RELATÓRIO DISCENTE DE ACOMPANHAMENTO

Campus Polo Cavalhada – Porto Alegre

DESENVOLVIMENTO FULL STACK

Missão Prática | Nível 3 | Mundo 4

RPG0017 - Vamos integrar sistemas

Turma: 9003

Semestre Letivo: 3º - 2024.1

GUILHERME BERNARDES BASTOS

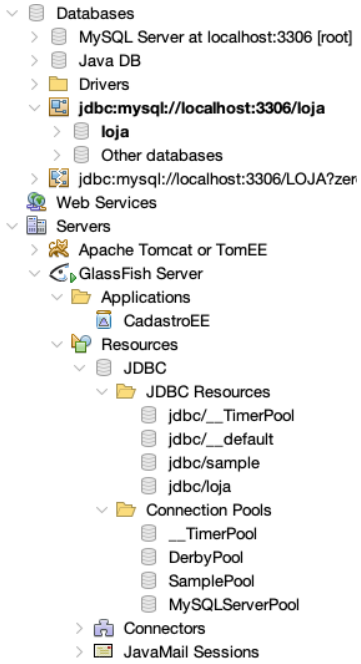
Repositório: https://github.com/guilhermebernardes96/Mundo3-Nivel4

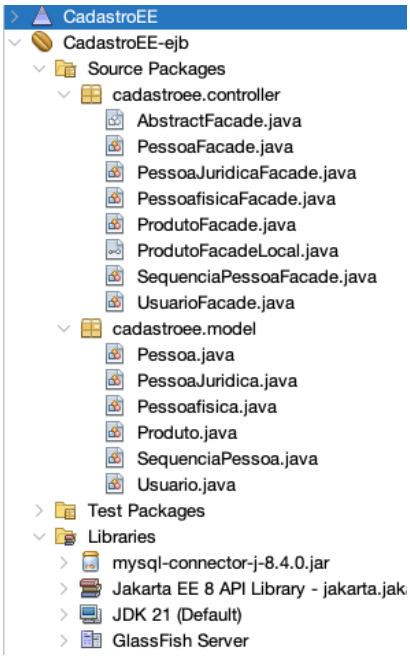
CRIANDO O BANCO DE DADOS

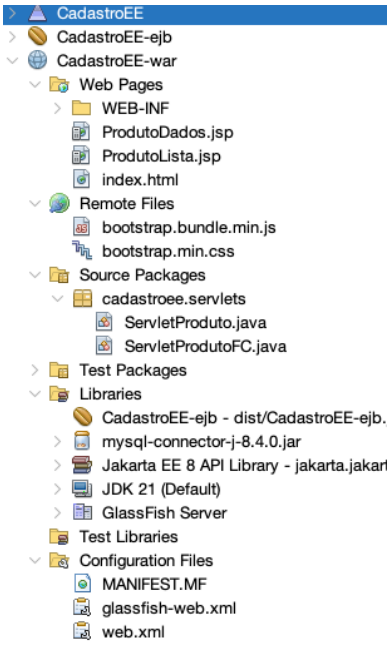
Objetivo da Prática

- Implementar persistência com base em JPA.
- Implementar regras de negocio na plataforma JEE.
- Utilizar a biblioteca Bootstrap para melhoria do design.

PROCEDIMENTOS








```java
package cadastroee.controller;
import java.util.List;
import jakarta.persistence.EntityManager;
public abstract class AbstractFacade<T> {
    private Class<T> entityClass;
    public AbstractFacade(Class<T> entityClass) {
        this.entityClass = entityClass;}
    protected abstract EntityManager getEntityManager();
    public void create(T entity) {
        getEntityManager().persist(entity);}
    public void edit(T entity) {
        getEntityManager().merge(entity);}
    public void remove(T entity) {
        getEntityManager().remove(getEntityManager().merge(entity));}
    public T find(Object id) {
        return getEntityManager().find(entityClass, id);}
    public List<T> findAll() {
        jakarta.persistence.criteria.CriteriaQuery cq = getEntityManager().getCriteriaBuilder().createQuery();
        cq.select(cq.from(entityClass));
        return getEntityManager().createQuery(cq).getResultList();}
    public List<T> findRange(int[] range) {
        jakarta.persistence.criteria.CriteriaQuery cq = getEntityManager().getCriteriaBuilder().createQuery();
        cq.select(cq.from(entityClass));
        jakarta.persistence.Query q = getEntityManager().createQuery(cq);
        q.setMaxResults(range[1] - range[0] + 1);
        q.setFirstResult(range[0]);
        return q.getResultList();}
    public int count() {
        jakarta.persistence.criteria.CriteriaQuery cq = getEntityManager().getCriteriaBuilder().createQuery();
        jakarta.persistence.criteria.Root<T> rt = cq.from(entityClass);
        cq.select(getEntityManager().getCriteriaBuilder().count(rt));
        jakarta.persistence.Query q = getEntityManager().createQuery(cq);
        return ((Long) q.getSingleResult()).intValue();}
}
```

```java
package cadastroee.controller;
import cadastroee.model.Pessoa;
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;

@Stateless
public class PessoaFacade extends AbstractFacade<Pessoa> {

    @PersistenceContext(unitName = "CadastroEE-ejbPU")
    private EntityManager em;

    @Override
    protected EntityManager getEntityManager() {
        return em;
    }

    public PessoaFacade() {
        super(Pessoa.class);
    }
}
```

```java
package cadastroee.controller;
import cadastroee.model.PessoaJuridica;
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;

@Stateless
public class PessoaJuridicaFacade extends AbstractFacade<PessoaJuridica> {

    @PersistenceContext(unitName = "CadastroEE-ejbPU")
    private EntityManager em;

    @Override
    protected EntityManager getEntityManager() {
        return em;
    }

    public PessoaJuridicaFacade() {
        super(PessoaJuridica.class);
    }
}
```

```java
package cadastroee.controller;
import cadastroee.model.Pessoafisica;
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;

@Stateless
public class PessoafisicaFacade extends AbstractFacade<Pessoafisica> {

    @PersistenceContext(unitName = "CadastroEE-ejbPU")
    private EntityManager em;

    @Override
    protected EntityManager getEntityManager() {
        return em;
    }

    public PessoafisicaFacade() {
        super(Pessoafisica.class);
    }
}
```

```java
package cadastroee.controller;
import cadastroee.model.Produto;
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import java.util.List;

@Stateless
public class ProdutoFacade extends AbstractFacade<Produto> implements ProdutoFacadeLocal {

    @PersistenceContext(unitName = "CadastroEE-ejbPU")
    private EntityManager em;

    @Override
    protected EntityManager getEntityManager() {
        return em;
    }

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

    public ProdutoFacade() {
        super(Produto.class);
    }

}
```

```java
package cadastroee.controller;
import cadastroee.model.Produto;
import jakarta.ejb.Local;
import java.util.List;

@Local
public interface ProdutoFacadeLocal {

    void create(Produto produto);

    void edit(Produto produto);

    void remove(Produto produto);

    Produto find(Object id);

    List<Produto> findAll();

    List<Produto> findRange(int[] range);

    int count();
}
```

```java
package cadastroee.controller;
import cadastroee.model.SequenciaPessoa;
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;

@Stateless
public class SequenciaPessoaFacade extends AbstractFacade<SequenciaPessoa> {

    @PersistenceContext(unitName = "CadastroEE-ejbPU")
    private EntityManager em;

    @Override
    protected EntityManager getEntityManager() {
        return em;
    }

    public SequenciaPessoaFacade() {
        super(SequenciaPessoa.class);
    }

}
```

```java
package cadastroee.controller;
import cadastroee.model.Usuario;
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
@Stateless
public class UsuarioFacade extends AbstractFacade<Usuario> {

    @PersistenceContext(unitName = "CadastroEE-ejbPU")
    private EntityManager em;

    @Override
    protected EntityManager getEntityManager() {
        return em;
    }

    public UsuarioFacade() {
        super(Usuario.class);
    }

}
```

```java
package cadastroee.model;
import jakarta.persistence.Basic;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.NamedQueries;
import jakarta.persistence.NamedQuery;
import jakarta.persistence.Table;
import java.io.Serializable;
@Entity
@Table(name = "pessoa")
@NamedQueries({
    @NamedQuery(name = "Pessoa.findAll", query = "SELECT p FROM Pessoa p"),
    @NamedQuery(name = "Pessoa.findByIdPessoa", query = "SELECT p FROM Pessoa p WHERE p.idPessoa = :idPessoa"),
    @NamedQuery(name = "Pessoa.findByNome", query = "SELECT p FROM Pessoa p WHERE p.nome = :nome"),
    @NamedQuery(name = "Pessoa.findByLogradouro", query = "SELECT p FROM Pessoa p WHERE p.logradouro = :logradouro"),
    @NamedQuery(name = "Pessoa.findByCidade", query = "SELECT p FROM Pessoa p WHERE p.cidade = :cidade"),
    @NamedQuery(name = "Pessoa.findByEstado", query = "SELECT p FROM Pessoa p WHERE p.estado = :estado"),
    @NamedQuery(name = "Pessoa.findByTelefone", query = "SELECT p FROM Pessoa p WHERE p.telefone = :telefone"),
    @NamedQuery(name = "Pessoa.findByEmail", query = "SELECT p FROM Pessoa p WHERE p.email = :email")})
public class Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "idPessoa")
    private Integer idPessoa;
    @Basic(optional = false)
    @Column(name = "nome")
    private String nome;
    @Basic(optional = false)
    @Column(name = "logradouro")
    private String logradouro;
    @Basic(optional = false)
    @Column(name = "cidade")
    private String cidade;
    @Basic(optional = false)
    @Column(name = "estado")
    private String estado;
```

```java
    @Basic(optional = false)
    @Column(name = "telefone")
    private String telefone;
    @Basic(optional = false)
    @Column(name = "email")
    private String email;

    public Pessoa() {}
    public Pessoa(Integer idPessoa) {
        this.idPessoa = idPessoa;
    }
    public Pessoa(Integer idPessoa, String nome, String logradouro, String cidade, String estado, String telefone, String email) {
        this.idPessoa = idPessoa;
        this.nome = nome;
        this.logradouro = logradouro;
        this.cidade = cidade;
        this.estado = estado;
        this.telefone = telefone;
        this.email = email;
    }
    public Integer getIdPessoa() {
        return idPessoa;
    }
    public void setIdPessoa(Integer idPessoa) {
        this.idPessoa = idPessoa;
    }
    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public String getLogradouro() {
        return logradouro;
    }
    public void setLogradouro(String logradouro) {
        this.logradouro = logradouro;
    }
    public String getCidade() {
        return cidade;
    }
```

```java
    public void setCidade(String cidade) {
        this.cidade = cidade;
    }
    public String getEstado() {
        return estado;
    }
    public void setEstado(String estado) {
        this.estado = estado;
    }
    public String getTelefone() {
        return telefone;
    }
    public void setTelefone(String telefone) {
        this.telefone = telefone;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idPessoa != null ? idPessoa.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        if (!(object instanceof Pessoa)) {
            return false;
        }
        Pessoa other = (Pessoa) object;
        if ((this.idPessoa == null && other.idPessoa != null) || (this.idPessoa != null && !this.idPessoa.equals(other.idPessoa))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "cadastroee.model.Pessoa[ idPessoa=" + idPessoa + " ]";
    }
}
```

```java
package cadastroee.model;
import jakarta.persistence.Basic;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.NamedQueries;
import jakarta.persistence.NamedQuery;
import jakarta.persistence.Table;
import java.io.Serializable;

@Entity
@Table(name = "pessoaJuridica")
@NamedQueries({
    @NamedQuery(name = "PessoaJuridica.findAll", query = "SELECT p FROM PessoaJuridica p"),
    @NamedQuery(name = "PessoaJuridica.findByIdPessoa", query = "SELECT p FROM PessoaJuridica p WHERE p.idPessoa = :idPessoa"),
    @NamedQuery(name = "PessoaJuridica.findByCnpj", query = "SELECT p FROM PessoaJuridica p WHERE p.cnpj = :cnpj")})
public class PessoaJuridica implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @Column(name = "idPessoa")
    private Integer idPessoa;
    @Basic(optional = false)
    @Column(name = "cnpj")
    private String cnpj;
    public PessoaJuridica() {}
    public PessoaJuridica(Integer idPessoa) {
        this.idPessoa = idPessoa;
    }
    public PessoaJuridica(Integer idPessoa, String cnpj) {
        this.idPessoa = idPessoa;
        this.cnpj = cnpj;
    }
    public Integer getIdPessoa() {
        return idPessoa;
    }
    public void setIdPessoa(Integer idPessoa) {
        this.idPessoa = idPessoa;
    }
    public String getCnpj() {
        return cnpj;
    }
```

```java
    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idPessoa != null ? idPessoa.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        if (!(object instanceof PessoaJuridica)) {
            return false;
        }
        PessoaJuridica other = (PessoaJuridica) object;
        if ((this.idPessoa == null && other.idPessoa != null) || (this.idPessoa != null && !this.idPessoa.equals(other.idPessoa))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "cadastroee.model.PessoaJuridica[ idPessoa=" + idPessoa + " ]";
    }
}
```

```java
package cadastroee.model;
import jakarta.persistence.Basic;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.NamedQueries;
import jakarta.persistence.NamedQuery;
import jakarta.persistence.Table;
import java.io.Serializable;

@Entity
@Table(name = "pessoafisica")
@NamedQueries({
    @NamedQuery(name = "Pessoafisica.findAll", query = "SELECT p FROM Pessoafisica p"),
    @NamedQuery(name = "Pessoafisica.findByIdPessoa", query = "SELECT p FROM Pessoafisica p WHERE p.idPessoa = :idPessoa"),
    @NamedQuery(name = "Pessoafisica.findByCpf", query = "SELECT p FROM Pessoafisica p WHERE p.cpf = :cpf")})
public class Pessoafisica implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @Column(name = "idPessoa")
    private Integer idPessoa;
    @Basic(optional = false)
    @Column(name = "cpf")
    private String cpf;
    public Pessoafisica() {}

    public Pessoafisica(Integer idPessoa) {
        this.idPessoa = idPessoa;
    }
    public Pessoafisica(Integer idPessoa, String cpf) {
        this.idPessoa = idPessoa;
        this.cpf = cpf;
    }
    public Integer getIdPessoa() {
        return idPessoa;
    }
    public void setIdPessoa(Integer idPessoa) {
        this.idPessoa = idPessoa;
    }
```

```java
    public String getCpf() {
        return cpf;
    }
    public void setCpf(String cpf) {
        this.cpf = cpf;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idPessoa != null ? idPessoa.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        if (!(object instanceof Pessoafisica)) {
            return false;
        }
        Pessoafisica other = (Pessoafisica) object;
        if ((this.idPessoa == null && other.idPessoa != null) || (this.idPessoa != null && !this.idPessoa.equals(other.idPessoa))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "cadastroee.model.Pessoafisica[ idPessoa=" + idPessoa + " ]";
    }
}
```

```java
package cadastroee.model;
import jakarta.persistence.Basic;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.NamedQueries;
import jakarta.persistence.NamedQuery;
import jakarta.persistence.Table;
import java.io.Serializable;

@Entity
@Table(name = "produto")
@NamedQueries({
    @NamedQuery(name = "Produto.findAll", query = "SELECT p FROM Produto p"),
    @NamedQuery(name = "Produto.findByIdProduto", query = "SELECT p FROM Produto p WHERE p.idProduto = :idProduto"),
    @NamedQuery(name = "Produto.findByNome", query = "SELECT p FROM Produto p WHERE p.nome = :nome"),
    @NamedQuery(name = "Produto.findByQuantidade", query = "SELECT p FROM Produto p WHERE p.quantidade = :quantidade"),
    @NamedQuery(name = "Produto.findByPrecoVenda", query = "SELECT p FROM Produto p WHERE p.precoVenda = :precoVenda")})
public class Produto implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "idProduto")
    private Integer idProduto;
    @Basic(optional = false)
    @Column(name = "nome")
    private String nome;
    @Basic(optional = false)
    @Column(name = "quantidade")
    private int quantidade;
    @Column(name = "precoVenda")
    private Float precoVenda;

    public Produto() {}

    public Produto(Integer idProduto) {
        this.idProduto = idProduto;
    }
```

```java
    public Produto(Integer idProduto, String nome, int quantidade) {
        this.idProduto = idProduto;
        this.nome = nome;
        this.quantidade = quantidade;
    }
    public Integer getIdProduto() {
        return idProduto;
    }
    public void setIdProduto(Integer idProduto) {
        this.idProduto = idProduto;
    }
    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public int getQuantidade() {
        return quantidade;
    }
    public void setQuantidade(int quantidade) {
        this.quantidade = quantidade;
    }
    public Float getPrecoVenda() {
        return precoVenda;
    }
    public void setPrecoVenda(Float precoVenda) {
        this.precoVenda = precoVenda;
    }
```

```java
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idProduto != null ? idProduto.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        if (!(object instanceof Produto)) {
            return false;
        }
        Produto other = (Produto) object;
        if ((this.idProduto == null && other.idProduto != null) || (this.idProduto != null && !this.idProduto.equals(other.idProduto)))
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "cadastroee.model.Produto[ idProduto=" + idProduto + " ]";
    }
}
```

```java
package cadastroee.model;
import jakarta.persistence.Basic;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.NamedQueries;
import jakarta.persistence.NamedQuery;
import jakarta.persistence.Table;
import java.io.Serializable;

@Entity
@Table(name = "sequenciaPessoa")
@NamedQueries({
    @NamedQuery(name = "SequenciaPessoa.findAll", query = "SELECT s FROM SequenciaPessoa s"),
    @NamedQuery(name = "SequenciaPessoa.findById", query = "SELECT s FROM SequenciaPessoa s WHERE s.id = :id")})
public class SequenciaPessoa implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id")
    private Integer id;

    public SequenciaPessoa() {}

    public SequenciaPessoa(Integer id) {
        this.id = id;
    }
    public Integer getId() {
        return id;
    }
    public void setId(Integer id) {
        this.id = id;
    }
```

```java
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (id != null ? id.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        if (!(object instanceof SequenciaPessoa)) {
            return false;
        }
        SequenciaPessoa other = (SequenciaPessoa) object;
        if ((this.id == null && other.id != null) || (this.id != null && !this.id.equals(other.id))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "cadastroee.model.SequenciaPessoa[ id=" + id + " ]";
    }
}
```

```java
package cadastroee.model;
import jakarta.persistence.Basic;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.NamedQueries;
import jakarta.persistence.NamedQuery;
import jakarta.persistence.Table;
import java.io.Serializable;

@Entity
@Table(name = "usuario")
@NamedQueries({
    @NamedQuery(name = "Usuario.findAll", query = "SELECT u FROM Usuario u"),
    @NamedQuery(name = "Usuario.findByIdUsuario", query = "SELECT u FROM Usuario u WHERE u.idUsuario = :idUsuario"),
    @NamedQuery(name = "Usuario.findByLogin", query = "SELECT u FROM Usuario u WHERE u.login = :login"),
    @NamedQuery(name = "Usuario.findBySenha", query = "SELECT u FROM Usuario u WHERE u.senha = :senha")})
public class Usuario implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @Column(name = "idUsuario")
    private Integer idUsuario;
    @Basic(optional = false)
    @Column(name = "login")
    private String login;
    @Basic(optional = false)
    @Column(name = "senha")
    private String senha;

    public Usuario() {}

    public Usuario(Integer idUsuario) {
        this.idUsuario = idUsuario;
    }
    public Usuario(Integer idUsuario, String login, String senha) {
        this.idUsuario = idUsuario;
        this.login = login;
        this.senha = senha;
    }
```

```java
    public Integer getIdUsuario() {
        return idUsuario;
    }
    public void setIdUsuario(Integer idUsuario) {
        this.idUsuario = idUsuario;
    }
    public String getLogin() {
        return login;
    }
    public void setLogin(String login) {
        this.login = login;
    }
    public String getSenha() {
        return senha;
    }
    public void setSenha(String senha) {
        this.senha = senha;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (idUsuario != null ? idUsuario.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        if (!(object instanceof Usuario)) {
            return false;
        }
        Usuario other = (Usuario) object;
        if ((this.idUsuario == null && other.idUsuario != null) || (this.idUsuario != null && !this.idUsuario.equals(other.idUsuario))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "cadastroee.model.Usuario[ idUsuario=" + idUsuario + " ]";
    }
}
```

```java
package cadastroee.servlets;
import cadastroee.controller.ProdutoFacadeLocal;
import cadastroee.model.Produto;
import jakarta.ejb.EJB;
import java.io.IOException;
import java.io.PrintWriter;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.util.List;

public class ServletProduto extends HttpServlet {
    @EJB
    ProdutoFacadeLocal facade;

    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>Servlet ServletProduto at " + request.getContextPath() + "</h1>");
                List<Produto> produtos = facade.findAll();
                out.println("<ul>");
                for (Produto produto : produtos) {
                    out.println("<li>" + produto.getNome()+ "</li>");
                }
                out.println("</ul>");
                out.println("</body>");
                out.println("</html>");
            }
        }

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

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

    @Override
    public String getServletInfo() {
        return "Short description";
    }
}
```

```java
package cadastroee.servlets;
import cadastroee.controller.ProdutoFacadeLocal;
import cadastroee.model.Produto;
import jakarta.ejb.EJB;
import jakarta.servlet.RequestDispatcher;
import java.io.IOException;
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.util.List;

@WebServlet(name = "ServletProdutoFC", urlPatterns = {"/ServletProdutoFC"})
public class ServletProdutoFC extends HttpServlet {
    @EJB
    private ProdutoFacadeLocal facade;
    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String acao = request.getParameter("acao");
        String destino = null;

        if (acao == null || acao.isEmpty()) {
            acao = "listar";
        }
        try {
            switch (acao) {
                case "listar":
                    List<Produto> produtos = facade.findAll();
                    request.setAttribute("produtos", produtos);
                    destino = "ProdutoLista.jsp";
                    break;
                case "formIncluir":
                    destino = "ProdutoDados.jsp";
                    break;
```

```java
                case "formAlterar":
                    try {
                        Integer id = Integer.parseInt(request.getParameter("id"));
                        Produto produto = facade.find(id);
                        request.setAttribute("produto", produto);
                        destino = "ProdutoDados.jsp";
                    } catch (NumberFormatException e) {
                        e.printStackTrace();
                    }
                    break;
                case "excluir":
                    try {
                        Integer id = Integer.parseInt(request.getParameter("id"));
                        Produto produto = facade.find(id);
                        facade.remove(produto);
                        request.setAttribute("produtos", facade.findAll());
                        destino = "ProdutoLista.jsp";
                    } catch (NumberFormatException e) {
                        e.printStackTrace();
                    }
                    break;
                case "alterar":
                    try {
                        Integer id = Integer.parseInt(request.getParameter("id"));
                        Produto produto = facade.find(id);
                        produto.setNome(request.getParameter("nome"));
                        produto.setQuantidade(Integer.parseInt(request.getParameter("quantidade")));
                        produto.setPrecoVenda(Float.parseFloat(request.getParameter("precoVenda")));
                        facade.edit(produto);
                        request.setAttribute("produtos", facade.findAll());
                        destino = "ProdutoLista.jsp";
                    } catch (NumberFormatException e) {
                        e.printStackTrace();
                    }
                    break;
```

```java
                case "incluir":
                    Produto novoProduto = new Produto();
                    novoProduto.setNome(request.getParameter("nome"));
                    novoProduto.setQuantidade(Integer.parseInt(request.getParameter("quantidade")));
                    novoProduto.setPrecoVenda(Float.parseFloat(request.getParameter("precoVenda")));
                    facade.create(novoProduto);
                    request.setAttribute("produtos", facade.findAll());
                    destino = "ProdutoLista.jsp";
                    break;
                default:
                    request.setAttribute("produtos", facade.findAll());
                    destino = "ProdutoLista.jsp";
                    break;
            }
        } catch (Exception e) {
            throw new ServletException("Erro ao processar a ação: " + acao, e);
        }

        RequestDispatcher rd = request.getRequestDispatcher(destino);
        rd.forward(request, response);
    }

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

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        processRequest(request, response);
    }

    @Override
    public String getServletInfo() {
        return "Short description";
    }
}
```

TODO write content

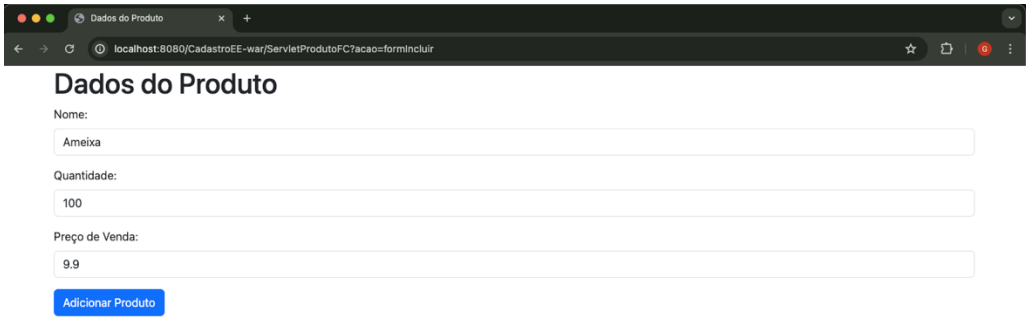
Servlet ServletProduto at /CadastroEE-war

- Laranja
- Abacate
- Bergamota
- Alface
- Tomate

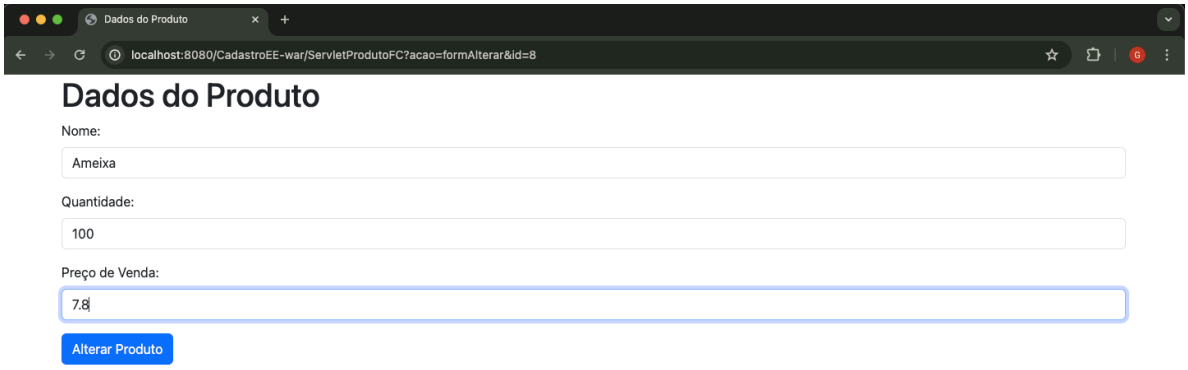
Listagem de Produtos

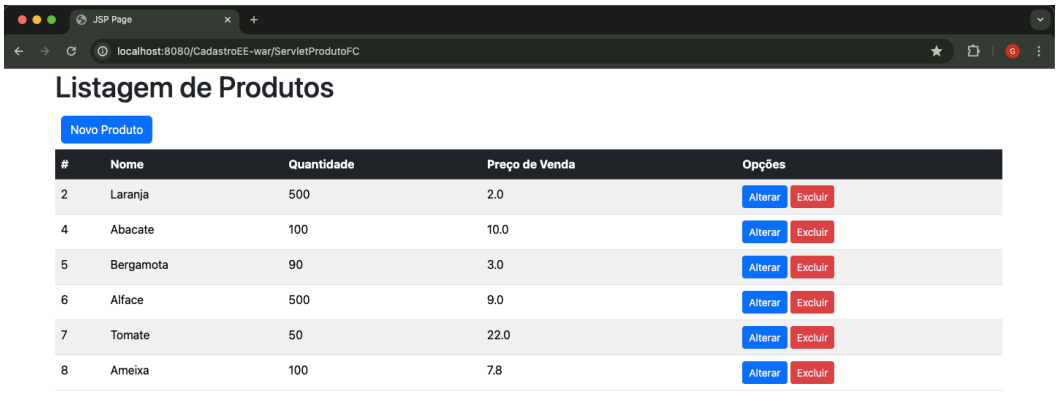
Novo Produto

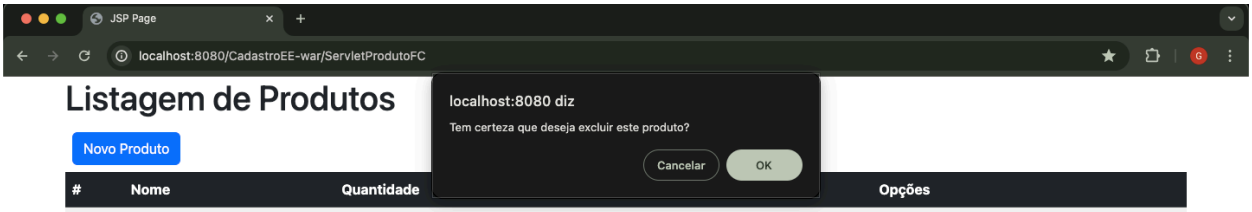
#	Nome	Quantidade	Preço de Venda	Opções
2	Laranja	500	2.0	Alterar Excluir
4	Abacate	100	10.0	Alterar Excluir
5	Bergamota	90	3.0	Alterar Excluir
6	Alface	500	9.0	Alterar Excluir
7	Tomate	50	22.0	Alterar Excluir

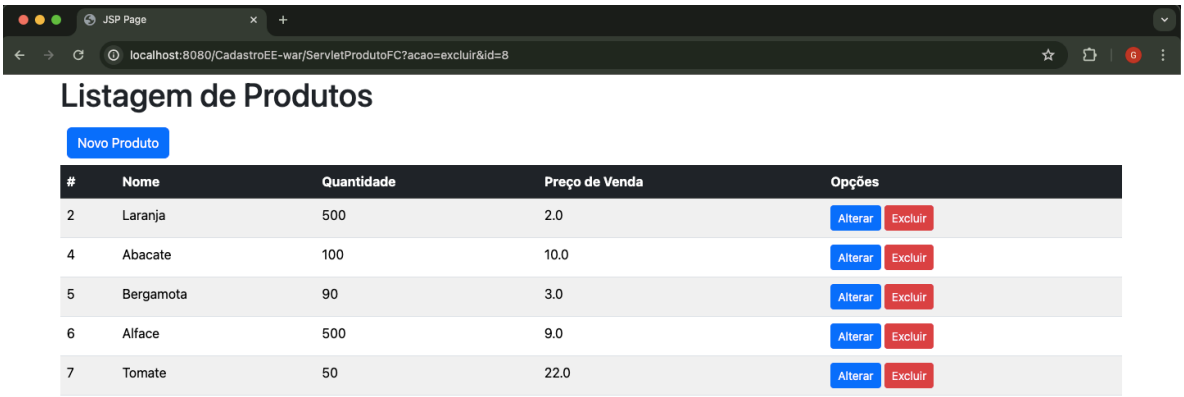

Dados do Produto
localhost:8080/CadastroEE-war/ServletProdutoFC?acao=formIncluir
Dados do Produto
Nome:
Ameixa
Quantidade:
100
Preço de Venda:
9.9
Adicionar Produto


JSP Page
localhost:8080/CadastroEE-war/ServletProdutoFC
Listagem de Produtos
Novo Produto
Nome Quantidade Preço de Venda Opções
2 Laranja 500 2.0 Alterar Excluir
4 Abacate 100 10.0 Alterar Excluir
5 Bergamota 90 3.0 Alterar Excluir
6 Alface 500 9.0 Alterar Excluir
7 Tomate 50 22.0 Alterar Excluir
8 Ameixa 100 9.9 Alterar Excluir


Dados do Produto
localhost:8080/CadastroEE-war/ServletProdutoFC?acao=formAlterar&id=8
Dados do Produto
Nome:
Ameixa
Quantidade:
100
Preço de Venda:
7.8
Alterar Produto


JSP Page
localhost:8080/CadastroEE-war/ServletProdutoFC
Listagem de Produtos
Novo Produto
Nome Quantidade Preço de Venda Opções
2 Laranja 500 2.0 Alterar Excluir
4 Abacate 100 10.0 Alterar Excluir
5 Bergamota 90 3.0 Alterar Excluir
6 Alface 500 9.0 Alterar Excluir
7 Tomate 50 22.0 Alterar Excluir
8 Ameixa 100 7.8 Alterar Excluir


JSP Page
localhost:8080/CadastroEE-war/ServletProdutoFC
Listagem de Produtos
localhost:8080 diz
Tem certeza que deseja excluir este produto?
Cancelar
OK
Novo Produto
Nome Quantidade Preço de Venda Opções
2 Laranja 500 2.0 Alterar Excluir
4 Abacate 100 10.0 Alterar Excluir
5 Bergamota 90 3.0 Alterar Excluir
6 Alface 500 9.0 Alterar Excluir
7 Tomate 50 22.0 Alterar Excluir
8 Ameixa 100 7.8 Alterar Excluir


JSP Page
localhost:8080/CadastroEE-war/ServletProdutoFC?acao=excluir&id=8
Listagem de Produtos
Novo Produto
Nome Quantidade Preço de Venda Opções
2 Laranja 500 2.0 Alterar Excluir
4 Abacate 100 10.0 Alterar Excluir
5 Bergamota 90 3.0 Alterar Excluir
6 Alface 500 9.0 Alterar Excluir
7 Tomate 50 22.0 Alterar Excluir

ANÁLISE E CONCLUSÃO

Procedimento 1

a. Como é organizado um projeto corporativo no NetBeans?

Um projeto corporativo é organizado utilizando módulos hierárquico, incluindo diferentes camadas de aplicação que geralmente é um projeto principal e outros subprojetos dependentes.

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

JPA é usado para gerenciar persistência dos dados e EJB são os componentes de servidor que encapsulam o código.

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

O NetBeans viabiliza a produtividade pois integram funcionalidades que facilitam o desenvolvimento do código, ele também simplifica o processo de teste e implantação do aplicativo. O editor de código “autocompleta”, facilitando a escrita em si.

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

Servlets são classes do Java para trabalhar com desenvolvimento web, executado em um servidor para atender requisições da aplicação.

O NetBeans, oferece um assistente para criação rápida de um Servlet, facilita a configuração de mapeamento de URL e fornece integração com servidores de aplicação.

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

A comunicação é geralmente feita por meio de injeção de dependência do EJB no Servlet ou por chamadas de métodos. Na injeção de dependência, se usa a anotação para que o Servlet obtenha uma referencia a um bean de sessão. Já pela chamada de método, a sessão bean já foi injetada e o Servlet pode chamar o método para executar a logica.

Procedimento 2

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

   Ele funciona com um único Servlet para lidar com todas as solicitações de entrada decidindo qual ação tomar ou recusar.

   Ele é implementado como controlador central, intermediando o cliente e o resto da aplicação, sendo responsável por rotear requisições para cada componente apropriado.

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

   Os dois são componentes Java usado para criação de aplicações web e são executados no servidor.

   Suas diferenças são que o Servlet são classes puras de Java que lidam com requisições HTTP e geram respostar programadas e o código pode ter alguns trechos de HTML para gerar interface ao usuário. As JSPs são paginas HTML que podem conter fragmentos de código Java embutidos.

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?

   O Redirecionamento simples usa o método sendRedirect do HttpServletResponse e envia uma nova solicitação HTTP ao cliente fazendo o navegador ir para uma nova url. O forward, por si so, usa o método forward do RequestDispatcher, encaminhando a solicitação para um JSP ou Servlet no servidor e o cliente não consegue saber que houve um encaminhamento pois o servidor trata a requisição internamente.

Procedimento 3

a. Como o framework Bootstrap é utilizado?

   Ele pode ser utilizado usando o link do CDN no próprio arquivo HTML ou baixando os arquivos diretamente do site oficial.

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

   Ele garante essa independência pois separa a apresentação e o estilo visual do conteúdo HTML.

c. Qual a relação entre o Bootstrap e a responsividade da pagina?

Sua relação sempre foi a responsividade desde que foi criado, sendo assim, sempre se adaptando a diferentes dispositivos e tamanhos de telas.