	<p align="center">Universidade Estácio</p> <p align="center">Polo São Lourenço da Mata</p> <p align="center">Desenvolvimento Full Stack</p> <p align="center">Semestre 2024.1</p>	<p>Disciplina: Vamos Integrar Sistemas.</p> <p>Aluno: Manoel José</p> <p>Matrícula: 202301361117</p> <p>Turma: 2023.1</p>
---	---	---

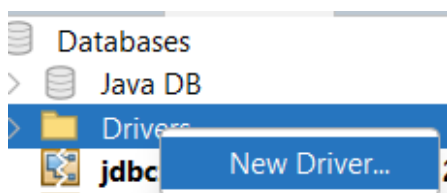
Implementação de sistema cadastral com interface Web, baseado nas tecnologias de Servlets, JPA e JEE.

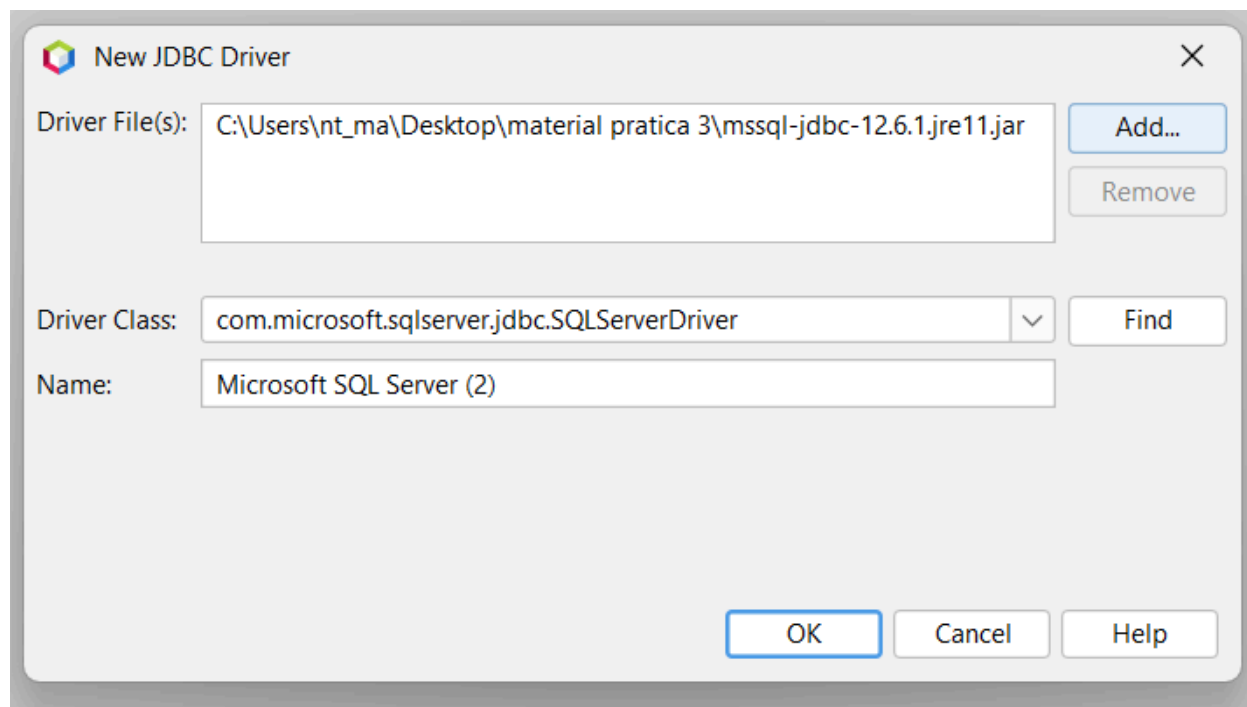
Objetivos da Prática:

1. Implementar persistência com base em JPA.
2. Implementar regras de negócio na plataforma JEE, através de EJBs.
3. Implementar sistema cadastral Web com base em Servlets e JSPs.
4. Utilizar a biblioteca Bootstrap para melhoria do design.
5. 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.

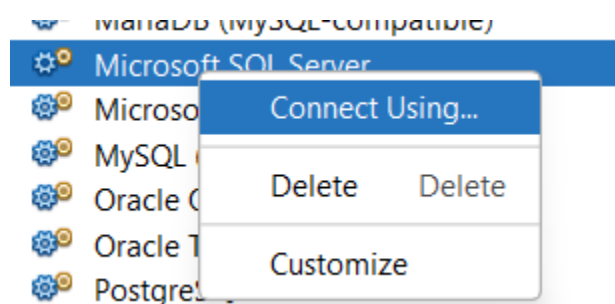
1º Procedimento | Camada de Persistência e Controle.

1- Configurando a conexão com SQL Server via NetBeans e o pool de conexões no GlassFish Server 6.2.1:





2- Definindo uma Conexão:



New Connection Wizard

Customize Connection

Driver Name: Microsoft SQL Server on Microsoft SQL Server (1) ▾

Host: localhost Port: 1433

Database: loja

Instance Name:


User Name: loja

Password: ••••

☐ Remember password

Connection Properties Test Connection

JDBC URL: :sqlserver:// localhost:1433;databaseName=loja;encrypt=true;trustServerCertificate=true;

 Connection Succeeded.

< Back Next > Finish Cancel Help

3- Configurando o GlassFish:


Add Server Instance

×

Steps

1. Choose Server

2. ...



Choose Server

Server:

Apache Tomcat or TomEE

GlassFish Server

Payara Server

WildFly Application Server

Name:

GlassFish Server

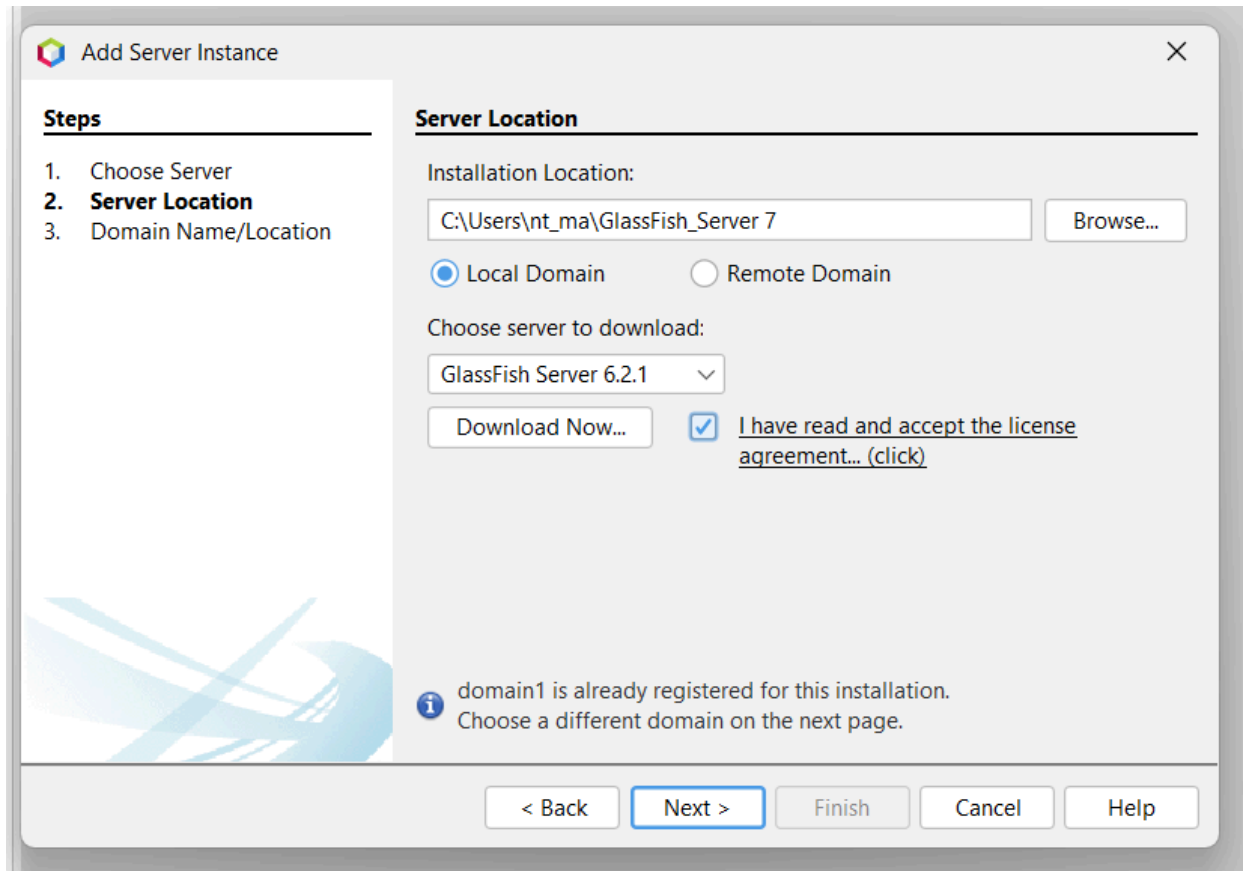
< Back

Next >

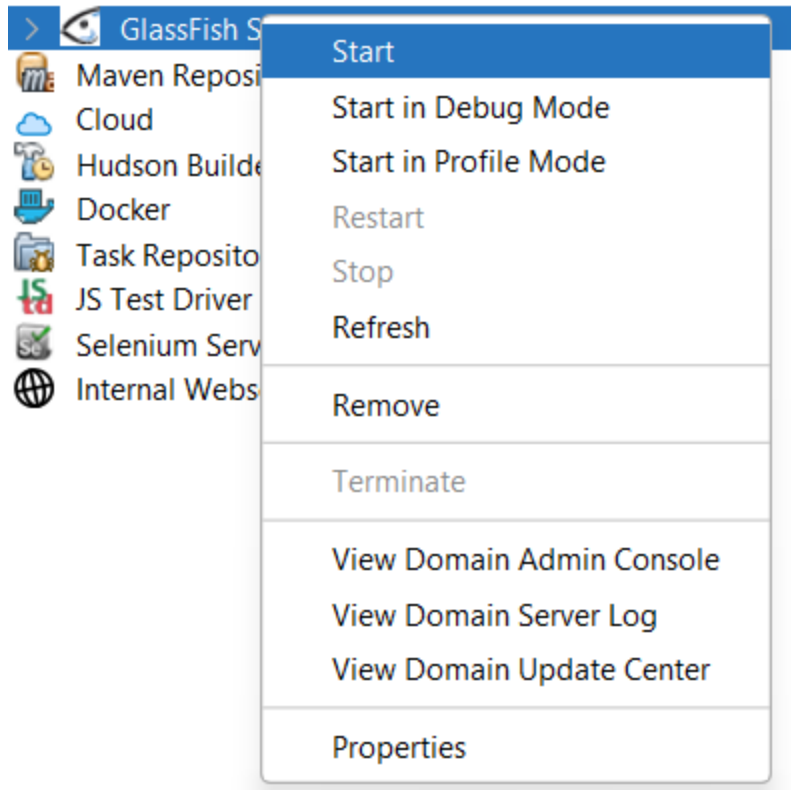
Finish

Cancel

Help



4- Iniciando o servidor do GlassFish a partir do NetBeans:



5- Através da linha de comando, executar o comando `asadmin`, no diretório `bin` do GlassFish e logo em seguida identificando o pool (`SQLServerPool`):

```
asadmin> create-jdbc-connection-pool --datasourceclassname com.microsoft.sqlserver.jdbc.SQLServerDataSource --restype javax.sql.DataSource --property driverClass=com.microsoft.sqlserver.jdbc.SQLServerDriver:portNumber=1433;password=loja:user=loja:serverName=localhost:databaseName=loja;trustServerCertificate=true;URL="jdbc:sqlserver://localhost:1433;databaseName=loja;encrypt=true;trustServerCertificate=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.
```

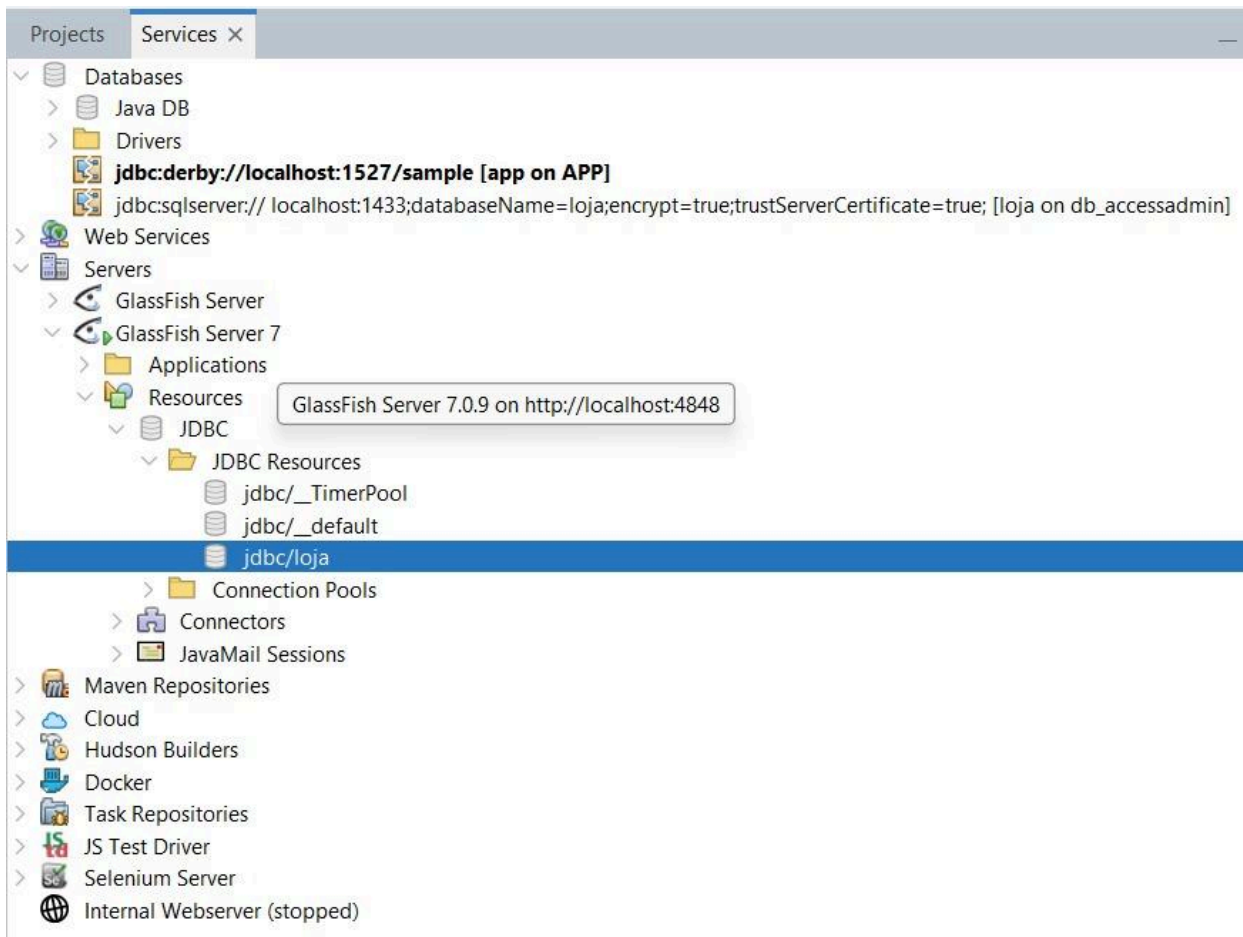
6- Testando o pool de conexões:

```
C:\Users\nt_ma\GlassFish_Server 7\glassfish\bin>asadmin
Use "exit" to exit and "help" for online help.
asadmin> ping-connection-pool SQLServerPool
CLI031: Warning: Option "target" is obsolete and will be ignored.
CLI031: Warning: Option "target" is obsolete and will be ignored.
Command ping-connection-pool executed successfully.
asadmin> |
```

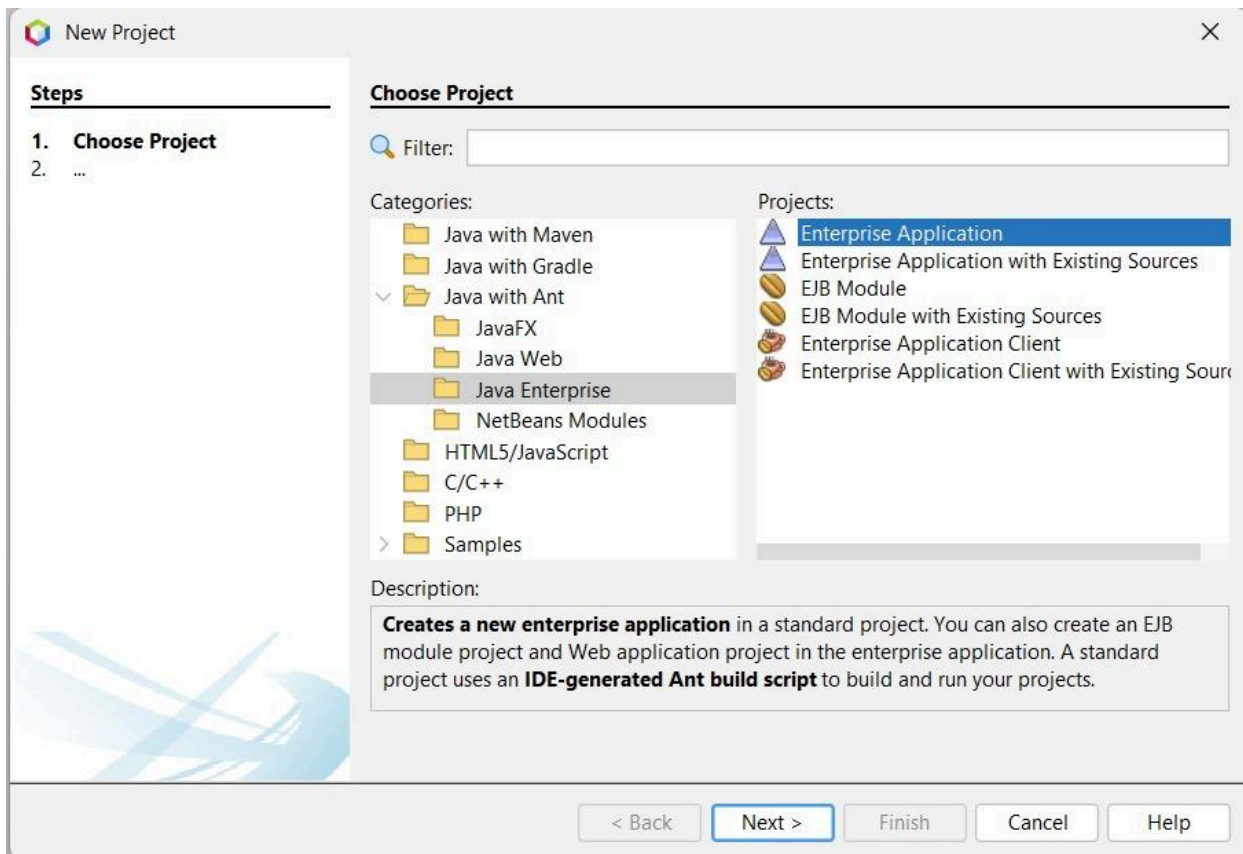
7- Criando o registro JNDI:

```
C:\Users\nt_ma\GlassFish_Server 7\glassfish\bin>asadmin
Use "exit" to exit and "help" for online help.
asadmin> create-jdbc-resource --connectionpoolid SQLServerPool jdbc/loja
JDBC resource jdbc/loja created successfully.
Command create-jdbc-resource executed successfully.
```

8- Configurações realizadas:



9- Criar o aplicativo corporativo no NetBeans:



**Steps**

1. Choose Project
- 2. Name and Location**
3. Server and Settings

Name and LocationProject Name: Project Location: Project Folder: ☐ Use Dedicated Folder for Storing LibrariesLibraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

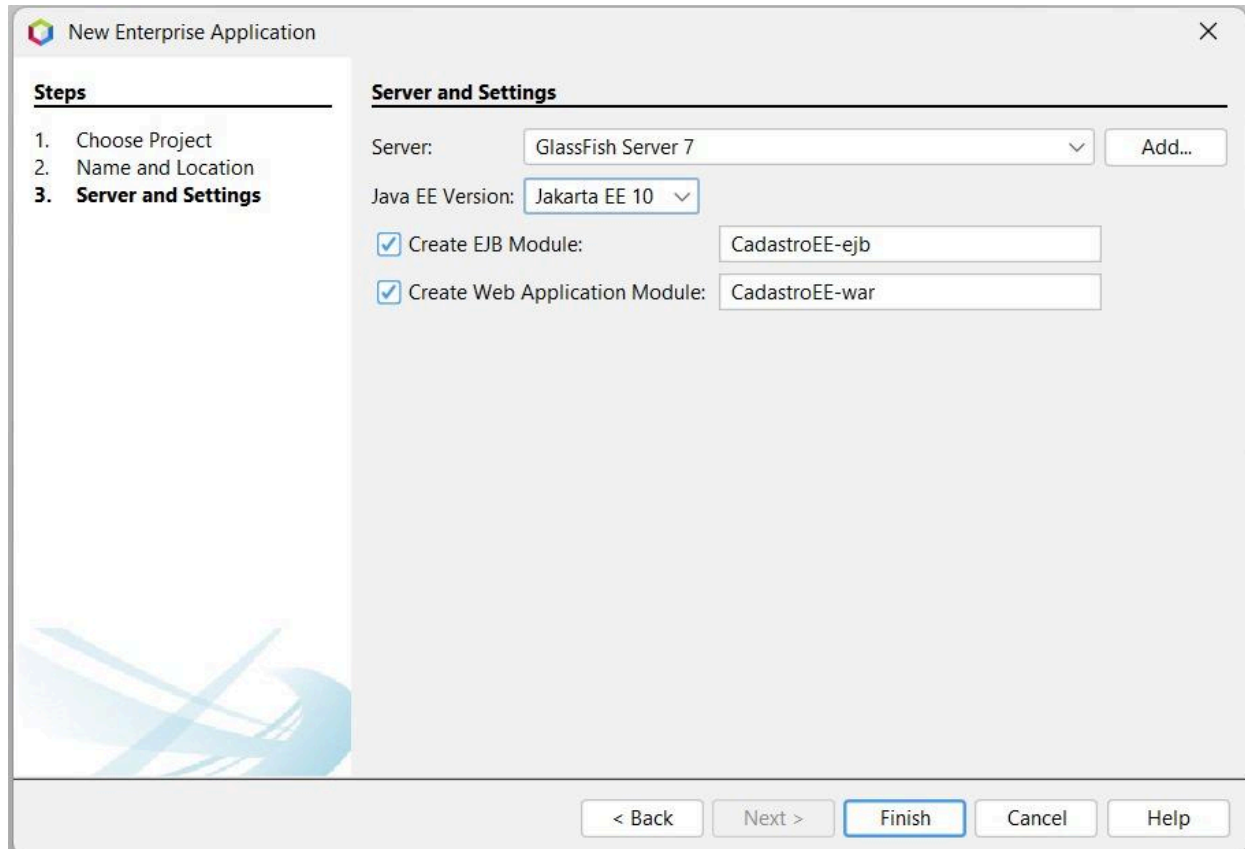
< Back

Next >

Finish

Cancel

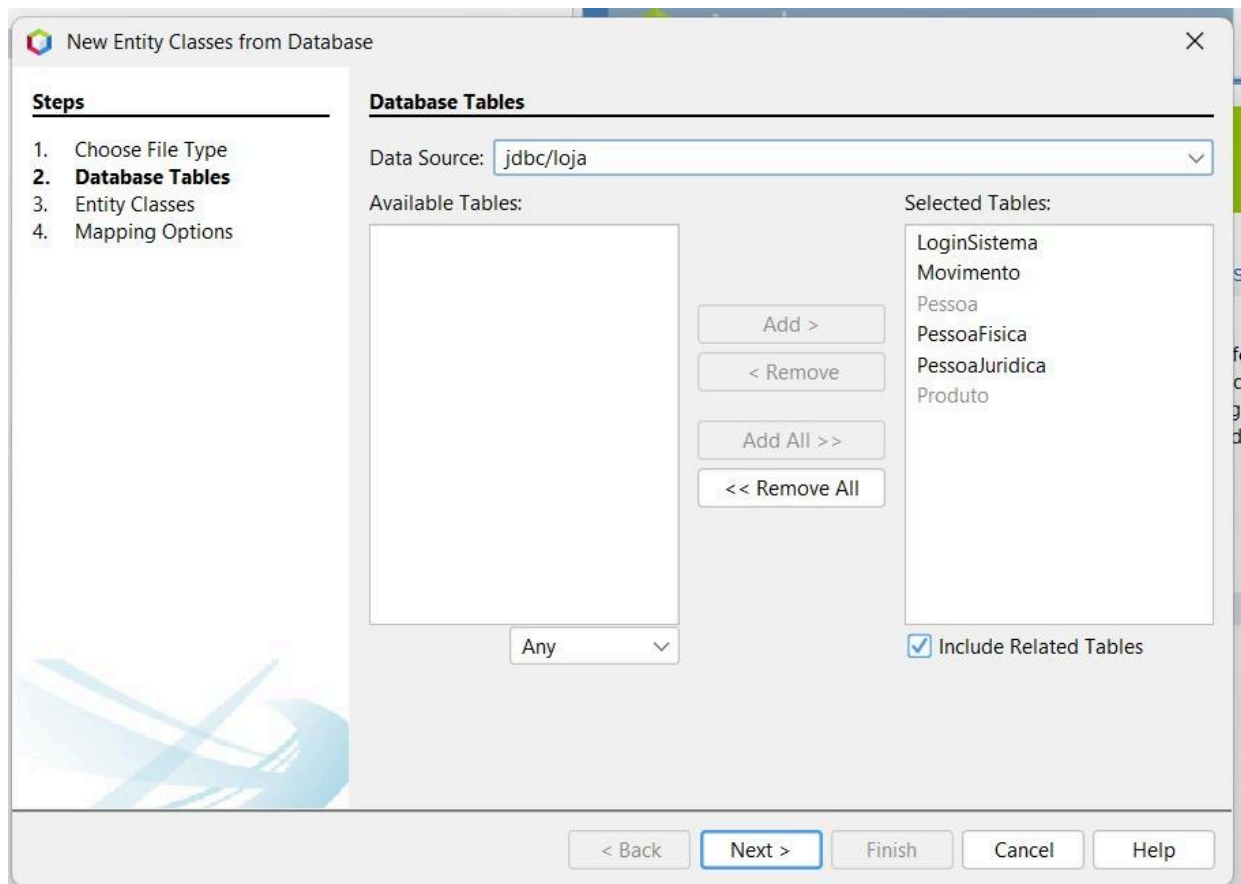
Help



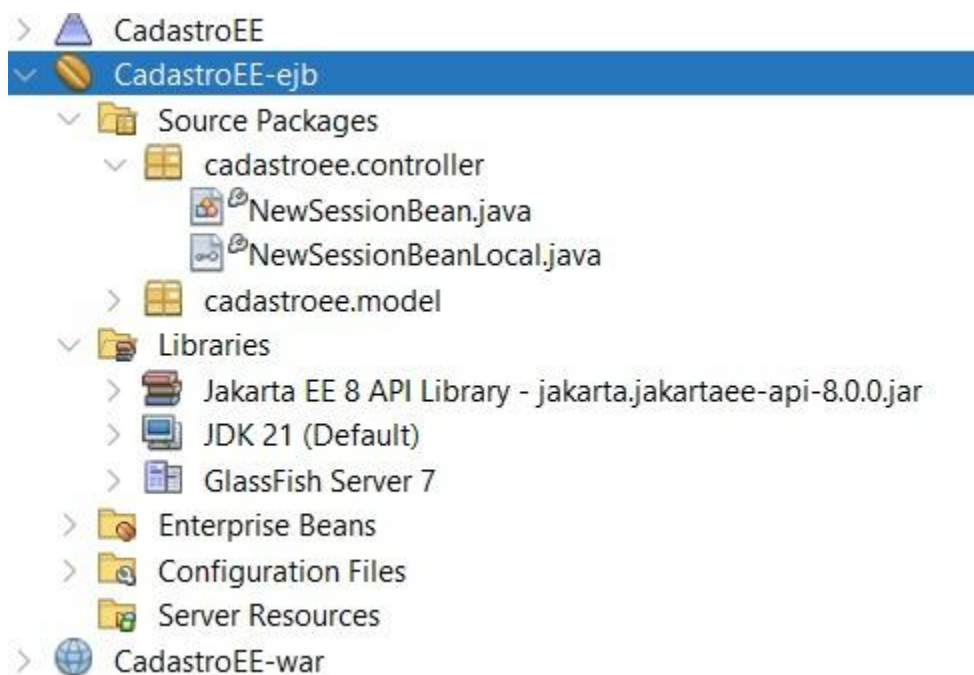
10- Os três projetos gerados:

- >  CadastroEE
- >  CadastroEE-ejb
- >  CadastroEE-war

11- Definindo as camadas de persistência e controle no projeto CadastroEE-ejb:



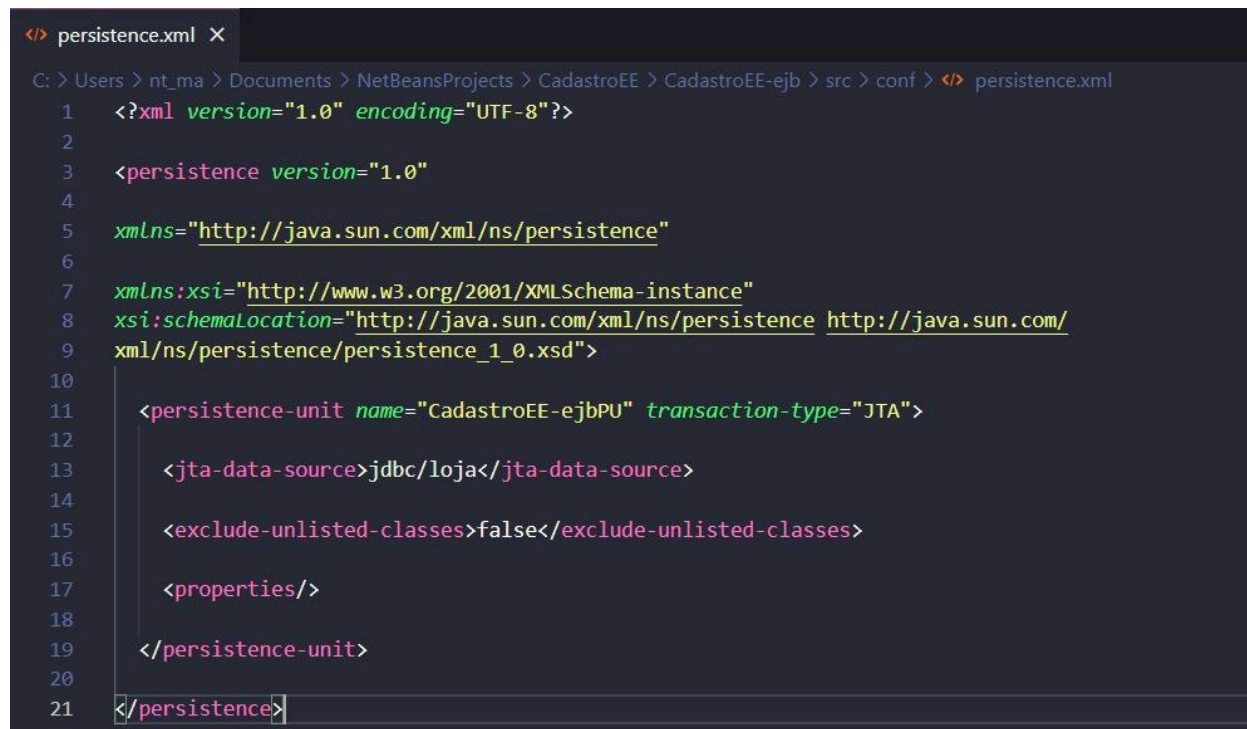
12– Componentes criados e bibliotecas ajustadas:



13- Modificando as importações de pacotes *javax* para *jakarta*, em todos os arquivos do projeto CadastroEE-ejb:

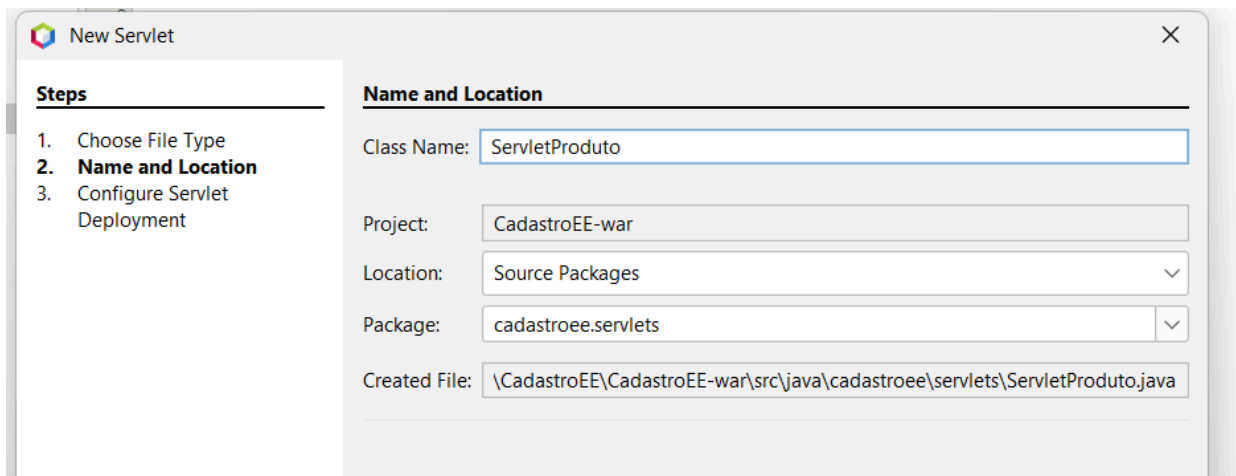
```
import jakarta.ejb.Stateless;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import java.util.List;
```

16- Modificando o arquivo persistence.xml:

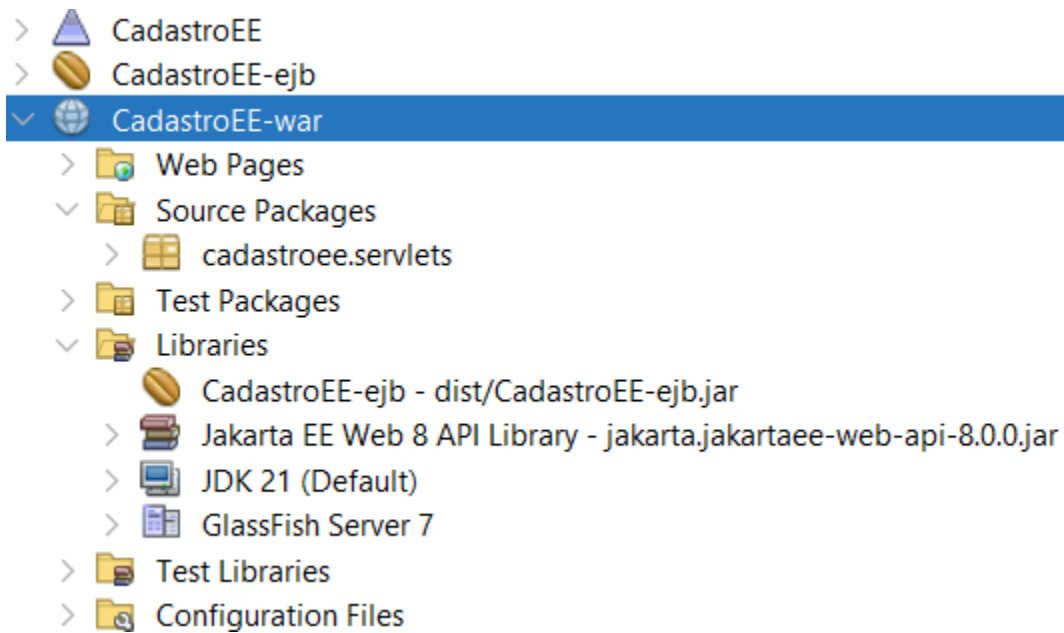


```
</> persistence.xml X
C: > Users > nt_ma > Documents > NetBeansProjects > CadastroEE > CadastroEE-ejb > src > conf > </> persistence.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <persistence version="1.0"
4
5  xmlns="http://java.sun.com/xml/ns/persistence"
6
7  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
8  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence http://java.sun.com/
9  xml/ns/persistence/persistence_1_0.xsd">
10
11  <persistence-unit name="CadastroEE-ejbPU" transaction-type="JTA">
12
13    <jta-data-source>jdbc/loja</jta-data-source>
14
15    <exclude-unlisted-classes>>false</exclude-unlisted-classes>
16
17    <properties/>
18
19  </persistence-unit>
20
21 </persistence>
```

14- Criando um Servlet de teste no projeto CadastroEE-war:



15- Criado o Servlet e ajustadas as bibliotecas:



16- Modificando o arquivo web.xml:

```
</> web.xml X
C: > Users > nt_ma > Documents > NetBeansProjects > CadastroEE > CadastroEE-war > web > WEB-INF > </> web.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <web-app version="4.0" xmlns="http://xmlns.jcp.org/xml/ns/javaee"
4
5  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/
7  javaee/web-app_4_0.xsd">
8
9  <servlet>
10
11    <servlet-name>ServletProduto</servlet-name>
12
13    <servlet-class>cadastroee.servlets.ServletProduto</servlet-class>
14
15  </servlet>
16
17  <servlet>
18
19    <servlet-name>ServletProdutoFC</servlet-name>
20
21    <servlet-class>cadastroee.servlets.ServletProdutoFC
22
23    </servlet-class>
24
25  </servlet>
26
27  <session-config>
28
29    <session-timeout>30</session-timeout>
30
31  </session-config>
32
33  </web-app>
```

17- Executar o projeto:

Análise e Conclusão:

A. Como é organizado um projeto corporativo no NetBeans?

R- A organização de um projeto corporativo no NetBeans envolve várias etapas, desde a criação do projeto até a configuração e implementação de recursos corporativos. Aqui estão os principais passos de forma resumida:

Criação do Projeto:

Novo Projeto: Vá para File > New Project.

Selecionar Tipo: Escolha Java EE ou outro tipo adequado (por exemplo, Web Application).

Nome e Localização: Defina o nome do projeto e a localização onde ele será salvo.

Configuração do Servidor:

Escolher Servidor: Durante a criação do projeto, escolha um servidor de aplicação como GlassFish ou Apache Tomcat.

Configuração: Configure as conexões e propriedades do servidor conforme necessário.

Estrutura de Pastas:

Src (Source Packages): Contém os pacotes e classes Java do projeto.

Web Pages: Diretório para páginas JSP, HTML, CSS, JavaScript e outros recursos web.

Libraries: Inclui bibliotecas externas e dependências do projeto.

Configuration Files: Contém arquivos de configuração como web.xml, persistence.xml, entre outros.

Dependências e Bibliotecas:

Adicionar JARs: Use Libraries para adicionar dependências externas ao projeto.

Maven/Gradle: Caso utilize Maven ou Gradle, configure o pom.xml ou build.gradle para gerenciar dependências.

Configuração de Banco de Dados:

Data Source: Configure a conexão com o banco de dados através do Persistence.xml ou diretamente no servidor de aplicação.

JPA (Java Persistence API): Utilize JPA para mapeamento objeto-relacional, configurando entidades e persistência.

Desenvolvimento de Código:

Java Classes: Implemente a lógica de negócio nas classes Java dentro do diretório src.

Serviços Web: Configure e implemente serviços RESTful ou SOAP se necessário.

Build e Deploy:

Build: Utilize as opções de build para compilar o projeto (Build > Build Main Project).

Deploy: Implemente o projeto no servidor configurado (Run > Run Main Project).

Testes e Debugging:

JUnit/TestNG: Configure testes unitários com JUnit ou TestNG.

Debugging: Utilize as ferramentas de debug do NetBeans para depurar e corrigir o código.

Integração Contínua:

CI/CD: Configure integração contínua e deploy contínuo utilizando ferramentas como Jenkins, Git, etc.

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

R- Na construção de um aplicativo para a plataforma Web no ambiente Java, JPA (Java Persistence API) e EJB (Enterprise JavaBeans) desempenham papéis essenciais e complementares. JPA é crucial para o gerenciamento da persistência de dados, facilitando a interação com bancos de dados relacionais por meio do mapeamento objeto-relacional (ORM). As entidades JPA representam tabelas do banco de dados como classes Java, permitindo que os desenvolvedores realizem operações de CRUD (Create, Read, Update, Delete) de forma transparente utilizando a EntityManager. JPA também oferece a Java Persistence Query Language (JPQL) para consultas sofisticadas e suporta transações para garantir que operações de persistência sejam atômicas e consistentes. Além disso, JPA inclui mecanismos de caching para melhorar a performance e reduzir o número de acessos ao banco de dados.

Por outro lado, EJBs são componentes essenciais para a implementação da lógica de negócios e serviços empresariais. Eles encapsulam a lógica de negócios em componentes distribuídos, escaláveis e reutilizáveis, facilitando a implementação de funcionalidades complexas. Existem diferentes tipos de EJBs: Session Beans, que podem ser Stateful (mantêm estado entre as chamadas) ou Stateless (não mantêm estado entre as chamadas), usados principalmente para a lógica de negócios, e Message-Driven Beans, usados para processamento assíncrono de mensagens, geralmente integrados com sistemas de mensagens como JMS (Java Message Service). EJBs oferecem gerenciamento automático de transações, simplificando a coordenação de operações de banco de dados e outras operações de negócio, garantindo consistência e integridade.

Portanto, enquanto JPA trata da persistência de dados, permitindo um acesso eficiente e transparente ao banco de dados, EJBs lidam com a lógica de negócios e a coordenação de transações, tornando a aplicação robusta e escalável. Juntas, essas tecnologias permitem a construção de aplicativos corporativos complexos e eficientes no ambiente Java.

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

R- O NetBeans melhora a produtividade ao lidar com tecnologias JPA e EJB através de várias funcionalidades integradas. Ele oferece assistentes de código e templates que geram automaticamente o código boilerplate necessário para JPA e EJB, como entidades JPA e beans EJB, facilitando a configuração inicial e reduzindo erros.

O suporte integrado a Maven e Gradle simplifica a gestão de dependências e a configuração de projetos complexos, incluindo a configuração automática de arquivos

como *persistence.xml*. Ferramentas visuais, como o editor gráfico de mapeamento de entidades JPA, permitem criar e modificar mapeamentos de banco de dados de maneira intuitiva, enquanto o assistente de criação de EJBs ajuda a configurar session beans e message-driven beans com facilidade.

Além disso, o NetBeans oferece um ambiente de depuração robusto que facilita a identificação e correção de problemas em código JPA e EJB, incluindo a capacidade de depurar transações e sessões distribuídas. A integração com servidores de aplicação como GlassFish e Tomcat permite a implantação e teste direto do ambiente de desenvolvimento, agilizando o ciclo de desenvolvimento e teste.

Com esses recursos, o NetBeans torna o desenvolvimento com JPA e EJB mais eficiente, reduzindo o tempo de configuração e permitindo aos desenvolvedores focar mais na lógica de negócios e menos em detalhes de infraestrutura.

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

R- Servlets são componentes Java que executam no lado do servidor e são usados para criar aplicações web dinâmicas, processando requisições de clientes e gerando respostas, como HTML, XML ou JSON. No NetBeans, você pode criar um projeto web com facilidade através do menu "File > New Project", escolhendo a categoria "Java Web" e o tipo "Web Application". O assistente guia você pela configuração inicial do projeto, incluindo a configuração do servidor de aplicação e a estrutura do projeto.

Para criar um Servlet, o NetBeans fornece um assistente que pode ser acessado clicando com o botão direito no projeto e selecionando "New > Servlet". Esse assistente permite configurar o nome do Servlet, o pacote e o URL de mapeamento. Ao criar um Servlet, o NetBeans gera automaticamente o código boilerplate necessário, incluindo a extensão da classe `HttpServlet` e a substituição dos métodos `doGet` e `doPost`.

O editor de código do NetBeans oferece recursos como realce de sintaxe, auto-completar e verificação de erros em tempo real, o que facilita a escrita e a manutenção do código dos Servlets. Além disso, o NetBeans integra-se com servidores de aplicação como Apache Tomcat e GlassFish, permitindo que você implemente e teste seus Servlets diretamente do ambiente de desenvolvimento, agilizando o ciclo de desenvolvimento e depuração.

Esses recursos combinados tornam o desenvolvimento de Servlets no NetBeans mais eficiente e produtivo, permitindo que os desenvolvedores se concentrem na lógica de negócios em vez de se preocuparem com detalhes de infraestrutura.

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

R- A comunicação entre Servlets e Session Beans do pool de EJBs é feita através de injeção de dependência. No Servlet, você usa a anotação `@EJB` para injetar a referência ao Session Bean. Isso permite que o Servlet chame métodos no Session Bean como se fosse qualquer outro objeto Java, facilitando a integração e a lógica de negócios distribuída. O contêiner de EJB cuida de fornecer a instância apropriada do Session Bean, gerenciando o ciclo de vida e as transações automaticamente.