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

Aluno: Caio Viana Castelo Branco

Matrícula: 202307465925

Disciplina: RPG0017 - Vamos Integrar Sistemas

Curso: Desenvolvimento Full Stack

Turma: 9001

Semestre Letivo: 2025.1

Campus: Parangaba

1. 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.
- 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.

2. Códigos utilizados na prática

1º Procedimento | Camadas de Persistência e Controle

- ServletProduto.java

```
/*

* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

* Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template

*/

package cadastroee.servlets;

import cadastroee.controller.ProdutoFacadeLocal;
import cadastroee.model.Produto;
import jakarta.ejb.EJB;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException; import java.io.PrintWriter;

/**

* @author caio

*/
```

@WebServlet("/ServletProduto") public class ServletProduto extends HttpServlet {

```
@EJB
ProdutoFacadeLocal facade;

/**
 * Processes requests for both HTTP GET and POST methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");

    try (PrintWriter out = response.getWriter()) {
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Lista de Produtos</title>");
        out.println("</head>");
        out.println("<body>");

        for (Produto p : facade.findAll()) {
            out.println("<p>" + p.getNome() + "</p>");
        }

        out.println("</body>");
        out.println("</html>");
    }
}

/**
 * Handles the HTTP GET method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}
```

```
/**
 * Handles the HTTP POST method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}


}
```

2º Procedimento | Interface Cadastral com Servlet e JSPs

- ServletProduto.java

```
/*

* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/licensedefault.txt to
change this license

* Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to
edit this template

*/

package cadastroee.servlets;
```

```
import cadastroee.controller.ProdutoFacadeLocal;
import cadastroee.model.Produto;

import jakarta.ejb.EJB;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

/** *

* @author caio

*/

@WebServlet("/ServletProduto") public class ServletProduto extends HttpServlet {

@EJB
ProdutoFacadeLocal facade;

/**
 * Processes requests for both HTTP <code>GET</code> and <code>POST</code>
methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {

    response.setContentType("text/html;charset=UTF-8");

    try (PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet ServletProduto</title>");
        out.println("</head>");
        out.println("<body>");
        // out.println("<h1>Servlet ServletProduto at " +
request.getContextPath() + "</h1>");
        // out.println(facade.findAll().getClass());
        // out.println(facade.find(1).getClass());

        for (Produto p : facade.findAll()) {
            out.println("<li>" + p.getNome() + "</li>");
        }

        out.println("</body>");
        out.println("</html>");
```

```
    }
}

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the
+ sign on the left to edit the code.">
/**
 * Handles the HTTP <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}
// </editor-fold>


}
```

- ServletProdutoFC.java

```
/*

* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/licensedefault.txt to change
```

this license

* Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this template

*/

package cadastroee.servlets;

import cadastroee.controller.ProdutoFacadeLocal; import cadastroee.model.Produto;

import jakarta.ejb.EJB; import jakarta.servlet.RequestDispatcher; import jakarta.servlet.ServletException; import jakarta.servlet.annotation.WebServlet; import jakarta.servlet.http.HttpServlet; import jakarta.servlet.http.HttpServletRequest; import jakarta.servlet.http.HttpServletResponse;

import java.io.IOException;

/** *

* @author caio

*/

@WebServlet(name = "ServletProdutoFC", urlPatterns = {"/ServletProdutoFC"}) public class ServletProdutoFC extends HttpServlet {

```
@EJB
ProdutoFacadeLocal facade;

int idAtual = 6;

public int aleatorio() {
    // Math.random() gera um número aleatório entre 0.0 e 0.999
    // Assim, Math.random()*5 estará entre 0.0 e 4.999
double doubleRandomNumber = Math.random() * 100;
    int randomNumber = (int) doubleRandomNumber;
    return randomNumber;
}

/**
 * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
protected void processRequest(HttpServletRequest request,
```

```
HttpServletResponse response)
        throws ServletException, IOException {

    String acao = request.getParameter("acao");
    String destino = "";

    if (acao != null) {
        switch (acao) {
            case "listar":
                request.setAttribute("lista", facade.findAll());
                destino = "ProdutoLista.jsp";
                break;

            case "excluir":
                int idProduto =
Integer.valueOf(request.getParameter("idproduto"));
                facade.remove(facade.find(idProduto));
                request.setAttribute("lista", facade.findAll());
                RequestDispatcher rdExcluir =
request.getRequestDispatcher("ProdutoLista.jsp");
                rdExcluir.forward(request, response);
                return;

            case "formIncluir":
                destino = "ProdutoDados.jsp";
                break;

            case "formAlterar":
                int id_produto =
Integer.valueOf(request.getParameter("idproduto"));
                request.setAttribute("lista",
facade.find(id_produto));
                destino = "ProdutoDados.jsp";
                break;
        }

        RequestDispatcher rd =
request.getRequestDispatcher(destino);
        rd.forward(request, response);
    } else {
        request.setAttribute("lista", facade.findAll());
        RequestDispatcher rd =
request.getRequestDispatcher("ProdutoLista.jsp");
        rd.forward(request, response);
    }
}

// <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
```

```
/**
 * Handles the HTTP <code>GET</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

/**
 * Handles the HTTP <code>POST</code> method.
 *
 * @param request servlet request
 * @param response servlet response
 * @throws ServletException if a servlet-specific error occurs
 * @throws IOException if an I/O error occurs
 */
@Override
protected void doPost(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {

    String acao = request.getParameter("acao");

    switch (acao) {
        case "alterar":
            int produtoID =
Integer.valueOf(request.getParameter("idproduto"));
            String nome = request.getParameter("nome");
            int quantidadeProduto =
Integer.valueOf(request.getParameter("quantidade"));
            float precoProduto =
Float.valueOf(request.getParameter("preco"));

            Produto produtoAlterar = facade.find(produtoID);
            produtoAlterar.setNome(nome);
            produtoAlterar.setQuantidade(quantidadeProduto);
            produtoAlterar.setPrecoVenda(precoProduto);

            facade.edit(produtoAlterar);
            request.setAttribute("lista", facade.findAll());

            RequestDispatcher rdAlterar =
```

```
request.getRequestDispatcher("ProdutoLista.jsp");
            rdAlterar.forward(request, response);
            break;

        case "incluir":
            int idNext = aleatorio();

            if (idNext != idAtual) {
                float preco =
Float.valueOf(request.getParameter("preco"));
                String nome2 = request.getParameter("nome");
                int quantidade =
Integer.valueOf(request.getParameter("quantidade"));

                Produto produto = new Produto(idNext, nome2,
quantidade, preco);
                facade.create(produto);

                request.setAttribute("lista", facade.findAll());
                idAtual = idNext;

                RequestDispatcher rdIncluir =
request.getRequestDispatcher("ProdutoLista.jsp");
                rdIncluir.forward(request, response);
                break;
            } else {
                idNext++;
                idAtual = idNext;

                float preco =
Float.valueOf(request.getParameter("preco"));
                String nome2 = request.getParameter("nome");
                int quantidade =
Integer.valueOf(request.getParameter("quantidade"));

                Produto produto = new Produto(idNext, nome2,
quantidade, preco);
                facade.create(produto);

                request.setAttribute("lista", facade.findAll());
                idAtual = idNext;

                RequestDispatcher rdIncluir =
request.getRequestDispatcher("ProdutoLista.jsp");
                rdIncluir.forward(request, response);
                break;
            }
    }
}
```

```
/**
 * Returns a short description of the servlet.
 *
 * @return a String containing servlet description
 */
@Override
public String getServletInfo() {
    return "Short description";
}
// </editor-fold>


}
```

- ProdutoLista.jsp

```
<%--
 Document : ProdutoLista
 Created on : 1 de jun. de 2025, 21:16:09
 Author : caio
--%>
<%@page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@page import="cadastroee.model.Produto"%>
<%@page import="java.util.ArrayList"%>
<%@page import="java.util.List"%>
<%@page import="cadastroee.controller.ProdutoFacadeLocal"%>
<!DOCTYPE html>
<html>
 <head>
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
 <title>JSP Page</title>
 </head>
```

```
<body>
<div>
<h1>Listagem de Produtos</h1>
<h3>
<a href="ServletProdutoFC?acao=formIncluir"> Novo Produto
</a>
<!-- <a href="ServletProdutoFC"> Atualizar Dados </a> !-->
</h3>

<table border="1" width="100%">
<tr>
<td> ID </td>
<td> Nome </td>
<td> Quantidade </td>
<td> Preco de Venda </td>
<td> Opções </td>
</tr>

<%
try{
List<Produto> lista = (List)
request.getAttribute("lista");
for(Produto p: lista){
%>
<tr>
<td>
<%=p.getIdproduto()%>
</td>
```

```
<td>
 <%=p.getNome()%>
 </td>
<td>
 <%=p.getQuantidade()%>
 </td>
<td>
 <%=p.getPrecoVenda()%>
 </td>
<td>
 <a
href="ServletProdutoFC?acao=formAlterar&idproduto=<%=p.getIdproduto()%>">
Alterar </a>
 <a
href="ServletProdutoFC?acao=excluir&idproduto=<%=p.getIdproduto()%>"> Excluir
</a>
 </td>
 </tr>
 <% }
 } catch(NullPointerException nexc){
 out.print("<h1>"+nexc.getMessage()+"</h1>");

 }
 %>

 </table>
 </div>
 </body>
```

```
</html>
```

- ProdutoDados.jsp

```
<%--
 Document : ProdutoDados
 Created on : 2 de jun. de 2025, 14:26:07
 Author : caio
--%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="cadastroee.model.Produto"%>
<%@page import="java.util.ArrayList"%>
<%@page import="java.util.List"%>
<%@page import="cadastroee.controller.ProdutoFacadeLocal"%>
<!DOCTYPE html>
<html>
 <head>
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
 <title>JSP Page</title>
 </head>
 <body>


 <%
 try{
 Produto produto = (Produto) request.getAttribute("lista");
 if (produto != null){
 %>


 <h1> Dados do Produto </h1>
```

```
<form action="ServletProdutoFC" method="post" >
<input type="hidden" name="acao" value="alterar">
<input type="hidden" name="idproduto"
value="<%=produto.getIdproduto()%>">
Nome: <input name="nome" value="<%=produto.getNome()%>"/>
Quantidade: <input name="quantidade"
value="<%=produto.getQuantidade()%>"/>
Preco de Venda: <input name="preco"
value="<%=produto.getPrecoVenda()%>"/>
<input type="submit" value="Alterar Produto"/>
</form>
<%
} else {
%>
<h1> Dados do Produto </h1>
<form action="ServletProdutoFC" method="post" >
<input type="hidden" name="acao" value="incluir">
Nome: <input name="nome"/>
Quantidade: <input name="quantidade"/>
Preco de Venda: <input name="preco"/>
<input type="submit" value="Adicionar Produto"/>
</form>
<%
}
} catch(ClassCastException nexc){
out.print("<h1>"+nexc.getMessage()+"</h1>");}
%>
</body>
```

```
</html>
```

3º Procedimento | Melhorando o Design da Interface

- ProdutoLista.jsp

```
<%--
 Document : ProdutoLista
 Created on : 2 de jun. de 2025, 18:21:02
 Author : caio
--%>
<%@page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@page import="cadastroee.model.Produto"%>
<%@page import="java.util.ArrayList"%>
<%@page import="java.util.List"%>
<%@page import="cadastroee.controller.ProdutoFacadeLocal"%>
<!DOCTYPE html>
<html>
 <head>
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
 <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css
" rel="stylesheet" integrity="sha384-
T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/
Dwwykc2MPK8M2HN"
crossorigin="anonymous"> <title>JSP Page</title>
 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.mi
n.js" integrity="sha384-
C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"
crossorigin="anonymous"></script>
 </head>

 <body class="container">
 <div>
 <h1>Listagem de Produtos</h1>
 <h3>
 <a class="btn btn-primary m-2"
href="ServletProdutoFC?acao=formIncluir"> Novo Produto </a>
 <!-- <a href="ServletProdutoFC"> Atualizar Dados </a> !-->
 </h3>

 <table class="table table-striped" border="1" width="100%">
 <tr class="table-dark">
 <td> ID </td>
 <td> Nome </td>
 <td> Quantidade </td>
 <td> Preco de Venda </td>
```

```
 <td> Opções </td>
 </tr>

 <%
 try{
List<Produto> lista = (List)
request.getAttribute("lista");
 for(Produto p: lista){
 %>
 <tr>
 <td>
 <%=p.getIdproduto()%>
 </td>
<td>
 <%=p.getNome()%>
 </td>
 <td>
 <%=p.getQuantidade()%>
 </td>
<td>
 <%=p.getPrecoVenda()%>
 </td>
<td>
 <a class="btn btn-primary btn-sm"
href="ServletProdutoFC?acao=formAlterar&idproduto=<%=p.getIdproduto()%>">
Alterar </a>
 <a class="btn btn-danger btn-sm"
href="ServletProdutoFC?acao=excluir&idproduto=<%=p.getIdproduto()%>"> Excluir
</a>
 </td>
 </tr>
 <% }
 } catch(NullPointerException nexc){
 out.print("<h1>"+nexc.getMessage()+"</h1>");

 }
 %>

 </table>
 </div>
 </body>
</html>
```

- ProdutoDados.jsp

```
<%--

Document : ProdutoDados

Created on : 2 de jun. de 2025, 19:02:54
```

```
 Author : caio
--%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="cadastroee.model.Produto"%>
<%@page import="java.util.ArrayList"%>
<%@page import="java.util.List"%>
<%@page import="cadastroee.controller.ProdutoFacadeLocal"%>
<!DOCTYPE html>
<html>
 <head>
 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
 <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css
" rel="stylesheet" integrity="sha384-
T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"
crossorigin="anonymous"> <title>JSP Page</title>
 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.mi
n.js" integrity="sha384-
C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"
crossorigin="anonymous"></script>
 </head>
 <body class="container">


 <%
 try{
 Produto produto = (Produto) request.getAttribute("lista");
 if (produto != null){
```

```
%>


<h1> Dados do Produto </h1>
<form class="form" action="ServletProdutoFC" method="post" >
<input type="hidden" name="acao" value="alterar">
<input type="hidden" name="idproduto"
value="<%=produto.getIdproduto()%>">
<div class=" mb-3">
<label class="form-label"> Nome: </label>
<input class="form-control" name="nome"
value="<%=produto.getNome()%>"/>
</div>
<div class=" mb-3">
<label class="form-label">Quantidade: </label>
<input class="form-control" name="quantidade"
value="<%=produto.getQuantidade()%>"/>
</div>
<div class=" mb-3">
<label class="form-label"> Preco de Venda: </label>
<input class="form-control" name="preco"
value="<%=produto.getPrecoVenda()%>"/>
</div>
<div class=" mb-3">
<input class=" btn btn-primary" type="submit"
value="Alterar Produto"/>
</div>
</form>
<%
```

```
 } else {
 %>
 <h1> Dados do Produto </h1>
 <form class="form" action="ServletProdutoFC" method="post" >
 <input type="hidden" name="acao" value="incluir">
 <div class=" mb-3">
 <label class="form-label">Nome: </label>
 <input class="form-control" name="nome"/>
 </div>
<div class=" mb-3">
 <label class="form-label">Quantidade: </label>
 <input class="form-control" name="quantidade"/>
 </div>
<div class=" mb-3">
 <label class="form-label">Preco de Venda: </label>
 <input class="form-control" name="preco"/>
 </div>
 <input class=" btn btn-primary" type="submit"
value="Adicionar Produto"/>
 </form>
 <%
 }
} catch(ClassCastException nexc){
 out.print("<h1>"+nexc.getMessage()+"</h1>");}
 %>
 </body>
</html>
```

3. Análise e Conclusão

## 1º Procedimento

a) Como é organizado um projeto corporativo no NetBeans?

No NetBeans, um projeto corporativo Java é geralmente organizado em módulos separados, como EJB Module, Web Module e Application Client Module, todos dentro de um Enterprise Application Project. Isso ajuda a separar responsabilidades — por exemplo, a lógica de negócio fica no módulo EJB, enquanto as interfaces com o usuário (como servlets e JSPs) ficam no módulo web. O NetBeans facilita isso com modelos prontos e estrutura padronizada, o que torna mais fácil organizar e manter o código.

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

JPA (Java Persistence API) é usada para o mapeamento objeto-relacional, ou seja, ela facilita o acesso e a manipulação de dados no banco de dados usando classes Java em vez de SQL diretamente.

- EJB (Enterprise JavaBeans) fornece os componentes de lógica de negócio e permite que a aplicação tenha transações, segurança, concorrência e injeção de dependência de forma automatizada.

Resumindo, JPA cuida da persistência dos dados e os EJBs cuidam das regras de negócio da aplicação.

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

O NetBeans tem várias funcionalidades que ajudam na produtividade, como:

- Geradores automáticos de código, como entidades JPA a partir de tabelas do banco.
- Assistente de criação de EJBs e classes de persistência, com estrutura pronta.
- Suporte a anotações, com autocomplete e validação.
- Deploy e testes integrados ao servidor GlassFish ou outro servidor compatível.

Isso ajuda a reduzir o tempo de codificação manual e evita muitos erros.

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

Servlets são classes Java que processam requisições HTTP e geram respostas, geralmente HTML. Eles funcionam como o controlador em uma aplicação web MVC.

O NetBeans oferece suporte à criação de Servlets com:

- Modelos prontos, que já criam a estrutura básica do `doGet()` e `doPost()`.
- Facilidade de mapeamento de URL com anotações como `@WebServlet`.
- Deploy automático e integração com servidores de aplicação para testes rápidos.

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

A comunicação é feita por meio de injeção de dependência com a anotação @EJB. No servlet, declaramos a variável com essa anotação e o contêiner do servidor injeta automaticamente uma instância do EJB correspondente. Assim, o servlet pode chamar métodos do EJB como se fosse uma instância local, e o contêiner cuida do resto, incluindo gerenciamento de transações e pooling.

2º Procedimento

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

É um padrão arquitetural que se comporta como um controlador tratando todas as solicitações para um site Web e então roteia para uma ação (ou comando). o Front Controller trata todas as chamadas vindas de um site web e é organizado em duas partes: através de um Manipulador Web e uma hierarquia de Comandos. O Manipulador Web é o objeto que efetivamente recebe as solicitações HTTP do tipo POST ou GET do servidor web. Ele extrai as informações necessárias da URL e das solicitações e então decide que tipo de ação iniciar e por fim delega a um objeto Comando para executar a ação.

b) Quais as diferenças e semelhanças entre Servlets e JSPs?

- Semelhanças: Ambos geram conteúdo dinâmico e rodam no Servlet container.
- Diferenças: Servlets focam na lógica de controle (Java puro); JSPs focam na apresentação (HTML + Java).

c) 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?

A diferença entre redirecionamento simples e com o uso do método foward é:

- Redirecionamento simples (response.sendRedirect):
  Quando você faz um redirecionamento simples, o servidor envia uma resposta para o navegador com um novo endereço URL. Aí o navegador faz uma nova requisição para essa URL. Isso significa que a URL no navegador muda, e o processo é dividido em duas requisições separadas (a original e a nova). Também dá para redirecionar para páginas fora do seu servidor.

- Forward com RequestDispatcher:
  Já o forward é um encaminhamento interno no servidor. A requisição que chegou é passada para outra página ou servlet dentro do mesmo servidor, sem que o navegador saiba disso. A URL no navegador não muda, e só ocorre uma única requisição. O processamento continua no servidor.

Em objetos HttpRequest Parâmetros são Dados da URL/formulário (request.getParameter()). Já Atributos são Objetos usados internamente para passar dados entre Servlet e JSP (request.setAttribute()).

3º Procedimento

a) Como o framework Bootstrap é utilizado?

O Bootstrap é utilizado para facilitar a criação de interfaces web bonitas e responsivas. A gente usa ele incluindo seus arquivos CSS e JavaScript no HTML (pelo CDN ou baixando). Depois disso, aplicamos as classes prontas que ele fornece nos elementos HTML para estilizar botões, menus, formulários, grids, entre outros, sem precisar escrever muito CSS manualmente.

b) Por que o Bootstrap garante a independência estrutural do HTML?

Porque o Bootstrap não depende de uma estrutura HTML específica. Ele funciona com classes reutilizáveis que você aplica diretamente nos elementos HTML, então você pode montar sua página da forma que quiser, usando essas classes para definir o estilo e o comportamento visual. Isso separa o conteúdo (HTML) do estilo (CSS/Bootstrap), seguindo boas práticas de desenvolvimento.

c) Qual a relação entre o Boostrap e a responsividade da página?

O Bootstrap já vem com um sistema de grid responsivo e várias classes utilitárias que se adaptam automaticamente ao tamanho da tela. Ou seja, quando a gente usa Bootstrap, ele já garante que o layout fique organizado e utilizável em diferentes dispositivos, sem precisar fazer tudo manualmente com media queries.