



Estácio

CAMPUS POLO AUSTIN - NOVA IGUAÇU – RJ

DESENVOLVIMENTO FULL STACK

Nível 4: Vamos Integrar Sistemas

RPG0017 9001

2024/2

SALOMÃO ISAAC CARVALHO GARCIA

**Implementação de Arquitetura MVC com Front Controller em
Aplicações Web Java**

Nova Iguaçu

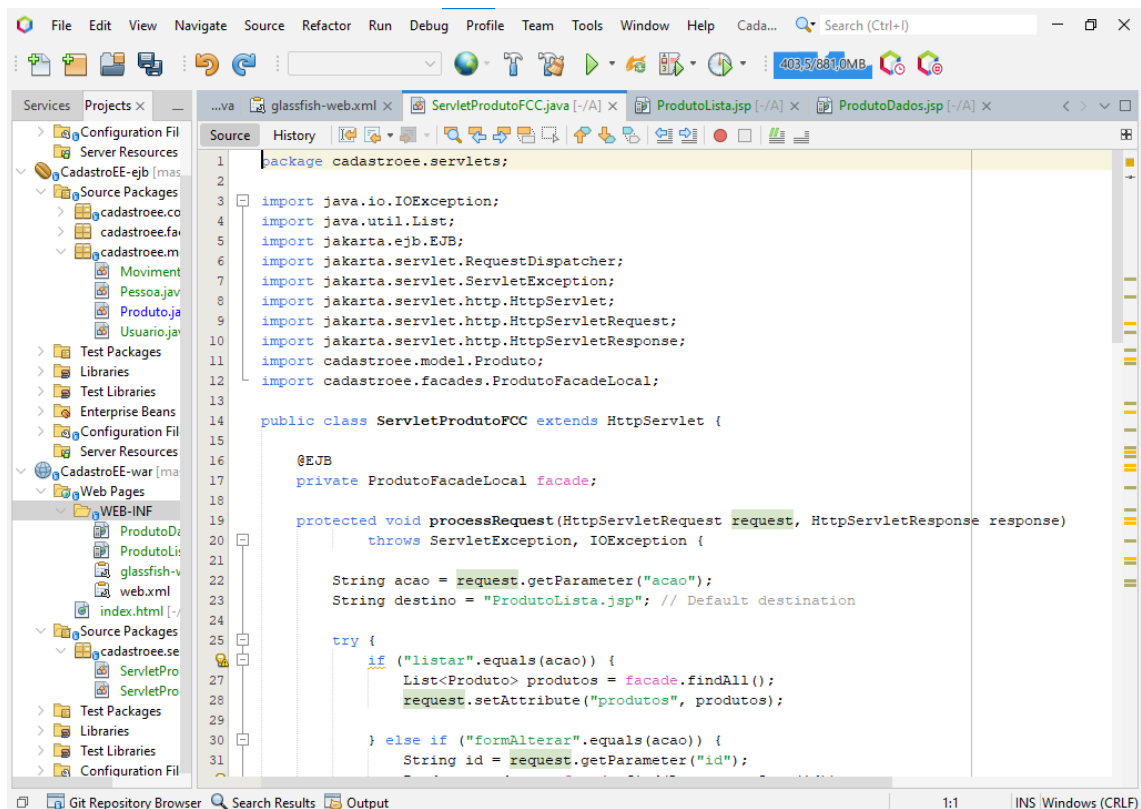
01/09/2024

Implementação de Arquitetura MVC com Front Controller em Aplicações Web Java

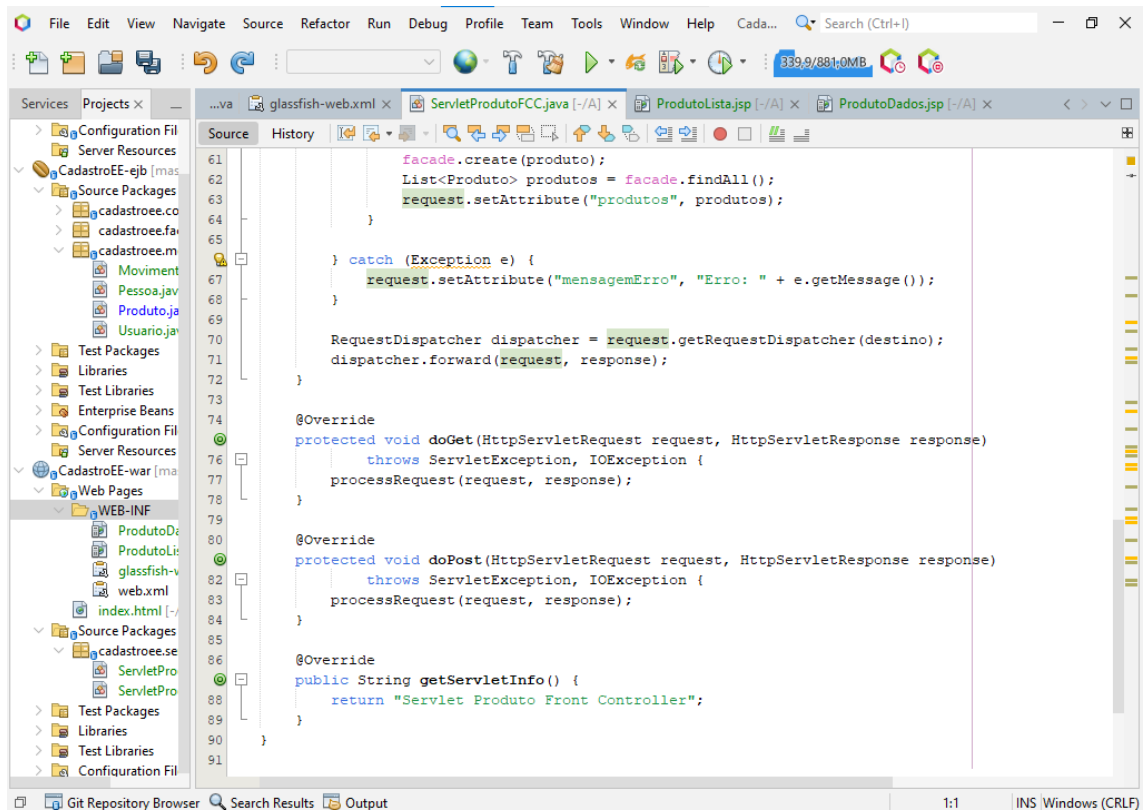
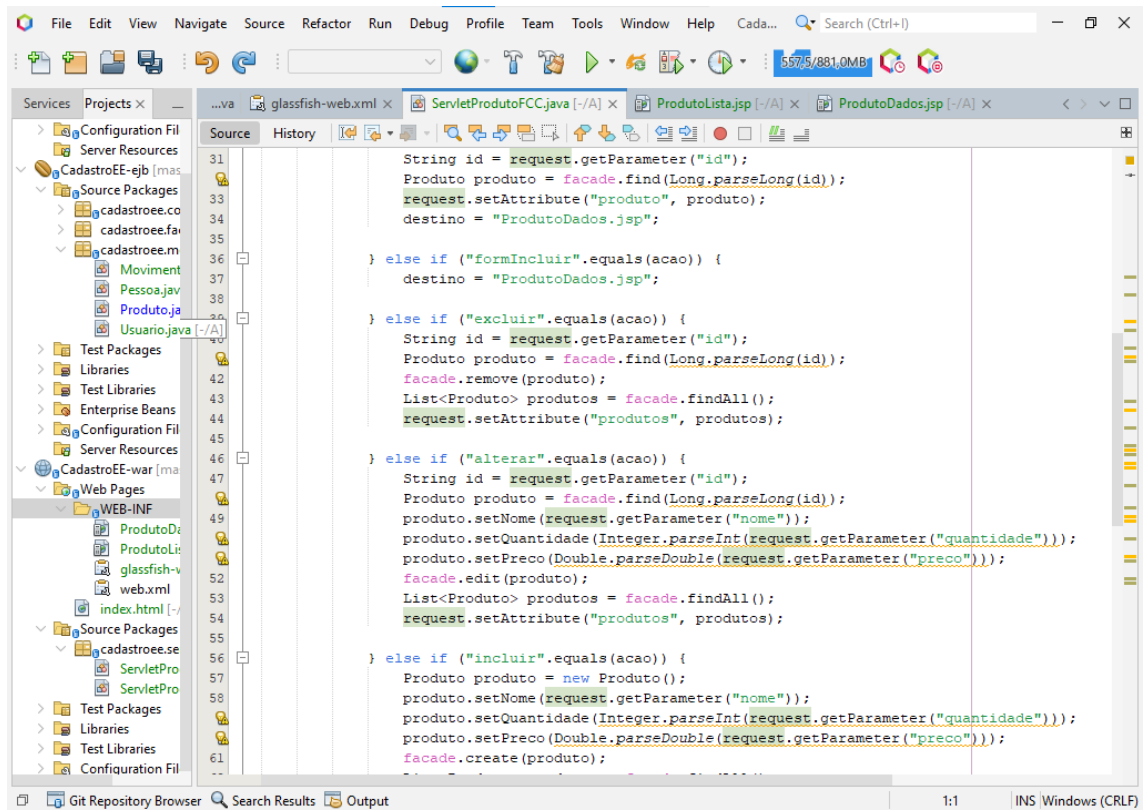
Objetivo da Prática: O objetivo desta prática é implementar um aplicativo web utilizando a arquitetura MVC (Model-View-Controller) com o padrão Front Controller. O foco é demonstrar a criação e gerenciamento de uma aplicação web Java com Servlets e JSPs, destacando a importância do padrão Front Controller e a utilização de técnicas como redirecionamento e forward.

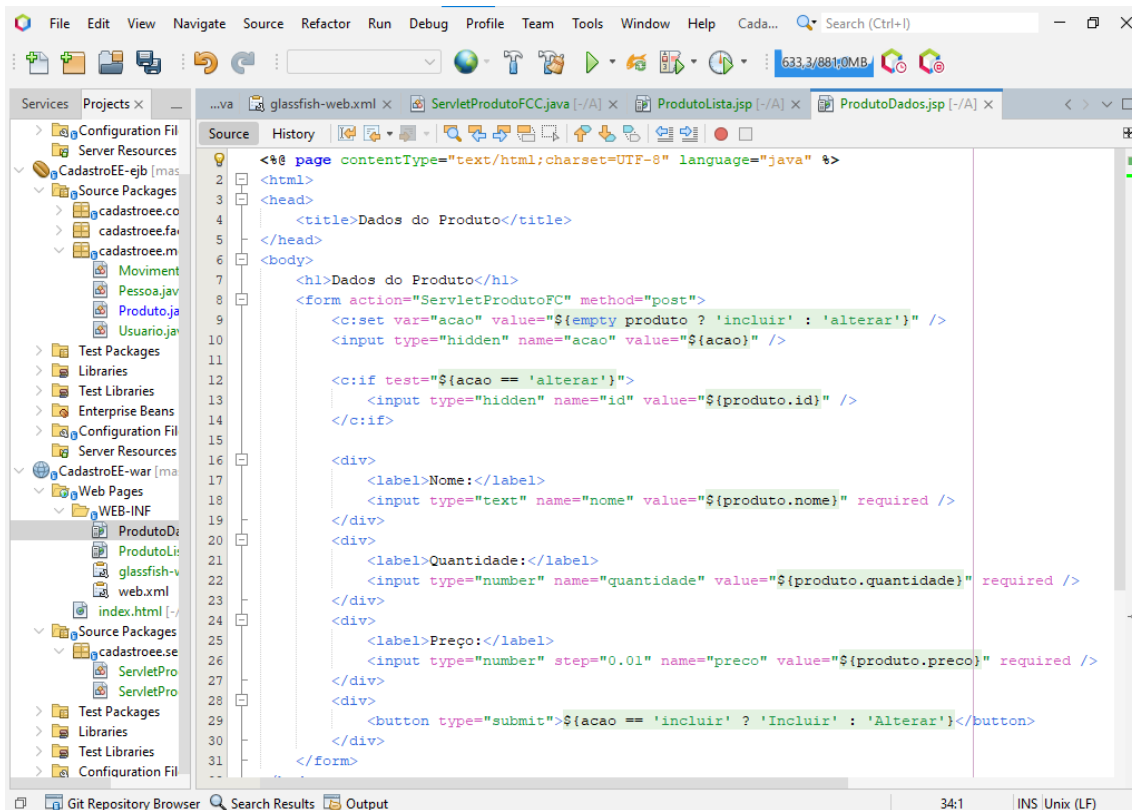
<https://github.com/SaloGarcia/Missao-Pratica-N4.git>

CÓDIGOS



```
1 package cadastroee.servlets;
2
3 import java.io.IOException;
4 import java.util.List;
5 import jakarta.ejb.EJB;
6 import jakarta.servlet.RequestDispatcher;
7 import jakarta.servlet.ServletException;
8 import jakarta.servlet.http.HttpServlet;
9 import jakarta.servlet.http.HttpServletRequest;
10 import jakarta.servlet.http.HttpServletResponse;
11 import cadastroee.model.Produto;
12 import cadastroee.facades.ProdutoFacadeLocal;
13
14 public class ServletProdutoFCC extends HttpServlet {
15
16     @EJB
17     private ProdutoFacadeLocal facade;
18
19     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
20         throws ServletException, IOException {
21
22         String acao = request.getParameter("acao");
23         String destino = "ProdutoLista.jsp"; // Default destination
24
25         try {
26             if ("listar".equals(acao)) {
27                 List<Produto> produtos = facade.findAll();
28                 request.setAttribute("produtos", produtos);
29
30             } else if ("formAlterar".equals(acao)) {
31                 String id = request.getParameter("id");
```

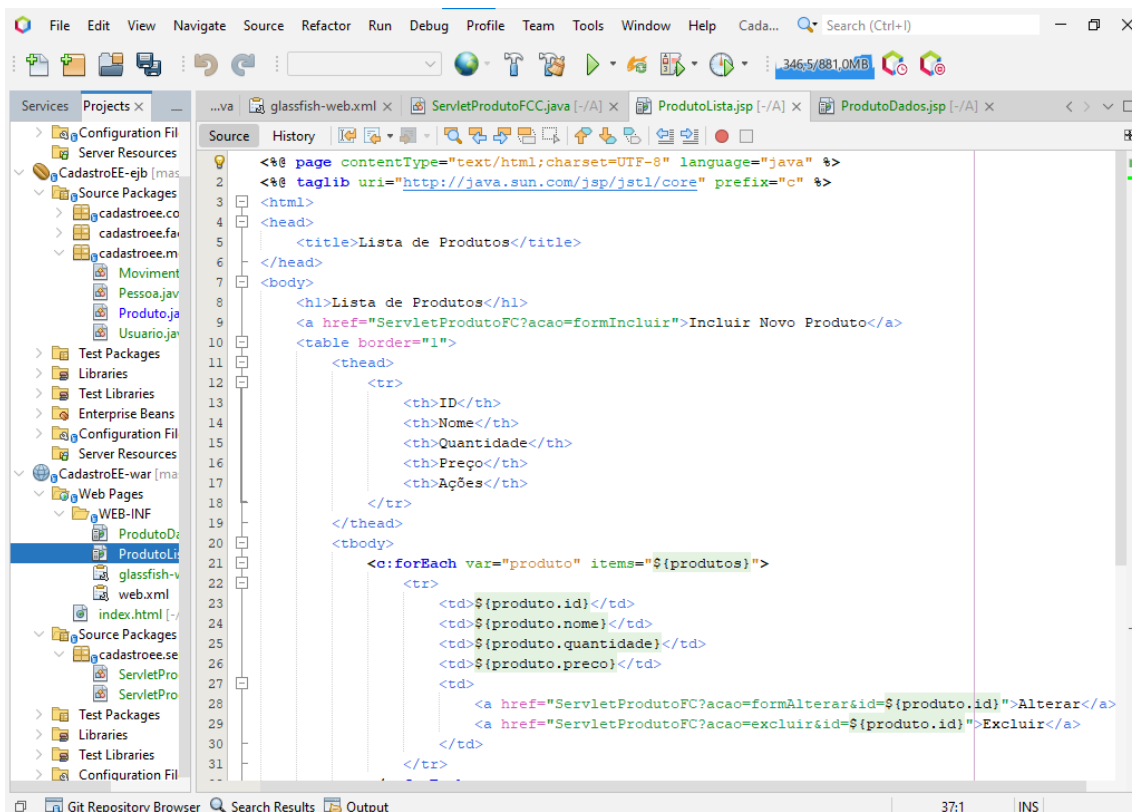




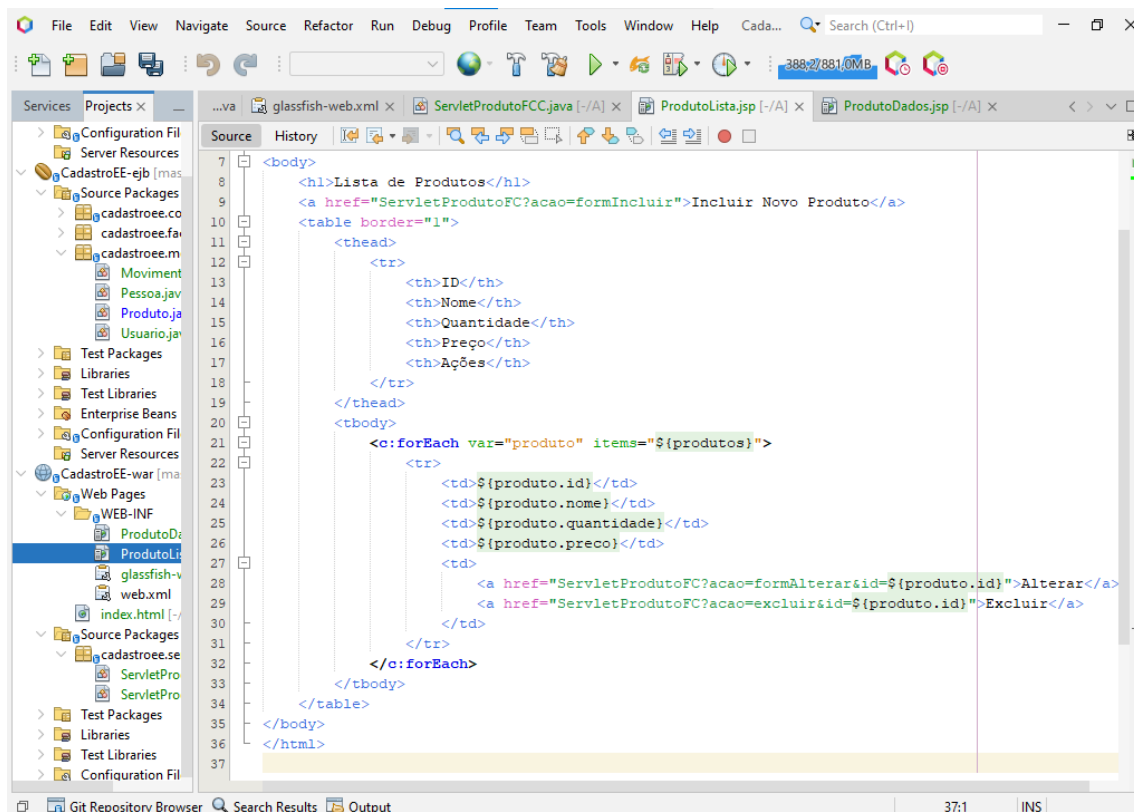
```
<%@ page contentType="text/html; charset=UTF-8" language="java" %>
<html>
<head>
<title>Dados do Produto</title>
</head>
<body>
<h1>Dados do Produto</h1>
<form action="ServletProdutoFC" method="post">
<c:set var="acao" value="${empty produto ? 'incluir' : 'alterar'}" />
<input type="hidden" name="acao" value="${acao}" />

<c:if test="${acao == 'alterar'}">
<input type="hidden" name="id" value="${produto.id}" />
</c:if>

<div>
<label>Nome:</label>
<input type="text" name="nome" value="${produto.nome}" required />
</div>
<div>
<label>Quantidade:</label>
<input type="number" name="quantidade" value="${produto.quantidade}" required />
</div>
<div>
<label>Preço:</label>
<input type="number" step="0.01" name="preco" value="${produto.preco}" required />
</div>
<div>
<button type="submit">${acao == 'incluir' ? 'Incluir' : 'Alterar'}</button>
</div>
</form>
```



```
<%@ page contentType="text/html; charset=UTF-8" language="java" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
<head>
<title>Lista de Produtos</title>
</head>
<body>
<h1>Lista de Produtos</h1>
<a href="ServletProdutoFC?acao=formIncluir">Incluir Novo Produto</a>
<table border="1">
<thead>
<tr>
<th>ID</th>
<th>Nome</th>
<th>Quantidade</th>
<th>Preço</th>
<th>Ações</th>
</tr>
</thead>
<tbody>
<c:forEach var="produto" items="${produtos}">
<tr>
<td>${produto.id}</td>
<td>${produto.nome}</td>
<td>${produto.quantidade}</td>
<td>${produto.preco}</td>
<td>
<a href="ServletProdutoFC?acao=formAlterar&id=${produto.id}">Alterar</a>
<a href="ServletProdutoFC?acao=excluir&id=${produto.id}">Excluir</a>
</td>
</tr>
</c:forEach>
</tbody>
</table>
```



Análise e Conclusão

1. 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 padrão de design que centraliza o processamento de todas as requisições em um único ponto de entrada, geralmente um servlet. Em um aplicativo Web Java com arquitetura MVC, o Front Controller Servlet atua como o controlador principal, direcionando as requisições para os diferentes controladores ou páginas JSP com base na URL ou parâmetros fornecidos. Isso permite uma melhor organização do código e facilita o gerenciamento da navegação.

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

- **Servlets:** São classes Java que processam requisições e geram respostas dinâmicas. Eles manipulam diretamente os dados da requisição e a lógica de negócios, geralmente gerando HTML para a resposta.

- **JSPs (JavaServer Pages):** São arquivos de texto que combinam HTML e código Java. Eles são compilados em servlets pelo servidor e são mais adequados para criar a interface de usuário. JSPs permitem uma separação mais clara entre a lógica de apresentação e a lógica de negócios.
- **Semelhanças:** Ambos são usados para gerar conteúdo dinâmico em uma aplicação web e podem interagir com Servlets para fornecer funcionalidade completa.

3. **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 (response.sendRedirect):** Faz uma nova requisição HTTP para o cliente, que então solicita a nova URL. Isso é útil quando você precisa que o cliente veja a nova URL e pode ser usado para enviar dados para outra página.
- **Forward (RequestDispatcher.forward):** Faz uma transferência interna de controle para outro recurso no servidor, sem que o cliente saiba. Isso é útil para delegar o processamento de uma requisição para outro servlet ou JSP dentro da mesma aplicação.
- **Parâmetros:** São dados enviados na URL ou no corpo da requisição e podem ser acessados via `HttpServletRequest.getParameter()`.
- **Atributos:** São usados para armazenar dados temporários entre diferentes componentes do servidor, como entre um servlet e um JSP, e são acessados via `HttpServletRequest.getAttribute()` e `setAttribute()`.