



## Vamos integrar sistemas

Ramon Barros Correa, 202208939635

**Inserir aqui o Campus:** Polo Sete lagoas (MG)

**Nível 4 – Vamos integrar sistemas, turma 2022.2, semestre 2024.1**

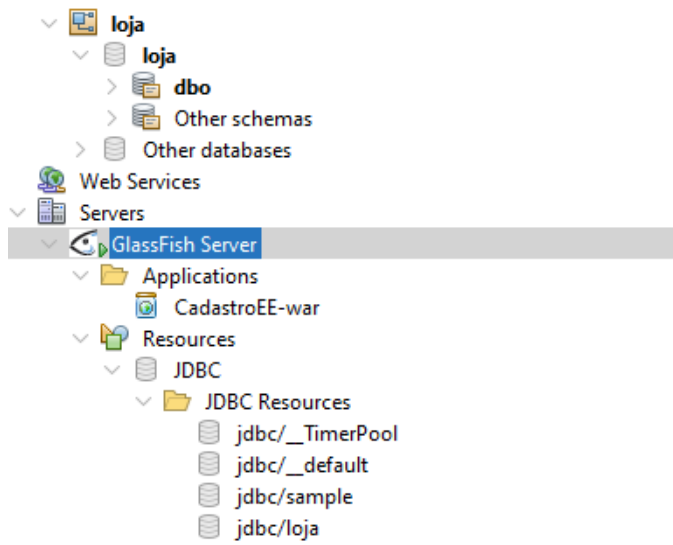
### Objetivo da Prática

O objetivo desta missão prática foi a criação de um banco de dados usando o Microsoft SQL Server, neste caso, foi usado inicialmente o SQL Server Express, mas posteriormente foi feita uma alteração para o SQL Server Developer, pois a versão anterior apresentava limitações para o seguimento de outras missões práticas















### 1º Procedimento | Camadas de Persistência e Controle

Inserir neste campo, **de forma organizada**, todos os códigos do roteiro do 1º Procedimento da Atividade Prática, os resultados da execução do código e a Análise e Conclusão:





















Configurar a conexão com SQL Server via NetBeans e o pool de conexões no GlassFish



## Criar o aplicativo corporativo no NetBeans

- ▼  CadastroEE-ejb
  - ▼  Source Packages
    - >  cadastroee.controler
    - >  cadastroee.model
  - >  Test Packages
  - ▼  Libraries
    - >  Java EE 8 API Library - javaee-api-8.0.jar
    - >  Jakarta EE Web 8 API Library - jakarta.jakartaee-web-api-8.0.0.jar
    - >  JDK 11
    - >  GlassFish Server
  - >  Test Libraries
  - >  Enterprise Beans
  - >  Configuration Files
  - >  Server Resources

## 2º e 3º procedimento

- ▼  CadastroEE-war
  - ▼  Web Pages
    - >  WEB-INF
      - >  Dados.html
      - >  index.html
  - >  Remote Files
  - ▼  Source Packages
    - ▼  cadastroee.servlets
      - >  ServletProduto.java
      - >  ServletProdutoFC.java
  - >  Test Packages
  - ▼  Libraries
    - >  CadastroEE-ejb - dist/CadastroEE-ejb.jar
    - >  Jakarta EE 10 API Library - jakarta.jakartaee-api-10.0.0.jar
    - >  Jakarta EE 8 API Library - jakarta.jakartaee-api-8.0.0.jar
    - >  Jakarta EE 9 API Library - jakarta.jakartaee-api-9.0.0.jar
    - >  JDK 11
    - >  GlassFish Server
  - >  Test Libraries
  - >  Configuration Files

Como é organizado um projeto corporativo no NetBeans?

Criação de um novo projeto; Estrutura de diretórios padrão; Gerenciamento de dependências; Construção e Implantação; Integração com sistemas de controle de versão, Ferramentas de depuração e teste; Documentação e colaboração

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

JPA (Java Persistence API):

A JPA é uma API do Java que define um conjunto de classes e métodos para mapear objetos Java para bancos de dados relacionais.

Com a JPA, os desenvolvedores podem escrever código Java para interagir com o banco de dados, em vez de escrever consultas SQL diretamente.

Ela simplifica o desenvolvimento de aplicativos ao oferecer uma abordagem baseada em objetos para trabalhar com dados persistentes.

No contexto de aplicativos Web, a JPA é frequentemente usada para mapear entidades de negócios para tabelas de banco de dados e fornecer operações de persistência, como criar, ler, atualizar e excluir (CRUD).

EJB (Enterprise JavaBeans):

EJB é um conjunto de especificações do Java EE (Enterprise Edition) para construir componentes de negócios em aplicativos corporativos.

Os EJBs são componentes de servidor que encapsulam a lógica de negócios e são executados em um ambiente de contêiner EJB.

Eles oferecem serviços como transações, segurança, pooling de conexões, concorrência, gerenciamento de ciclo de vida, etc.

Os EJBs podem ser de diferentes tipos, como Session Beans (stateless, stateful e singleton), Entity Beans (agora substituídos pela JPA para mapeamento de entidades) e Message-Driven Beans.

No contexto de aplicativos Web, os EJBs são usados para implementar a lógica de negócios que pode ser acessada por vários clientes, como aplicativos da Web, aplicativos de desktop, serviços da Web, etc.

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

Assistentes de Criação de Entidades; Suporte a Anotações; Ferramentas de Consulta JPQL; Integração com Banco de Dados; Gerenciamento de Persistência; Assistentes de Criação de EJBs; Suporte a Anotações e Interface; Testes e Debugging; Deploy em Servidores de Aplicação; Templates e Snippets de Código e etc.

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 estendem as capacidades de um servidor, permitindo que ele responda a solicitações. Eles são usados principalmente para criar aplicações web dinâmicas. Um servlet recebe uma solicitação HTTP, processa-a (geralmente interagindo com um banco de dados ou outros serviços), e envia uma resposta de volta ao cliente (tipicamente um navegador web).

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

A comunicação entre Servlets e Session Beans (EJBs) é uma prática comum em aplicações Java EE, onde Servlets atuam como controladores de requisições HTTP e os Session Beans encapsulam a lógica de negócios. O NetBeans facilita essa integração através de várias ferramentas e recursos.

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 dos padrões de design mais usados na arquitetura de aplicações web, especialmente em conjunto com o padrão MVC (Model-View-Controller). O objetivo principal do Front Controller é centralizar o controle das requisições para um único ponto de entrada, onde essas requisições podem ser tratadas de maneira consistente antes de serem encaminhadas para os componentes apropriados. O padrão Front Controller, quando implementado em uma aplicação web Java usando a arquitetura MVC, proporciona uma maneira estruturada e eficiente de gerenciar requisições e respostas. O NetBeans oferece suporte robusto para esse tipo de desenvolvimento, com ferramentas que facilitam a criação de servlets, JSPs e a integração de lógica de negócios, promovendo uma arquitetura limpa e bem organizada.

Quais as diferenças e semelhanças entre Servlets e JSPs?

Parte da Tecnologia Java EE:

Ambos fazem parte da especificação Java EE e são usados em servidores de aplicação compatíveis com Java EE.

Geram Código Java:

Tanto servlets quanto JSPs são convertidos em classes Java pelo contêiner servlet, e posteriormente compilados em bytecode.

No caso das JSPs, o contêiner gera um servlet correspondente para processar a página.

Manipulam Requisições HTTP:

Ambos podem manipular requisições e respostas HTTP, usando objetos `HttpServletRequest` e `HttpServletResponse`.

Suporte a Sessões:

Ambos podem trabalhar com sessões HTTP para manter o estado entre diferentes requisições do cliente.

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 (HTTP Redirect)

Como funciona: Utiliza o método `sendRedirect` do objeto `HttpServletResponse`.

Comportamento:

O navegador do cliente recebe uma resposta HTTP com um código de status de redirecionamento (geralmente 302) e uma nova URL.

O navegador então faz uma nova requisição HTTP para a nova URL.

URL visível: A URL no navegador é atualizada para a nova URL.

Novo ciclo de requisição/resposta: Uma nova requisição HTTP é feita pelo navegador.

Quando usar:

Quando você deseja redirecionar o cliente para um recurso externo ou uma URL diferente da aplicação atual.

Quando você deseja que a URL no navegador seja atualizada.

Quando você deseja evitar o reenvio de formulários.

Forward (Encaminhamento)

Como funciona: Utiliza o método forward do objeto RequestDispatcher.

Comportamento:

O encaminhamento é feito no lado do servidor, sem o conhecimento do cliente.

O servidor encaminha a mesma requisição e resposta para outro recurso (como uma servlet, JSP, etc.).

URL invisível: A URL no navegador não muda, pois a operação é transparente para o cliente.

Mesmo ciclo de requisição/resposta: Não há uma nova requisição HTTP; a mesma requisição é passada adiante.

Quando usar:

Quando você deseja encaminhar a requisição para outro recurso dentro do mesmo servidor.

Quando você deseja compartilhar dados entre recursos usando a mesma requisição e resposta.

Quando a URL não precisa ser alterada no navegador.

Como o framework Bootstrap é utilizado?

Bootstrap é um framework front-end poderoso e amplamente utilizado para o desenvolvimento de sites e aplicativos web responsivos e modernos. Ele oferece uma vasta coleção de componentes CSS e JavaScript prontos para uso, o que facilita a criação de interfaces de usuário consistentes e esteticamente agradáveis.

Por que o Bootstrap garante a independência estrutural do HTML?

O Bootstrap garante a independência estrutural do HTML ao fornecer uma estrutura robusta e flexível que permite aos desenvolvedores criar páginas web responsivas e esteticamente consistentes sem depender fortemente da estruturação manual e específica do HTML.

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

Bootstrap facilita a criação de páginas web responsivas por meio de seu sistema de grid flexível, breakpoints predefinidos, componentes adaptáveis e classes utilitárias. Esses recursos permitem que desenvolvedores criem layouts que se ajustam automaticamente a diferentes tamanhos de tela e dispositivos, garantindo uma experiência de usuário consistente e otimizada. A capacidade de Bootstrap de separar a estrutura e o estilo do HTML também contribui para um desenvolvimento mais rápido e eficiente, mantendo o código limpo e organizado.