

CAMPUS: Estácio Via Corpvs

CURSO: Desenvolvimento FullStack

DISCIPLINA: Vamos integrar sistemas

TURMA: 2025.1

SEMESTRE LETIVO: Primeiro Semestre (2025)

ALUNO: Francisco Ernaldo Vieira

MATRÍCULA: 202401191817

**Título da prática:**

Implementação de sistema cadastral com interface web, baseado nasd tecnologias de servlet, JPA e JEE.

**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.

**Procedimento 1:**

**Códigos usados neste roteiro:**

**Procedimento01Application.java**

package com.estacio.procedimento01;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class Procedimento01Application {  
  
 public static void main(String[] args) {  
 SpringApplication.run(Procedimento01Application.class, args);  
 }  
  
}

**Análise e Conclusão:**

**Como é organizado um projeto corporativo no NetBeans?**

Um projeto corporativo no NetBeans (Java EE ou Jakarta EE) é estruturado em **módulos separados**, facilitando o desenvolvimento e manutenção de aplicações empresariais. Os principais módulos são:

* **Projeto EAR (Enterprise Application)**: projeto principal que reúne os módulos Web e EJB. Gera um .ear para deploy.
* **Módulo EJB**: contém a lógica de negócio (serviços, entidades JPA, interfaces locais e remotas). Gera um .jar.
* **Módulo Web**: interface da aplicação (JSP, Servlets, JSF, REST). Gera um .war.
* **Cliente Aplicação (opcional)**: utilizado para aplicações desktop que consomem EJBs.

Geralmente usa-se **Maven** ou **Ant** para gerenciamento de build e dependências, e a aplicação é implantada em um **servidor de aplicação** como GlassFish ou WildFly.

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

**JPA (Java Persistence API)** e **EJB (Enterprise JavaBeans)** são tecnologias fundamentais da plataforma Java EE (atualmente Jakarta EE), utilizadas no desenvolvimento de aplicações corporativas para a web:

* **JPA**:
  + Responsável pela **persistência de dados** em bancos relacionais.
  + Utiliza anotações como @Entity, @Id e @Table para mapear classes Java para tabelas no banco de dados.
  + Substitui o uso direto de JDBC, simplificando o acesso e a manipulação de dados.
  + Trabalha com o EntityManager para realizar operações como persist, merge, remove e find.
* **EJB**:
  + Fornece **componentes de negócio** reutilizáveis, transacionais e seguros.
  + Os **Session Beans** (por exemplo, @Stateless) encapsulam a lógica de negócio e podem ser injetados em outras partes da aplicação.
  + Automatiza aspectos como controle de transações, segurança e concorrência.
  + Facilita a criação de aplicações distribuídas e escaláveis.

### Em conjunto:

* O **EJB** executa a lógica de negócio (como regras, validações e cálculos).
* O **JPA** gerencia a interação com o banco de dados.
* Ambos permitem a separação de responsabilidades e melhor organização do código na construção de **aplicações web robustas e manuteníveis**.

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

O NetBeans aumenta a produtividade ao trabalhar com JPA e EJB por meio de:

* **Geração automática de código**, como entidades JPA e Session Beans.
* **Suporte a servidores de aplicação**, facilitando o deploy e testes.
* **Autocompletar e validação de anotações**, reduzindo erros.
* **Ferramentas visuais** para navegação entre classes e mapeamentos.
* **Hot redeploy**, permitindo alterações rápidas sem reiniciar o servidor.

Esses recursos automatizam tarefas e aceleram o desenvolvimento de aplicações Java corporativas.

**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 processam requisições HTTP em aplicações web. Eles são usados para gerar respostas dinâmicas, como páginas HTML ou dados JSON.

O **NetBeans** oferece suporte ao desenvolvimento de servlets com:

* Criação automática de arquivos com métodos doGet() e doPost().
* Suporte a anotações como @WebServlet.
* Deploy integrado com servidores como Tomcat e GlassFish.
* Ferramentas de depuração e edição de web.xml.

Esse suporte agiliza o desenvolvimento e facilita a manutenção de aplicações web Java.

**Como é feita a comunicação entre os Servlets e os SessionBeans do pool de EJBs?**

A comunicação entre **Servlets** e **SessionBeans** (como @Stateless ou @Stateful) ocorre por meio da **injeção de dependência**, utilizando a anotação @EJB.

O **container de EJB** gerencia um **pool de SessionBeans**, e o servlet acessa esses componentes para executar regras de negócio.

O que o container faz:

* Injeta automaticamente a instância do EJB no servlet.
* Cuida do ciclo de vida, concorrência e transações.
* Usa um **lookup interno** (JNDI) para localizar o EJB correto.

**Procedimento 2:**

**Análise e Conclusão:**

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

O O **padrão Front Controller** centraliza todas as requisições em um único **Servlet**, que decide qual lógica executar e qual **View (JSP)** exibir.

Na arquitetura **MVC**:

* O Front Controller (Servlet) é o **Controlador principal**.
* Ele chama o **modelo** (serviços, EJBs, JPA) e encaminha para a **visão** (JSP).

Esse padrão **organiza melhor o fluxo**, facilita o controle de acesso e melhora a manutenção da aplicação web.

**Quais as diferenças e semelhanças entre Servlets e JSPs?**

**Semelhanças**: Ambos são usados para gerar conteúdo dinâmico em Java Web e rodam no servidor.

**Diferenças**:

* + **Servlet**: foca na **lógica** da aplicação (controle).
  + **JSP**: foca na **apresentação** (interface HTML).
  + **Servlet** usa só Java; **JSP** mistura HTML com Java.

Juntos, são usados no padrão **MVC**: Servlet como controlador e JSP como visão.

**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?**

**Diferença entre Redirecionamento e Forward**

* **Redirecionamento (response.sendRedirect)**:
  + Envia uma nova requisição para o navegador.
  + A URL no navegador muda.
  + O controle sai do servidor e volta para o cliente.
  + Pode redirecionar para outro domínio.
* **Forward (RequestDispatcher.forward)**:
  + A requisição continua no servidor.
  + A URL no navegador **não muda**.
  + Mais rápido e interno (usado entre páginas da mesma aplicação).

### ****Parâmetros vs. Atributos no HttpServletRequest****

* **Parâmetros (**request.getParameter**)**:
  + São valores enviados pelo cliente (formulários, query string).
  + Sempre do tipo **String**.
  + Ex: request.getParameter("nome").
* **Atributos (**request.setAttribute**)**:
  + São objetos criados no servidor e compartilhados entre servlets/JSPs.
  + Usados com **forward** para passar dados.
  + Ex: request.setAttribute("cliente", objCliente).

**Procedimento 3:**

**Análise e Conclusão:**

**Como o framework Bootstrap é utilizado?**

**Bootstrap** é um framework CSS e JavaScript usado para criar páginas web responsivas e com visual moderno, de forma rápida.

#### Como é usado:

* Importa-se o **CSS** e o **JavaScript** do Bootstrap via CDN ou arquivos locais.
* Usa-se **classes prontas** no HTML para aplicar estilos, como:
  + Layouts com container, row, col.
  + Botões com btn btn-primary.
  + Menus, formulários e alertas com classes específicas.

#### Vantagens:

* Design responsivo automático.
* Reduz o uso de CSS personalizado.
* Agiliza o desenvolvimento web.

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

O **Bootstrap** garante independência estrutural do HTML porque **separa o conteúdo da apresentação**. Isso é feito usando **classes CSS prontas**, que aplicam o estilo sem modificar a estrutura ou lógica do HTML.

#### Em outras palavras:

* O HTML define **o que** será exibido.
* O Bootstrap (via classes) define **como** será exibido.
* Isso permite mudar o visual sem alterar o conteúdo ou a estrutura base.

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

O **Bootstrap** oferece um sistema de **grid responsivo** que permite que páginas web se adaptem automaticamente a diferentes tamanhos de tela (celular, tablet, desktop).

#### Como isso acontece:

* Usa classes como col-sm, col-md, col-lg para definir o comportamento dos elementos em cada tipo de tela.
* Inclui **media queries** e componentes ajustáveis por padrão.

Resultado: com Bootstrap, a página fica **visualizável e funcional em qualquer dispositivo**, sem precisar reescrever o código.

**LINK DO GITHUB:**

https://github.com/ernaldo777/semestre3---missao-pratica-4