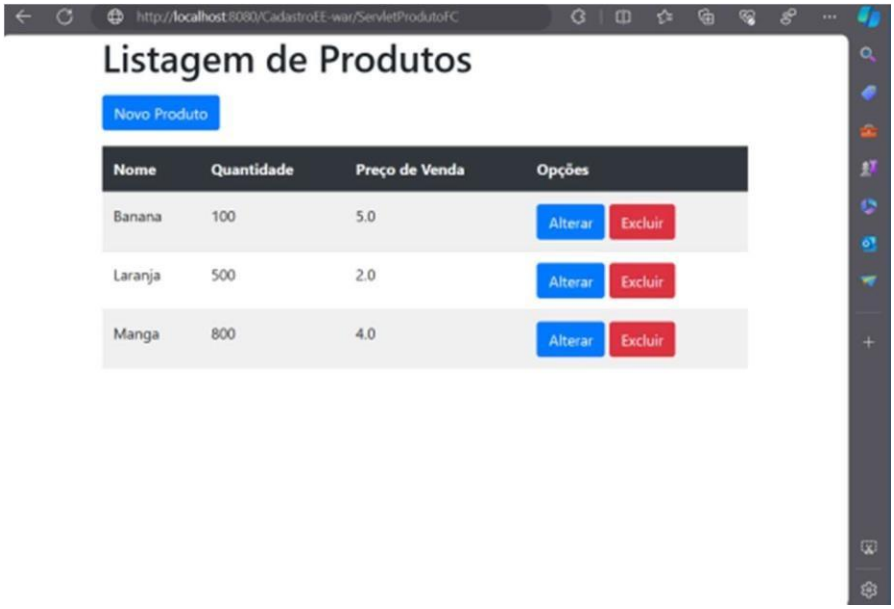
1 - Camadas de persistência e controle

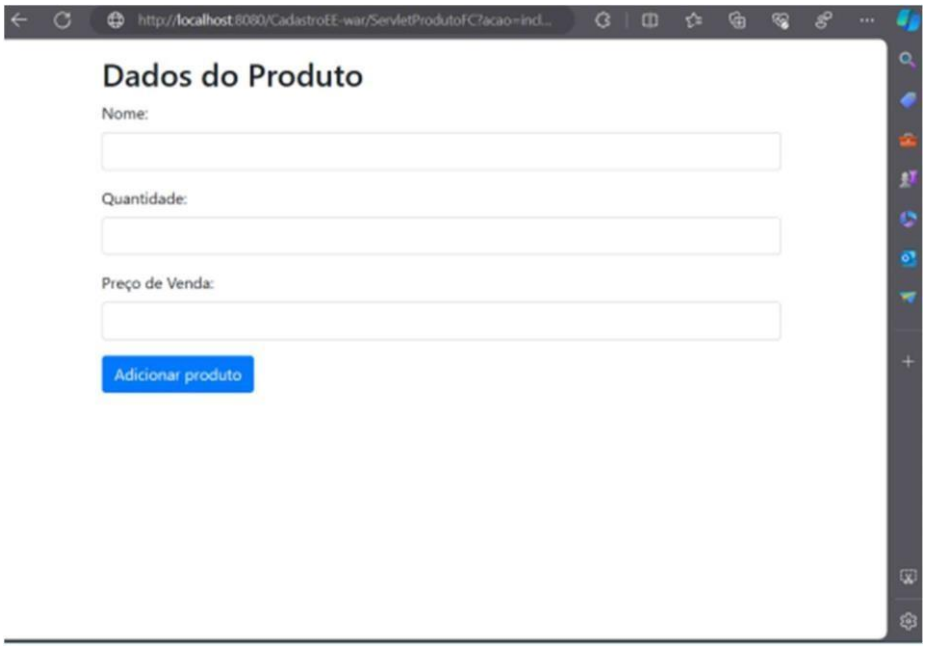


2 - Interface cadastral com Servlet e JSPs



3 - Melhorando o design da interface





Análise e Conclusão

1. Organizando um projeto corporativo no NetBeans?

No NetBeans, um projeto corporativo é organizado hierarquicamente, separando as camadas e componentes da aplicação em módulos. Ao iniciar um projeto de aplicação corporativa Java, o NetBeans cria subprojetos: um para o arquivo EAR (Enterprise Archive) principal e outros para EJB e camada web. Essa estrutura modular facilita o desenvolvimento, a manutenção e a escalabilidade da aplicação, promovendo uma organização clara e separação de responsabilidades.

Por exemplo, ao criar um projeto do tipo Java Enterprise Application no NetBeans, são gerados três subprojetos: um para o arquivo EAR (Enterprise Archive), que é o principal, e dois outros projetos dependentes, um para o EJB (Enterprise JavaBeans) e outro para a camada web. Essa estrutura permite uma organização clara e separação de responsabilidades, facilitando o desenvolvimento, manutenção e escalabilidade da aplicação. Dentro de cada subprojeto, os componentes são organizados de acordo com a sua função. Por exemplo, no projeto EJB, as entidades JPA (Java Persistence API), os Session Beans e outras classes relacionadas à lógica de negócio são agrupadas em pacotes específicos. No projeto web, os Servlets, JSPs (JavaServer Pages), recursos estáticos como CSS e JavaScript, entre outros, são organizados de forma semelhante, em diretórios e pacotes adequados. Além disso, o NetBeans oferece recursos de gerenciamento de dependências, facilitando a inclusão de bibliotecas externas, e integração com sistemas de controle de versão como o Git, para um controle eficiente do código fonte.

Essa organização permite que desenvolvedores trabalhem de forma colaborativa em projetos corporativos, seguindo boas práticas de desenvolvimento e garantindo a qualidade do software entregue.

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

O JPA atua na camada de persistência de dados, permitindo interações orientadas a objetos com o banco de dados, sem precisar de SQL direto. Já os EJBs fornecem um ambiente seguro e transacional para a lógica de negócios, facilitando o controle de transações e gerenciamento de concorrência. Com JPA e EJB, segue-se o padrão MVC (Model-View-Controller), que promove a separação de responsabilidades.

A JPA é responsável pela camada de persistência dos dados, permitindo que os desenvolvedores interajam com o banco de dados de forma orientada a objetos, sem a necessidade de lidar diretamente com SQL. Ela simplifica o acesso e manipulação dos dados por meio de entidades, que representam objetos mapeados para tabelas do banco de dados. Com a JPA, é possível realizar operações de CRUD (Create, Read, Update, Delete) de forma eficiente, além de oferecer recursos avançados como consultas JPQL (Java Persistence Query Language) e relacionamentos entre entidades. Por outro lado, os EJBs fornecem um ambiente de execução seguro e transacional para a lógica de negócios da aplicação. Eles são componentes distribuídos que encapsulam a lógica de negócios e podem ser implantados em servidores de aplicativos Java EE.

Os EJBs oferecem recursos como controle de transações, segurança, escalabilidade e gerenciamento de concorrência, tornando-os ideais para implementar a lógica empresarial complexa de uma aplicação web. Ao utilizar JPA em conjunto com EJB, os desenvolvedores podem criar aplicativos web que seguem o padrão arquitetural MVC (Model-View-Controller), onde a JPA atua como o modelo, representando os dados da aplicação, e os EJBs atuam como o controlador, implementando a lógica de negócios. Essa abordagem permite uma separação clara de responsabilidades e facilita a manutenção e evolução da aplicação ao longo do tempo.

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

O NetBeans permite integração direta com servidores Java EE, assistência de código, geração rápida de código, e recursos de depuração e gerenciamento de banco de dados. Ferramentas integradas possibilitam criar entidades JPA e EJBs com rapidez, otimizando tempo e reduzindo erros. Além disso, o NetBeans proporciona assistência durante o processo de codificação, fornecendo recursos como sugestões de código, preenchimento automático e realce de sintaxe.

Essas ferramentas auxiliam os desenvolvedores a escrever código de forma mais rápida e precisa, reduzindo erros e aumentando a produtividade. A geração de código é simplificada por meio de ferramentas integradas no NetBeans. Os desenvolvedores podem criar rapidamente entidades JPA a partir de tabelas de banco de dados existentes e EJBs a partir dessas entidades. Isso economiza tempo e esforço, permitindo que os desenvolvedores se concentrem em aspectos mais críticos do desenvolvimento. A capacidade de depuração também é fundamental.

O NetBeans oferece suporte completo para depurar aplicativos que utilizam EJB e JPA, facilitando a identificação e correção de problemas de lógica de negócios e acesso a dados. Isso ajuda a garantir a qualidade do software desenvolvido e reduz o tempo gasto na resolução de bugs. A integração com ferramentas de gerenciamento de banco de dados é outra vantagem. O NetBeans permite que os desenvolvedores criem e mantenham bancos de dados utilizados por aplicativos JPA de forma eficiente. Isso é importante para garantir a integridade e o desempenho

dos dados manipulados pela aplicação. Por fim, o gerenciamento de dependências e bibliotecas é simplificado no NetBeans.

Os desenvolvedores podem facilmente adicionar bibliotecas relacionadas a JPA e EJB aos seus projetos, garantindo a compatibilidade e a atualização adequada das dependências. Essa facilidade de uso contribui para uma experiência de desenvolvimento mais fluida e eficaz.

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

Servlets são componentes Java que permitem o processamento de requisições HTTP em servidores web. O NetBeans simplifica a criação, desenvolvimento e teste de Servlets com ferramentas de modelagem, depuração e suporte a ciclo de vida, tornando o desenvolvimento web mais prático. Os Servlets são executados no servidor e podem lidar com diversas tarefas, como receber dados de formulários, acessar bancos de dados, gerar conteúdo HTML dinâmico e muito mais.

O NetBeans oferece um conjunto robusto de ferramentas para a construção de Servlets em projetos web. Ele simplifica o processo de criação, desenvolvimento e implantação de Servlets, tornando o desenvolvimento web mais eficiente e produtivo. O suporte do NetBeans inclui a criação de Servlets por meio de assistentes e modelos predefinidos, integração com o ciclo de vida do projeto, realce de sintaxe e assistência à codificação, depuração e teste.

Todas essas funcionalidades combinadas proporcionam um ambiente de desenvolvimento integrado completo para a criação de Servlets, aumentando a eficiência dos desenvolvedores e facilitando o desenvolvimento de aplicativos web robustos e dinâmicos.

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

A comunicação entre Servlets e EJBs ocorre via injeção de dependência (anotação @EJB) ou pelo lookup via JNDI. Essas opções promovem a separação de responsabilidades e a reutilização de código, essenciais para aplicações corporativas robustas e escaláveis. A injeção de dependência é o método preferido, pois é mais simples e menos propenso a erros. No caso da injeção de dependência, o Servlet declara uma dependência no Session Bean usando a anotação @EJB.

O contêiner EJB gerencia a criação e a injeção do EJB no Servlet durante o ciclo de vida da aplicação. Isso permite que o Servlet acesse métodos e atributos do Session Bean de forma transparente, como se fosse um componente local. Por outro lado, o lookup é uma abordagem mais manual na qual o Servlet obtém uma referência para o Session Bean do pool de EJBs usando um contexto de inicialização JNDI (Java Naming and Directory Interface).

O Servlet precisa conhecer o nome JNDI do EJB para realizar o lookup. Uma vez obtida a referência, o Servlet pode chamar métodos no Session Bean normalmente. Independentemente do método utilizado, a comunicação entre Servlets e Session Beans do pool de EJBs permite que os Servlets deleguem tarefas de negócio complexas para os EJBs, promovendo a separação de preocupações e a reutilização de código. Isso facilita o desenvolvimento de aplicativos corporativos robustos e escaláveis no ambiente Java EE.

6. Como funciona o padrão Front Controller, e como ele é implementado em um aplicativo eb Java, na arquitetura MVC?

No padrão MVC, o Front Controller centraliza a recepção e roteamento das requisições HTTP, facilitando autenticação, autorização e tratamento de erros. Isso organiza o fluxo e melhora a manutenção e extensão do aplicativo. Ele atua como um intermediário entre o cliente e o resto da aplicação, sendo responsável por rotear as requisições para o componente apropriado, geralmente um controlador, e coordenar a execução do fluxo de trabalho. Na arquitetura MVC (Model-View-Controller) em um aplicativo web Java, o padrão Front Controller é implementado como o controlador frontal. O controlador frontal recebe todas as requisições HTTP do cliente e decide qual controlador deve ser acionado para processar a requisição. Ele também pode executar tarefas comuns, como autenticação, autorização e manipulação de erros.

Os controladores na arquitetura MVC são responsáveis por processar as requisições específicas de acordo com as regras de negócio da aplicação. Eles interagem com o modelo de dados (Model) para obter ou atualizar informações e com a camada de visualização (View) para renderizar a resposta adequada para o cliente. O Front Controller geralmente utiliza um mecanismo de mapeamento de URLs para associar uma requisição a um controlador específico. Isso pode ser feito por meio de configurações declarativas em um arquivo de configuração, como o web.xml, ou por meio de anotações diretamente no código fonte. Ao centralizar o controle de fluxo da aplicação em um único ponto, o padrão Front Controller facilita a manutenção, a extensão e o teste do aplicativo web.

Ele promove a separação de preocupações e a coesão ao agrupar a lógica relacionada ao processamento de requisições em um local específico, tornando o código mais organizado e fácil de entender.

7. Quais as diferenças e semelhanças entre Servlets e JSPs?

Servlets focam em lógica de negócios e processamento de requisições, enquanto JSPs combinam HTML com Java para criar interfaces dinâmicas. Em projetos grandes, Servlets ajudam a separar a lógica de negócios, e JSPs são ideais para o front-end dinâmico.

Tanto Servlets quanto JSPs (JavaServer Pages) são tecnologias fundamentais para o desenvolvimento de aplicativos web em Java, mas elas diferem em sua abordagem e finalidade. Ambos são utilizados para criar aplicativos web dinâmicos baseados na plataforma Java EE, são executados no servidor e respondem a solicitações HTTP dos clientes, e permitem a geração de conteúdo HTML dinâmico com base em dados do servidor. Servlets são classes Java que estendem a funcionalidade de servidores web para processar diretamente requisições HTTP. Eles são responsáveis por toda a lógica de negócios e processamento de requisições, geralmente manipulando a lógica de controle.

JSPs, por outro lado, são arquivos de texto que misturam código Java com marcação HTML para criar páginas dinâmicas. Eles são mais focados na apresentação do conteúdo e permitem a fácil incorporação de código Java dentro do HTML, facilitando a criação de páginas web dinâmicas. Em Servlets, a lógica de negócios e a apresentação normalmente são separadas, o que pode levar a uma melhor organização do código e manutenção em projetos maiores. Em

JSPs, a lógica de negócios e a apresentação tendem a estar mais misturadas, o que pode tornar o código mais difícil de manter e entender, especialmente em projetos grandes e complexos. Enquanto Servlets são mais adequados para processamento de formulários, acesso a banco de dados e outras tarefas de negócios complexas, JSPs são mais adequados para a criação de interfaces de usuário dinâmicas e interativas. Em termos de desempenho, Servlets tendem a ter um desempenho ligeiramente melhor do que JSPs, uma vez que os Servlets são pré-compilados em bytecode Java durante a compilação, enquanto JSPs precisam ser compilados em Servlets antes de serem executados.

Em resumo, enquanto Servlets e JSPs têm objetivos semelhantes de criar aplicativos web dinâmicos, suas abordagens e áreas de foco são diferentes. A escolha entre eles depende das necessidades específicas do projeto e das preferências do desenvolvedor.

8. 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?

O redirecionamento (sendRedirect) inicia nova solicitação, enquanto o forward encaminha internamente a resposta para outro recurso no servidor. A escolha entre eles depende do tipo de solicitação e se novos dados precisam ser enviados ao servidor.

Um redirecionamento simples, geralmente feito usando o método sendRedirect() da interface HttpServletResponse, envia uma resposta ao cliente com um código de status de redirecionamento (geralmente 302) e a URL para a qual o cliente deve redirecionar. Isso faz com que o cliente emita uma nova solicitação para a URL fornecida. O redirecionamento é transparente para o cliente, que percebe apenas a mudança na URL. Por outro lado, o método forward do RequestDispatcher (geralmente acessado usando getRequestDispatcher().forward()) encaminha a solicitação e a resposta internamente para outro recurso no servidor antes de enviar a resposta final ao cliente. Isso é feito sem que o cliente saiba, pois a URL permanece a mesma no navegador.

O recurso de destino pode ser um Servlet, uma JSP ou até mesmo um arquivo estático. Em termos de funcionalidade, o redirecionamento simples é útil quando é necessário direcionar o cliente para outra página, especialmente em casos onde a lógica de processamento ocorre em um servidor diferente ou quando uma nova solicitação precisa ser feita.

Por outro lado, o encaminhamento (forward) é mais adequado quando a resposta precisa ser processada adicionalmente no servidor antes de ser enviada ao cliente, como quando várias Servlets ou JSPs estão envolvidas em uma cadeia de processamento. Quanto aos parâmetros e atributos nos objetos HttpRequest, eles são usados para passar dados entre diferentes partes da aplicação dentro do mesmo ciclo de solicitação-resposta. Parâmetros são geralmente usados para passar informações de entrada do cliente para o servidor, como dados de formulário ou parâmetros de URL. Eles podem ser acessados usando os métodos getParameter() ou getParameterMap().

Por outro lado, os atributos são usados para armazenar dados que podem ser acessados e compartilhados entre diferentes componentes da aplicação durante o ciclo de vida da solicitação. Eles podem ser definidos usando o método setAttribute() e acessados usando getAttribute(). Esses atributos são úteis, por exemplo, para passar dados entre Servlets ou entre Servlets e JSPs durante o encaminhamento (forward).

9. Como o framework Bootstrap é utilizado?

O Bootstrap é usado para criar interfaces responsivas e esteticamente agradáveis, através de componentes pré-estilizados e um sistema de grid flexível, simplificando o desenvolvimento.

Amplamente utilizado no desenvolvimento web para criar interfaces de usuário responsivas e visualmente atraentes. Ele fornece um conjunto de ferramentas e componentes pré-estilizados, como botões, formulários, barras de navegação, cards e grids, que podem ser facilmente incorporados em páginas web. Para utilizar o Bootstrap, os desenvolvedores geralmente incluem os arquivos CSS e JavaScript do Bootstrap em suas páginas HTML. Isso pode ser feito baixando os arquivos diretamente do site oficial do Bootstrap ou usando um link CDN (Content Delivery Network) para acessar os arquivos hospedados em servidores externos. Uma vez que os arquivos do Bootstrap são incluídos, os desenvolvedores podem começar a usar as classes CSS e componentes JavaScript fornecidos pelo Bootstrap em seu código HTML.

Por exemplo, eles podem aplicar classes como "btn" para estilizar botões, "form-control" para estilizar inputs de formulários, ou "container" para criar layouts responsivos. Além disso, o Bootstrap é altamente personalizável, permitindo que os desenvolvedores ajustem facilmente o visual e o comportamento dos componentes de acordo com as necessidades específicas de seus projetos. Eles podem modificar o código CSS ou usar as variáveis SASS fornecidas pelo Bootstrap para personalizar cores, fontes, espaçamentos e outros aspectos visuais.

Em resumo, o Bootstrap é utilizado como uma ferramenta para acelerar o desenvolvimento de interfaces de usuário web, fornecendo um conjunto de componentes pré-estilizados e responsivos que podem ser facilmente integrados em projetos web para criar layouts consistentes e visualmente atraentes.

10. Por que o Bootstrap garante a independência estrutural do HTML?

O Bootstrap separa estilo e conteúdo HTML, permitindo interfaces atraentes e limpas, facilitando a manutenção e a criação de layouts flexíveis para diferentes dispositivos. Ele permite que os desenvolvedores criem interfaces de usuário atraentes e responsivas sem alterar a estrutura semântica do HTML subjacente. Isso é alcançado através do uso de classes CSS predefinidas que podem ser aplicadas aos elementos HTML para estilizá-los de acordo com as diretrizes de design do Bootstrap.

Ao usar o Bootstrap, os desenvolvedores podem manter o HTML limpo e focado no conteúdo, enquanto as classes CSS do Bootstrap são usadas para controlar a aparência e o layout dos elementos. Isso resulta em um código HTML mais legível, modular e fácil de manter, já que as preocupações de estilo e apresentação são separadas do conteúdo. Além disso, o Bootstrap oferece um sistema de grid flexível que permite que os desenvolvedores criem layouts responsivos sem precisar definir manualmente estilos CSS para diferentes tamanhos de tela. Isso significa que o mesmo código HTML pode ser usado para criar interfaces de usuário que se adaptam automaticamente a diferentes dispositivos e tamanhos de tela, garantindo uma experiência consistente para os usuários finais.

Em resumo, o Bootstrap promove a independência estrutural do HTML ao fornecer uma estrutura de estilos predefinida que pode ser aplicada aos elementos HTML sem afetar sua estrutura semântica. Isso permite que os desenvolvedores criem interfaces de usuário visualmente atraentes e responsivas, mantendo o HTML limpo e focado no conteúdo.

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

Bootstrap proporciona uma página responsiva com seu sistema de grid, que adapta layouts automaticamente a diferentes dispositivos, melhorando a experiência do usuário.

O Bootstrap foi projetado desde o início com o objetivo de criar páginas web responsivas, ou seja, que se adaptam automaticamente a diferentes dispositivos e tamanhos de tela, como desktops, tablets e smartphones. O Bootstrap alcança essa responsividade através de seu sistema de grid flexível. Esse sistema permite que os desenvolvedores criem layouts de página dividindo-a em linhas e colunas, e especificando como essas colunas devem se comportar em diferentes tamanhos de tela.

Por exemplo, é possível definir que determinadas colunas devem ocupar 50% da largura da tela em dispositivos grandes, mas apenas 100% da largura em dispositivos pequenos. Além do sistema de grid, o Bootstrap oferece classes e componentes responsivos que se ajustam automaticamente com base no tamanho da tela. Por exemplo, existem classes CSS que podem ser aplicadas para ocultar ou exibir determinados elementos dependendo do tamanho da tela, ou componentes como barras de navegação e carrosséis que são projetados para funcionar bem em dispositivos de todos os tamanhos.

Portanto, o Bootstrap facilita a criação de páginas web responsivas, fornecendo uma estrutura de estilos e componentes que se ajustam dinamicamente para proporcionar uma experiência consistente e agradável ao usuário, independentemente do dispositivo que esteja sendo utilizado