



FASAM – Faculdade Sul-Americana

Prof.: Telmo Queiroz da Silva

E-mail: telmo.silva@fasam.edu.br

Back-end

Criando projeto SpringBoot

1. Documentação do *SpringBoot*:

<https://docs.spring.io/spring-boot/docs/current/reference/html/using-spring-boot.html>

2. Baixar Eclipse IDE:












The screenshot shows the Eclipse Foundation website's 'Eclipse IDE 2021-03 R Packages' page. The main content area features two cards for different IDE versions. The 'Eclipse IDE for Enterprise Java and Web Developers' card is highlighted with a red border. To the right, a sidebar contains a callout box stating 'The Eclipse Installer 2021-03 R now includes a JRE for macOS, Windows and Linux.' Below this, there are buttons for 'Download x86_64' and 'Download AArch64'. At the bottom right, a 'RELATED LINKS' section lists several links: 'Compare & Combine Packages', 'New and Noteworthy', 'Install Guide', 'Documentation', and 'Updating Eclipse'.

3. Baixar o Oracle JDK 11:

https://www.oracle.com/br/java/technologies/javase-jdk11-downloads.html

Java SE Development Kit 11.0.11

This software is licensed under the [Oracle Technology Network License Agreement for Oracle Java SE](#)

Product / File Description	File Size	Download
Linux ARM 64 Debian Package	145.61 MB	 jdk-11.0.11_linux-aarch64_bin.deb
Linux ARM 64 RPM Package	152.16 MB	 jdk-11.0.11_linux-aarch64_bin.rpm
Linux ARM 64 Compressed Archive	169.53 MB	 jdk-11.0.11_linux-aarch64_bin.tar.gz
Linux x64 Debian Package	149.39 MB	 jdk-11.0.11_linux-x64_bin.deb
Linux x64 RPM Package	156.07 MB	 jdk-11.0.11_linux-x64_bin.rpm
Linux x64 Compressed Archive	173.46 MB	 jdk-11.0.11_linux-x64_bin.tar.gz
macOS Installer	167.15 MB	 jdk-11.0.11_osx-x64_bin.dmg
macOS Compressed Archive	167.67 MB	 jdk-11.0.11_osx-x64_bin.tar.gz
Solaris SPARC Compressed Archive	184.51 MB	 jdk-11.0.11_solaris-sparcv9_bin.tar.gz
Windows x64 Installer	152.05 MB	 jdk-11.0.11_windows-x64_bin.exe
Windows x64 Compressed Archive	171.53 MB	 jdk-11.0.11_windows-x64_bin.zip

4. Instalar, configurar e criar banco de dados:

The screenshot shows the 'Download' section of the pgAdmin 4 website. On the left, there is a 'Quick Links' sidebar with icons for 'Online Demo', 'Download', 'FAQ', 'Latest Docs', 'Get Help', and 'Screenshots'. The main content area is titled 'Download' and contains the following text:

pgAdmin is a free software project released under the [PostgreSQL licence](#). The software is available in source and binary format from the [PostgreSQL mirror network](#). Because compiling from source requires technical knowledge, we recommend installing binary packages whenever possible.

The pages in this section give additional details about each binary package available as well as more direct download links. In addition, you can download source tarballs and pgAgent for your servers to enable additional functionality.

pgAdmin 4

pgAdmin 4 is a complete rewrite of pgAdmin, built using Python and Javascript/JQuery. A desktop runtime written in NW.js allows it to run standalone for individual users, or the web application code may be deployed directly on a web server for use by one or more users through their web browser. The software has the look and feels of a desktop application whatever the runtime environment is, and vastly improves on pgAdmin III with updated user interface elements, multi-user/web deployment options, dashboards, and a more modern design.

Below this text are two rows of icons representing different operating systems and package managers:

- Row 1: Container, macOS, Python, APT, RPM, Source Code, Windows.
- Row 2: APT, macOS, RPM, Windows, Source Code.

Below the second row, there is a section for 'pgAgent' with the following text:

pgAgent

pgAgent is a job scheduler for PostgreSQL which may be managed using pgAdmin. Prior to pgAdmin v1.9, pgAgent shipped as part of pgAdmin. From pgAdmin v1.9 onwards, pgAgent is shipped as a separate application.

Below this text are icons for APT, macOS, RPM, Windows, and Source Code.

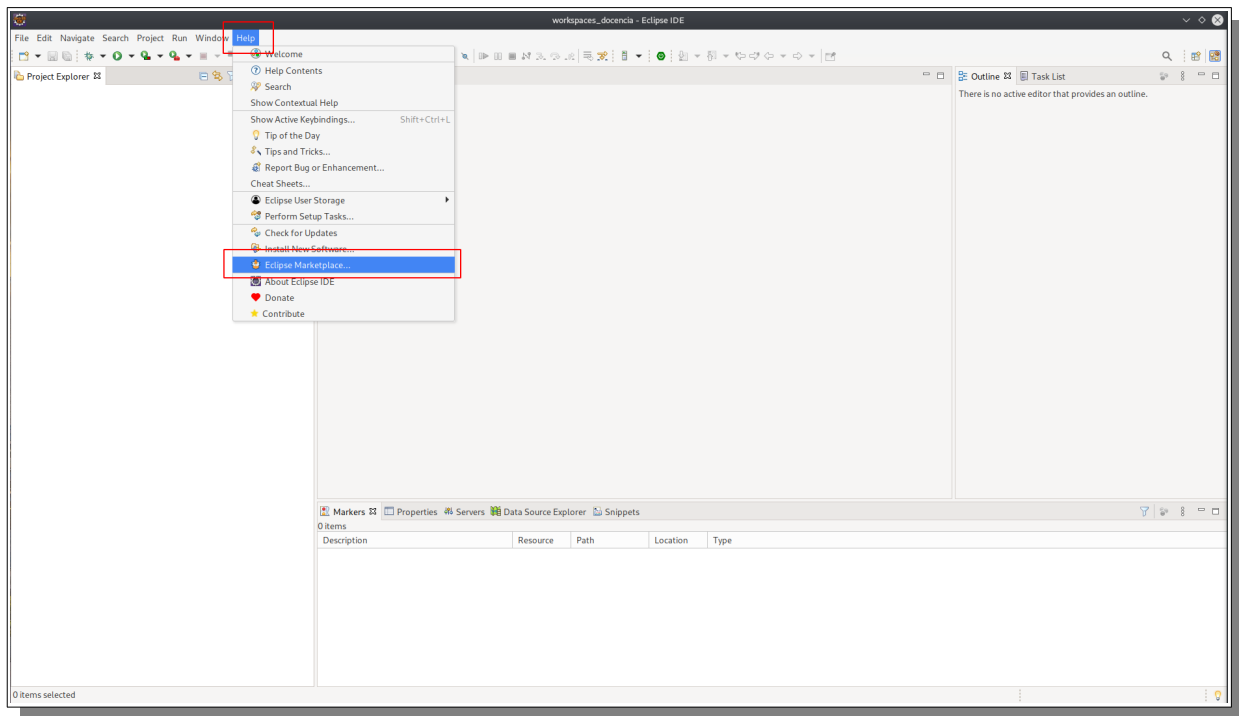
The screenshot shows the pgAdmin 4 web interface. The top navigation bar includes 'File', 'Object', 'Tools', and 'Help'. The main interface is divided into two panes. The left pane, titled 'Browser', shows a tree view of the database structure. The right pane, titled 'Dependents', shows a message: 'No dependents'.

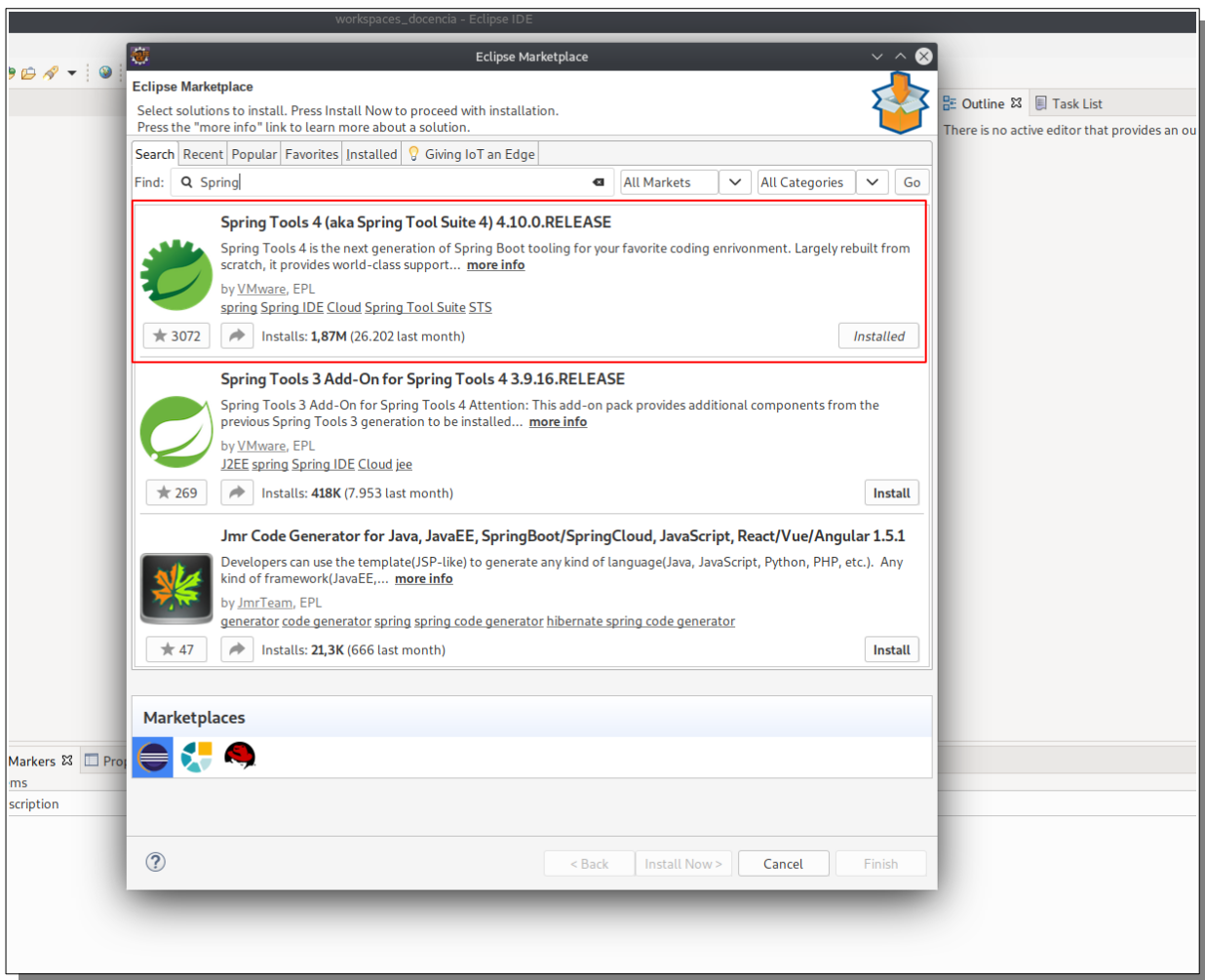
The 'Browser' pane shows the following structure:

- demo
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Publications
 - Schemas (1)
 - public
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Procedures
 - Sequences
 - Tables
 - Trigger Functions
 - Types
 - Views

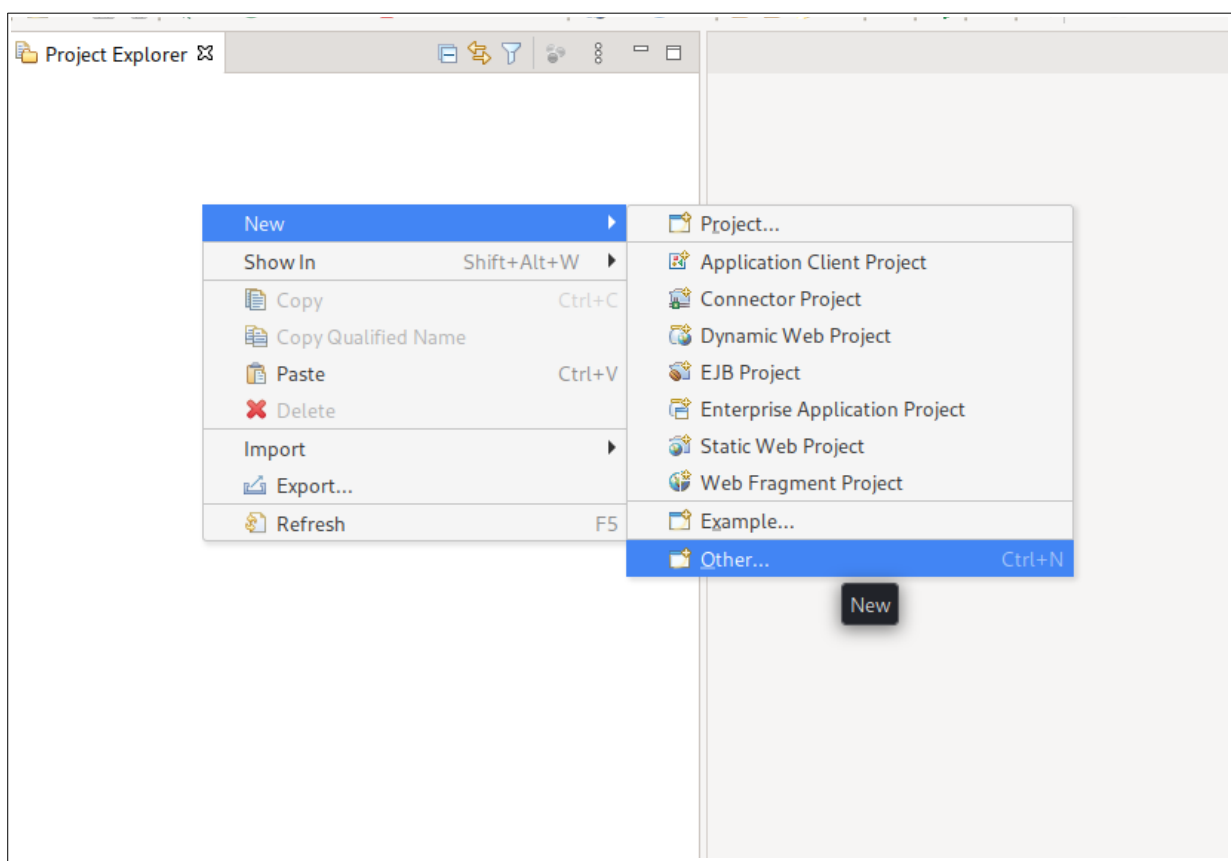
The 'Dependents' pane shows a message: 'No dependents'.

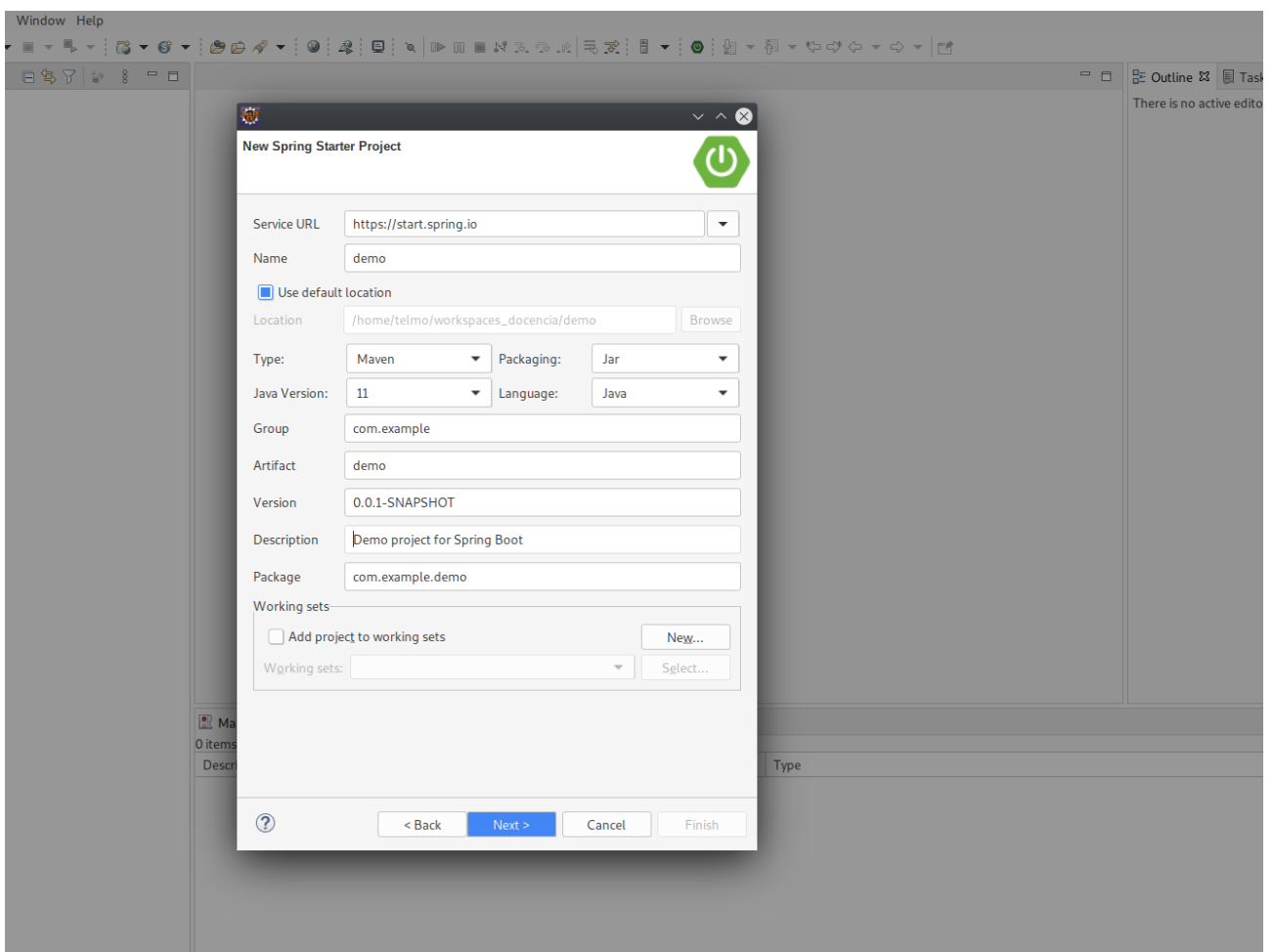
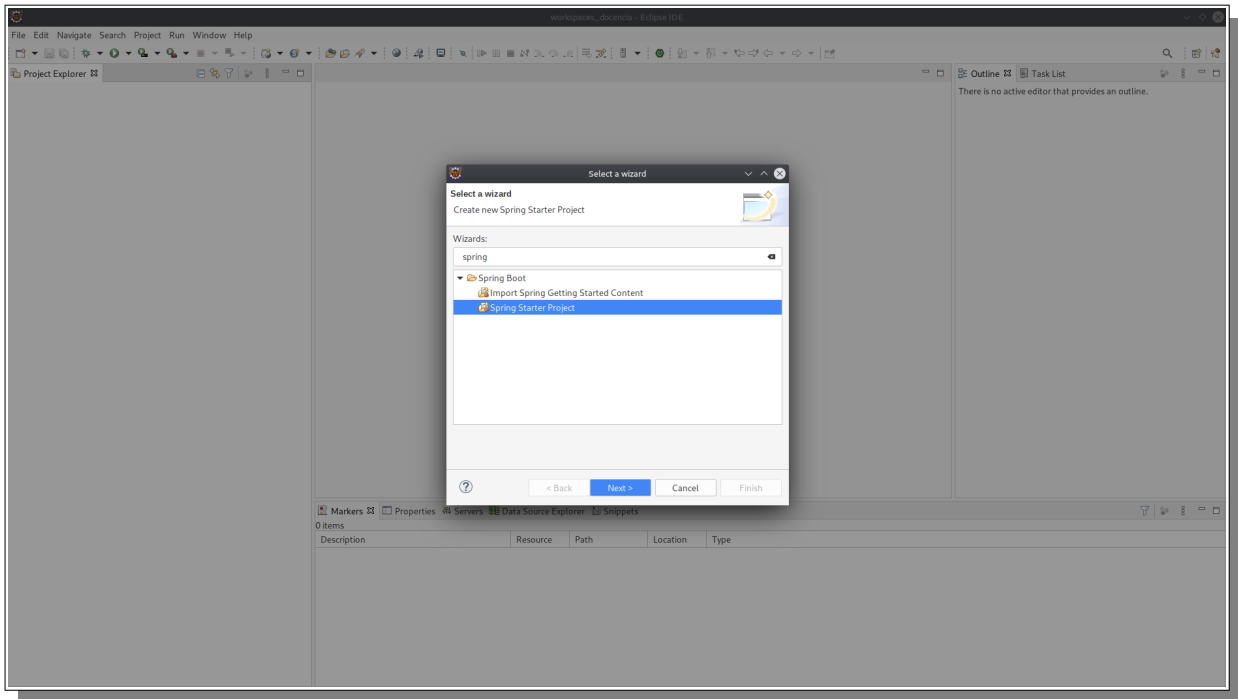
5. Opcionalmente, pode instalar *plugin* do *Spring* no Eclipse



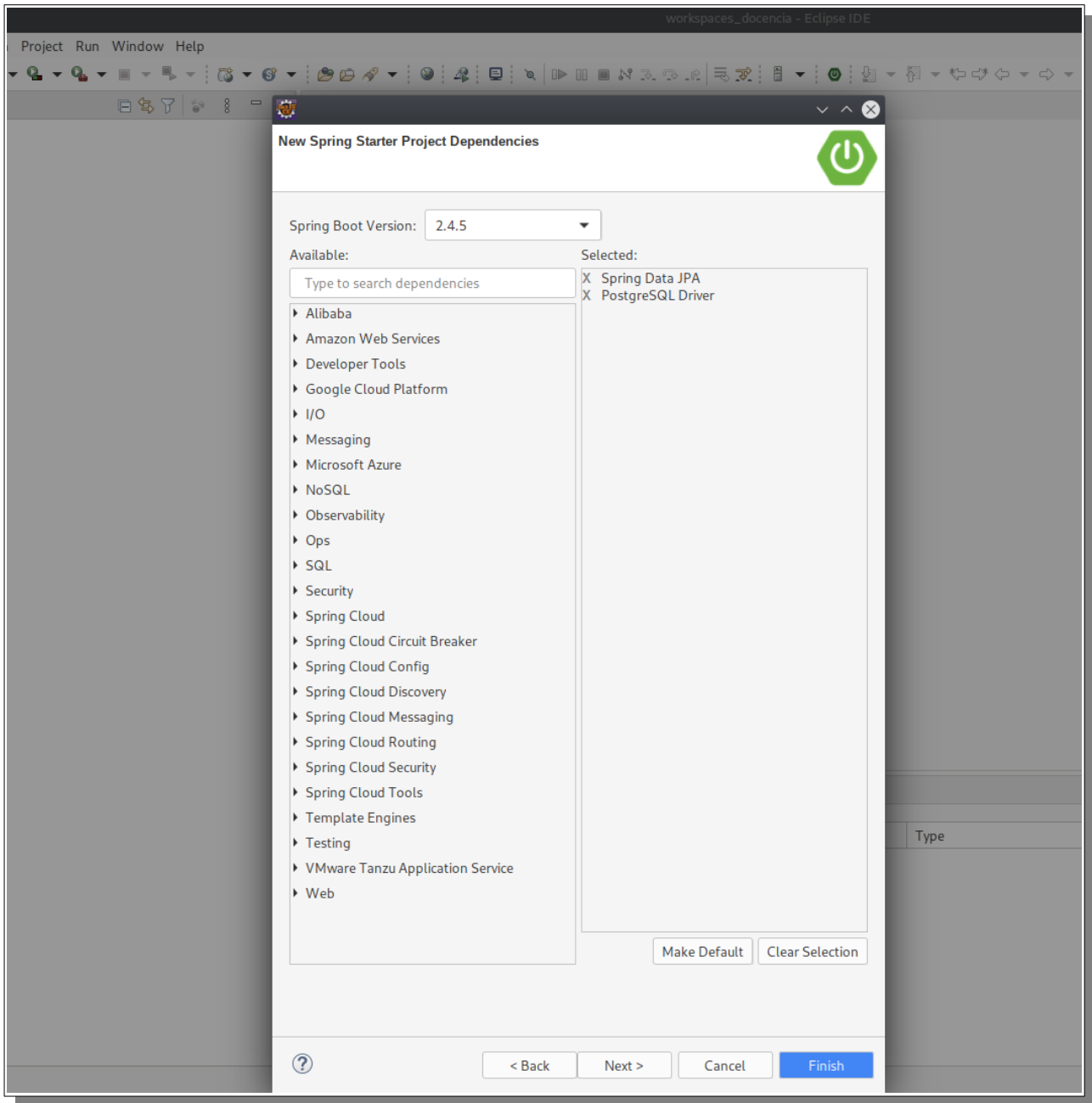


6. Criar novo projeto:

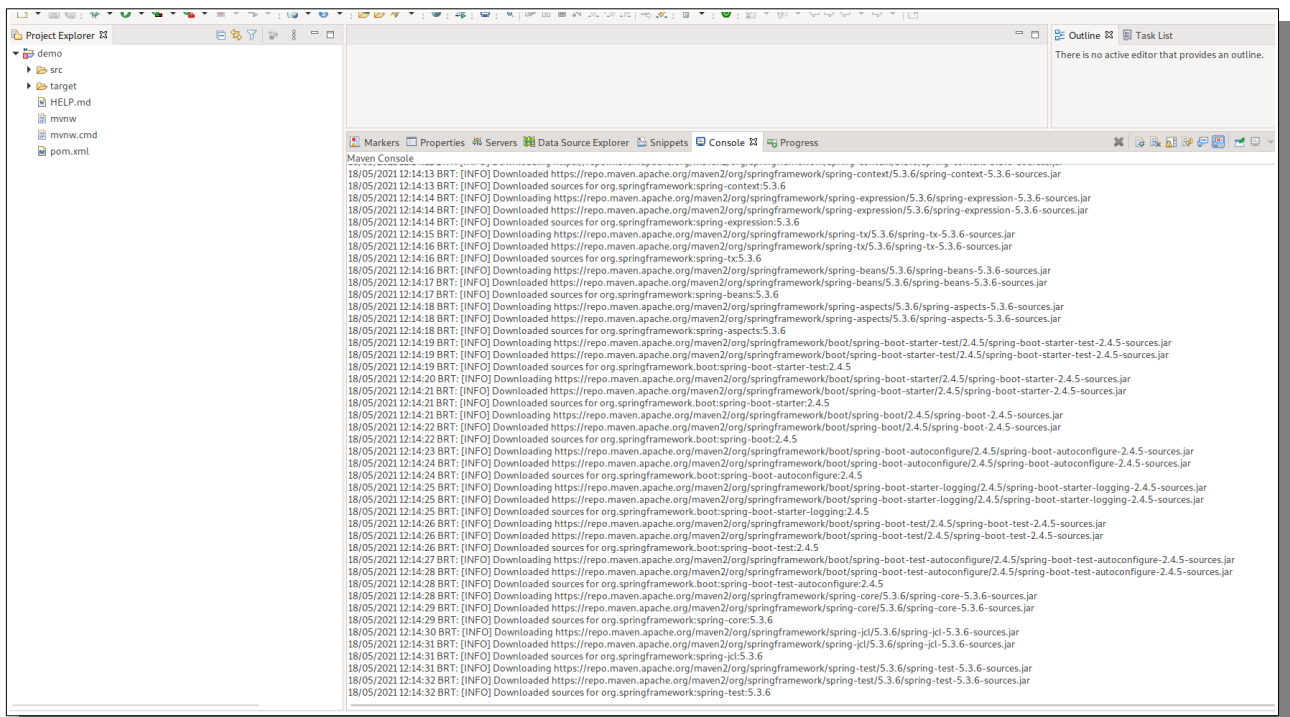
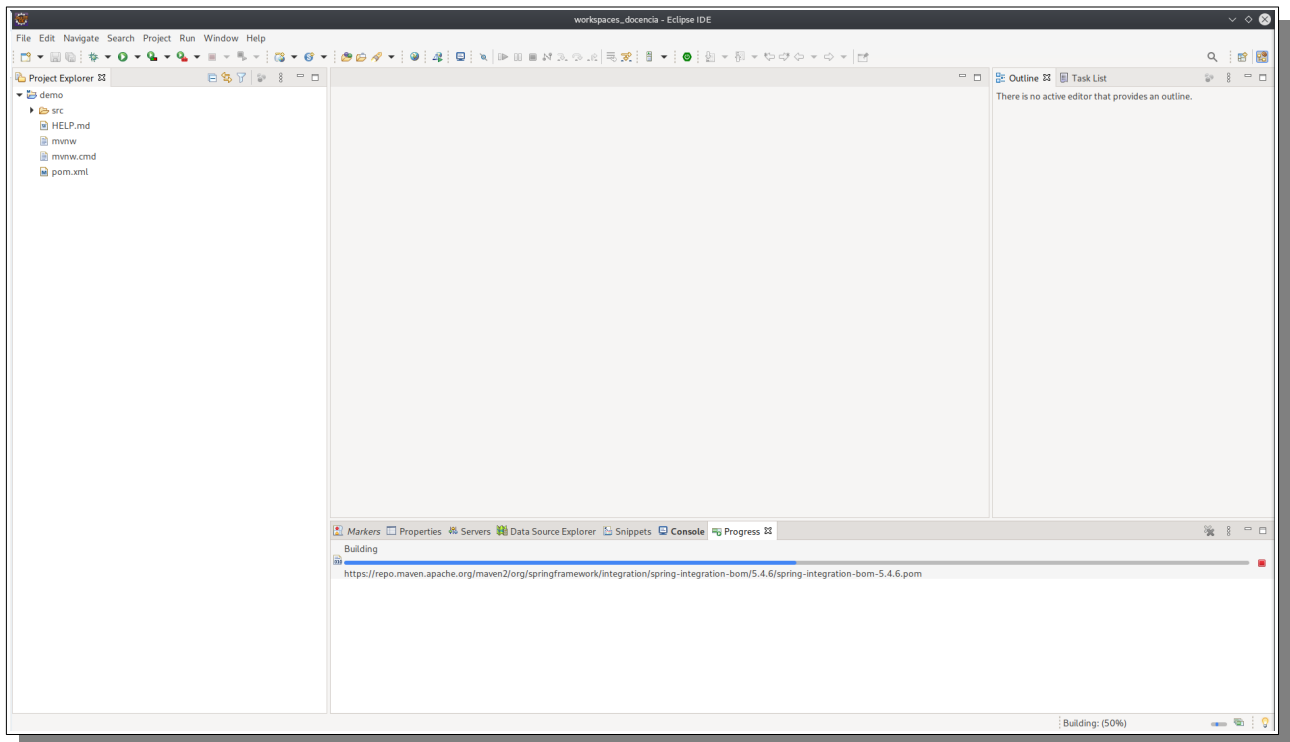




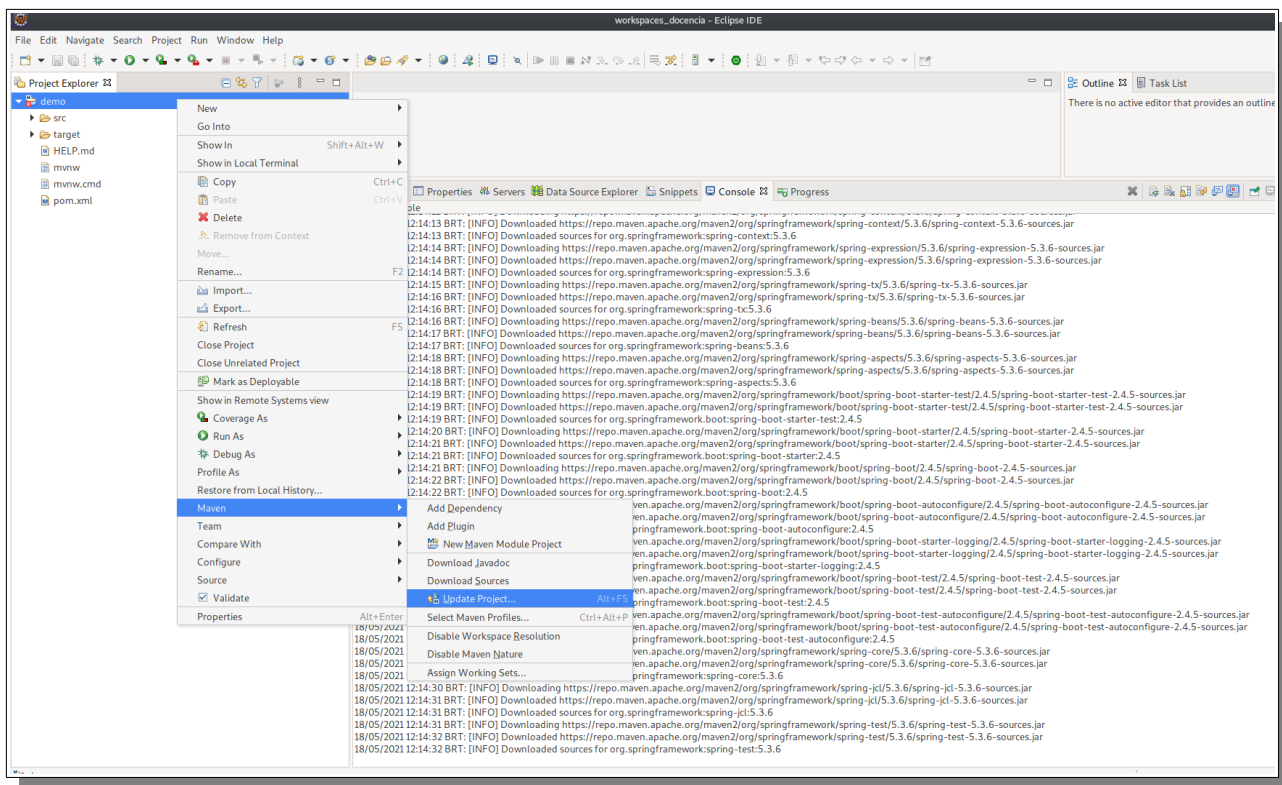
7. Seleção das API's/frameworks:



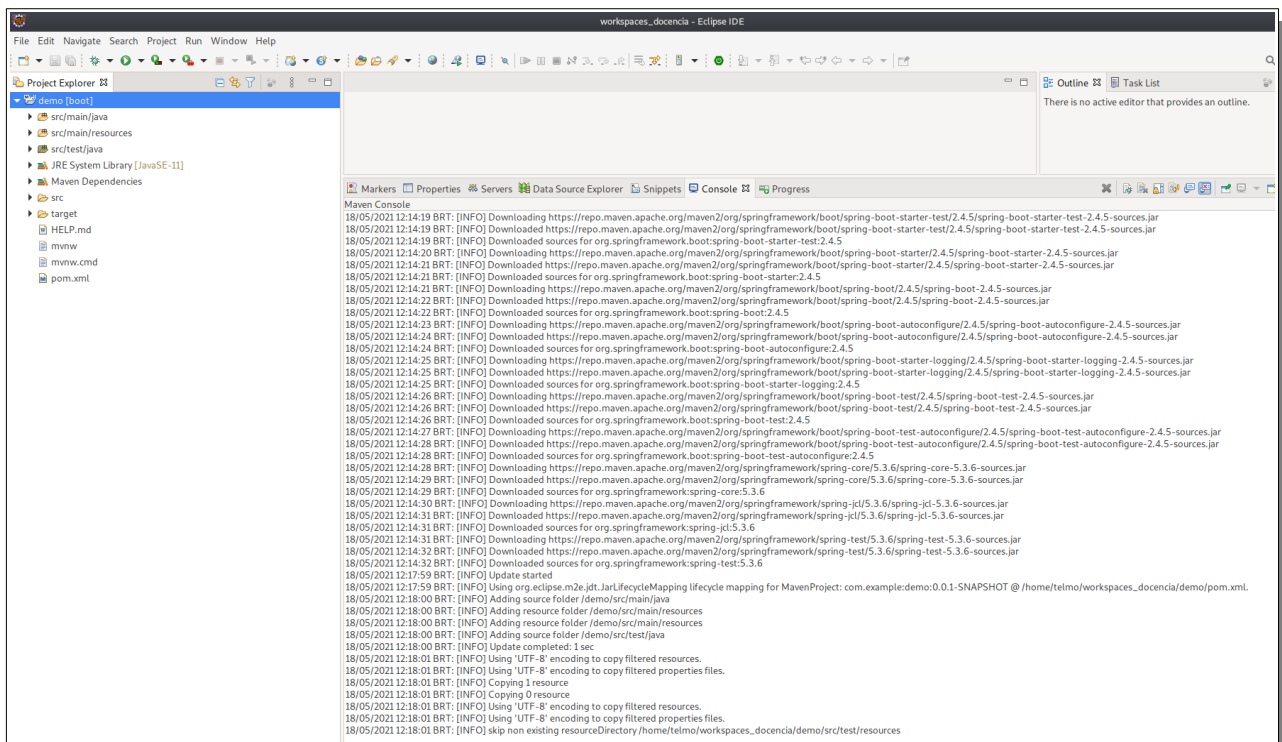
8. Aguardar o *download* das bibliotecas:



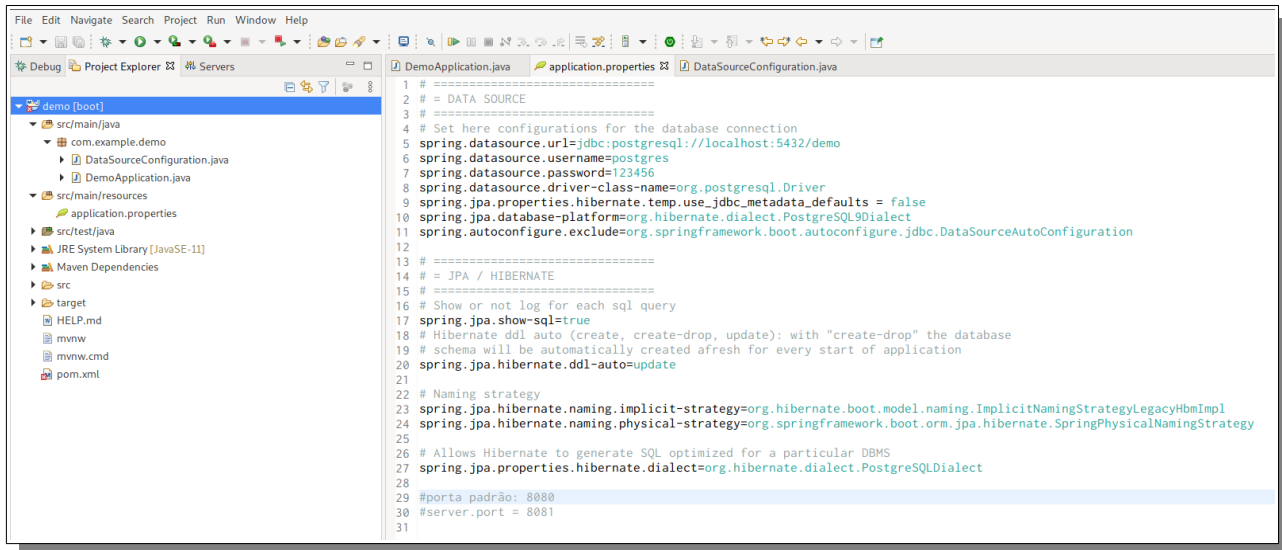
9. Atualizar novamente as listas das bibliotecas:



10. Projeto Pronto:

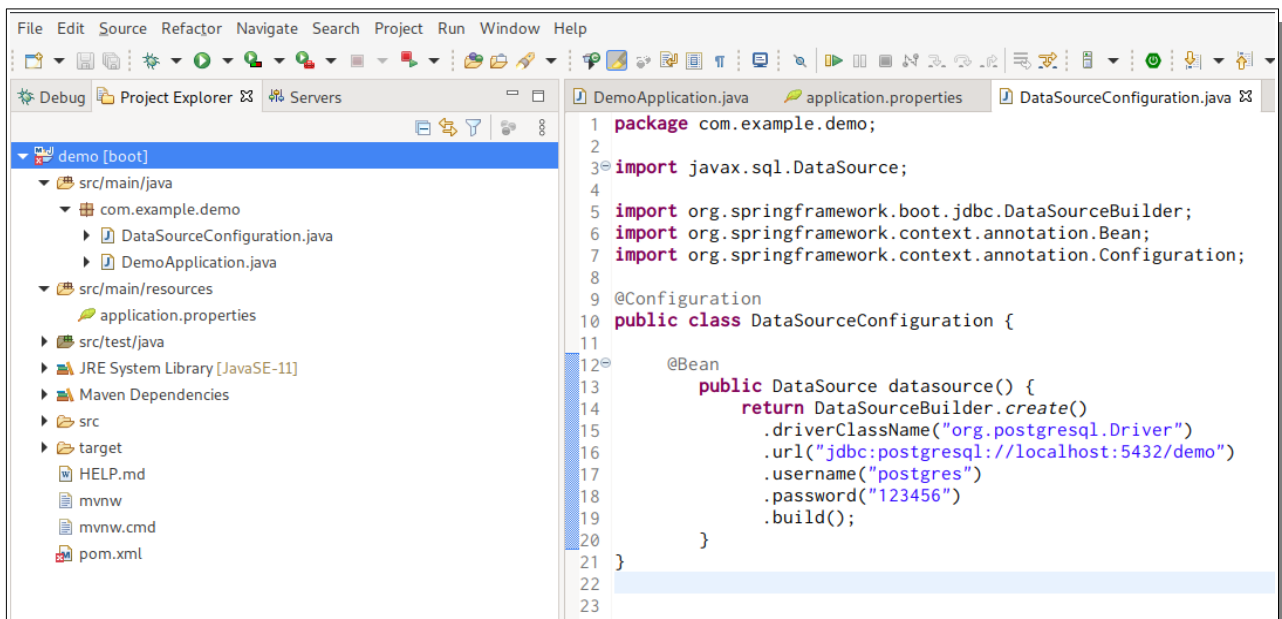


11. Configurar a conexão com o banco de dados:



The screenshot shows an IDE with the `application.properties` file open. The file contains configuration for a PostgreSQL database connection using Spring Boot. The Project Explorer on the left shows the project structure with `src/main/java/com/example/demo` containing `DataSourceConfiguration.java` and `DemoApplication.java`.

```
1 # =====
2 # = DATA SOURCE
3 # =====
4 # Set here configurations for the database connection
5 spring.datasource.url=jdbc:postgresql://localhost:5432/demo
6 spring.datasource.username=postgres
7 spring.datasource.password=123456
8 spring.datasource.driver-class-name=org.postgresql.Driver
9 spring.jpa.properties.hibernate.temp.use_jdbc_metadata_defaults = false
10 spring.jpa.database-platform=org.hibernate.dialect.PostgreSQL9Dialect
11 spring.autoconfigure.exclude=org.springframework.boot.autoconfigure.jdbc.DataSourceAutoConfiguration
12
13 # =====
14 # = JPA / HIBERNATE
15 # =====
16 # Show or not log for each sql query
17 spring.jpa.show-sql=true
18 # Hibernate ddl auto (create, create-drop, update): with "create-drop" the database
19 # schema will be automatically created afresh for every start of application
20 spring.jpa.hibernate.ddl-auto=update
21
22 # Naming strategy
23 spring.jpa.hibernate.naming.implicit-strategy=org.hibernate.boot.model.naming.ImplicitNamingStrategyLegacyHbmImpl
24 spring.jpa.hibernate.naming.physical-strategy=org.springframework.boot.orm.jpa.hibernate.SpringPhysicalNamingStrategy
25
26 # Allows Hibernate to generate SQL optimized for a particular DBMS
27 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
28
29 #porta padrão: 8080
30 #server.port = 8081
31
```



The screenshot shows the same IDE with the `DataSourceConfiguration.java` file open. The code defines a Spring Boot configuration class for the database connection. The Project Explorer on the left shows the project structure.

```
1 package com.example.demo;
2
3 import javax.sql.DataSource;
4
5 import org.springframework.boot.jdbc.DataSourceBuilder;
6 import org.springframework.context.annotation.Bean;
7 import org.springframework.context.annotation.Configuration;
8
9 @Configuration
10 public class DataSourceConfiguration {
11
12     @Bean
13     public DataSource datasource() {
14         return DataSourceBuilder.create()
15             .driverClassName("org.postgresql.Driver")
16             .url("jdbc:postgresql://localhost:5432/demo")
17             .username("postgres")
18             .password("123456")
19             .build();
20     }
21 }
22
23
```

createDDL.sql

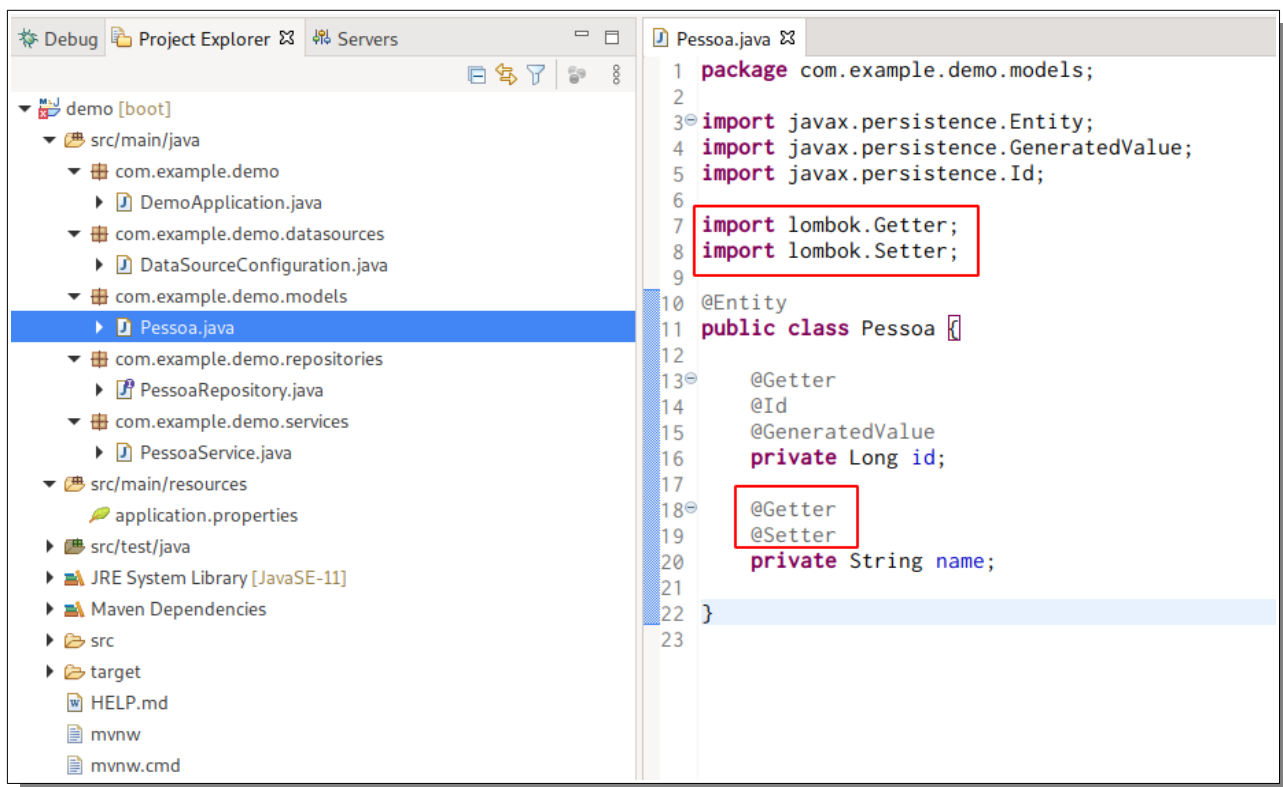
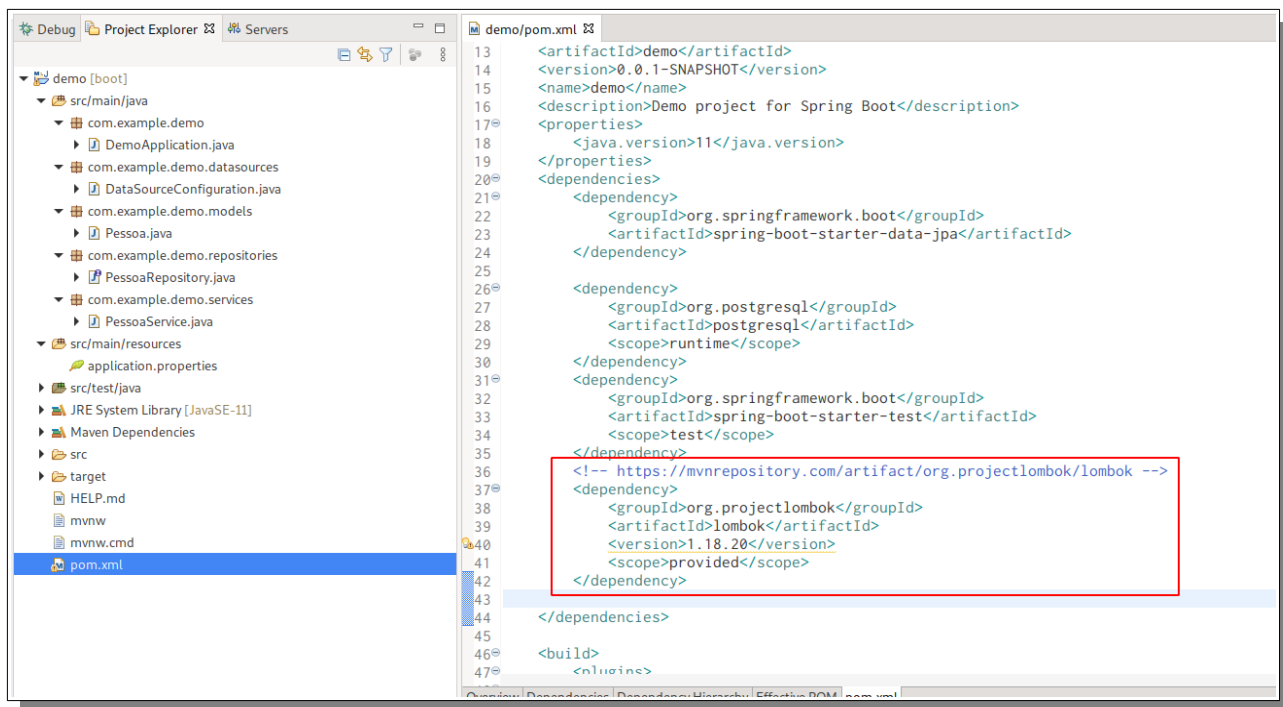
Connection profile

Type:

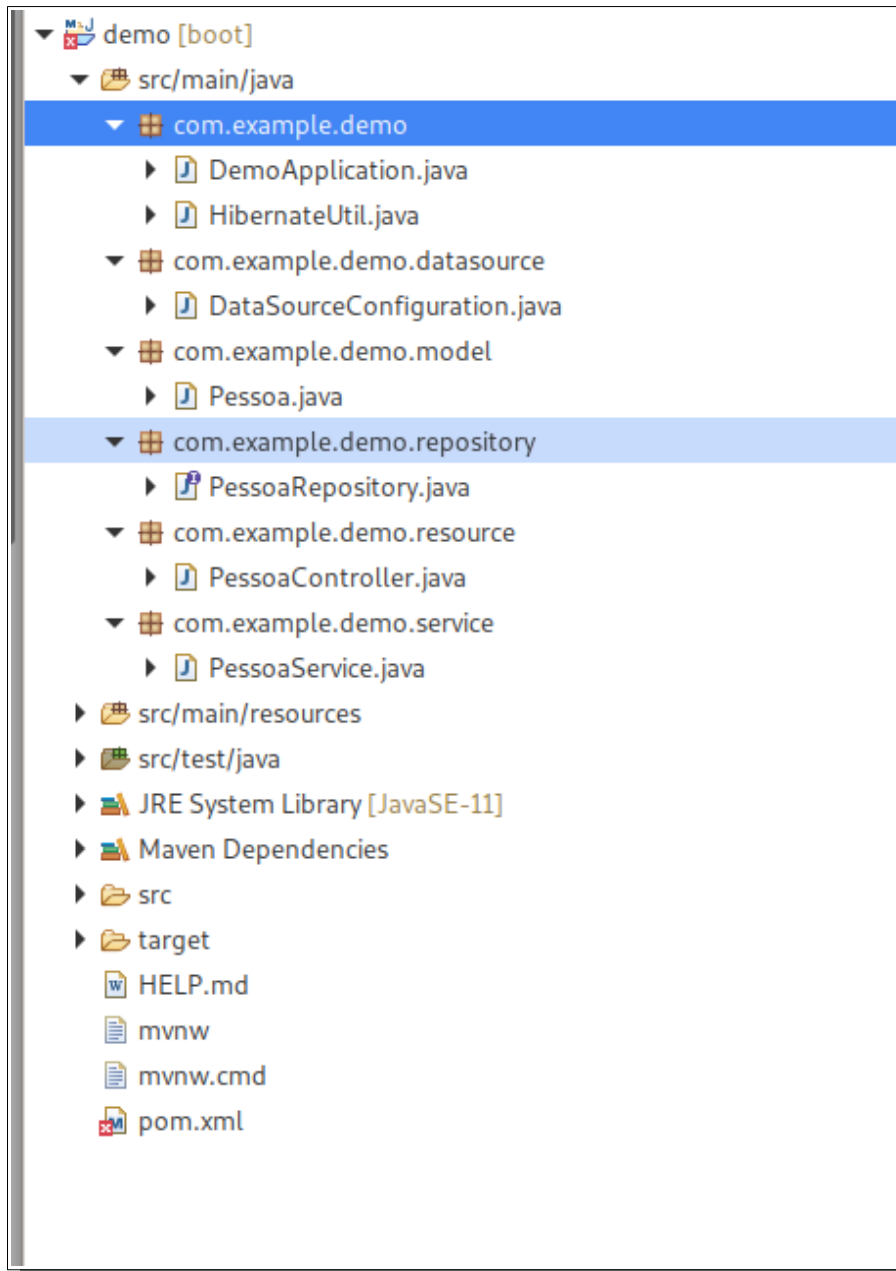
Name:

```
1 CREATE TABLE public.pessoa (  
2     id int4 NOT NULL,  
3     nome varchar(255) NOT NULL,  
4     email varchar(50) NOT NULL  
5 );  
6  
7 CREATE SEQUENCE public.gerador_pessoa_seq  
8     INCREMENT BY 1  
9     MINVALUE 1  
10    MAXVALUE 9223372036854775807  
11    START 1  
12    CACHE 1  
13    NO CYCLE;  
14  
15 INSERT INTO public.pessoa  
16 (id, nome, email)  
17 VALUES(1, 'Telmo Q Silva', 'telmoqs@gmail.com');  
18
```

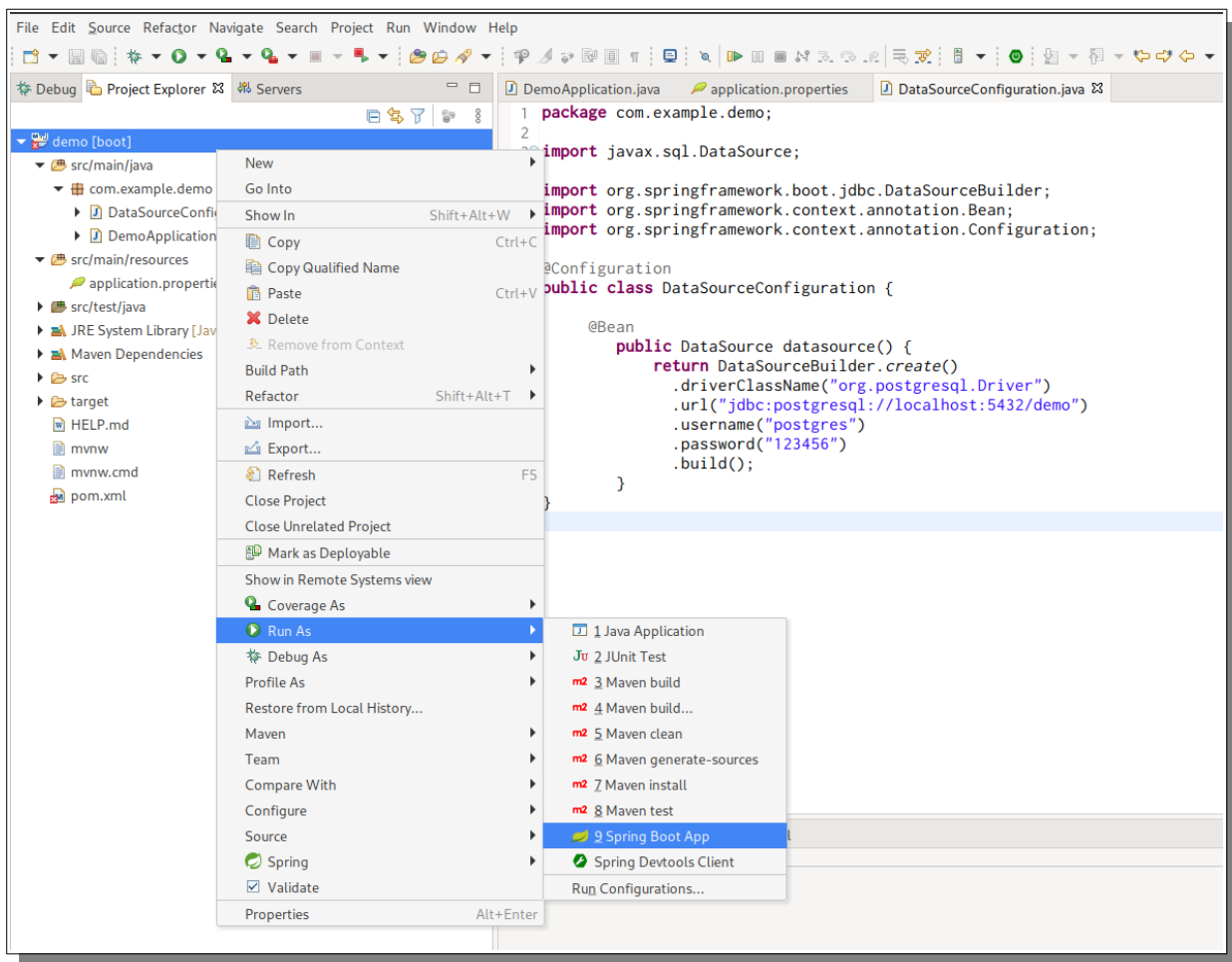
12. Biblioteca Lombok:



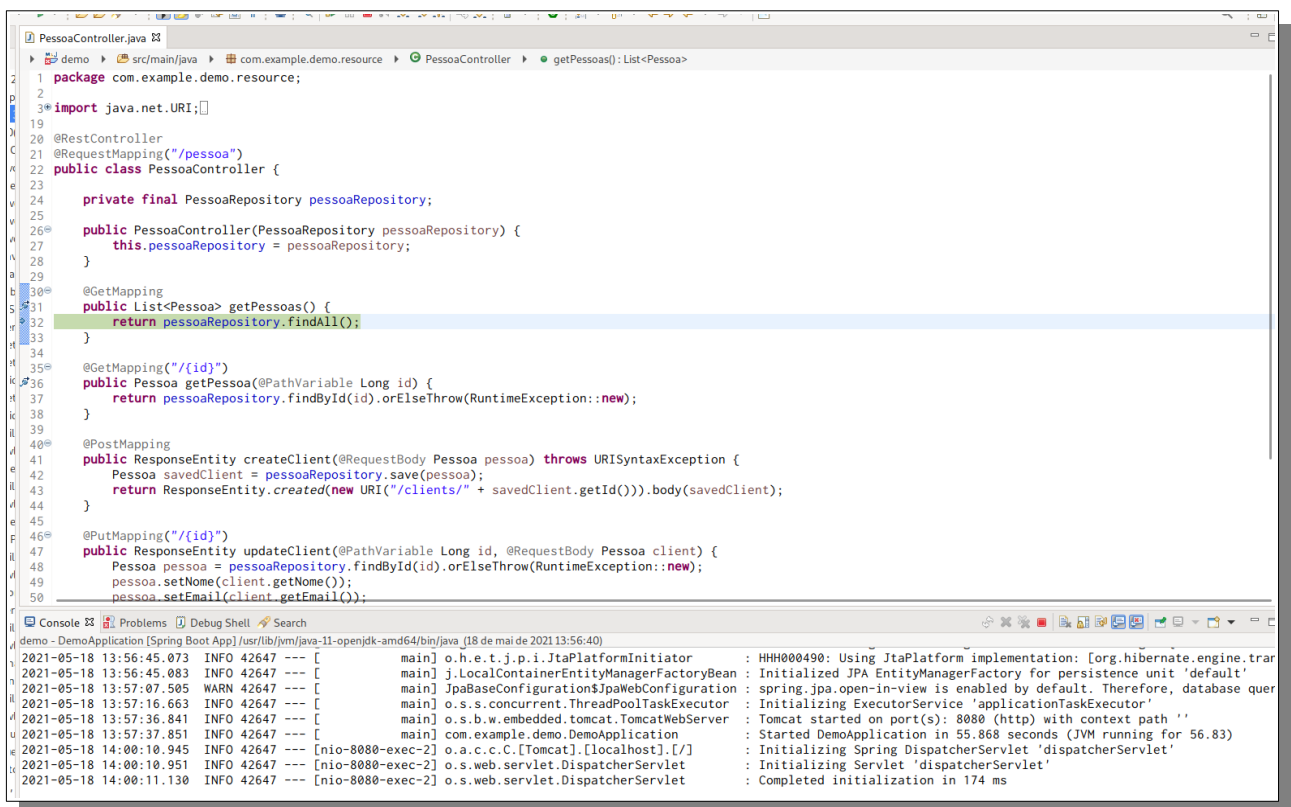
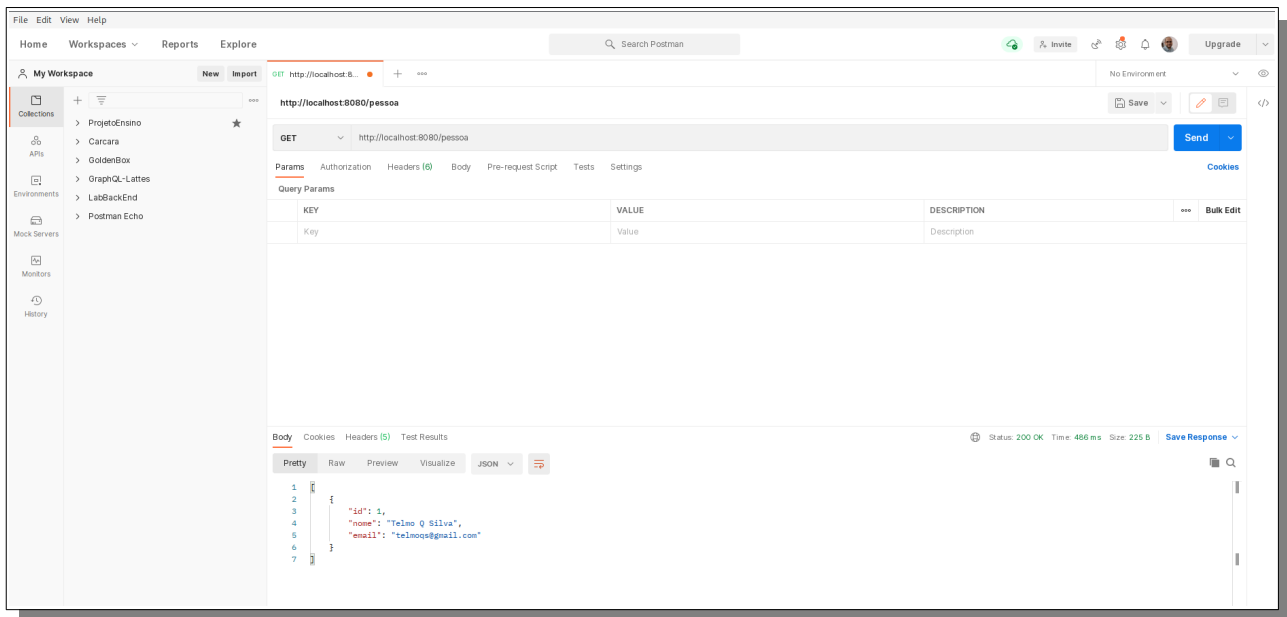
13.Estrutura do projeto:



14. Executando o projeto:



15. Testando com o Postman:



FileEditViewHelp

HomeWorkspacesReportsExplore

My WorkspaceNewImport

Collections

APIs

Environments

Mock Servers

Monitors

History

ProjetoEnsino

Carcara

GoldenBox

GraphQL-Lattes

LabBackEnd

Postman Echo

GEThttp://localhost:8080/pessoa/1

http://localhost:8080/pessoa/1

GEThttp://localhost:8080/pessoa/1

ParamsAuthorizationHeaders(6)BodyPre-request ScriptTestsSettings

Query Params

KEY	VALUE
Key	Value

BodyCookiesHeaders(5)Test Results

PrettyRawPreviewVisualizeJSON

```
1{"id": 1,
2  "nome": "Telmo Q Silva",
3  "email": "telmoqs@gmail.com"
4  }
5
```