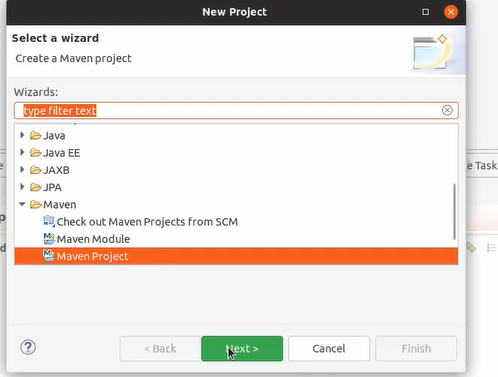
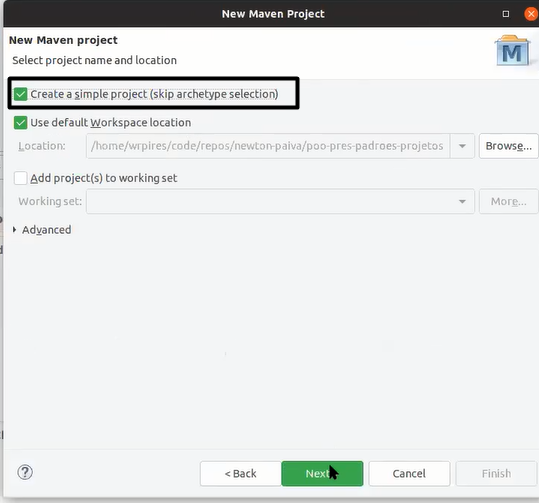
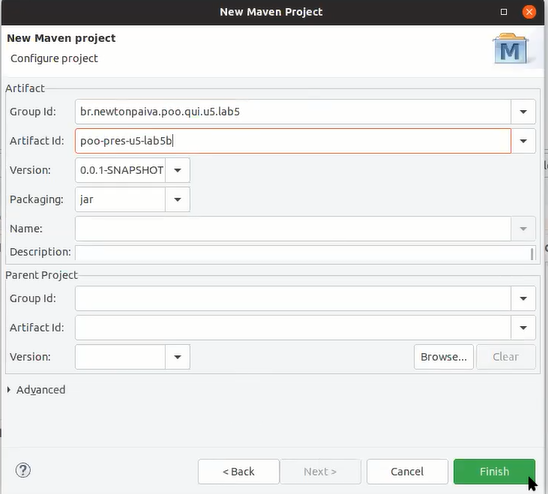
Feito no eclipse:

1- Criar um projeto Maven a gente consegue gerenciar as dependências pro nosso projeto

de uma forma mais simplificada

Criando um artefato simples

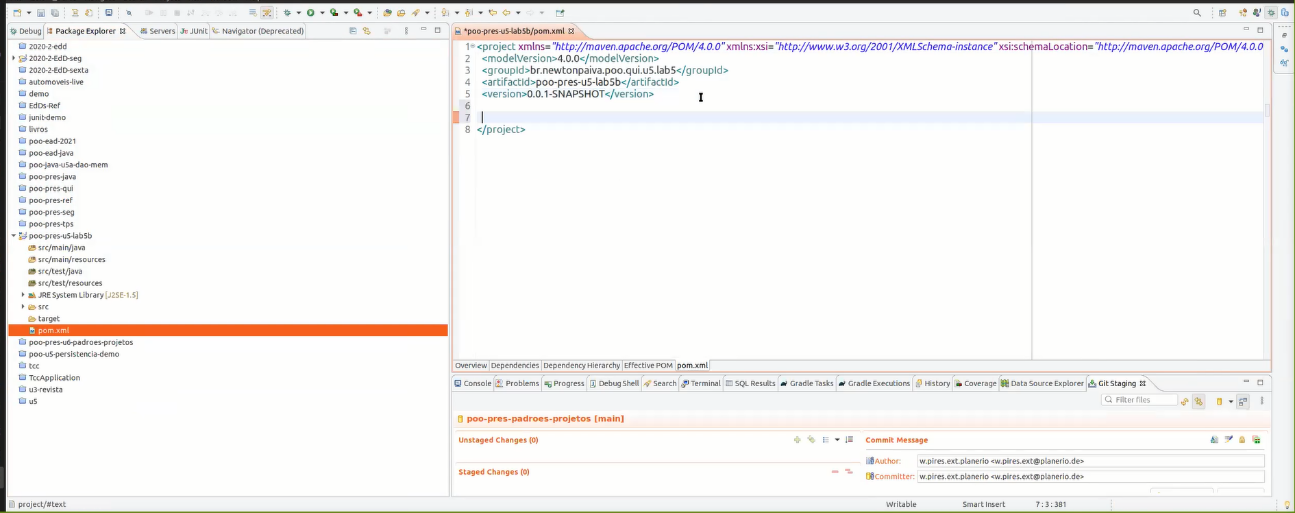




Projeto Maven criado esta vazio, o arquivo importante para nos é o Pom.xml



Esse arquivo Xml que vai nos ajudar a colocar todas as dependências que o nosso projeto vai precisar



**As bibliotecas/Dependencias que a gente precisa pra esse lab5b funcionar:**

Precisa do Hibernet 🡪Ele é uma biblioteca de mapeamento objeto relacional e o hibernet foi feito para trabalhar com diversos Banco de dados, so que cada banco de dados tem o seu protocolo específico pra conectar, então para o hibernet conversar com o MySql Server ele precisa de uma biblioteca pra conectar com o MySql Server que justamente é o Connector

Conector do MySql 🡪Connector/J ou jar ou JDBC 🡪 JDBC Driver for MySQL (Connector/J)

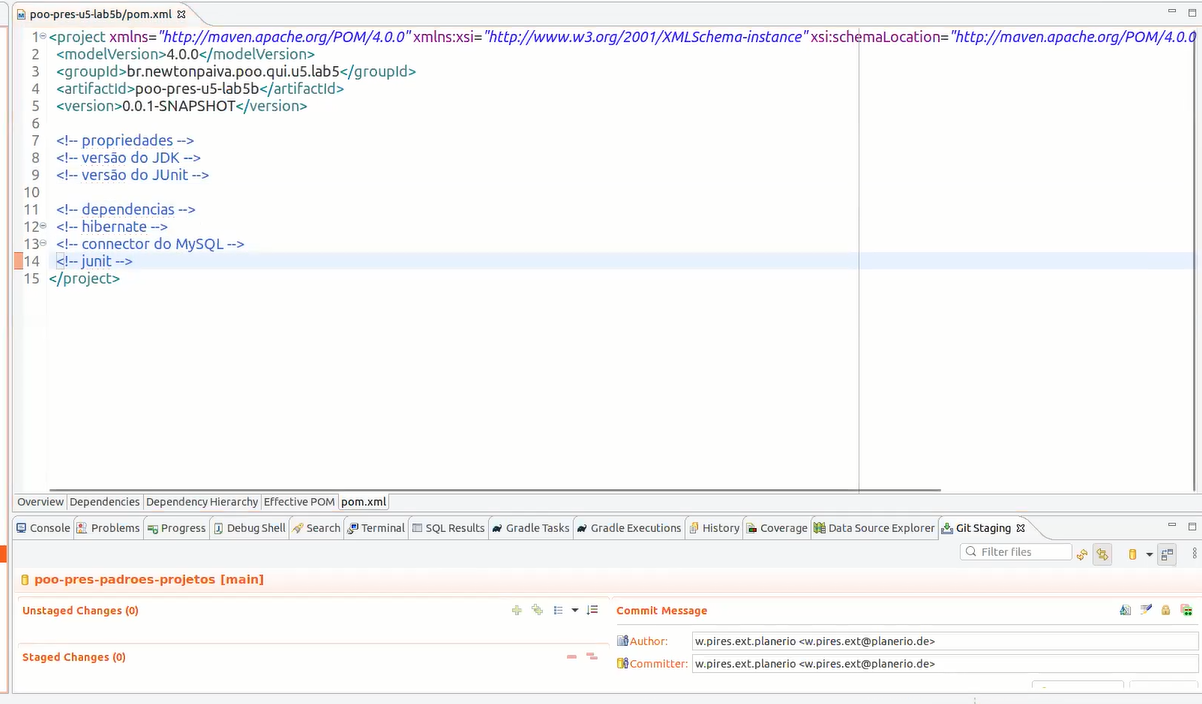
Junit 🡪 implementar teste unitário

**Algumas propriedades:**

Versão do [Java SE - JDK](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwi2lJ2qnfDwAhXyL7kGHQ_mANAQFjAAegQIAxAD&url=https%3A%2F%2Fwww.oracle.com%2Fbr%2Fjava%2Ftechnologies%2Fjavase-downloads.html&usg=AOvVaw0lJ0lVq2-JsoNGLJJfpL_N)

Versão do JUnit pra definir qual JUnit utilizar

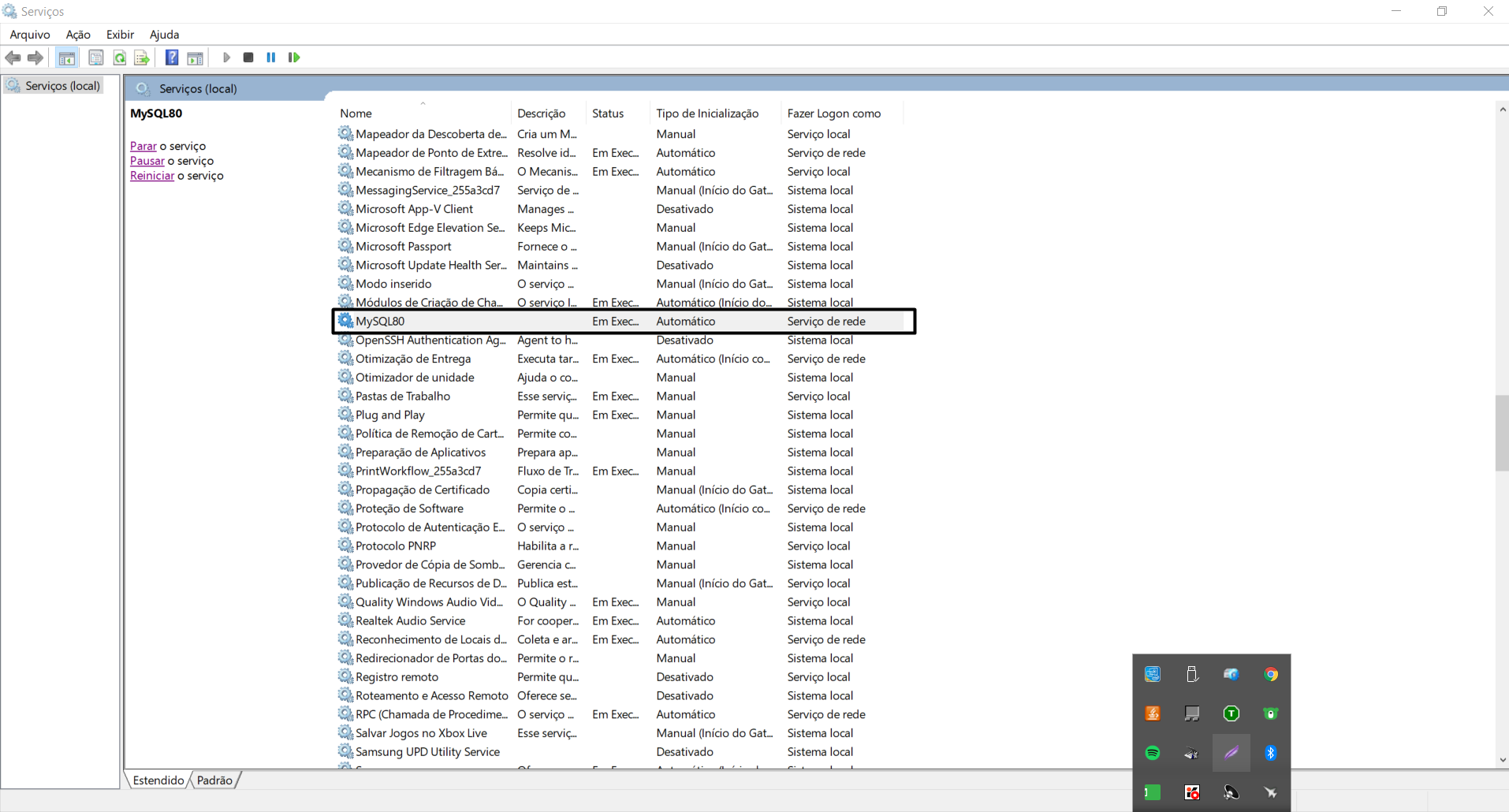
Então esse arquivo XML vai servir para nos ajudar a gerenciar essas dependências



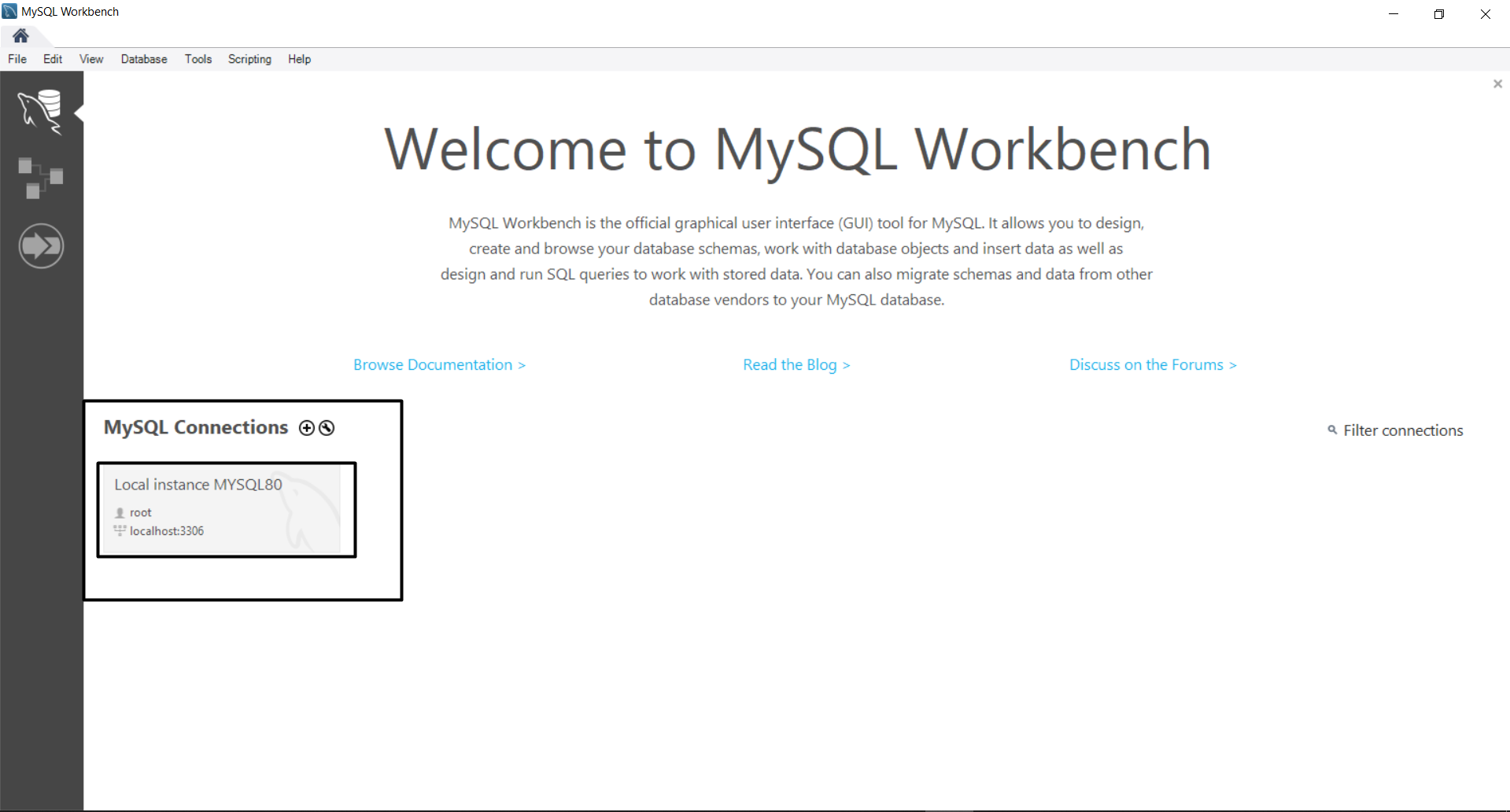
Quem faz o download das bibliotecas é o Maven a partir desse XML, o Maven consome o XML

Pra rodas esse lab5b é necessário o MySql, o serviço do MySql instalado 🡪 <https://dev.mysql.com/downloads/connector/j/> ou <https://dev.mysql.com/downloads/windows/installer/8.0.html> , por exemplo o MySQL Workbench 8.0 CE e o MySql Workbench não precisa de estar aberto para funcionar, mas pelo menos o Serviço do MySql tem que estar aberto/instanciado no servidor ou local e com o BD criado que foi definido no objeto de persistência persistence.xml: <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/bancoHolerite?useTimezone=true&amp;serverTimezone=UTC" />:

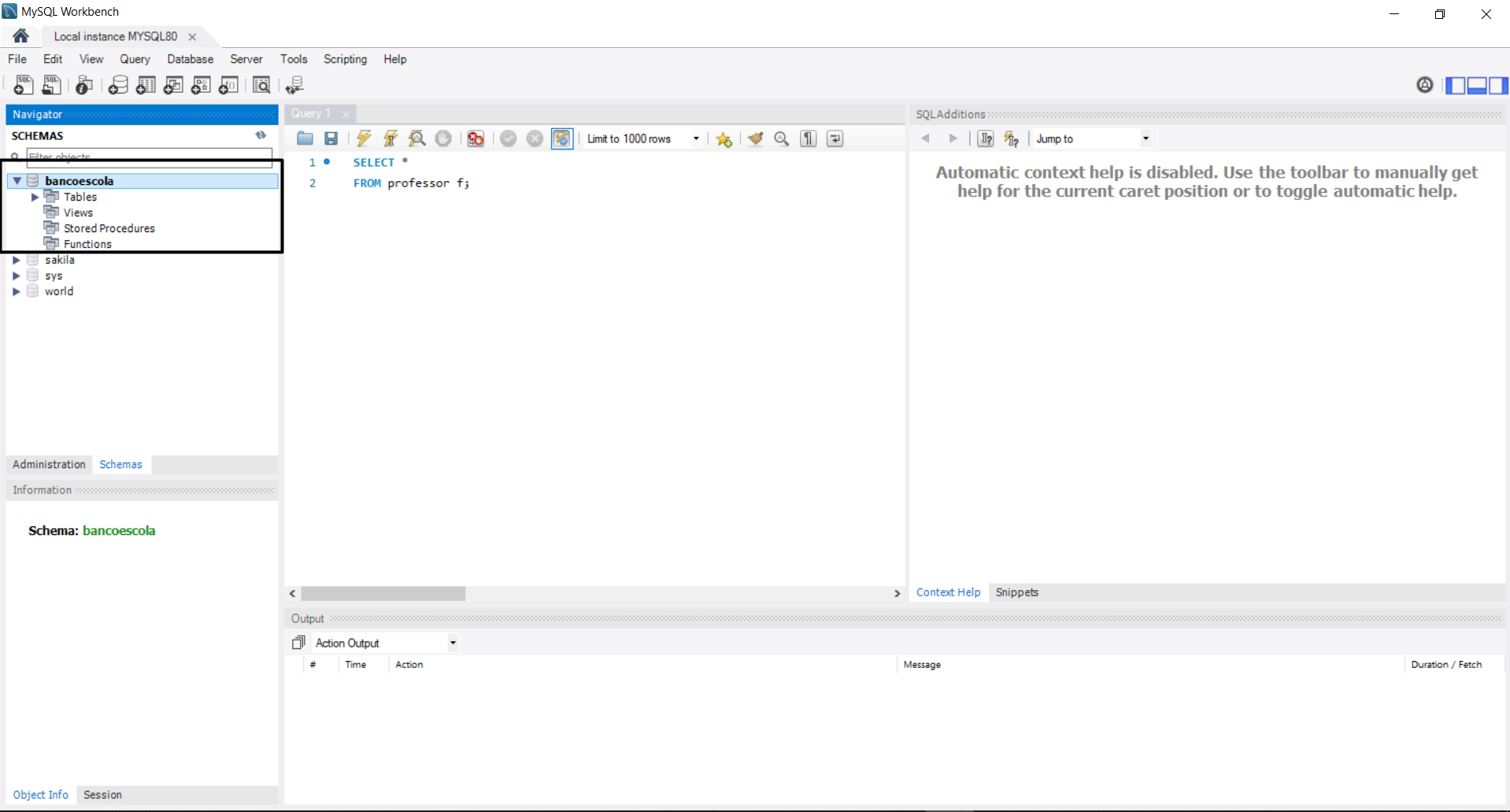
Obs: Se quiser usar o XAMPP tem que parar/pausar o MySql80 pra instanciar um novo servidor na maquina local 🡪 <https://www.apachefriends.org/pt_br/download.html>



Instancia do servidor local



Banco de dados(schema) criado no servidor local



POM.XML

Propriedades:

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding> 🡪 Qual é a codificação do texto, dos arquivos do código fonte, estamos usando o UTF-8 pra padronizar a codificação dos arquivos

<java.version>11</java.version> 🡪 Versão do Java 🡪 Java11

**Algumas versões:**

<maven.compiler.source>${java.version}</maven.compiler.source>🡪 versão fonte de compilação 11

<maven.compiler.target>${java.version}</maven.compiler.target>🡪 versão destino é 11

<maven.compiler.release>${java.version}</maven.compiler.release> 🡪 versão de liberação da biblioteca do Jar é 11

Isso tudo é para garantir que a mesma versão esteja sendo usada em todos os processos de compilação

**Versão do Junit:**

<junit>5.6.2</junit>

Versões de Plugin que o Maven usa:

<!-- Plugin versions -->

<maven.shade>3.2.2</maven.shade>

<maven.clean>3.1.0</maven.clean> <maven.resources>3.1.0</maven.resources>

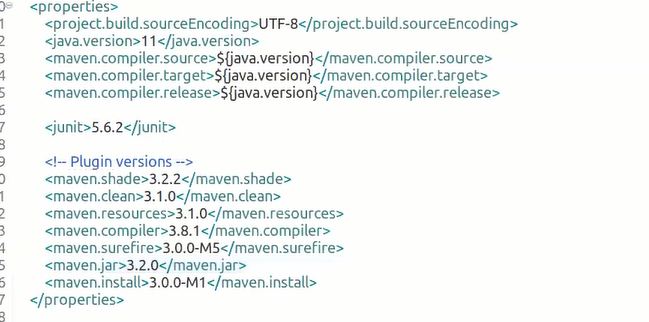
<maven.compiler>3.8.1</maven.compiler>

<maven.surefire>3.0.0-M5</maven.surefire>

<maven.jar>3.2.0</maven.jar>

<maven.install>3.0.0-M1</maven.install>

</properties>



Agora as dependências/bibliotecas que o nosso projeto depende

<dependencies>

1ºO Hibernet:

<dependency>

<groupId>org.hibernate</groupId> 🡪 Definição da biblioteca

<artifactId>hibernate-core</artifactId>

<version>5.4.32.Final</version> 🡪 Versão do Hibernet que vai ser usada 🡪 Link das versões: <https://mvnrepository.com/artifact/org.hibernate/hibernate-core>

</dependency>

O maven vai carregar isso tudo automaticamente para mim e adicionar isso como biblioteca no meu projeto, então é assim que eu consigo instanciar um:

// Fabrica de persistencia //Usando o objeto de persistencia

private static EntityManagerFactory entityManagerFactory = Persistence.createEntityManagerFactory("BancoMySql");

// Gerenciador de persistencia

private static EntityManager entityManager = entityManagerFactory.createEntityManager();

O EntityManager e o EntityManagerFactory esta dentro do <artifactId>hibernate-core</artifactId>

2º Dependência com o connector do MySql:

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.16</version>

</dependency>



3º Dependência com o JUnit, são 3 bibliotecas que compõem o JUnit 5:

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>${junit}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>${junit}</version>

<scope>test</scope>

</dependency>

<dependency>

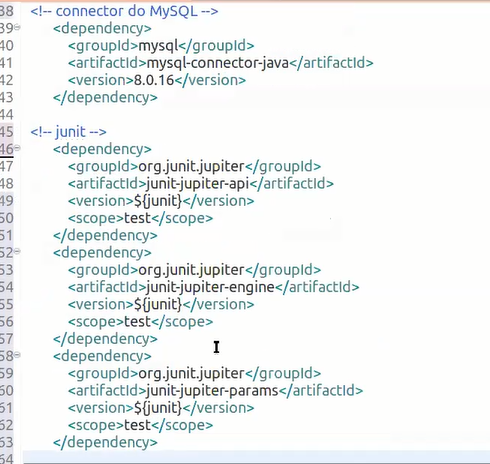
<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-params</artifactId>

<version>${junit}</version>

<scope>test</scope>

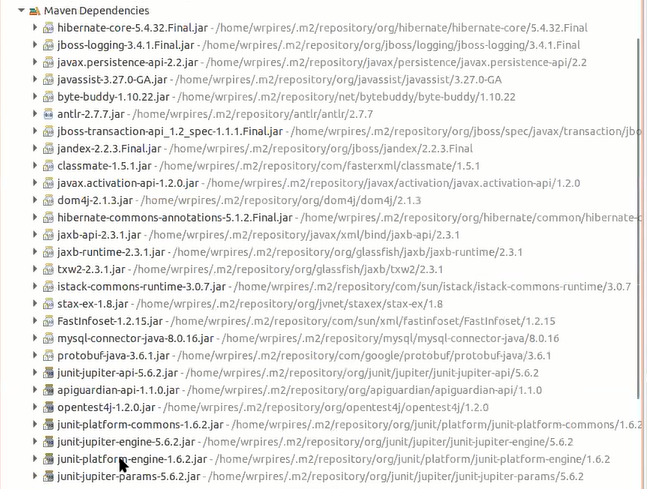
</dependency>



</dependencies>

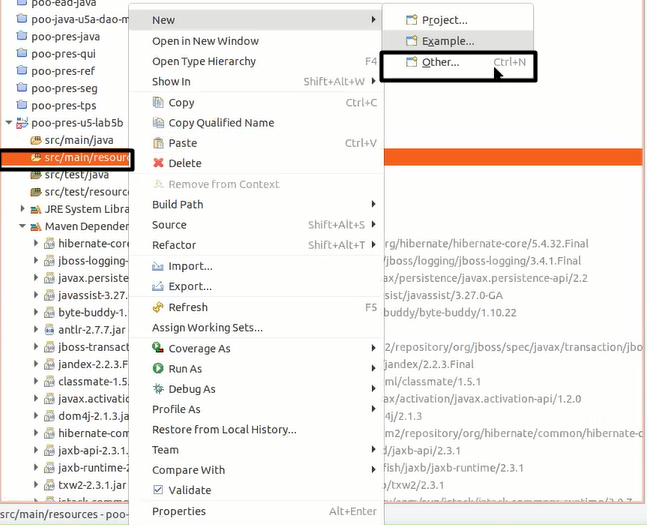
Esse arquivo então é a onde a gente coloca as dependências/bibliotecas e propriedades do nosso projeto, que eu dependo

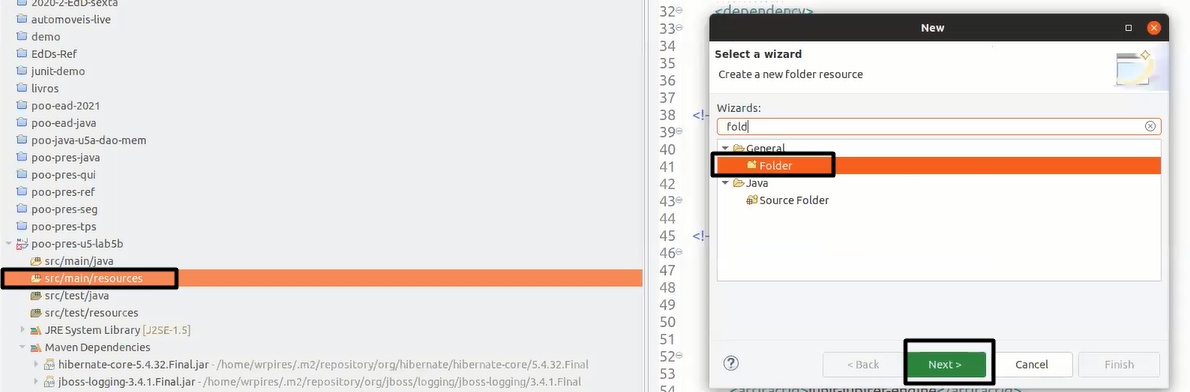
Todas as dependências listadas:

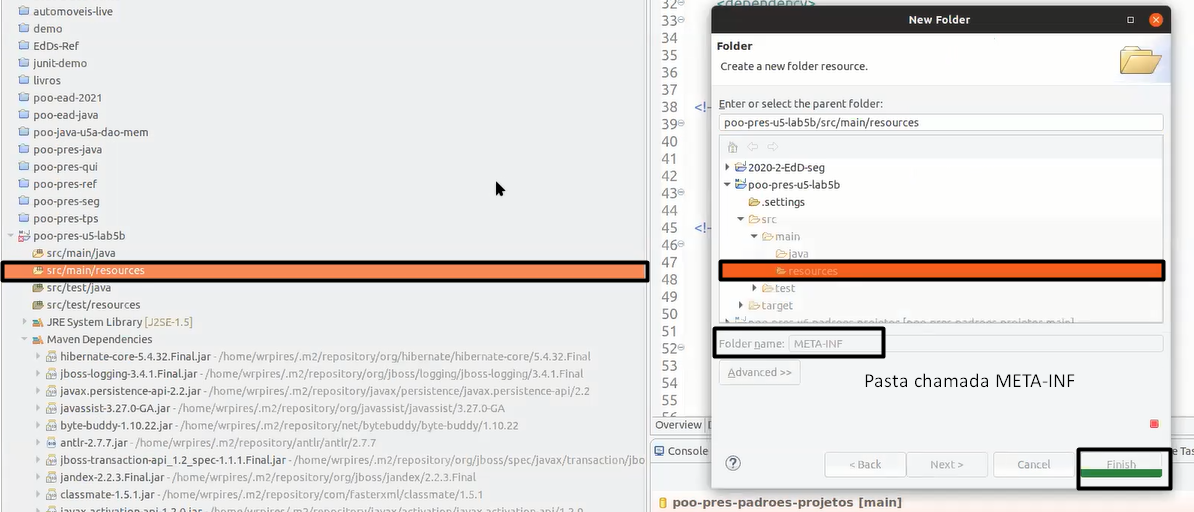


Depois do POM.XML agora é configurar o **objeto de persistência** que é as unidades de persistência, pra isso a gente cria uma pasta **META-INF**



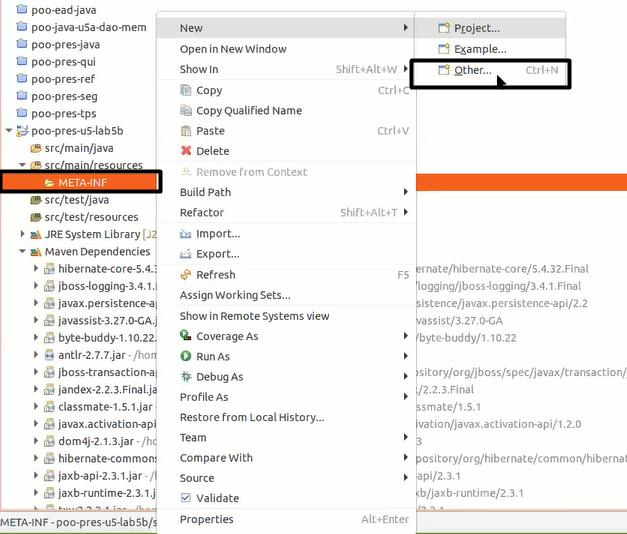


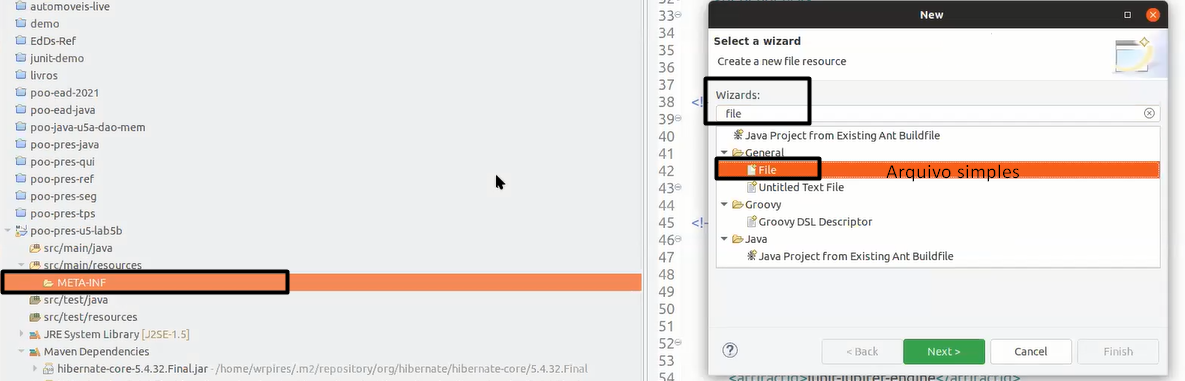


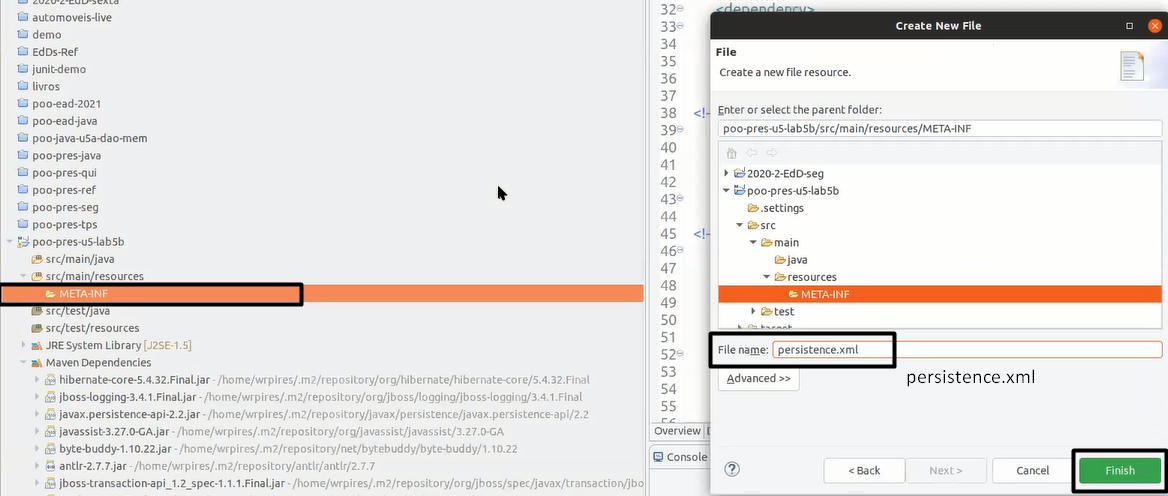




Dentro dessa pasta META-INF a gente cria um arquivo chamado persistence.xml que é o nosso objeto de persistência é a onde configura as unidades de persistência(MySql, H2, etc) que a gente vai usar







**A unidade de persistência que foi utilizada nesse projeto foi o MySql, mas poderia usar o H2 também, os dois juntos e separado também**

<?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\_2\_2.xsd"

version="2.2">

<!—Aqui são os parâmetros de conexão que o banco recebe para processar -->

<!-- Aqui eu defino qual unidade de persistência que eu vou utilizar no meu projeto java-->

<!—Nome da minha unidade de persistência-->

<persistence-unit name="BancoMySqlHolerite">

<!—Propriedades pra conectar com o banco-->

<properties>

<!— Essa propriedade é qual a URL que é vou conectar com o banco, nesse caso o meu MySql está rodando na porta 3306, depois coloca o nome do banco, lembrando que a porta tem que estar certa e o banco com o nome colocado já tem que estar criado no server local, **useTimezone=true&amp;** 🡪 definição do fuso horário **serverTimezone=UTC** 🡪 e qual fuso horário que o banco vai usar que é o UTC que é a hora universal -->

<property name="javax.persistence.jdbc.url"

value="jdbc:mysql://localhost:3306/bancoHolerite?useTimezone=true&amp;serverTimezone=UTC" />

<!— Essa propriedade abaixo é igual a anterior so que com essa é possível criar o Banco de dados do 0 a partir do código java, sem precisa ter criado o BD no server local 🡪 **createDatabaseIfNotExist=true&amp;** -->

<!--<property *name*="javax.persistence.jdbc.url"

*value*="jdbc:mysql://localhost:43306/holerite?createDatabaseIfNotExist=true&amp;useTimezone=true&amp;serverTimezone=UTC" />-->

<!—Informação de conexão usuário e senha que esta configurado no meu banco, isso vai de banco para banco -->

<property name="javax.persistence.jdbc.user" value="root" />

<property name="javax.persistence.jdbc.password"

value="123456" />

<!—Aqui é o driver de conexão com o MySql que o hibernet vai usar através do MySql connector JDBC e isso justamente é a dependência que foi adicionada no nosso projeto-->

<property name="javax.persistence.jdbc.driver"

value="com.mysql.cj.jdbc.Driver" />

<!—Dialeto de comunicação entre o hibernet e o banco, isso aqui é importante, aqui esta sendo usado o dialeto MySql8 que é o padrão-->

<property name="hibernate.dialect"

value="org.hibernate.dialect.MySQL8Dialect" />

<!—Aqui são informações pra exibir na saída padrão-->

<!—Sql vai vim formatado-->

<property name="hibernate.format\_sql" value="true" />

<!—Exibe o Sql-->

<property name="hibernate.show\_sql" value="true" />

<!—Exibe comentários sobre Sql-->

<property name="hibernate.use\_sql\_comments" value="true" />

# <!--Essa propriedade é importante porque ela faz com que o hibernet gere as tabelas quando é conectado no banco, ele limpa(Drop table) todo o banco de dados, apaga todos os dados e tabelas, e recria as tabelas do zero, isso é interessante quando é feito modificações nas classes de entidade, quando eu mudo a classe de entidade e mando executar ele vai regerar as tabelas do banco mas os dados armazenados são excluídos, isso é so utilizado no desenvolvimento da aplicação e não durante o uso mesmo da aplicação, [**Como funciona hibernate.hbm2ddl.auto?**](https://pt.stackoverflow.com/questions/90723/como-funciona-hibernate-hbm2ddl-auto)

-->

<property name="hibernate.hbm2ddl.auto" value="create" />

</properties>

</persistence-unit>

</persistence>

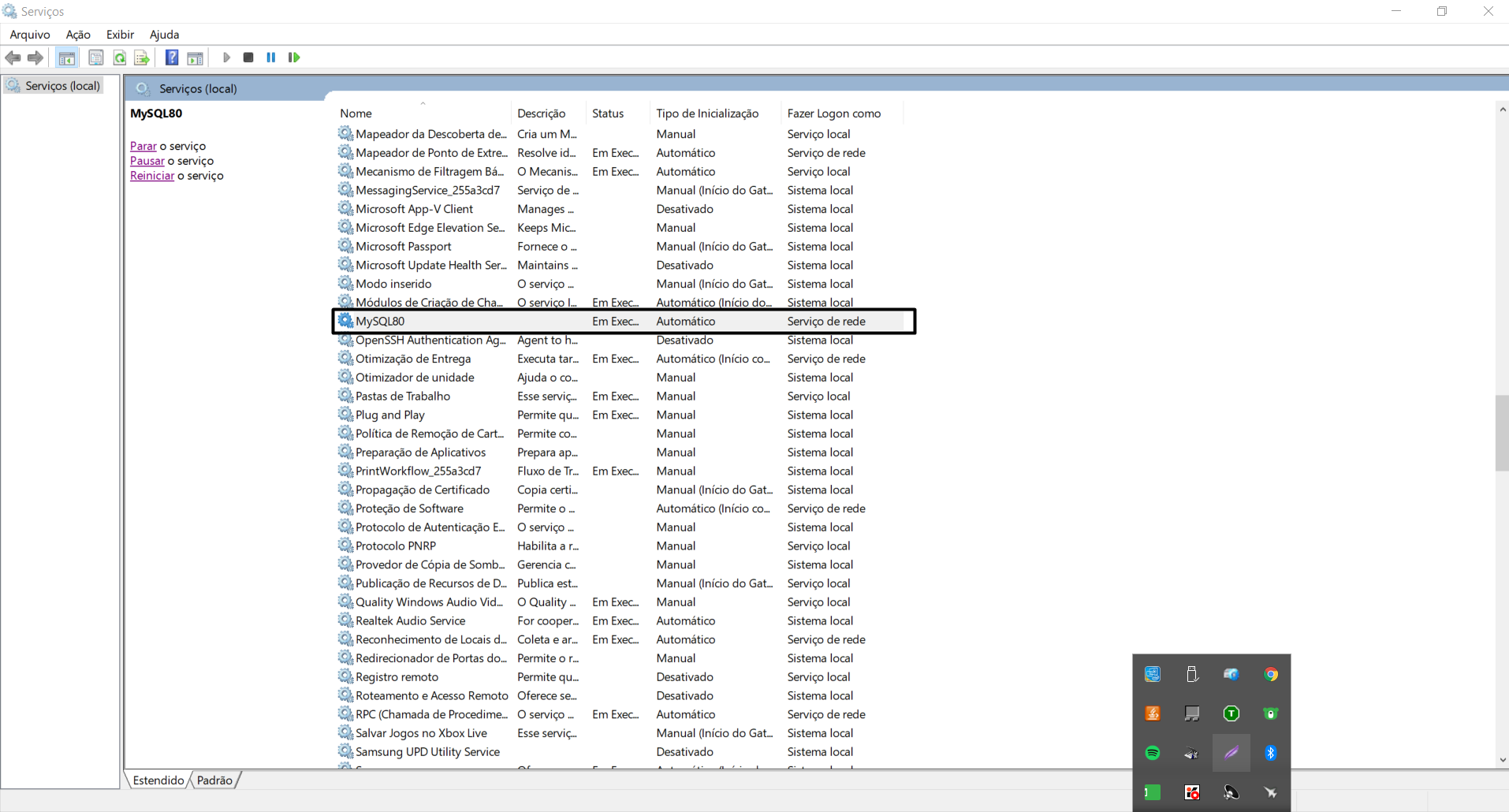
E pronto já pode começar o mapeamento das classes em java para classes de entidade usando anotações JPA/Hibernet 🡪 @

Esses exemplos estão na pasta 🡪 **2- Ajuda do Waldir**

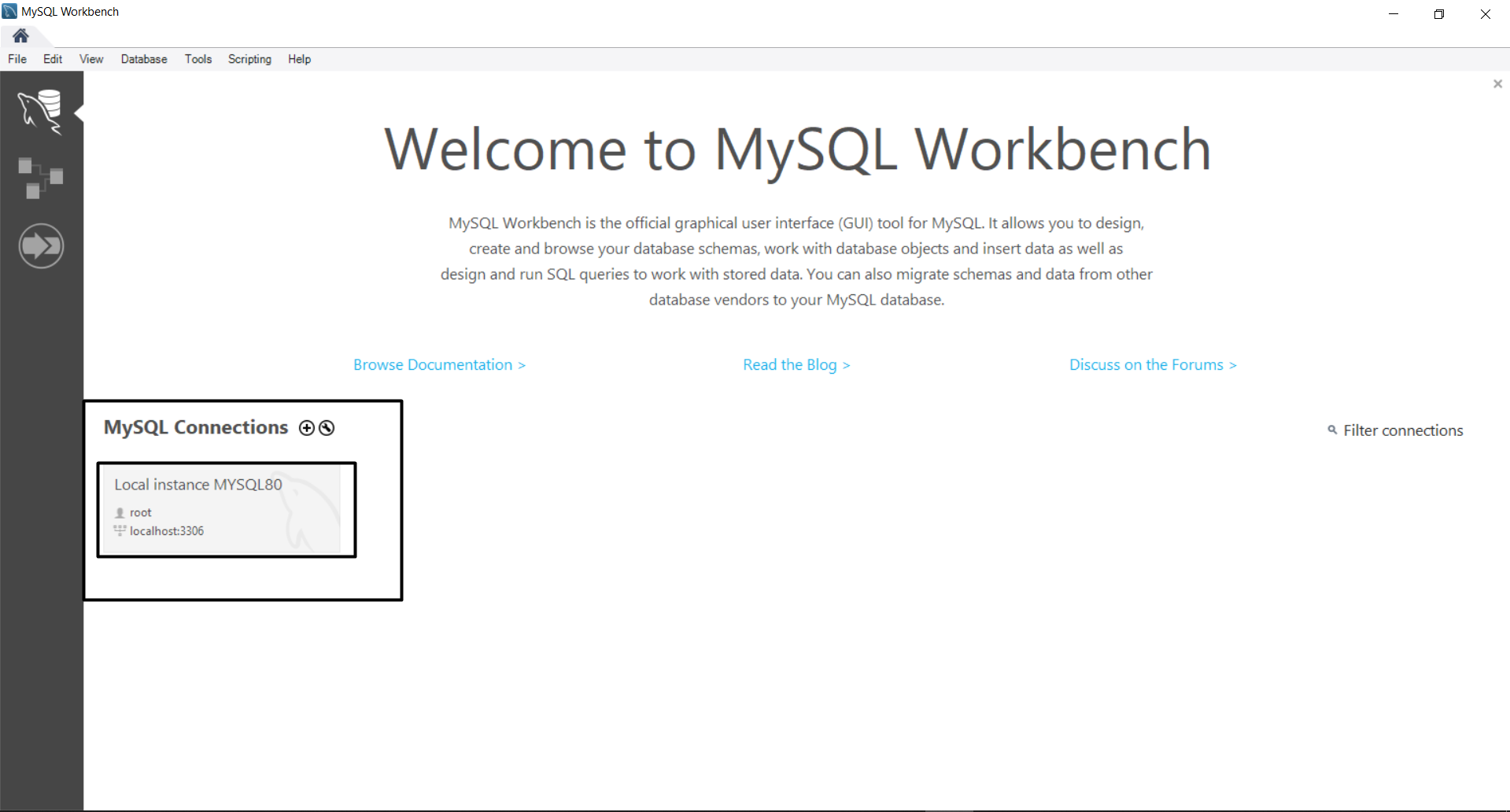
Obs: Serviço do MySql tem que estar on e o banco de dados tem que estar criado, se não tiver criado e se não quiser criar o BD antes e sim na hora de iniciar o código java é so usar isso na URL do objeto de persistência 🡪 <!--<property *name*="javax.persistence.jdbc.url"

*value*="jdbc:mysql://localhost:43306/holerite?createDatabaseIfNotExist=true&amp;useTimezone=true&amp;serverTimezone=UTC" />-->

Serviço on:



Instancia do servidor local



Banco de dados(schema) criado no servidor local

