



**Nome:** Wallace Felipe Tavares Moreira

**Matrícula:** 202109237331

**Universidade:** UNIVERSIDADE ESTÁCIO DE SÁ

**Curso:** Desenvolvimento Full Stack

**Campus:** Jardim América – Itaguaí/RJ

**Disciplina:** Nível 4 – Vamos integrar sistemas

**Semestre Letivo:** Terceiro Semestre

**Prática:** Implementação de sistema cadastral com interface Web, baseado nas tecnologias de Servlets, JPA e JEE.

### **Objetivo 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.
- 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.



## Códigos Solicitados:

### Classe AbstractFacade<T>

- **Descrição:**

- Esta é uma classe abstrata genérica que fornece métodos genéricos para operações de persistência em entidades de banco de dados. É projetada para ser estendida por classes específicas que representam entidades de domínio.

- **Atributos:**

- entityClass: Armazena a classe da entidade que está sendo gerenciada.

- **Métodos Principais:**

- create(T entity): Cria uma nova entidade no banco de dados.
- edit(T entity): Edita uma entidade existente no banco de dados.
- remove(T entity): Remove uma entidade existente do banco de dados.
- find(Object id): Localiza uma entidade no banco de dados com base em sua chave primária.
- findAll(): Retorna uma lista de todas as entidades do tipo gerenciado.
- findRange(int[] range): Retorna uma lista de entidades em um intervalo específico.
- count(): Retorna o número total de entidades do tipo gerenciado no banco de dados.

- **Método Abstrato:**

- protected abstract EntityManager getEntityManager(): Deve ser implementado pelas subclasses para fornecer um EntityManager necessário para interagir com o banco de dados.



### Classe ProdutoFacade

- **Descrição:** Esta classe específica estende AbstractFacade para gerenciar operações de persistência para a entidade Produto.
- **Anotações:**
  - @Stateless: Indica que esta classe é um bean sem estado do EJB.
  - @PersistenceContext: Define a unidade de persistência a ser usada.
- **Método Principal:**
  - getEntityManager(): Implementa o método abstrato da classe pai e fornece o EntityManager necessário.
- **Construtor:**
  - ProdutoFacade(): Inicializa a classe para gerenciar a entidade Produto.

### Interface ProdutoFacadeLocal

- **Descrição:** Uma interface que define métodos para operações de persistência relacionadas a produtos. Esses métodos são implementados pela classe ProdutoFacade.

### Classe Produto

- **Descrição:** Representa a entidade Produto do banco de dados.
- **Anotações:**
  - @Entity: Indica que esta classe é uma entidade JPA.
  - @Table: Define o nome da tabela no banco de dados.
  - @NamedQueries: Define consultas nomeadas para a entidade.



- **Campos:**
  - codProduto: Chave primária da entidade.
  - nome: Nome do produto.
  - quantidade: Quantidade em estoque.
  - precoVenda: Preço de venda do produto.

### **Páginas JSP**

- IncluirProduto.jsp: Uma página JSP para incluir um novo produto no banco de dados.
- EditarProduto.jsp: Uma página JSP para editar um produto existente no banco de dados.
- ListaProduto.jsp: Uma página JSP para listar todos os produtos existentes no banco de dados.

### **Classe ProdutoStrategy**

- **Descrição:** Uma classe que implementa uma estratégia para lidar com ações relacionadas a produtos, como listar, incluir, editar e excluir produtos.
- **Método Principal:** executar(String acao, HttpServletRequest request): Recebe uma ação e executa a lógica correspondente com base nessa ação.

### **Classe Strategy<K>**

- **Descrição:** Uma classe abstrata que define uma estrutura genérica para estratégias. A classe ProdutoStrategy é uma implementação concreta dessa estrutura.

### **Classe CadastroFC**

- **Descrição:** Uma servlet que gerencia as ações recebidas por meio de parâmetros e delega a execução de ações específicas para as estratégias apropriadas com base na ação recebida.



## **Análise e Conclusão:**

### **1. Como é organizado um projeto corporativo no NetBeans?**

- Projetos corporativos no NetBeans são divididos em módulos para separar a camada de apresentação, lógica de negócios e acesso a dados.

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

- JPA mapeia objetos Java para bancos de dados, enquanto EJB simplifica o desenvolvimento de componentes empresariais, como session beans.

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

- O NetBeans aprimora a produtividade por meio de assistentes e geração de código automática, incluindo a criação de classes de entidade e session beans.

### **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 processam solicitações HTTP em aplicativos web. O NetBeans facilita sua criação e mapeamento.

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



- O NetBeans permite a injeção de session beans em servlets, simplificando a comunicação para chamar métodos de lógica de negócios.

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

- O Front Controller é um padrão de design que centraliza o processamento de solicitações em um único ponto. Em aplicativos web Java com arquitetura MVC (Model-View-Controller), o Front Controller é geralmente implementado por meio de um servlet que direciona solicitações para controladores adequados.

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

- Servlets e JSPs são ambos componentes Java usados em aplicativos web. A principal diferença é que os servlets são classes Java que respondem a solicitações HTTP diretamente, enquanto as JSPs são páginas HTML com código Java embutido. Ambos podem ser usados para criar a lógica de apresentação.

**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?**

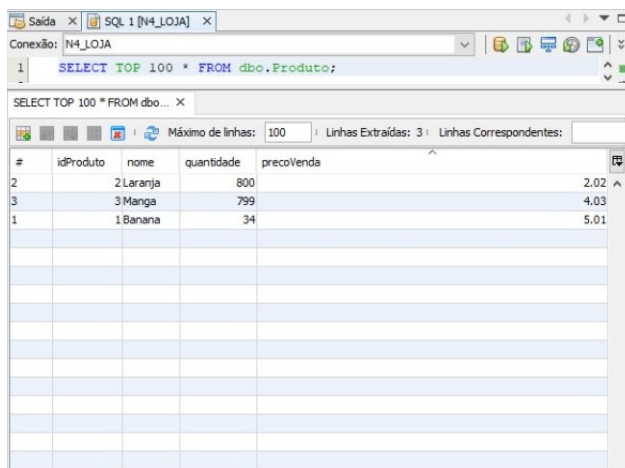
- Um redirecionamento simples leva o navegador a uma nova URL, enquanto o método forward permite que o servidor direcione a solicitação para outra parte do mesmo aplicativo. Parâmetros e atributos em objetos HttpRequest são usados para transmitir dados entre componentes e partes diferentes do aplicativo web. Parâmetros são geralmente visíveis na URL,



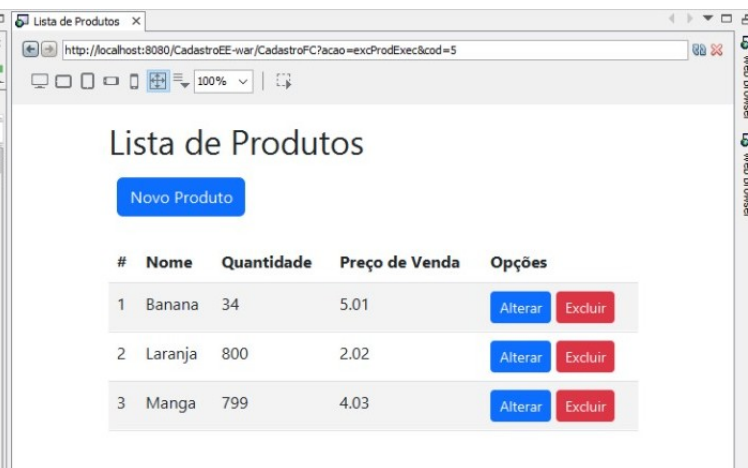
enquanto atributos são usados para armazenar informações temporárias acessíveis somente no contexto da solicitação.

Execução:

### Lista de Produtos



#	idProduto	nome	quantidade	precoVenda
2		2 Laranja	800	2.02
3		3 Manga	799	4.03
1		1 Banana	34	5.01



#	Nome	Quantidade	Preço de Venda	Opções
1	Banana	34	5.01	<button>Alterar</button> <button>Excluir</button>
2	Laranja	800	2.02	<button>Alterar</button> <button>Excluir</button>
3	Manga	799	4.03	<button>Alterar</button> <button>Excluir</button>



## Inserindo um Produto Tomate

SQL 1 [N4\_LOJA] x

Conexão: N4\_LOJA

```
SELECT TOP 100 * FROM dbo.Produto;
```

SELECT TOP 100 \* FROM dbo... x

Máximo de linhas: 100 | Linhas Extraídas: 3 | Linhas Correspondentes:

#	idProduto	nome	quantidade	precoVenda
2	2	Laranja	800	2.02
3	3	Manga	799	4.03
1	1	Banana	34	5.01

Incluir Produto x

http://localhost:8080/CadastroEE-war/CadastroFC?acao=incProd

Nome do Produto:

Quantidade:

Preço Venda:

SQL 1 [N4\_LOJA] x

Conexão: N4\_LOJA

```
SELECT TOP 100 * FROM dbo.Produto;
```

SELECT TOP 100 \* FROM dbo... x

Máximo de linhas: 100 | Linhas Extraídas: 4 | Linhas Correspondentes:

#	idProduto	nome	quantidade	precoVenda
2	2	Laranja	800	2.02
3	3	Manga	799	4.03
1	1	Banana	34	5.01
4	11	Tomate	20	30.0

Lista de Produtos x

http://localhost:8080/CadastroEE-war/CadastroFC

Lista de Produtos

#	Nome	Quantidade	Preço de Venda	Opções
1	Banana	34	5.01	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
2	Laranja	800	2.02	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
3	Manga	799	4.03	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
11	Tomate	20	30.0	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>





## Alterando produto Tomate para Abacaxi

The screenshot shows two windows. On the left, SQL Server Enterprise Manager displays a query result for 'SELECT TOP 100 \* FROM dbo.Produto;'. The results table is as follows:

#	idProduto	nome	quantidade	precoVenda
2	2	Laranja	800	2.02
3	3	Manga	799	4.03
1	1	Banana	34	5.01
4	11	Tomate	20	30.0

On the right, the 'Editar Produto' web form is shown. The 'Nome do Produto' field contains 'Abacaxi'. The 'Quantidade' field contains '20' and the 'Preço de Venda' field contains '30.0'. There are 'Salvar' and 'Cancelar' buttons at the bottom.

The screenshot shows two windows. On the left, SQL Server Enterprise Manager displays the same query result as above, but the product name in the last row has been updated from 'Tomate' to 'Abacaxi'.

On the right, the 'Lista de Produtos' web page is shown. It features a 'Novo Produto' button and a table listing the products:

#	Nome	Quantidade	Preço de Venda	Opções
1	Banana	34	5.01	<a href="#">Alterar</a> <a href="#">Excluir</a>
2	Laranja	800	2.02	<a href="#">Alterar</a> <a href="#">Excluir</a>
3	Manga	799	4.03	<a href="#">Alterar</a> <a href="#">Excluir</a>
11	Abacaxi	20	30.0	<a href="#">Alterar</a> <a href="#">Excluir</a>

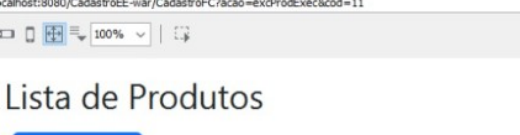


The screenshot shows the Microsoft Access interface. At the top, there are tabs for 'Saída', 'SQL 1 [N4\_LOJA]', and 'Objetos'. The 'SQL 1 [N4\_LOJA]' tab is active, displaying a SQL query in the design grid:

```
SELECT TOP 100 * FROM dbo.Produto;
```

Below the query, the status bar indicates 'Máximo de linhas: 100', 'Linhas Extraídas: 3', and 'Linhas Correspondentes: 100'. The results grid shows the following data:

#	idProduto	nome	quantidade	precoVenda
2	2	Laranja	800	2.02
3	3	Manga	799	4.03
1	1	Banana	34	5.01



Lista de Produtos

[Novo Produto](#)

#	Nome	Quantidade	Preço de Venda	Opções
1	Banana	34	5.01	<a href="#">Alterar</a> <a href="#">Excluir</a>
2	Laranja	800	2.02	<a href="#">Alterar</a> <a href="#">Excluir</a>
3	Manga	799	4.03	<a href="#">Alterar</a> <a href="#">Excluir</a>