RPG0017 - Vamos integrar sistemas

Ricardo Alves dos Santos Junior - 202208681158

Universidade Estácio de Sá

I Nível 4: Vamos Integrar Sistemas – 2023.3 -Terceiro Semestre

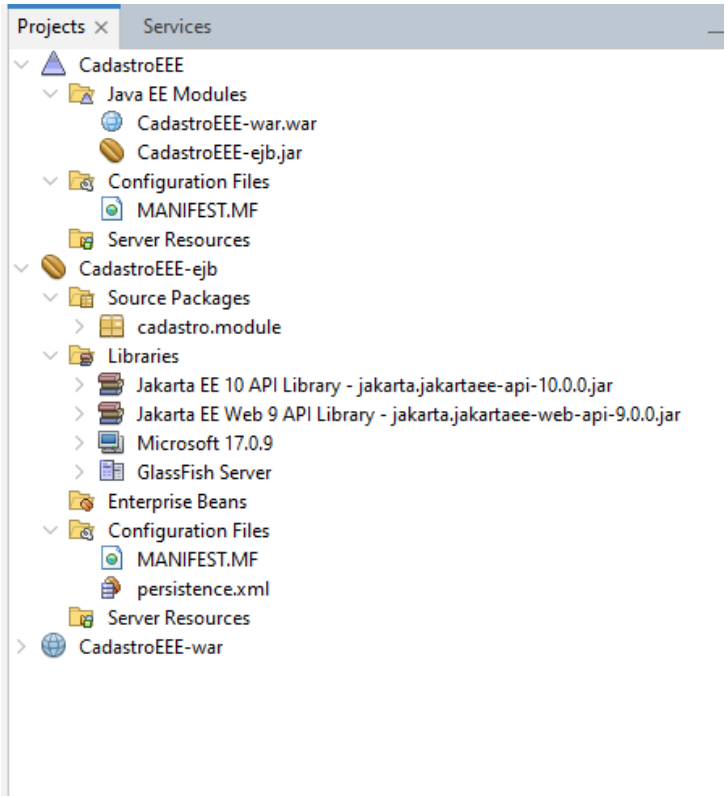
Objetivo da Prática

- 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.

Repositório no GitHub: https://github.com/RicardoASJunior/CadastroEEE

1º Procedimento | Camadas de Persistência e Controle

Criação do Projeto:



a) Como é organizado um projeto corporativo no NetBeans?

R: Um projeto corporativo no NetBeans é um tipo de projeto que permite criar aplicações Java EE (Enterprise Edition) que podem ser executadas em um servidor de aplicações, como o GlassFish, o Tomcat ou o JBoss. Um projeto corporativo consiste em um módulo EAR (Enterprise Application Archive) que contém um ou mais módulos EJB (Enterprise JavaBeans) e um ou mais módulos Web. Cada módulo pode ter suas próprias dependências, configurações e recursos. Um projeto corporativo no NetBeans pode ser criado usando um assistente que facilita a escolha dos componentes e das tecnologias a serem utilizados.

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

R: JPA (Java Persistence API) é uma especificação Java para acessar, persistir e gerenciar dados entre objetos Java e um banco de dados relacional. JPA permite mapear as entidades Java para as tabelas do banco de dados, usando anotações ou arquivos XML. JPA também oferece uma linguagem de consulta (JPQL) e uma API de critérios para realizar consultas dinâmicas. JPA facilita o desenvolvimento de aplicações que precisam manipular dados persistentes de forma padronizada e portátil.

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

R: O NetBeans oferece suporte à criação, configuração, teste e implantação de projetos que utilizam JPA e EJB. O NetBeans permite gerar as classes de entidade JPA a partir de um banco de dados existente, ou vice-versa, usando assistentes e ferramentas gráficas. O NetBeans também permite criar e testar consultas JPQL e critérios usando uma janela de consulta integrada.

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

R: Servlets são classes Java que podem tratar requisições HTTP recebidas de clientes Web, como os navegadores. Em uma Servlet, um método service recebe os objetos HttpServletRequest e HttpServletResponse, que representam a requisição e a resposta, respectivamente. Uma Servlet pode capturar os parâmetros da requisição, efetuar algum processamento e devolver uma página HTML ou outro tipo de conteúdo. Servlets são executadas em um container Web, que é responsável por gerenciar o seu ciclo de vida e invocá-las de acordo com o mapeamento definido em um arquivo web.xml.

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

R: A comunicação entre as Servlets e os Session Beans do pool de EJBs é feita através da injeção de dependências ou da localização dos EJBs usando o serviço de nomes (JNDI). A injeção de dependências é um mecanismo que permite ao container EJB injetar uma referência a um EJB em um campo ou método de uma Servlet, usando anotações como @EJB ou @Inject. A localização dos EJBs usando o JNDI é um mecanismo que permite à Servlet obter uma referência a um EJB a partir de um nome global, usando a interface InitialContext e o método lookup. Em ambos os casos, a Servlet pode invocar os métodos do EJB como se fosse um objeto local, mas na verdade a comunicação é feita de forma remota, usando protocolos como RMI ou IIOP.

2º Procedimento | Interface Cadastral com Servlet e JSPs

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

R: O padrão Front Controller é um design pattern utilizado no desenvolvimento de aplicações web que centraliza a gestão do fluxo de requisições (requests) e respostas (responses). Ele funciona como um ponto de entrada único para uma aplicação, recebendo todas as requisições e direcionando-as para os componentes apropriados para processamento.

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

R: Servlets: São classes Java que estendem a funcionalidade de um servidor web, lidando diretamente com requisições HTTP, processando-as e gerando respostas. Geralmente, são usados para implementar a lógica de controle em uma arquitetura MVC, recebendo solicitações, interagindo com os modelos e direcionando para as visualizações apropriadas.

JSPs (JavaServer Pages): São arquivos que misturam código Java e marcações HTML, permitindo a criação de páginas web dinâmicas. JSPs são usados para criar as visualizações em uma aplicação web, permitindo a inclusão de código Java embutido para acessar modelos (dados) e processar informações antes de renderizar a resposta para o cliente.

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?

R: Redirecionamento simples: É um processo em que o servidor web responde à requisição do cliente com um código de status de redirecionamento (geralmente 3xx), instruindo o navegador do cliente a fazer uma nova solicitação para outra URL. Isso geralmente envolve uma nova solicitação HTTP.

Forward utilizando o método RequestDispatcher: O forward é uma operação realizada no servidor que permite direcionar o processamento de uma requisição para outro recurso (como um Servlet ou uma JSP) no mesmo contexto de aplicação. Ele é usado para encaminhar a requisição e resposta para outro componente interno do servidor sem que o cliente seja envolvido. Isso é feito através do método forward() do RequestDispatcher, o que permite o encaminhamento da requisição atual para outro recurso no servidor.

Parâmetros: São informações enviadas do cliente para o servidor como parte da requisição HTTP. Por exemplo, quando um formulário HTML é enviado, os dados inseridos nele são enviados como parâmetros na requisição HTTP.

Atributos: São usados para armazenar informações na requisição, sessão ou contexto da aplicação no lado do servidor. Eles podem ser definidos durante a manipulação da requisição e persistidos para uso posterior durante o ciclo de vida da requisição ou sessão.

3º Procedimento | Melhorando o Design da Interface

a) Como o framework Bootstrap é utilizado?

R: O framework Bootstrap é utilizado para facilitar e acelerar o desenvolvimento de sites responsivos, que se adaptam a diferentes tamanhos de tela e dispositivos. O Bootstrap oferece uma coleção de componentes, utilitários, ícones e plugins pré-prontos em HTML, CSS e JavaScript, que podem ser personalizados e combinados de acordo com as necessidades do projeto.

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

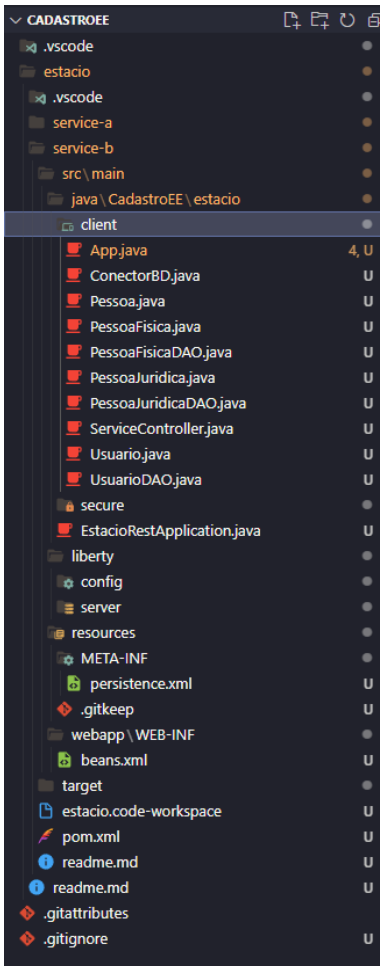
R: O Bootstrap garante a independência estrutural do HTML porque ele usa classes CSS para definir o estilo e o comportamento dos elementos HTML, sem alterar o seu conteúdo ou significado. Assim, o HTML fica mais limpo e semântico, e o CSS fica mais modular e reutilizável. Além disso, o Bootstrap usa um sistema de grid flexível, que permite organizar os elementos HTML em colunas e linhas responsivas, sem precisar de tabelas ou outras tags estruturais.

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

R: A relação entre o Bootstrap e a responsividade da página é que o Bootstrap foi criado com o conceito de mobile first, ou seja, ele prioriza a experiência do usuário em dispositivos móveis, e depois adapta o layout para telas maiores. O Bootstrap usa media queries, breakpoints e classes responsivas para ajustar o tamanho, a posição e a visibilidade dos elementos HTML de acordo com a largura da tela. O Bootstrap também oferece componentes responsivos, como menus, carrosséis, modais, etc., que se comportam de forma adequada em diferentes dispositivos.

Conclusão

Não consegui implementar o código por conta da integração com o servidor do glassfish. Ao tentar criar as classes o código estava gerando diversos erros de compilação. O Código está disponível no github, porém não está funcionando. Tentei fazer o exercício usando o vsCode:



Utilizando a extensão MicroProfile, Maven , OpenLiberty, Hibernate , Microsoft SQL e a plataforma JakartaEE. Porém tive problemas de compilação ao tentar integrar o Banco de Dados com JPA. O projeto não estava reconhecendo o arquivo persistence.xml criado:

```
Exception in thread "main" jakarta.persistence.PersistenceException: No Persistence provider for EntityManager named LojaPU
        at jakarta.persistence.Persistence.createEntityManagerFactory(Persistence.java:86)
        at jakarta.persistence.Persistence.createEntityManagerFactory(Persistence.java:55)
        at CadastroEE.estacio.client.UsuarioDAO.main(UsuarioDAO.java:15)
```

persistence.xml U

estacio > service-b > src > main > resources > META-INF > persistence.xml > xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
                                 http://xmlns.jcp.org/xml/ns/persistence/persistence_3_0.xsd"
             version="3.0">

    <persistence-unit name="LojaPU" >
        <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider> <!-- or your specific JPA provider -->
        <!-- Other configuration options, if any -->
        <properties>
            <property name="javax.persistence.jdbc.driver" value="com.microsoft.sqlserver.jdbc.SQLServerDriver" />
            <property name="javax.persistence.jdbc.url" value="jdbc:sqlserver://localhost:1433;databaseName=Loja;encrypt=true;trustServerCertificate=true;" />
            <property name="javax.persistence.jdbc.user" value="loja" />
            <property name="javax.persistence.jdbc.password" value="loja" />

            <!-- Configurações do Hibernate -->
            <property name="hibernate.dialect" value="org.hibernate.dialect.SQLServerDialect" />
            <property name="hibernate.show_sql" value="true" />
            <property name="hibernate.hbm2ddl.auto" value="update" />
        </properties>
    </persistence-unit>

</persistence>
```