

# C207 - BD

Arthur Openheimer

# Integração Java e MySQL

Java

[JDK \(Java Development Kit\)](#)



IntelliJ

[IntelliJ IDEA Community version](#)



# Integração Java e MySQL

Verificando se o Java está instalado:

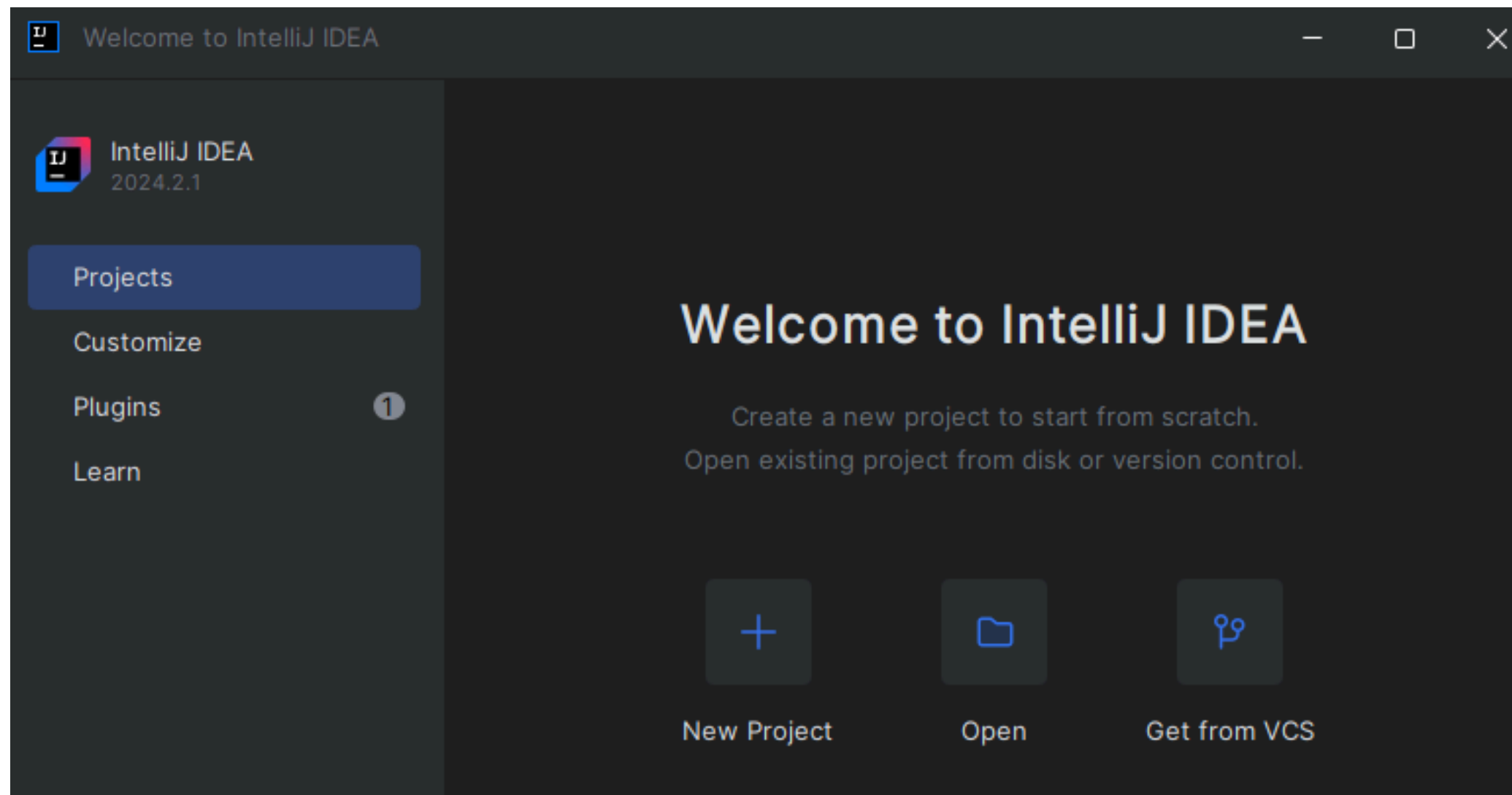
1. Na aba de pesquisa digite “Prompt de Comando”
2. Digite “java -version”
3. Deve aparecer a versão instalada:

```
java version "1.8.0_333"  
Java(TM) SE Runtime Environment (build 1.8.0_333-b02)  
Java HotSpot(TM) 64-Bit Server VM (build 25.333-b02, mixed mode)
```

O uso do IntelliJ é opcional, use a IDE de sua preferência mas lembre-se que é mais fácil gerenciar projetos em Java no IntelliJ!

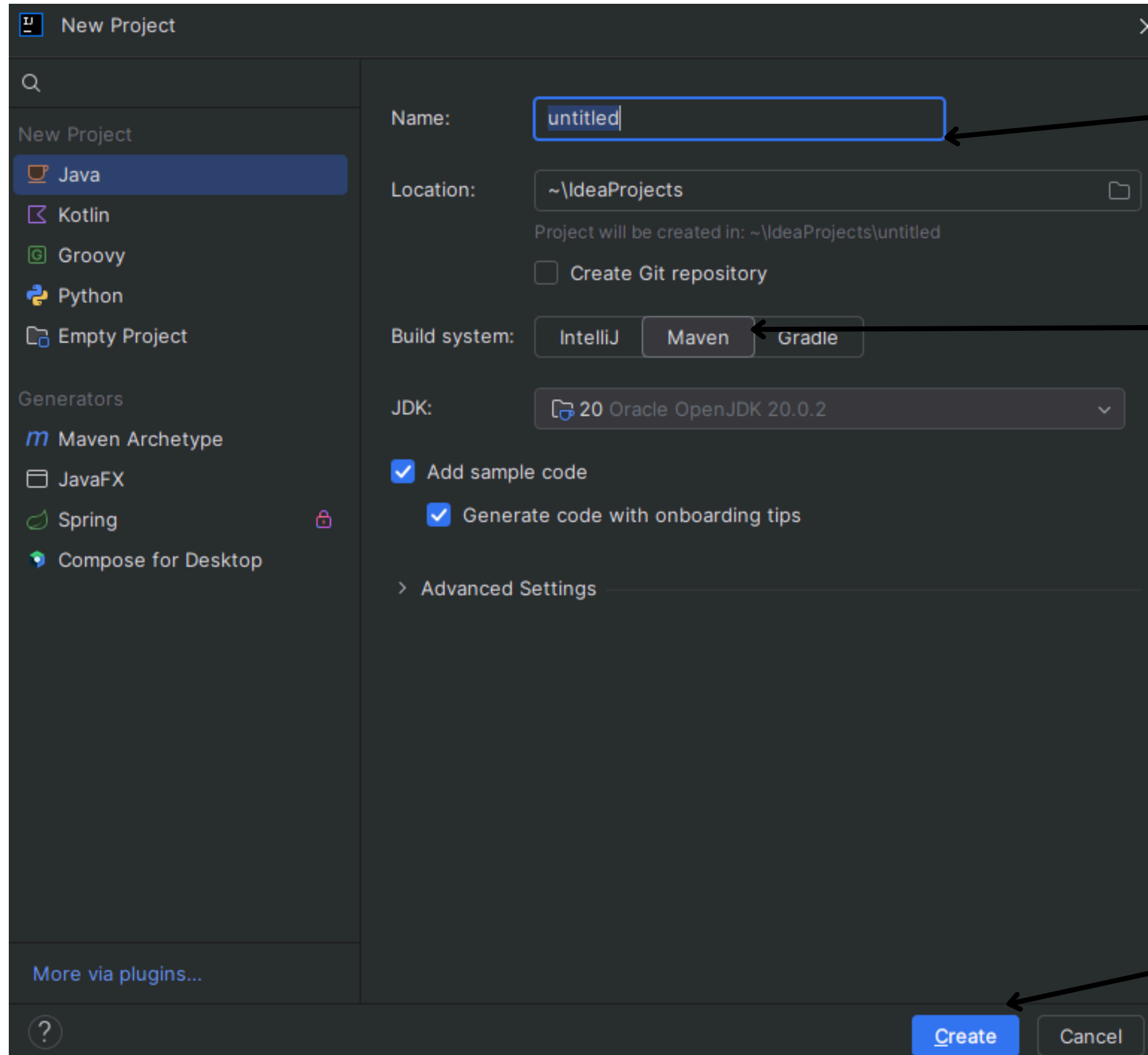
# Criando um projeto no IntelliJ

Ao iniciar o IntelliJ, se não tiver criado nenhum projeto anteriormente, a seguinte tela será exibida:



Clique em “New Project” para criar um novo projeto. Se tiver com projeto já aberto, vá em “File”->“Close Project”, será exibida a janela acima.

# Criando um projeto no IntelliJ

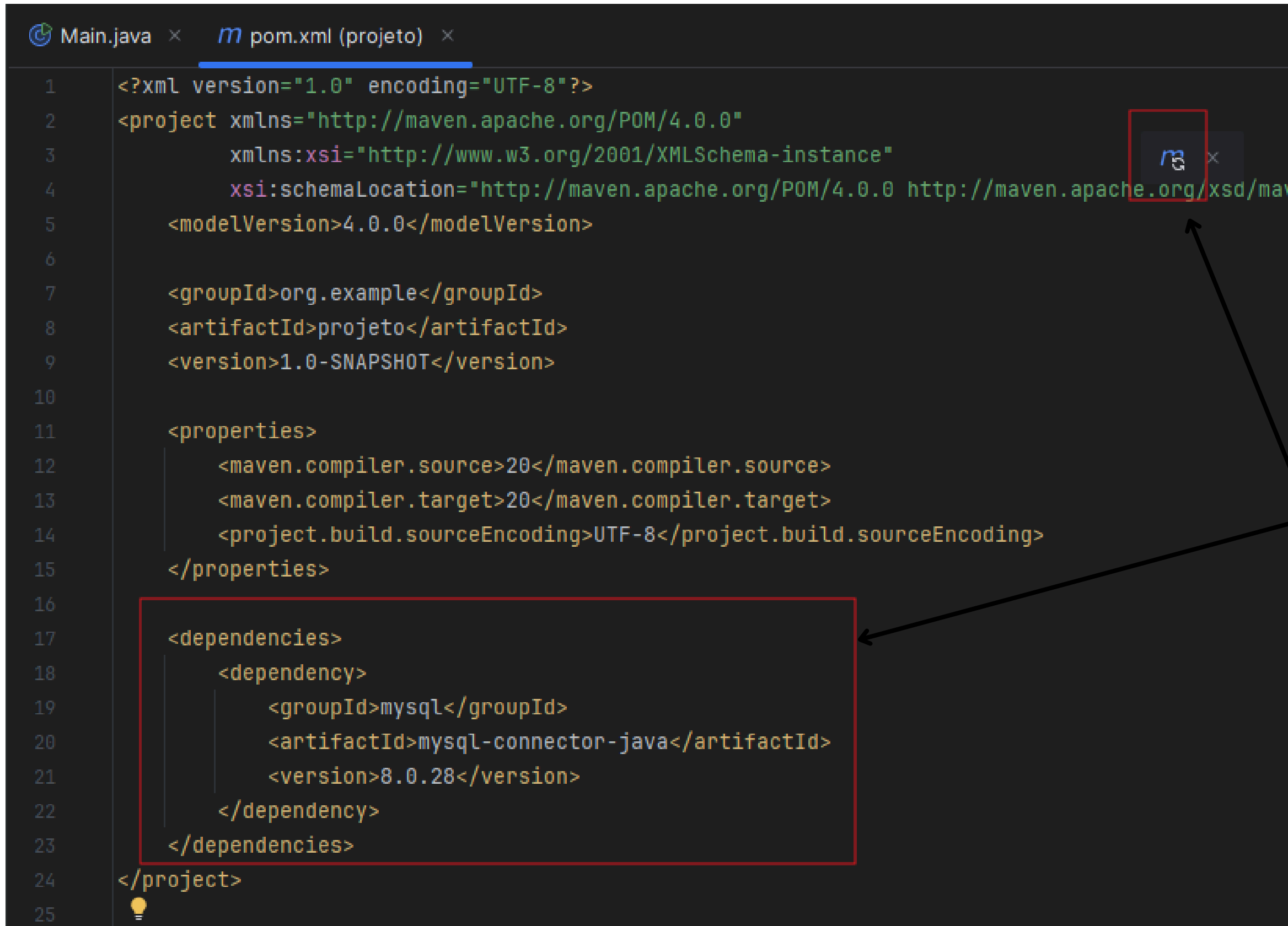


Escolha um nome pro projeto

Selecione a opção Maven

Crie o projeto

# Criando um projeto no IntelliJ



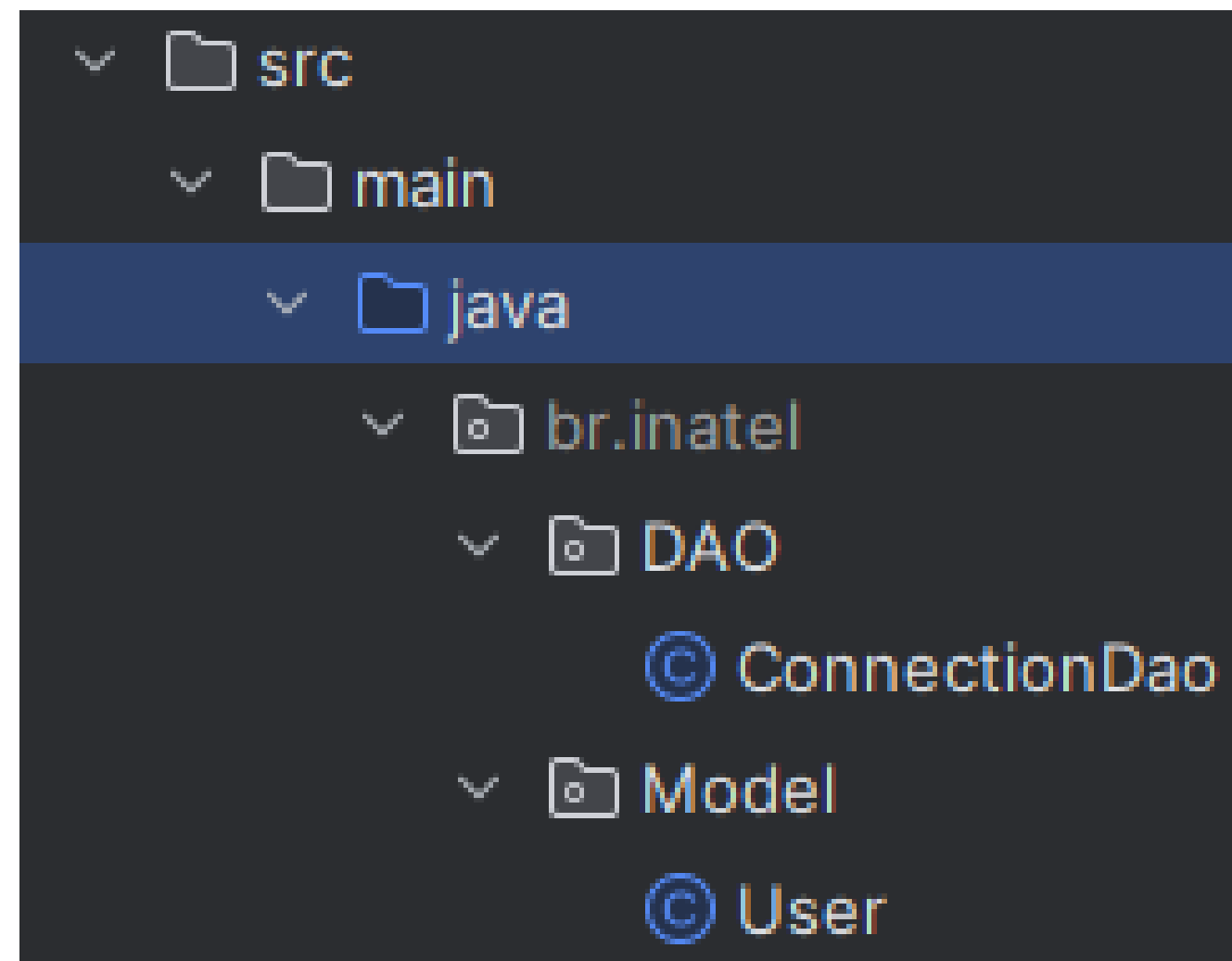
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5      <modelVersion>4.0.0</modelVersion>
6
7      <groupId>org.example</groupId>
8      <artifactId>projeto</artifactId>
9      <version>1.0-SNAPSHOT</version>
10
11     <properties>
12         <maven.compiler.source>20</maven.compiler.source>
13         <maven.compiler.target>20</maven.compiler.target>
14         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
15     </properties>
16
17     <dependencies>
18         <dependency>
19             <groupId>mysql</groupId>
20             <artifactId>mysql-connector-java</artifactId>
21             <version>8.0.28</version>
22         </dependency>
23     </dependencies>
24 </project>
```

Quando for finalizado a criação do projeto, o IntelliJ irá abrir o pom.xml, ele é o arquivo onde o Maven (Gerenciador do projeto Java) instala todas as dependências do projeto de maneira automática

Escreva esse bloco para que as dependências do MySQL sejam instaladas e em seguida clique no botão do canto superior para instalar

# Criando as classes para integração

1. No menu lateral, clique com o botão direito na pasta java em src-> main e crie uma nova pasta(package) chamada “br.inatel”
2. Nessa pasta, clique com o botão direito e crie uma nova classe chamada “DAO.ConnectionDAO”
3. Depois, na pasta br.inatel, crie outra classe chamada “Model.User”
4. A estrutura final deve ser assim:



# Criando as classes para integração

Tabela “User” em SQL

Classe “User” preenchida

```
1 • drop database if exists integracao;
2 • create database integracao;
3 • use integracao;
4
5 • ○ create table usuario(
6     id int not null auto_increment primary key,
7     nome varchar(45) not null,
8     cpf varchar(45) not null
9 );
```

```
package br.inatel.Model;

public class User { no usages

    String nome; 1 usage

    String cpf; 1 usage

    int id; 1 usage

    public User(String nome, String cpf, int id){ no usages
        this.nome = nome;
        this.cpf = cpf;
        this.id = id;
    }
}
```



# Fazendo a conexão com o banco

```
m pom.xml  ConnectionDao.java  User.java
1 package br.inatel.DAO;
2
3 import java.sql.*;
4
5 public abstract class ConnectionDao { no usages
6
7     Connection con; //conexão 1 usage
8     PreparedStatement pst; //declaração(query) preparada - código em sql no usages
9     Statement st; //declaração(query) - código em sql no usages
10    ResultSet rs; //resposta do banco no usages
11
12    String database = "integracao"; //nome do BD 1 usage
13    String user = "root"; 1 usage
14    String password = "root"; 1 usage
15    String url = "jdbc:mysql://localhost:3306/" + database 1 usage
16        + "?useTimezone=true&serverTimezone=UTC&useSSL=false&allowPublicKeyRetrieval=true";
17
18    public void connectToDb() { no usages
19        try {
20            con = DriverManager.getConnection(url, user, password);
21            System.out.println("Conectado com sucesso!");
22        } catch (SQLException exc) {
23            System.out.println("Erro: " + exc.getMessage());
24        }
25    }
26 }
```

Classe “ConnectionDAO”  
preenchida com o bloco de  
código para se conectar no  
banco

# Fazendo a conexão com o banco

Na pasta DAO, crie uma nova classe “UserDAO” e a preencha como no bloco abaixo, para cada tabela do banco, criaremos um Modelo para representá-lo e um DAO para se conectar com o banco e executar seus comandos

```
package br.inatel.DAO;

public class UserDAO extends ConnectionDAO{

    public void testConnection(){ //usage
        connectToDb();
    }

}
```

Método auxiliar que iremos usar para testar a conexão, mais tarde iremos criar os métodos para os comandos SQL aqui

# Testando a conexão

Crie um arquivo “Main” na pasta br.inatel e coloque o seguinte bloco de código nela:

```
package br.inatel;  
import br.inatel.DAO.UserDAO;  
  
public class Main {  
    public static void main(String[] args) {  
        UserDAO userDAO = new UserDAO();  
        userDAO.testConnection();  
    }  
}
```

Após isso execute o SQL para criar a tabela e após isso, execute o arquivo Main e a mensagem da conexão do banco deve ser mostrada:

**Conectado com sucesso!**

# Comandos SQL

# Insert

```
public boolean insertUser(User user){ no usages
    connectToDb();

    boolean sucesso;
    String sql = "INSERT INTO usuario (nome, cpf) VALUES (?, ?)";
    try {
        pst = con.prepareStatement(sql);
        pst.setString(parameterIndex: 1, user.getNome());
        pst.setString(parameterIndex: 2, user.getCpf());
        pst.execute();
        sucesso = true;
    } catch (SQLException exc) {
        System.out.println("Erro: " + exc.getMessage());
        sucesso = false;
    } finally {
        try {
            con.close();
            pst.close();
        } catch (SQLException exc) {
            System.out.println("Erro: " + exc.getMessage());
        }
    }
    return sucesso;
}
```

# Update

```
public boolean updateUser(int id, User user){ no usages
    connectToDb();

    boolean sucesso;
    String sql = "UPDATE usuario SET nome = ?, cpf = ? WHERE id = ?";
    try {
        pst = con.prepareStatement(sql);
        pst.setString(parameterIndex: 1, user.getNome());
        pst.setString(parameterIndex: 2, user.getCpf());
        pst.setInt(parameterIndex: 3, id);
        pst.execute();
        sucesso = true;
    } catch (SQLException exc) {
        System.out.println("Erro: " + exc.getMessage());
        sucesso = false;
    } finally {
        try {
            con.close();
            pst.close();
        } catch (SQLException exc) {
            System.out.println("Erro: " + exc.getMessage());
        }
    }
    return sucesso;
}
```

# Delete

```
public boolean deleteUser(int id){ no usages
    connectToDb();

    boolean sucesso;
    String sql = "DELETE FROM usuario WHERE id = ?";
    try {
        pst = con.prepareStatement(sql);
        pst.setInt( parameterIndex: 1, id);
        pst.execute();
        sucesso = true;

    } catch (SQLException exc) {
        System.out.println("Erro: " + exc.getMessage());
        sucesso = false;
    } finally {
        try {
            con.close();
            pst.close();
        } catch (SQLException exc) {
            System.out.println("Erro: " + exc.getMessage());
        }
    }
    return sucesso;
}
```

# Select

```
import java.util.ArrayList;
```

```
public ArrayList<User> selectUser() {
    connectToDb();

    ArrayList<User> users = new ArrayList<>();
    String sql = "SELECT * FROM usuario";
    try {
        st = con.createStatement();
        rs = st.executeQuery(sql);
        System.out.println("Lista de usuários:");
        while (rs.next()) {
            User userAux = new User(rs.getString("nome"), rs.getString("cpf"), rs.getInt("id"));
            System.out.println("Nome: " + userAux.getNome() + " CPF: " + userAux.getCpf());
            System.out.println("-----");
            users.add(userAux);
        }
    } catch (SQLException exc) {
        System.out.println("Erro: " + exc.getMessage());
    } finally {
        try {
            con.close();
            st.close();
            rs.close();
        } catch (SQLException exc) {
            System.out.println("Erro: " + exc.getMessage());
        }
    }
    return users;
}
```