Trabalho Prático | DGT2811 Desenvolvimento Back-End Corporativo com Java E Cloud

Curso: Desenvolvimento Full Stack

Polo: Polo - Vila Nova - Itu - Sp

Matrícula: 202404575268

Aluno: Rafael Lima de Medeiros

Turma: 2025.4 (5º Semestre)

Data: 14/02/2026

Github:
https://github.com/Rafaldm/Desenv-Back-end-Corporativo-Com-Java-e-Cloud

Objetivo: Criar uma aplicação corporativa usando a plataforma Jakarta EE. Nela, os dados são salvos no banco usando JPA, as regras do sistema são organizadas com EJB, que aparece para o usuário funciona com Servlets. A aplicação também foi conectada ao banco de dados SQL Server e publicada no servidor GlassFish.

Análise

Como é organizado um projeto corporativo no NetBeans?
Um projeto corporativo no NetBeans é dividido em módulos para separar as responsabilidades da aplicação.
Normalmente ele é organizado em:
EAR → Projeto principal que junta tudo
EJB → Onde ficam as regras de negócio e a conexão com o banco
WAR → Parte Web (Servlet, páginas, interface)
Essa separação facilita a organização do código, deixa o projeto mais profissional e mais fácil de manter.
Cada parte tem sua função bem definida.

Qual o papel das tecnologias JPA e EJB na construção de um aplicativo Web?
A JPA é responsável por fazer a comunicação com o banco de dados.
Ela permite salvar, buscar e atualizar informações usando classes Java, sem precisar escrever muito SQL direto no código.
O EJB é responsável pela regra de negócio da aplicação.
Ele funciona como uma camada intermediária entre a parte Web e o banco de dados.
Resumindo:
JPA → cuida dos dados
EJB → cuida da lógica da aplicação

Como o NetBeans melhora a produtividade com JPA e EJB?
O NetBeans ajuda bastante porque ele cria muita coisa automaticamente.
Por exemplo:
Gera entidades JPA a partir do banco
Cria Session Beans automaticamente
Configura arquivos como persistence.xml
Integra direto com o servidor GlassFish
Isso economiza tempo e evita erros de configuração manual.

O que são Servlets?
Servlets são classes Java que rodam no servidor e respondem às requisições do navegador.
Quando o usuário acessa uma URL, o Servlet recebe essa requisição, processa as informações e devolve uma resposta (como uma página HTML).
O NetBeans facilita isso porque:
Permite criar Servlet com poucos cliques
Gera a estrutura básica automaticamente
Faz a integração direta com o servidor

Como é feita a comunicação entre Servlets e Session Beans (EJB)?
A comunicação é feita por injeção automática.
No Servlet usamos:

```
@EJB
private ProdutoFacadeLocal facade;
```

Isso faz com que o servidor injete automaticamente o EJB dentro do Servlet.
Assim, o Servlet consegue chamar métodos do EJB como:

```
facade.findAll();
```

Sem precisar criar conexão manual ou instanciar objeto.
O servidor gerencia tudo internamente.

CÓDIGOS

Produto.java

```
package br.com.loja.entidade;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;

@Entity
@Table(name = "produto")
public class Produto {

   @Id
   private int id;
   private String nome;
   private double preco;

   public Produto() {
   }

   public Produto(int id, String nome, double preco) {
      this.id = id;
      this.nome = nome;
      this.preco = preco;
   }

   public int getId() {
      return id;
   }

   public void setId(int id) {
      this.id = id;
   }

   public String getNome() {
      return nome;
   }

   public void setNome(String nome) {
      this.nome = nome;
   }

   public double getPreco() {
      return preco;
   }

   public void setPreco(double preco) {
      this.preco = preco;
   }
}
```

persistence.xml

```
<persistence xmlns="https://jakarta.ee/xml/ns/persistence"
          version="3.0">

   <persistence-unit name="lojaPU" transaction-type="JTA">
      <jta-data-source>jdbc/loja</jta-data-source>

      <properties>
         <property name="jakarta.persistence.schema-
generation.database.action"
                   value="none"/>
      </properties>
   </persistence-unit>

</persistence>
```

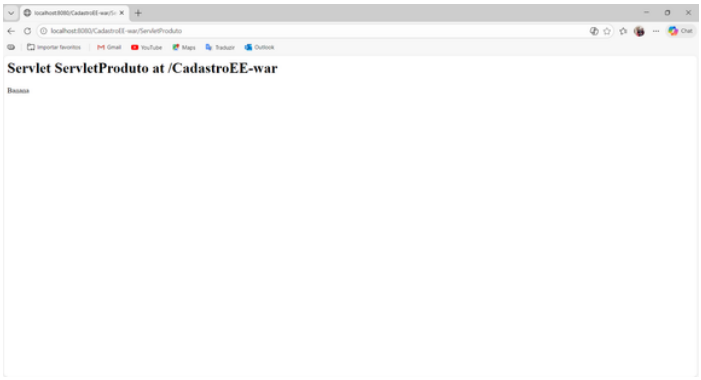
ProdutoBean.java

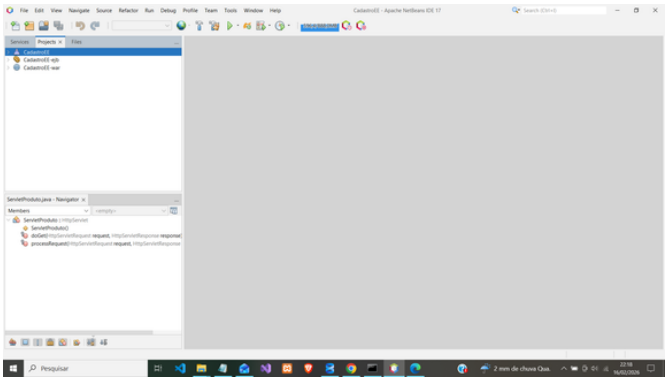
```
package br.com.loja.bean;

import br.com.loja.entidade.Produto;
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import java.util.List;

@Stateless
public class ProdutoBean {

   @PersistenceContext
   private EntityManager em;

   public List<Produto> listarProdutos() {
      return em.createQuery("SELECT p FROM Produto p", Produto.class)
            .getResultList();
   }
}
```

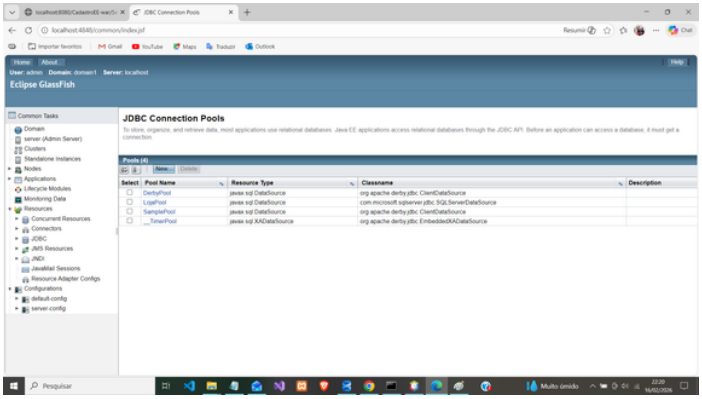
ServletProduto.java

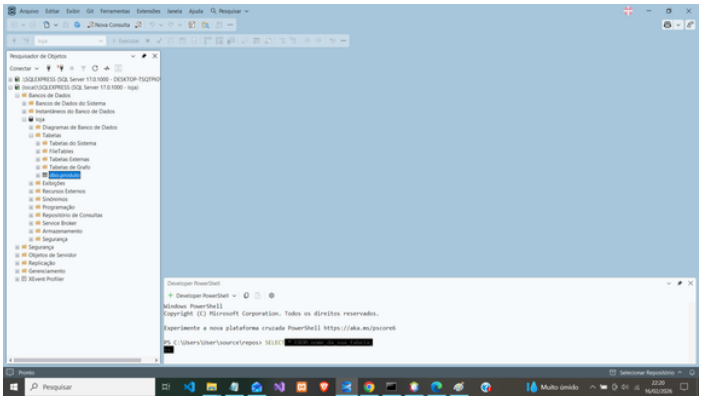
```
package br.com.loja.servlet;

import br.com.loja.bean.ProdutoBean;
import br.com.loja.entidade.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;
import java.util.List;

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

   @EJB
   private ProdutoBean produtoBean;

   @Override
   protected void doGet(HttpServletRequest request,
HttpServletResponse response)
         throws ServletException, IOException {

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

      List<Produto> lista = produtoBean.listarProdutos();

      try (PrintWriter out = response.getWriter()) {

         out.println("<h1>Lista de Produtos</h1>");

         for (Produto p : lista) {
            out.println(p.getId() + " - "
                  + p.getNome() + " - "
                  + p.getPreco() + "<br>");
         }
      }
   }
}
```

PRINTS


Servlet ServletProduto at /CadastroEE-war
Baunilha




Eclipse GlassFish
JDBC Connection Pools



Conclusão

Durante a prática foi possível entender como funciona a estrutura de uma aplicação corporativa em Java.
A separação em camadas (Web, negócio e banco) deixa o sistema organizado e mais profissional.
A JPA facilita o acesso ao banco, o EJB organiza as regras de negócio e o Servlet faz a comunicação com o usuário.
O NetBeans ajuda bastante no processo, pois automatiza várias etapas e integra diretamente com o servidor, tornando o desenvolvimento mais rápido e simples.