UNIVERSIDADE FEDERAL DO CEARÁ

11, Julho 2025

Sistemas Distribuídos - 01A - 2025.1

Lab. EJB (Enterprise JavaBeans)

Monalisa Silva Bezerra - 535614

Professor: Dr. Antonio Rafael Braga

1. Objetivo

O laboratório teve como objetivo criar um projeto Java com EJB utilizando a IDE VS Code, o Maven como gerenciador de dependências e o deploy de um servidor de aplicação compatível com EJB, como o WildFLy. O sistema desenvolvido inclui um EJB simples e um servlet que invoca via injeção de dependência.

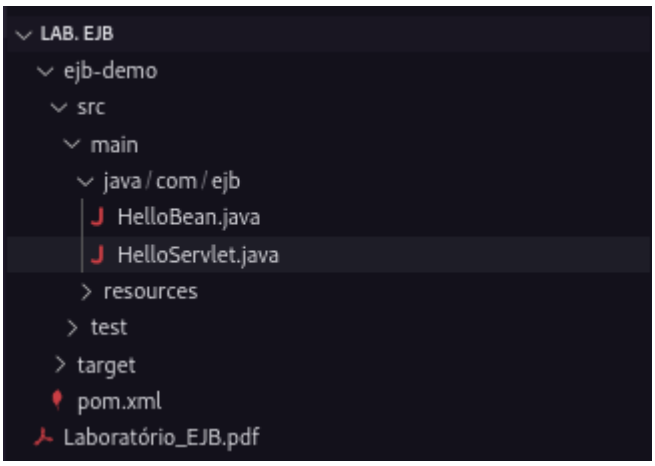
2. Etapas Realizadas

1. Instalação do Java.

2. Instalação do Maven.

3. Criação do projeto no VS Code:

- Utilizou-se o comando Java: Create Java Project.
- Foi escolhido o tipo Maven.
- Definiu-se o groupId como com.ejb e o artifactId como ejb-demo.



4. Configuração do pom.xml

O arquivo `pom.xml` foi editado para incluir a dependência da API de EJB:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
                             http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <!-- Coordenadas do projeto -->
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.ejb</groupId>
    <artifactId>ejb-demo</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>war</packaging>

    <name>EJB + Servlet Demo</name>
    <description>Projeto de exemplo com EJB Stateless e Servlet</description>

    <properties>
        <!-- Ajuste para a sua versão de Java -->
        <maven.compiler.source>17</maven.compiler.source>
        <maven.compiler.target>17</maven.compiler.target>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <dependencies>
        <!-- API de Servlet (é fornecida pelo container no runtime) -->
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>javax.servlet-api</artifactId>
            <version>4.0.1</version>
            <scope>provided</scope>
        </dependency>

        <!-- API de EJB (também fornecida pelo container) -->
        <dependency>
            <groupId>javax.ejb</groupId>
            <artifactId>javax.ejb-api</artifactId>
            <version>3.2</version>
            <scope>provided</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <!-- Compilador Java -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.8.1</version>
                <configuration>
                    <source>${maven.compiler.source}</source>
                    <target>${maven.compiler.target}</target>
                </configuration>
            </plugin>

            <!-- Gera um WAR pronto para deploy -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-war-plugin</artifactId>
                <version>3.3.2</version>
                <configuration>
                    <failOnMissingWebXml>false</failOnMissingWebXml>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

5. Criação do EJB

Foi criado um bean chamado `HelloBean` anotado com `@Stateless`, contendo um método simples que retorna uma frase:

```java
package com.ejb;

import javax.ejb.Stateless;

@Stateless
public class HelloBean  {
    public String sayHello(String name) {
        return "Olá " + name + ", este é um EJB!";
    }
}
```

6. Criação do Servlet

Um servlet chamado `HelloServlet` foi implementado com anotação `@WebServlet`, e o EJB `HelloBean` foi injetado nele com `@EJB`. O servlet responde a requisições GET retornando o resultado do EJB:

```java
package com.ejb;

import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import java.io.IOException;

@WebServlet("/hello")
public class HelloServlet extends HttpServlet {
    @EJB
    HelloBean helloBean;

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        response.setContentType("text/plain");
        response.getWriter().write(helloBean.sayHello("usuário"));
    }

}
```

7. Empacotamento e Deploy

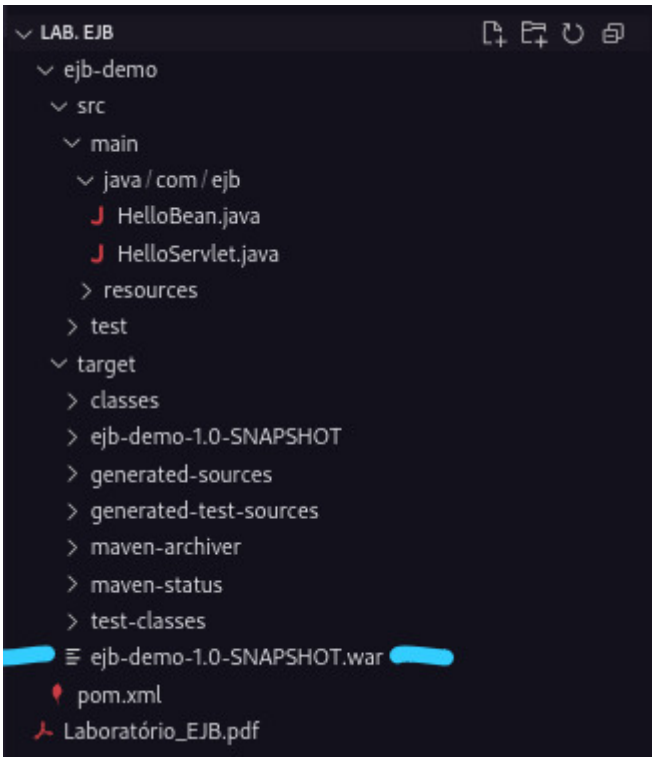
O projeto foi empacotado com o comando:

mvn clean install

```
/ejb-demo$ mvn clean install

[INFO]
[INFO] -------------------------< com.ejb:ejb-demo >-------------------------
[INFO] Building EJB + Servlet Demo 1.0-SNAPSHOT
[INFO] --------------------------------[ war ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ ejb-demo ---
[INFO] Deleting                                                  /ejb-demo/target
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ ejb-demo ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ ejb-demo ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to                                  'ejb-demo/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ ejb-demo ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory                                  /ejb-demo/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ ejb-demo ---
[INFO] Changes detected - recompiling the module!
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ ejb-demo ---
[INFO]
[INFO] --- maven-war-plugin:3.3.2:war (default-war) @ ejb-demo ---
[INFO] Packaging webapp
[INFO] Assembling webapp [ejb-demo] in                                  /ejb-demo/target/ejb-demo-1.0-SNAPSHOT]
[INFO] Processing war project
[INFO] Building war:                                          /ejb-demo/target/ejb-demo-1.0-SNAPSHOT.war
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ ejb-demo ---
[INFO] Installing                                             /ejb-demo/target/ejb-demo-1.0-SNAPSHOT.war to /home/monalisa/.m2/repository/com/ejb/ejb-demo/1.0-SNAPSHOT/ejb-demo-1.0-SNAPSHOT.war
[INFO] Installing                                             /ejb-demo/pom.xml to /home/monalisa/.m2/repository/com/ejb/ejb-demo/1.0-SNAPSHOT/ejb-demo-1.0-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.640 s
[INFO] Finished at: 2025-07-08T15:40:12-03:00
[INFO] ------------------------------------------------------------------------
```

Isso gerou um arquivo .war dentro da pasta `target`.



O `.war` foi copiado para a pasta de deploy do servidor de aplicação (como o WildFly), que fez o deploy automático do aplicativo.

8. Testes

- A aplicação foi testada acessando a URL: `http://localhost:8080/ejb-demo/hello`
- O retorno foi: Ola usuário, este é um EJB!

3. Conclusão e Resultados Obtidos

- O projeto EJB foi criado com sucesso utilizando Maven no VS Code.
- A lógica do EJB foi corretamente exposta por meio de um servlet.
- A aplicação foi empacotada e implantada com sucesso em um servidor de aplicação compatível.
- A resposta esperada foi obtida ao acessar a URL, confirmando o funcionamento correto da injeção do EJB e do servlet.