**Campus polo Rua Teresa - Desenvolvimento Full Stack**

**RPG0017 - Vamos integrar sistemas**
**Turma 9001 - 3o semestre**
**Aluno: Marcos Valerio R S de Carvalho**
**Github: https://github.com/Voidrrrr/Mundo-3-nivel-4**

# RPG 0017 - Vamos manter as informações?

**Objetivos da prática:**

- **Implementar persistência com base em JPA.**
- **Implementar regras do 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.**
- **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.**

## Análise e Conclusão:

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

Um projeto corporativo no NetBeans é organizado em módulos, geralmente divididos em:

- **Web**: Interface do usuário (JSP, HTML, CSS, JavaScript).
- **EJB**: Regras de negócio e transações.
- **Libraries**: Dependências externas.
- **Persistence**: Configuração de bancos de dados (JPA, entidades).

O NetBeans facilita a criação e integração dessas camadas, promovendo uma estrutura modular e reutilizável.

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

- **JPA:** Gerencia a persistência de dados, mapeando objetos Java para tabelas de banco de dados. Simplifica consultas e operações com dados.
- **EJB:** Implementa regras de negócio, transações e segurança de forma robusta e escalável, suportando aplicações distribuídas.

Ambos trabalham juntos para criar aplicativos web eficientes e organizados.

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

  O NetBeans melhora a produtividade com **JPA** e **EJB** através de geração automática de código, configuração simplificada, integração com servidores e ferramentas de debugging e deploy eficientes.

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

**Servlets**: Componentes Java que processam requisições HTTP e geram respostas dinâmicas.
**NetBeans**: Oferece suporte com assistentes para criação, edição, configuração automática do web.xml e integração com servidores para teste e deploy.

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

- **Comunicação**: Feita por injeção de dependência (`@EJB`) ou lookup via JNDI.
- **Objetivo**: O Servlet utiliza os Session Beans para acessar a lógica de negócio.

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

O padrão Front Controller centraliza o processamento de requisições em um único ponto de entrada, geralmente um Servlet. Ele recebe todas as requisições, decide qual lógica de controle executar e delega para o controlador adequado. No padrão MVC em Java:

1. **Front Controller (Servlet):** Recebe todas as requisições.
2. **Controller:** Processa a lógica da aplicação, interage com o Model e seleciona a View para resposta.
3. **View:** Exibe a interface para o usuário.

O Front Controller facilita a organização e manutenção do código, centralizando o fluxo de controle.

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

- **Servlets:** São classes Java que controlam o fluxo de requisições e geram respostas dinâmicas, geralmente atuando como controladores na arquitetura MVC.
- **JSPs:** Páginas HTML com código Java embutido, usadas para gerar conteúdo dinâmico e criar a interface do usuário, funcionando como a "view".

**Semelhanças:** Ambos são tecnologias Java para aplicações web, lidam com requisições HTTP e são usados em servidores Java EE.

**Diferença principal:** Servlets controlam a lógica da aplicação, enquanto JSPs geram a apresentação dinâmica.

- **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 (`response.sendRedirect`):** O servidor envia uma nova requisição para o cliente, mudando a URL no navegador e criando uma segunda requisição.
- **Forward (`RequestDispatcher.forward`):** O servidor encaminha a mesma requisição para outro recurso sem alterar a URL no navegador, mantendo a requisição original.

---

**Parâmetros e Atributos no `HttpRequest`:**

- **Parâmetros:** Dados enviados na URL ou no corpo da requisição (ex: `request.getParameter("nome")`).
- **Atributos:** Informações armazenadas no ciclo de vida da requisição (ex: `request.setAttribute("chave", valor)`).