Estácio - Unidade São pedro
RJ 140 Km 2, 512 loja 1, São Pedro da Aldeia - RJ, 28941-182

Desenvolvimento Full Stack
Classe: Missão Prática | Nível 4 | Mundo 3
3º Semestre
Marvin de Almeida Costa

Título da Prática: 1º Procedimento | Camadas de Persistência e Controle

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.

Todos os códigos solicitados neste roteiro de aula:

web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app version="4.0" xmlns="http://xmlns.jcp.org/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
>

```
 <servlet>
  <servlet-name>ServletProduto</servlet-name>
  <servlet-class>cadastroee.servlets.ServletProduto</servlet-class>
 </servlet>
 <servlet>
  <servlet-name>ServletProdutoFC</servlet-name>
  <servlet-class>cadastroee.servlets.ServletProdutoFC
  </servlet-class>
 </servlet>
 <session-config>
  <session-timeout>30</session-timeout>
 </session-config>
<servlet-mapping>
    <servlet-name>ServletProduto</servlet-name>
    <url-pattern>/ServletProduto</url-pattern>
</servlet-mapping>

</web-app>
```

ServletProduto.java

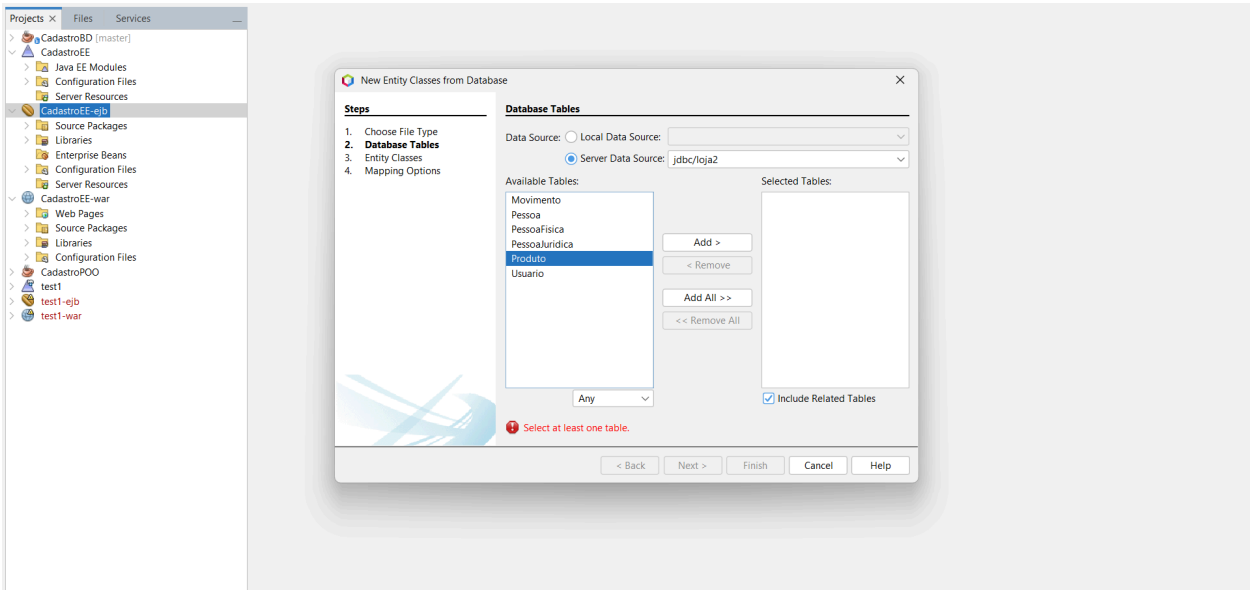
```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
license
 * Click nbfs://nbhost/SystemFileSystem/Templates/JSP_Servlet/Servlet.java to edit this
template
 */
package cadastroee.servlets;

import java.io.IOException;
import java.io.PrintWriter;
```

```
import java.util.List;
import jakarta.ejb.EJB;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import cadatroee.controller.ProdutoFacadeLocal;
import cadastroee.model.Produto;

/**
 *
 * @author marvin
 */
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()) {
            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>Lista de Produtos</h1>");
            out.println("<ul>");

            // Utilizar o facade para recuperar os dados
            List<Produto> produtos = facade.findAll();

            // Apresentar os dados em uma lista HTML
```

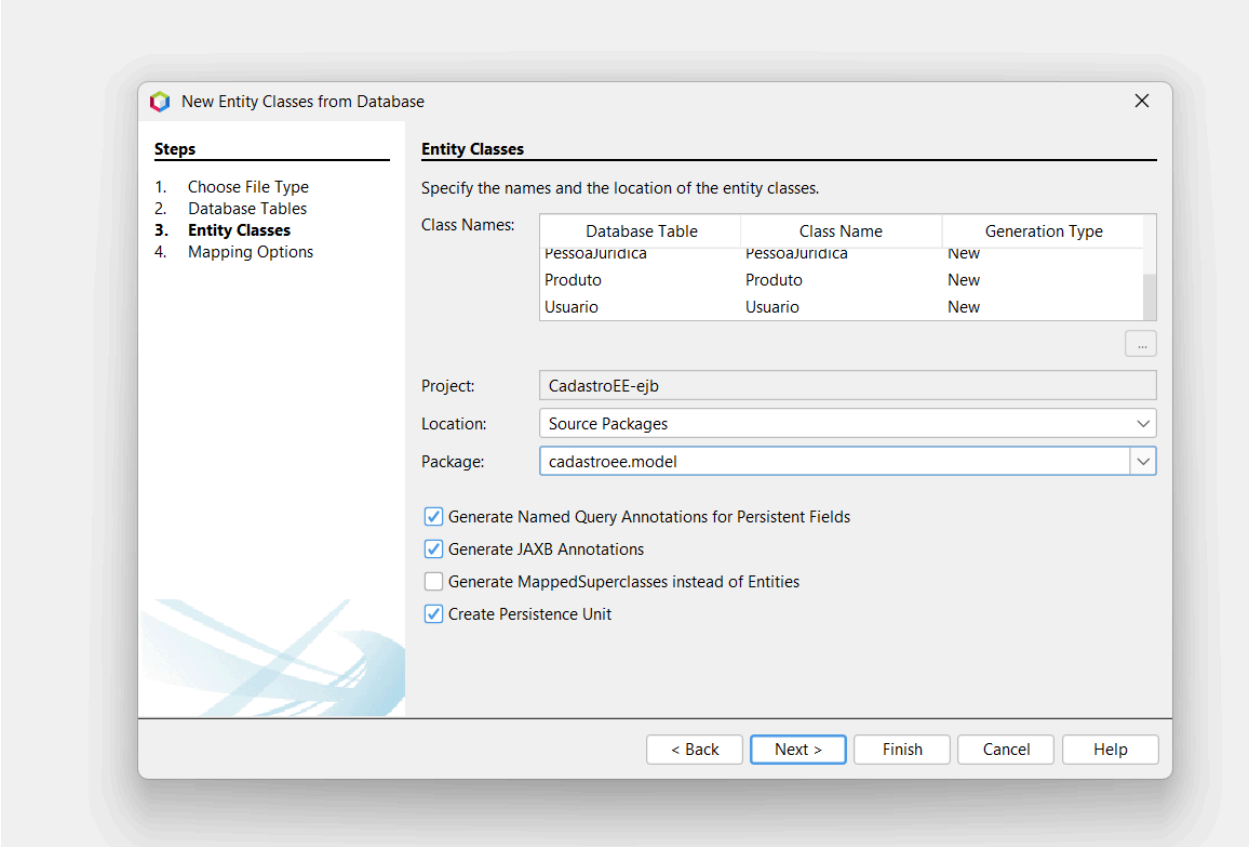
```
            for (Produto produto : produtos) {
                out.println("<li>" + produto.getNome() + "</li>");
            }

            out.println("</ul>");
            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
```

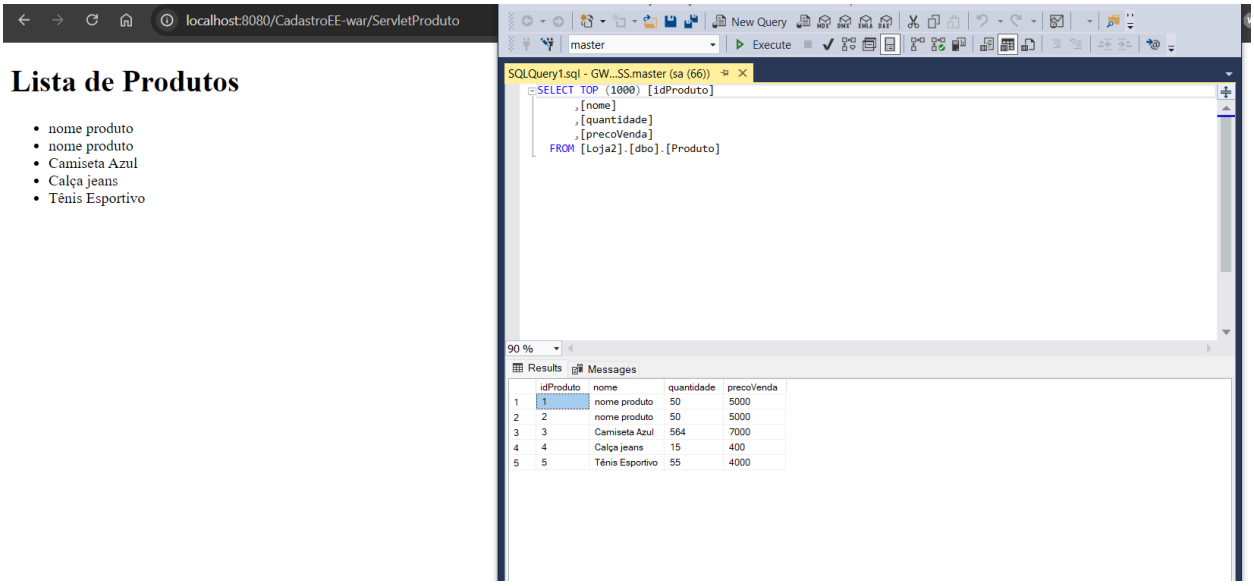
```
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Os resultados da execução dos códigos também devem ser apresentados;




```
asadmin> create-jdbc-connection-pool --datasourceclassname com.microsoft.sqlserver.jdbc.SQLServerDataSou
rce --restype javax.sql.DataSource --property driverClass=com.microsoft.sqlserver.jdbc.SQLServerDriver:p
ortNumber=1433:password=1234:user=sa:serverName=localhost:databaseName=Loja2:trustServerCertificate=true
:URL="jdbc\\:sqlserver\\://localhost\\:1433\\;databaseName\\=Loja2\\;encrypt\\=true\\;trustServerCertifi
cate\\=true\\;"
Enter the value for the jdbc_connection_pool_id operand> SQLServerPool
JDBC connection pool SQLServerPool created successfully.
Command create-jdbc-connection-pool executed successfully.
asadmin> ping-connection-pool SQLServerPool
Command ping-connection-pool executed successfully.
asadmin> create-jdbc-resource --connectionpoolid SQLServerPool jdbc/loja2
JDBC resource jdbc/loja2 created successfully.
Command create-jdbc-resource executed successfully.
asadmin>
```


New Entity Classes from Database
Steps
1. Choose File Type
2. Database Tables
3. Entity Classes
4. Mapping Options
Entity Classes
Specify the names and the location of the entity classes.
Class Names:
Database Table
Class Name
Generation Type
PessoaJuridica
PessoaJuridica
New
Produto
Produto
New
Usuario
Usuario
New
Project:
CadastroEE-ejb
Location:
Source Packages
Package:
cadastroee.model
Generate Named Query Annotations for Persistent Fields
Generate JAXB Annotations
Generate MappedSuperclasses instead of Entities
Create Persistence Unit
< Back
Next >
Finish
Cancel
Help


localhost:8080/CadastroEE-war/ServletProduto
Lista de Produtos
nome produto
nome produto
Camiseta Azul
Calça jeans
Tênis Esportivo
New Query
master
Execute
SQLQuery1.sql - GW...SS.master (sa (66))
SELECT TOP (1000) [idProduto]
,[nome]
,[quantidade]
,[precoVenda]
FROM [Loja2].[dbo].[Produto]
90 %
Results
Messages
idProduto nome quantidade precoVenda
1 1 nome produto 50 5000
2 2 nome produto 50 5000
3 3 Camiseta Azul 564 7000
4 4 Calça jeans 15 400
5 5 Tênis Esportivo 55 4000

Análise e Conclusão:

Como é organizado um projeto corporativo no NetBeans?

Um projeto corporativo no NetBeans é organizado de forma estruturada e eficiente para facilitar o desenvolvimento e gerenciamento de aplicações complexas. Vamos explorar como os projetos são organizados neste ambiente IDE:

A estrutura básica de um projeto corporativo no NetBeans geralmente inclui:

- Páginas JSP (JavaServer Pages)
- Classes Java
- Arquivos CSS e JavaScript
- Configurações do aplicativo (ex: web.xml)

Esta estrutura permite que desenvolvedores trabalhem em diferentes aspectos da aplicação de forma organizada.

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

As tecnologias JPA (Java Persistence API) e EJB (Enterprise JavaBeans) desempenham papéis importantes na construção de aplicações web em Java. Vamos explorar cada uma delas separadamente:

JPA (Java Persistence API)

O JPA é uma especificação da Sun Microsystems que define uma API padrão para persistência de dados em Java. Seu principal objetivo é fornecer uma maneira consistente de mapear objetos Java para tabelas de banco de dados.

Papéis do JPA:

- Abstrair a complexidade de acesso a bancos de dados
- Facilitar o desenvolvimento de aplicações com base em objetos
- Melhorar a portabilidade entre diferentes sistemas de gerenciamento de banco de dados

EJB (Enterprise JavaBeans)

O EJB é uma especificação da Sun Microsystems que define um framework para desenvolver aplicações de negócios distribuídas em Java. Ele fornece recursos como gerenciamento de transações, segurança e persistência de estado.

Papéis do EJB:

- Fornecer uma estrutura padrão para desenvolvimento de aplicações empresariais
- Gerenciar transações automaticamente
- Implementar segurança e autenticação
- Facilitar o desenvolvimento de aplicações distribuídas

Integração entre JPA e EJB

As tecnologias JPA e EJB são frequentemente usadas juntas na construção de aplicações web em Java. O JPA fornece a camada de acesso aos dados, enquanto o EJB fornece a lógica de negócios e gerencia os recursos compartilhados.

Vantagens desta integração:

- Separação clara das responsabilidades
- Melhor reutilização de código
- Facilita o desenvolvimento de aplicações escaláveis e robustas

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

O NetBeans oferece várias funcionalidades que ajudam a melhorar a produtividade ao trabalhar com tecnologias como Java Persistence API (JPA) e Enterprise JavaBeans (EJB). Vamos explorar algumas das principais maneiras pelas quais o NetBeans facilita o desenvolvimento com essas tecnologias:

Integração com JPA

1. Editor de Entidades:

- O NetBeans fornece um editor especializado para criar e editar classes de entidades JPA.
- Ele oferece suporte à autocompletação, validação em tempo real e sugestões inteligentes.

2. Geração de código:

- O IDE pode gerar automaticamente os métodos padrão da JPA (como toString(), equals(), etc.) para suas entidades.
- Isso economiza tempo e garante que seu código seja consistente com as melhores práticas.

3. Visualização de relacionamentos:

- O NetBeans permite visualizar os relacionamentos entre entidades através de diagramas UML.
- Isso ajuda a entender rapidamente a estrutura do banco de dados e a lógica de negócios.

Suporte a EJB

1. Criação de componentes EJB:

- O IDE oferece templates para criar facilmente novos componentes EJB, incluindo sessões, entidades, e serviços.

2. Integração com contêiner EJB:

- O NetBeans configura automaticamente o contêiner EJB necessário para executar seus aplicativos.

3. Depuragem avançada:

- O IDE fornece recursos de depuragem específicos para EJB, permitindo uma análise mais profunda dos problemas durante a execução.

Melhorias na produtividade

1. Autocomplete e sugestões inteligentes:

- O NetBeans oferece autocomplete e sugestões baseadas em contexto para APIs JPA e EJB.
- Isso reduz significativamente o tempo gasto em digitação e pesquisa.

2. Refatoração automática:

- O IDE pode sugerir e realizar refatorações automáticas, melhorando a manutenibilidade do código.

3. Integração com bancos de dados:

- O NetBeans permite conectar-se diretamente aos bancos de dados, facilitando a criação e manipulação de scripts DDL.

4. Teste unitário integrado:

- O IDE inclui ferramentas de teste unitário que podem ser usadas com JUnit e Mockito, facilitando o desenvolvimento de testes para componentes JPA e EJB.

5. Integração com frameworks populares:

- O NetBeans tem suporte nativo para frameworks como Spring Boot, Hibernate, e outros que frequentemente são usados em conjunto com JPA e EJB.

6. Gerenciamento de dependências:

- O IDE oferece ferramentas para gerenciar dependências, incluindo a resolução de dependências e atualização de versões.

7. Visualização de logs:

- O NetBeans fornece uma interface gráfica para visualizar logs de aplicativo, facilitando o diagnóstico de problemas.

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

Os Servlets são uma parte fundamental da tecnologia web Java, permitindo a criação de aplicações web dinâmicas e escaláveis. Vamos explorar o conceito de Servlets e como o NetBeans oferece suporte para desenvolver esses componentes em projetos web.

O que são Servlets?

Servlets são classes Java que processam requisições HTTP e geram respostas HTTP. Eles formam a base do framework JSP (JavaServer Pages) e são amplamente utilizados na construção de aplicações web Java Enterprise (JEE).

Principais características dos Servlets:

- São executados no servidor, não no cliente
- Processam dados enviados pelo navegador através de formulários ou outras fontes
- Podem modificar dinamicamente conteúdo HTML antes de enviar para o cliente
- Permitem interação com bancos de dados e outros recursos do servidor

Como o NetBeans oferece suporte à construção de Servlets

O NetBeans fornece várias ferramentas e funcionalidades para facilitar o desenvolvimento de Servlets em projetos web:

1. Criação automática de arquivos Servlet:

- Ao criar um novo projeto web, o NetBeans gera automaticamente um arquivo HelloWorldServlet.java
- Você pode personalizar este arquivo conforme necessário

2. Integração com o GlassFish Server:

- O NetBeans inclui um servidor web embutido (GlassFish) para executar e depurar Servlets localmente
- Você pode configurar e gerenciar o servidor diretamente através da IDE
-

3. Suporte ao Maven ou Gradle:

- As ferramentas de build são integradas para gerenciar dependências e configurações do projeto
- Facilitam a adição de bibliotecas necessárias para desenvolvimento de Servlets

4. Visualização de código-fonte:

- O NetBeans oferece uma visão clara do código-fonte dos seus Servlets
- Inclui recursos como autocompletar, refatoração e navegação entre arquivos

5. Depuração em tempo real:

- Permite depurar seu código Servlet em tempo real, colocando breakpoints e seguindo o fluxo de execução

6. Geração de código JSP:

- Ao criar um novo arquivo JSP, o NetBeans gera automaticamente os componentes necessários no backend (como classes de controlador)

7. Integração com bancos de dados:

- Oferece suporte para conexão com bancos de dados populares, facilitando a integração de operações de banco de dados nos Servlets

8. Teste unitário:

- Inclui ferramentas para escrever e executar testes unitários para seus Servlets

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

A comunicação entre os Serviços (Services) e as Beans de Sessão (Session Beans) no pool de Enterprise JavaBeans (EJBs) é um aspecto crucial da arquitetura EJB. Esta comunicação permite que os serviços externos interajam com as funcionalidades encapsuladas nas beans de sessão. Vamos explorar como esta comunicação é feita:

Interface Remota

A base para a comunicação entre serviços externos e beans de sessão é a interface remota. As beans de sessão expõem suas operações através de uma interface remota, que é implementada pelo contêiner EJB.

- O desenvolvedor define uma interface Java que especifica os métodos que serão acessados remotamente.
- Esta interface é anotada com @Remote ou @Local, dependendo se será acessada remotamente ou localmente.

Injeção de Dependência

O contêiner EJB é responsável por injetar a referência da bean de sessão nos serviços externos:

- Quando um serviço externo solicita uma instância da interface remota, o contêiner EJB fornece essa instância.
- A referência à bean de sessão é criada e gerenciada pelo contêiner.

Segurança e Autenticação
Para garantir a segurança das chamadas remotas, o EJB usa:

- Autenticação: Verifica se o cliente tem permissão para acessar o método específico.
- Autorização: Controla quais operações o cliente pode realizar.

Transações
As transações são gerenciadas automaticamente pelo contêiner EJB:

- Os métodos da bean de sessão são marcados com anotações como @TransactionAttribute.
- O contêiner gerencia as transações, garantindo consistência dos dados.