

Swagger

- ▶ Swagger es un conjunto de herramientas de software de código abierto para diseñar, construir, documentar, y utilizar servicios web RESTful. Fue desarrollado por SmartBear Software e incluye documentación automatizada, generación de código, y generación de casos de prueba.
- ▶ Es una herramienta muy importante para la documentación de nuestra API REST.
- ▶ <https://swagger.io/tools/swagger-ui/>

Buscamos la versión mas estable para gradle

The screenshot shows a web browser displaying the Maven Repository website at mvnrepository.com/artifact/org.springdoc/springdoc-openapi-ui/1.6.4. The page title is "Interfaz de usuario OpenAPI de SpringDoc". The main content area includes a chart showing the number of projects indexed by year, a table of artifact details, and a note about a new version. The bottom section contains build tool support information and a code snippet for Gradle.

Artefactos indexados (30,4 millones)

Projects (millions) | Year

Year	Projects (millions)
2006	0.5
2008	1.0
2010	1.5
2012	2.5
2014	4.0
2015	5.0
2016	7.0
2017	10.0
2018	13.0

Categorías Populares

- Marcos de prueba
- Paquetes de Android
- Marcos de registro
- Especificaciones Java
- Bibliotecas JSON
- Utilidades principales
- Lenguajes JVM
- Burlón
- Tiempo de ejecución de
- Activos web
- Bibliotecas de anotaciones
- Puentes de registro
- Clientes HTTP

Interfaz de usuario OpenAPI de SpringDoc » 1.6.4

Interfaz de usuario OpenAPI de SpringDoc

Licencia	apache 2.0
Etiquetas	primavera abrirapi interfaz de usuario API
Fecha	06 de enero de 2022
archivos	pom (1 KB) tarro (15 KB) Ver todo
Repositorios	Central Mulesoft
Clasificación	#1285 en MvnRepository (ver artefactos principales)
Usado por	325 artefactos

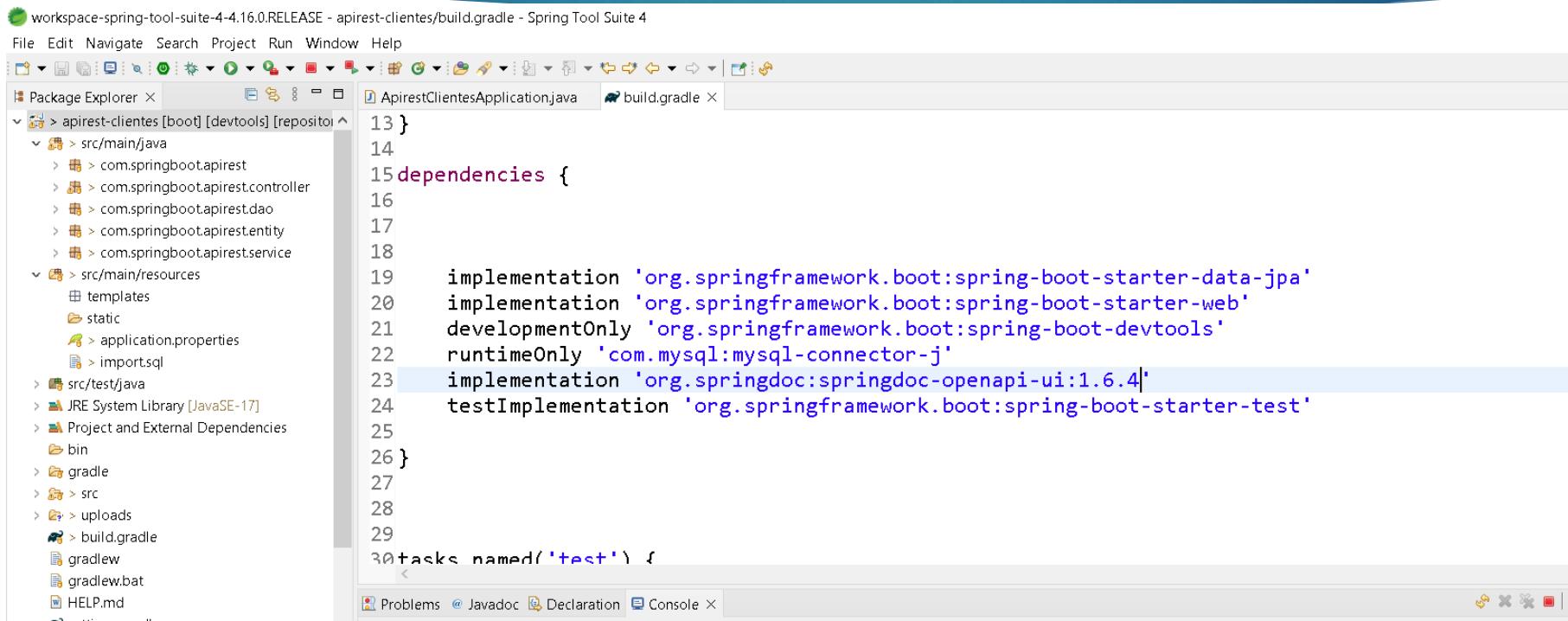
Nota: Hay una nueva versión para este artefacto.

Nueva versión	1.6.12
---------------	--------

Experto | gradle | Gradle (corto) | Gradle (Kotlin) | SBT | Hiedra | Uva | Leiningen | Construir

```
// https://mvnrepository.com/artifact/org.springdoc/springdoc-openapi-ui
implementation 'org.springdoc:springdoc-openapi-ui:1.6.4'
```

Añadimos la dependencia en gradle

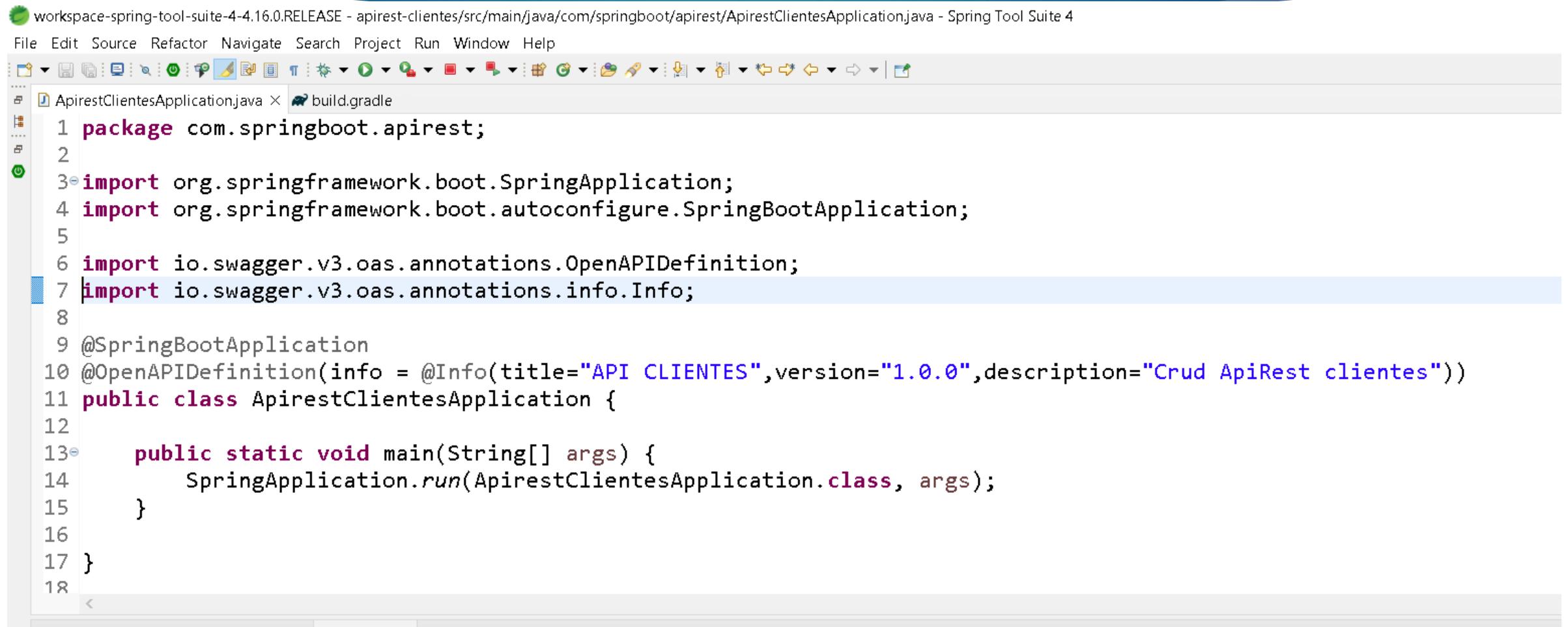


The screenshot shows the Spring Tool Suite 4 interface with the following details:

- Title Bar:** workspace-spring-tool-suite-4-4.16.0.RELEASE - apirest-clientes/build.gradle - Spring Tool Suite 4
- Menu Bar:** File Edit Navigate Search Project Run Window Help
- Toolbar:** Includes icons for file operations, search, and project navigation.
- Package Explorer:** Shows the project structure:
 - src/main/java
 - com.springboot.apirest
 - com.springboot.apirest.controller
 - com.springboot.apirest.dao
 - com.springboot.apirest.entity
 - com.springboot.apirest.service
 - src/main/resources
 - templates
 - static
 - application.properties
 - import.sql
 - src/test/java
 - JRE System Library [JavaSE-17]
 - Project and External Dependencies
 - bin
 - gradle
 - src
 - uploads
 - build.gradle
 - gradlew
 - gradlew.bat
 - HELP.md
- Editor:** The build.gradle file is open, showing the following code:

```
13 }
14
15 dependencies {
16
17
18
19     implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
20     implementation 'org.springframework.boot:spring-boot-starter-web'
21     developmentOnly 'org.springframework.boot:spring-boot-devtools'
22     runtimeOnly 'com.mysql:mysql-connector-j'
23     implementation 'org.springdoc:springdoc-openapi-ui:1.6.4'
24     testImplementation 'org.springframework.boot:spring-boot-starter-test'
25
26 }
27
28
29
30 tasks named('test') {
```
- Bottom Bar:** Problems, Javadoc, Declaration, Console

Configuramos la clase principal



The screenshot shows the Spring Tool Suite 4 interface with the title bar "workspace-spring-tool-suite-4-4.16.0.RELEASE - apirest-clientes/src/main/java/com/springboot/apirest/ApirestClientesApplication.java - Spring Tool Suite 4". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Build. The left sidebar shows project navigation with "ApirestClientesApplication.java" selected. The code editor displays the following Java code:

```
1 package com.springboot.apirest;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 import io.swagger.v3.oas.annotations.OpenAPIDefinition;
7 import io.swagger.v3.oas.annotations.info.Info;
8
9 @SpringBootApplication
10 @OpenAPIDefinition(info = @Info(title="API CLIENTES",version="1.0.0",description="Crud ApiRest clientes"))
11 public class ApirestClientesApplication {
12
13     public static void main(String[] args) {
14         SpringApplication.run(ApirestClientesApplication.class, args);
15     }
16
17 }
```

<http://localhost:8087/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config>

The screenshot shows the Swagger UI interface for a REST API. At the top, the URL is `localhost:8087/swagger-ui/index.html?configUrl=/v3/api-docs/swagger-config`. The main title is "API CLIENTES" version 1.0.0, OAS3. Below it, there's a link to `/v3/api-docs` and a note "Crud ApiRest clientes". The "Explore" button is visible on the right.

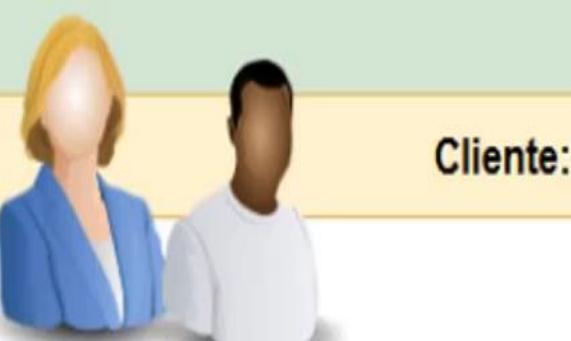
The "Servers" section shows a generated server URL: `http://localhost:8087 - Generated server url`.

The "cliente-controller" section lists the following endpoints:

- GET /api/clientes/{id}** (blue background)
- PUT /api/clientes/{id}** (orange background)
- DELETE /api/clientes/{id}** (red background)
- GET /api/clientes** (blue background)
- POST /api/clientes** (green background)

Que es JWT

- ▶ En el mismo se define un mecanismo para poder propagar entre dos partes, y de forma segura, la identidad de un determinado usuario, además con una serie de claims o privilegios.
- ▶ Estos privilegios están codificados en objetos de tipo JSON, que se incrustan dentro de del payload o cuerpo de un mensaje que va firmado digitalmente.
- ▶ <https://jwt.io/>



Cliente: Frontend



Servidor: API Rest

Authorization Server

POST: /api/login con username y password

Error Credenciales: 401 Unauthorized

Retorna el JWT al cliente: 200 OK

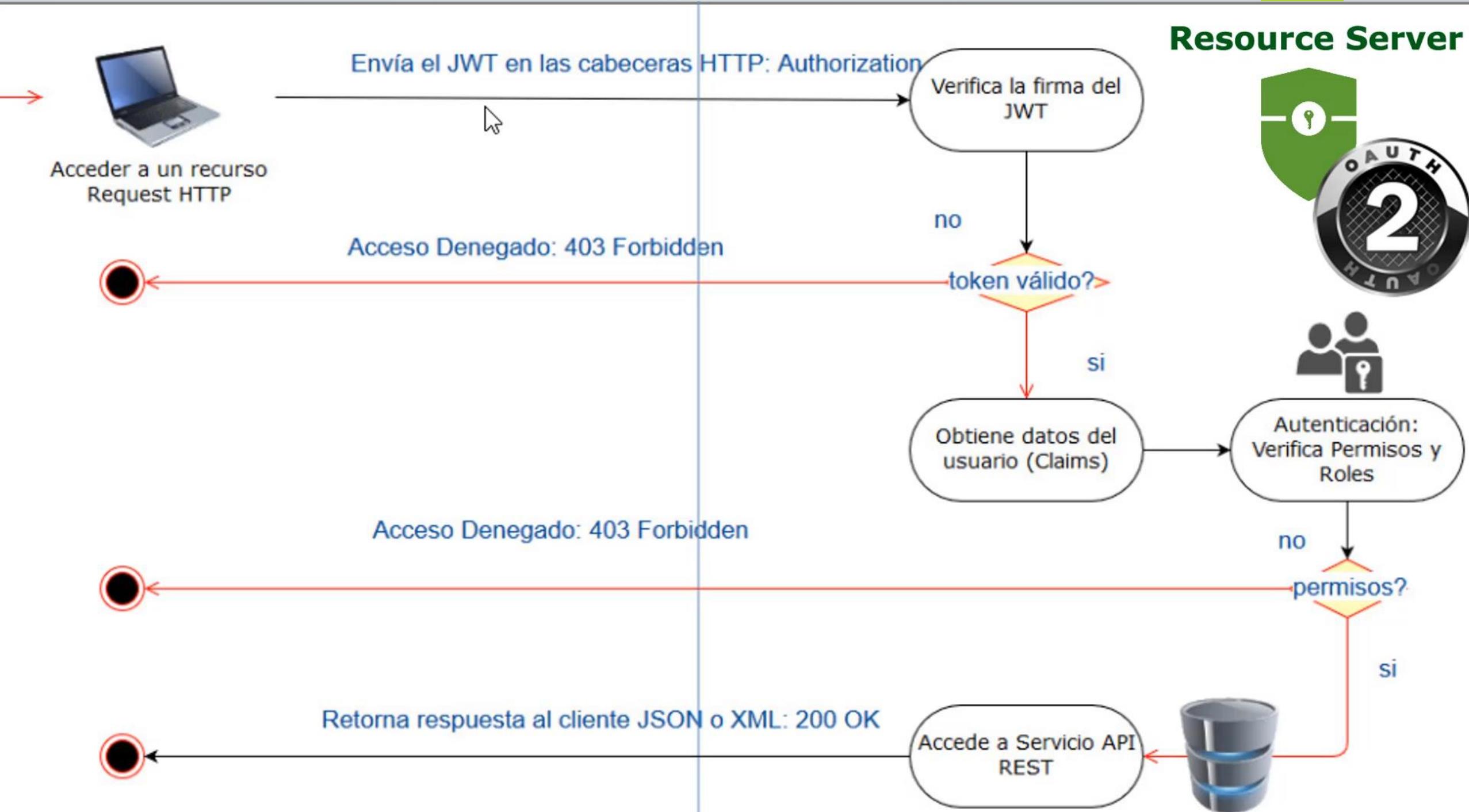
Valida autenticación
attemptAuthentication

autenticado?

si

Crea un JWT con un
SecretAlmacena JWT
LocalStore o
SessionStore

Resource Server



[Debugger](#) [Libraries](#) [Introduction](#) [Ask](#)

Crafted by auth0



JSON Web Tokens are an open, industry standard [RFC 7519](#) method for representing claims securely between two parties.

JWT.IO allows you to decode, verify and generate JWT.

[LEARN MORE ABOUT JWT](#)[SEE JWT LIBRARIES](#)

→ C jwt.io

Debugger Libraries Introduction Ask

Crafted by auth0 ?

Algorithm HS256

Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6I  
kpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ  
.SfIKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_a  
dQssw5c
```

Decoded

EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

PAYLOAD: DATA

```
{"  
  "sub": "1234567890",  
  "name": "John Doe",  
  "iat": 1516239022  
}
```

VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  your-256-bit-secret  
)  secret base64 encoded
```

[Spring Boot](#)[Spring Framework](#)[Spring Data](#) >[Spring Cloud](#) >[Spring Cloud Data Flow](#)[Spring Security](#) >[Spring Security Kerberos](#)[Spring Security OAuth](#)[Spring Security SAML](#)[Spring GraphQL](#)[Spring Session](#) >[Spring Integration](#)[Spring HATEOAS](#)

Spring Security OAuth

2.5.1.RELEASE

[OVERVIEW](#)[LEARN](#)

Deprecation Notice

The Spring Security OAuth project is deprecated. The latest OAuth 2.0 support is provided by Spring Security. See the [OAuth 2.0 Migration Guide](#) for further details.

Spring Security OAuth provides support for using Spring Security with OAuth (1a) and OAuth2 using standard Spring and Spring Security programming models and configuration idioms.

Features

- Support for OAuth providers and OAuth consumers

Características de Spring Security

- ▶ Provee características de seguridad para aplicaciones empresariales Java EE
- ▶ Maneja componentes de Autenticación y autorización

Spring Security

- ▶ Autenticación
- ▶ Autorización(Control de acceso)

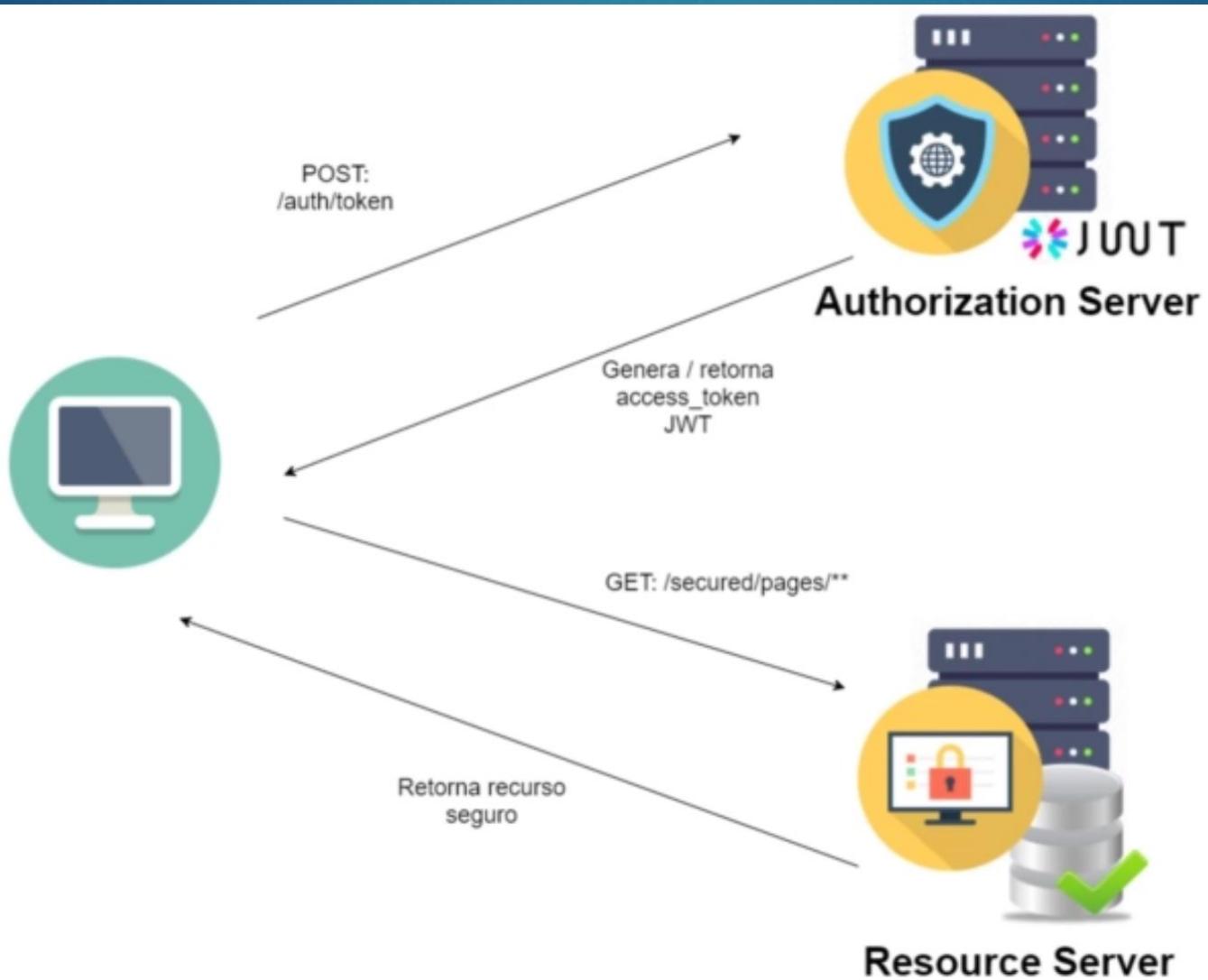
Autenticación

- ▶ Se refiere al proceso de establecer un principal(un principal significa un usuario, dispositivo o algún otro sistema el cual puede ejecutar alguna acción en nuestro sistema), en general permite a los principal autenticarse en base a cualquier proveedor de seguridad. Base de datos relacional principalmente y autenticación http.

Autorización

- ▶ Se refiere al proceso de decidir si se otorga acceso a un usuario para realizar una acción dentro de la aplicación, es decir para controlar el acceso a los recursos de la aplicación por medio de la asignación de roles y permisos a grupos de usuarios

OAuth



url: POST /auth/token

header:

- Authorization: Basic Base64(client_id:client_secret)
- Content-Type: application/x-www-form-urlencoded

body:

- grant_type = password
- username = andres
- password = 12345



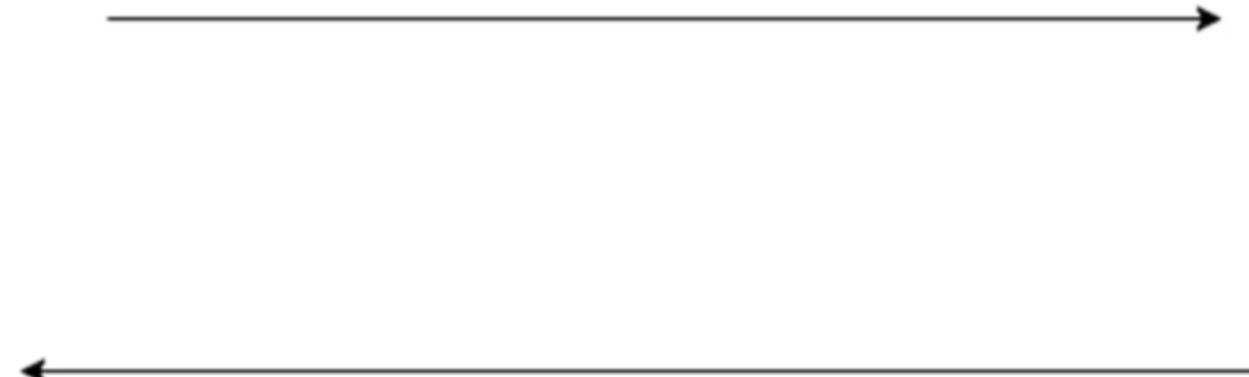
```
{  
  "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9...",  
  "token_type": "bearer",  
  "refresh_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9...",  
  "expires_in": 3599,  
  "scope": "read write",  
  "jti": "58efb674-46e6-4f6b-bbf0-e92e21e4b34a"  
}
```



Authorization Server

url: GET /api/clients

header: Authorization Bearer access_token

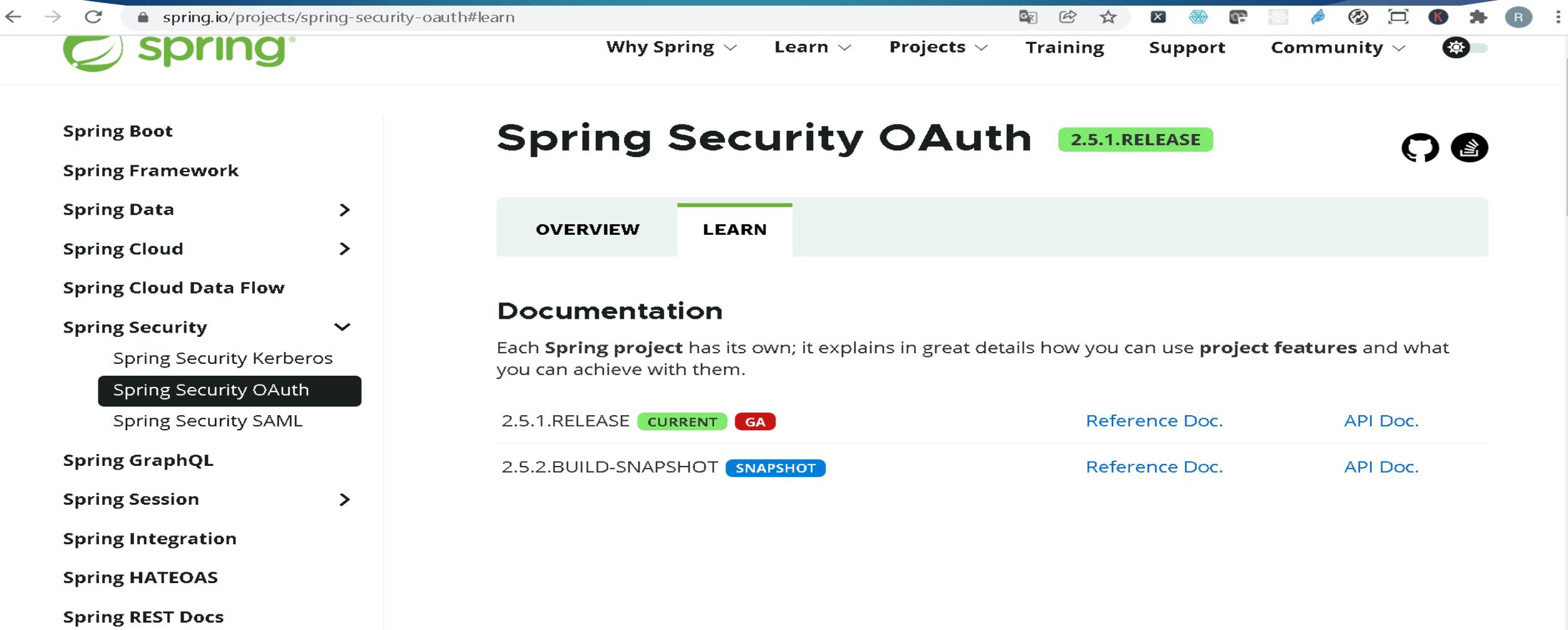


```
[  
  {  
    "id": 1,  
    "nombre": "Andrés",  
    "apellido": "Guzmán",  
    "email": "a@gmail.com"  
  
  },  
  {  
    "id": 2,  
    "nombre": "Mr. John",  
    "apellido": "Doe",  
    "email": "doe@gmail.com"  
  }  
]
```



Resource Server

Implementamos a nuestro proyecto



Spring Security OAuth x +

projects.spring.io/spring-security-oauth/docs/Home.html

The Spring Security OAuth project is deprecated. The latest OAuth 2.0 support is provided by Spring Security. See the [OAuth 2.0 Migration Guide](#) for further details.

Welcome

OAuth for Spring Security provides an [OAuth](#) implementation for [Spring Security](#). Support is provided for the implementation of OAuth providers and OAuth consumers. There is support for [Oauth 1\(a\)](#) (including [two-legged OAuth](#), a.k.a. "Signed Fetch") and for [OAuth 2.0](#).

Applying security to an application is not for the faint of heart, and OAuth is no exception. Before you get started, you're going to want to make sure you understand OAuth and the problem it's designed to address. There is good documentation at [the OAuth site](#). You will also want to make sure you understand how [Spring](#) and [Spring Security](#) work.

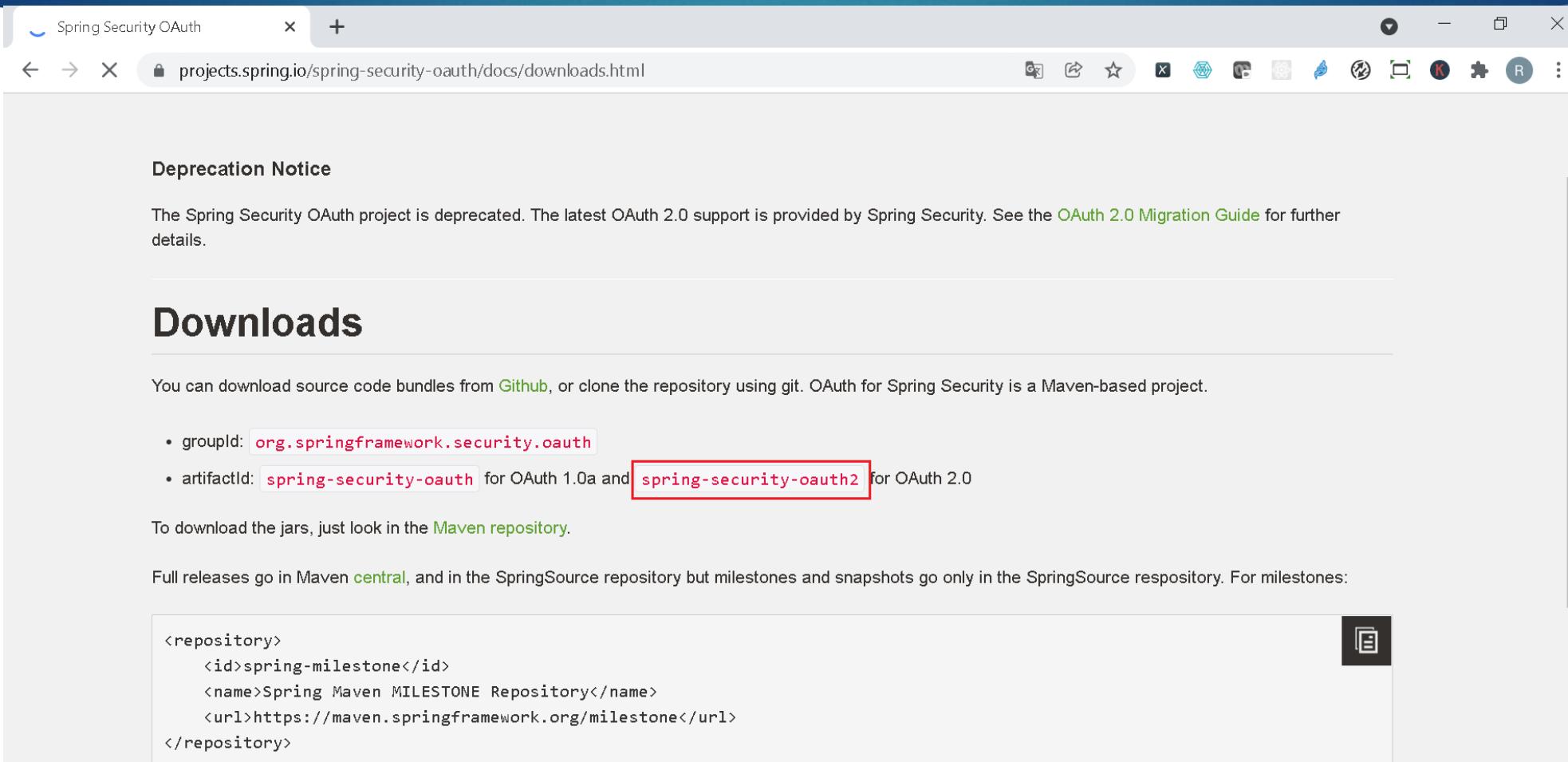
You're going to want to be quite familiar with both [OAuth](#) (and/or [OAuth2](#)) and [Spring Security](#), to maximize the effectiveness of this developers guide. OAuth for Spring Security is tightly tied to both technologies, so the more familiar you are with them, the more likely you'll be to recognize the terminology and patterns that are used.

With that, you're ready to get started. Here are some useful links:

- For access to the binaries, use Maven ([instructions here](#))
- Source code is in github [at spring-projects/spring-security-oauth](#).
- You'll want to see OAuth for Spring Security in action, so here is a tutorial

<https://projects.spring.io/spring-security-oauth/docs/downloads.html>

Copiamos esto



The screenshot shows a web browser window with the title "Spring Security OAuth". The address bar contains the URL "projects.spring.io/spring-security-oauth/docs/downloads.html". The page content includes a "Deprecation Notice" section stating that the project is deprecated and pointing to the "OAuth 2.0 Migration Guide". Below this is a "Downloads" section with instructions for Maven-based projects, listing group ID and artifact ID options. A code block shows Maven repository configuration for milestones.

Deprecation Notice

The Spring Security OAuth project is deprecated. The latest OAuth 2.0 support is provided by Spring Security. See the [OAuth 2.0 Migration Guide](#) for further details.

Downloads

You can download source code bundles from [Github](#), or clone the repository using git. OAuth for Spring Security is a Maven-based project.

- groupId: `org.springframework.security.oauth`
- artifactId: `spring-security-oauth` for OAuth 1.0a and `spring-security-oauth2` for OAuth 2.0

To download the jars, just look in the [Maven repository](#).

Full releases go in Maven [central](#), and in the SpringSource repository but milestones and snapshots go only in the SpringSource repository. For milestones:

```
<repository>
  <id>spring-milestone</id>
  <name>Spring Maven MILESTONE Repository</name>
  <url>https://maven.springframework.org/milestone</url>
</repository>
```

and for snapshots:

Y lo buscamos en el repositorio Maven

mvnrepository.com/search?q=spring-security-oauth2

spring-security-oauth2

Search

Categories | Popular | Contact Us

Repository

- Central 21.6k
- Sonatype 4.6k
- Spring Plugins 3.2k
- Spring Lib M 2.9k
- JCenter 925
- Alfresco 792
- Spring Releases 533
- Spring Milestones 510

Group

- com.github 3.1k
- org.springframework.org.apache 3.0k
- io.github 846
- org.jboss 259
- org.wso2 233
- org.wildfly 185
- org.kie 171

Category

- Web App 333

Found 26952 results

Sort: relevance | popular | newest

1. OAuth2 For Spring Security
org.springframework.security.oauth > spring-security-oauth2
Module for providing OAuth2 support to Spring Security
Last Release on Apr 9, 2021

402 usages Apache

2. Spring Security OAuth2 AutoConfigure
org.springframework.security.oauth.boot > spring-security-oauth2-autoconfigure
spring-security-oauth2-autoconfigure
Last Release on Nov 2, 2021

108 usages Apache

3. Spring Security OAuth2 JOSE
org.springframework.security > spring-security-oauth2-jose
Spring Security
Last Release on Nov 15, 2021

101 usages Apache

4. Spring Security OAuth2 Client
org.springframework.security > spring-security-oauth2-client

76 usages Apache

VULTR

Stop overpaying for AWS - try Vultr's powerful Cloud Compute for free instead! Claim your \$100 credit.

ADS VIA CARBON

Indexed Repositories (1351)

- Central
- Sonatype
- Atlassian
- Spring Plugins
- Hortonworks
- Spring Lib M
- JCenter
- Atlassian Public
- JBossEA
- BeDataDriven

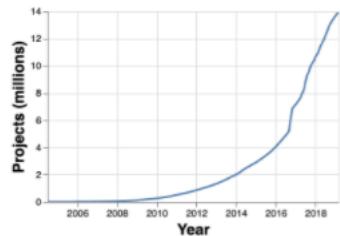


Search for groups, artifacts, categories

Search

Categories | Popular | Contact Us

Indexed Artifacts (24.5M)



Popular Categories

- Aspect Oriented
- Actor Frameworks
- Application Metrics
- Build Tools
- Bytecode Libraries
- Command Line Parsers
- Cache Implementations
- Cloud Computing
- Code Analyzers
- Collections
- Configuration Libraries
- Core Utilities
- Date and Time Utilities
- Dependency Injection

[Home](#) » [org.springframework.security.oauth](#) » [spring-security-oauth2](#)

OAuth2 For Spring Security

Module for providing OAuth2 support to Spring Security

License	Apache 2.0
Categories	OAuth Libraries
Tags	security spring authentication oauth
Used By	402 artifacts

[Central \(54\)](#)[Spring Releases \(1\)](#)[Spring Plugins \(20\)](#)[ICM \(2\)](#)[EBIPublic \(1\)](#)[SpringFramework \(2\)](#)

	Version	Repository	Usages	Date
2.5.x	2.5.1.RELEASE	Central	13	Apr, 2021
	2.5.0.RELEASE	Central	30	May, 2020
2.4.x	2.4.2.RELEASE	Central	0	Oct, 2021
	2.4.1.RELEASE	Central	11	Apr, 2020
	2.4.0.RELEASE	Central	28	Nov, 2019
	2.3.8.RELEASE	Central	25	Nov, 2019
	2.3.7.RELEASE	Central	18	Oct, 2019



Ship your code to production in just a few clicks. Get \$100 free credit.

ADS VIA CARBON

Indexed Repositories (1351)

- Central
- Sonatype
- Atlassian
- Spring Plugins
- Hortonworks
- Spring Lib M
- JCenter
- Atlassian Public
- JBossEA
- BeDataDriven

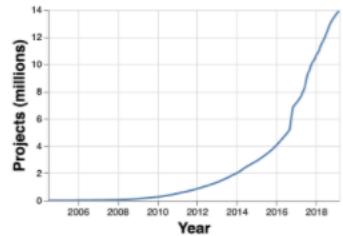


Search for groups, artifacts, categories

Search

Categories | Popular | Contact Us

Indexed Artifacts (24.5M)



Popular Categories

Aspect Oriented

Actor Frameworks

Application Metrics

Build Tools

Bytecode Libraries

Command Line Parsers

Cache Implementations

Cloud Computing

Code Analyzers

Collections

Configuration Libraries

Core Utilities

Date and Time Utilities

Dependency Injection

Home » org.springframework.security.oauth » spring-security-oauth2 » 2.5.1.RELEASE



OAuth2 For Spring Security » 2.5.1.RELEASE

Module for providing OAuth2 support to Spring Security

License

Apache 2.0

Categories

OAuth Libraries

Date

(Apr 09, 2021)

Files

jar (491 KB) [View All](#)

Repositories

Central | Scala-SBT

Used By

402 artifacts

Maven

Gradle

Gradle (Short)

Gradle (Kotlin)

SBT

Ivy

Grape

Leiningen

Buildr

```
<!-- https://mvnrepository.com/artifact/org.springframework.security.oauth/spring-security-oauth2 -->
<dependency>
    <groupId>org.springframework.security.oauth</groupId>
    <artifactId>spring-security-oauth2</artifactId>
    <version>2.5.1.RELEASE</version>
</dependency>
```

 Include comment with link to declaration

Copied to clipboard!



What if Your Project Management Tool Was Fast and Intuitive? Try Shortcut (formerly Clubhouse).

ADS VIA CARBON

Indexed Repositories (1351)

Central

Sonatype

Atlassian

Spring Plugins

Hortonworks

Spring Lib M

JCenter

Atlassian Public

JBossEA

BeDataDriven

Repository

- Central 21.6k
- Sonatype 4.5k
- Spring Plugins 3.2k
- Spring Lib M 2.9k
- JCenter 883
- Alfresco 792
- Spring Releases 533
- Spring Milestones 510

Group

- com.github 3.1k
- org.springframework 3.0k
- org.apache 1.8k
- io.github 845
- org.jboss 260
- org.wso2 204
- org.wildfly 192
- org.kie 171

Category

- Web App 326

Found 26910 resultsSort: [relevance](#) | [popular](#) | [newest](#)**1. Spring Security JWT Library**[org.springframework.security > spring-security-jwt](#)

128 usages

Apache

Spring Security JWT is a small utility library for encoding and decoding JSON Web Tokens. It belongs to the family of Spring Security crypto libraries that handle encoding and decoding text as a general, useful thing to be able to do.

Last Release on May 28, 2020

**2. Spring Context**[org.springframework > spring-context](#)

11,155 usages

Apache

Spring Context

Last Release on Nov 11, 2021

**3. Spring Core**[org.springframework > spring-core](#)

6,925 usages

Apache

Spring Core

Last Release on Nov 11, 2021



Shortcut

Shortcut provides speedy task management, reporting, and collaboration for developers. Try it free today.

ADS VIA CARBON

Indexed Repositories (1351)

- Central
- Sonatype
- Atlassian
- Spring Plugins
- Hortonworks
- Spring Lib M
- JCenter
- Atlassian Public
- JBossEA
- BeDataDriven

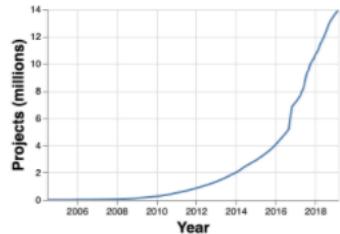


Search for groups, artifacts, categories

Search

Categories | Popular | Contact Us

Indexed Artifacts (24.5M)



Popular Categories

- Aspect Oriented
- Actor Frameworks
- Application Metrics
- Build Tools
- Bytecode Libraries
- Command Line Parsers
- Cache Implementations
- Cloud Computing
- Code Analyzers
- Collections
- Configuration Libraries
- Core Utilities
- Date and Time Utilities
- Dependency Injection

Home » org.springframework.security » spring-security-jwt



Spring Security JWT Library

Spring Security JWT is a small utility library for encoding and decoding JSON Web Tokens. It belongs to the family of Spring Security crypto libraries that handle encoding and decoding text as a general, useful thing to be able to do.

License	Apache 2.0
Categories	JWT Libraries
Tags	security json spring jwt
Used By	128 artifacts

[Central \(14\)](#) [Spring Plugins \(3\)](#) [ICM \(2\)](#)

	Version	Repository	Usages	Date
1.1.x	1.1.1.RELEASE	Central	18	May, 2020
	1.1.0.RELEASE	Central	14	Nov, 2019
	1.0.11.RELEASE	Central	14	Oct, 2019
	1.0.10.RELEASE	Central	36	Jan, 2019
	1.0.9.RELEASE	Central	25	Dec, 2017
	1.0.8.RELEASE	Central	19	May, 2017
	1.0.7.RELEASE	Central	15	Nov, 2016



Pick a React Data Grid that makes your life easier - and get the power, speed and support your app needs.

ADS VIA CARBON

Indexed Repositories (1351)

- Central
- Sonatype
- Atlassian
- Spring Plugins
- Hortonworks
- Spring Lib M
- JCenter
- Atlassian Public
- JBossEA
- BeDataDriven

← → C mvnrepository.com/artifact/org.springframework.security/spring-security-jwt/1.1.1.RELEASE

MVNREPOSITORY

Search for groups, artifacts, categories

Search

Categories | Popular | Contact Us

Indexed Artifacts (24.5M)

Year	Projects (millions)
2006	0.5
2008	1.0
2010	1.5
2012	2.5
2014	5.0
2016	8.0
2018	13.0

Home » org.springframework.security » spring-security-jwt » 1.1.1.RELEASE

Spring Security JWT Library » 1.1.1.RELEASE

Spring Security JWT is a small utility library for encoding and decoding JSON Web Tokens. It belongs to the family of Spring Security crypto libraries that handle encoding and decoding text as a general, useful thing to be able to do.

License	Apache 2.0
Categories	JWT Libraries
Organization	SpringSource
HomePage	https://github.com/spring-projects/spring-security-oauth
Date	(May 28, 2020)
Files	jar (37 KB) View All
Repositories	Central
Used By	128 artifacts

Maven Gradle Gradle (Short) Gradle (Kotlin) SBT Ivy Gape Leiningen Buildr

```
<!-- https://mvnrepository.com/artifact/org.springframework.security/spring-security-jwt -->
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-jwt</artifactId>
    <version>1.1.1.RELEASE</version>
</dependency>
```

Try for free and transform a proof-of-concept into a large-scale deployment powering mission-critical systems.

ADS VIA CARBON

Indexed Repositories (1351)

- Central
- Sonatype
- Atlassian
- Spring Plugins
- Hortonworks
- Spring Lib M
- JCenter
- Atlassian Public
- JBossEA
- BeDataDriven

mvnrepository.com/search?q=jaxb

Categories | Popular | Contact Us

Repository

- Central 683
- Sonatype 191
- Spring Lib M 125
- Spring Plugins 120
- Redhat GA 50
- JBoss Releases 36
- IBiblio 30
- Spring Lib Release 27

Group

- org.apache 57
- org.jvnet 46
- com.helger 37
- com.sun 36
- com.github 34
- org.glassfish 31
- io.github 17
- (s) janstey.sources 17

Category

jaxb

Search

Found 943 results

Sort: relevance | popular | newest

1. **JAXB API**
javax.xml.bind » jaxb-api
JAXB API
Last Release on Aug 30, 2018

4,741 usages
CDDL

2. **Old JAXB Runtime**
com.sun.xml.bind » jaxb-impl
Old JAXB Runtime module. Contains sources required for runtime processing.
Last Release on Aug 3, 2021

2,746 usages
EDL

3. **JAXB Runtime**
org.glassfish.jaxb » jaxb-runtime
JAXB (JSR 222) Reference Implementation
Last Release on Aug 3, 2021

1,549 usages
EDL

4. **Old JAXB Core**
sun.xml.bind » jaxb-core
1,149 usages
EDL

ADS VIA CARBON

elastic
Search that just works
Get started

The Elastic Search Platform is an all-in-one data store, search engine, and analytics solution.

Learn More!

Indexed Repositories (1351)

- Central
- Sonatype
- Atlassian
- Spring Plugins
- Hortonworks
- Spring Lib M
- JCenter
- Atlassian Public
- JBossEA
- BeDataDriven

<https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api>

Para jaxb-api y jaxb-runtime no necesitamos especificar la version

eclipse-workspace - springboot-apirest/pom.xml - Eclipse IDE

File Edit Navigate Project Run Window Help

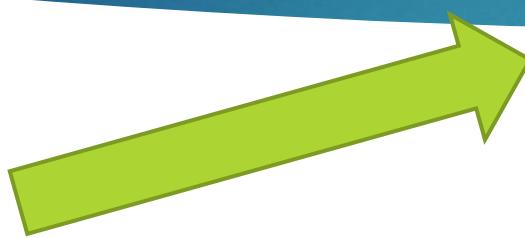
Project Explorer *springboot-apirest/pom.xml

```
<!-- https://mvnrepository.com/artifact/org.springframework.security.oauth/spring-security-oauth2 -->
<dependency>
    <groupId>org.springframework.security.oauth</groupId>
    <artifactId>spring-security-oauth2</artifactId>
    <version>2.5.1.RELEASE</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework.security/spring-security-jwt -->
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-jwt</artifactId>
    <version>1.1.1.RELEASE</version>
</dependency>

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->
<dependency>
    <groupId>javax.xml.bind</groupId>
    <artifactId>jaxb-api</artifactId>
</dependency>

<!-- https://mvnrepository.com/artifact/org.glassfish.jaxb/jaxb-runtime -->
<dependency>
    <groupId>org.glassfish.jaxb</groupId>
    <artifactId>jaxb-runtime</artifactId>
</dependency>
```

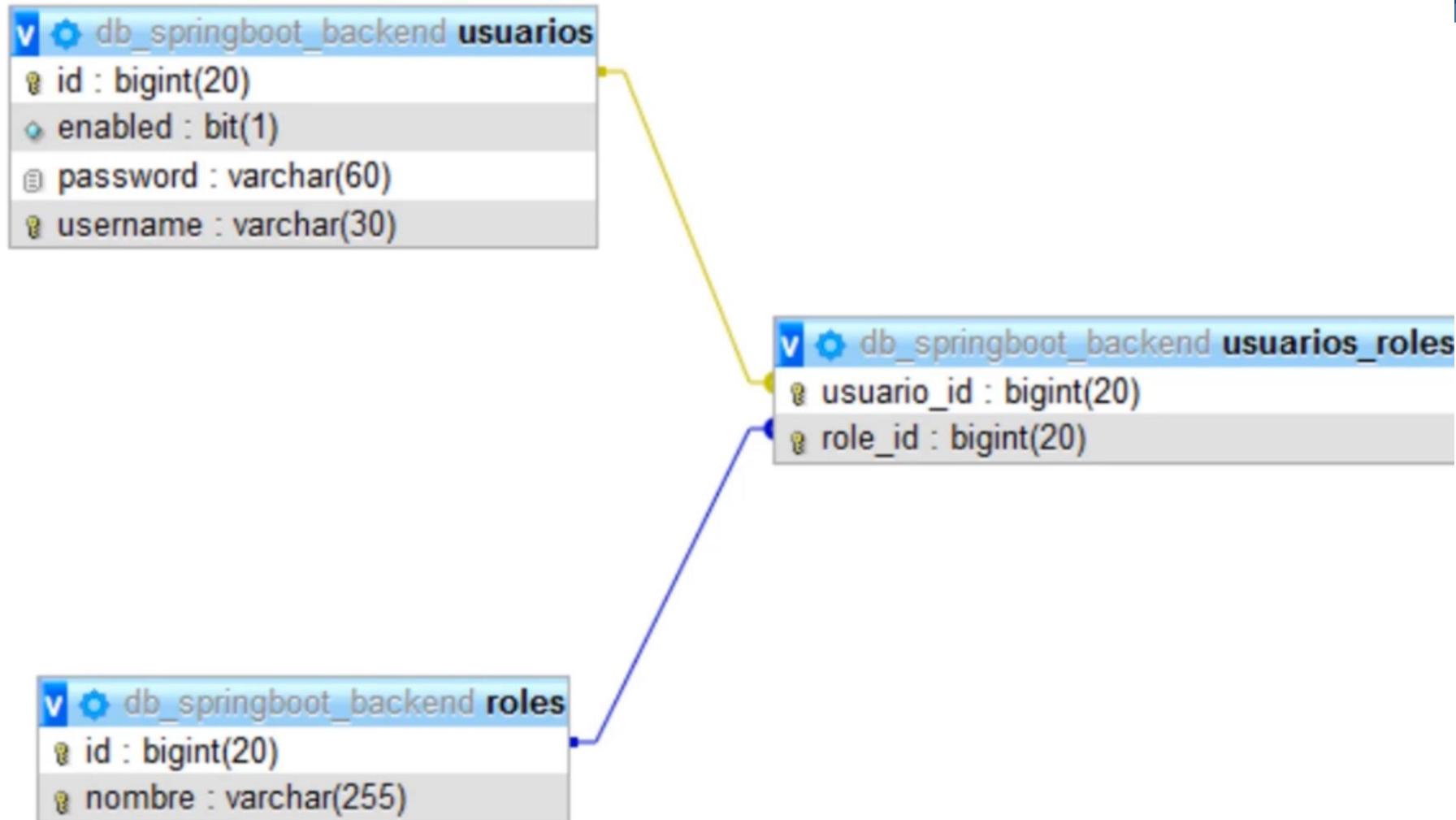
Actualizamos las dependencias



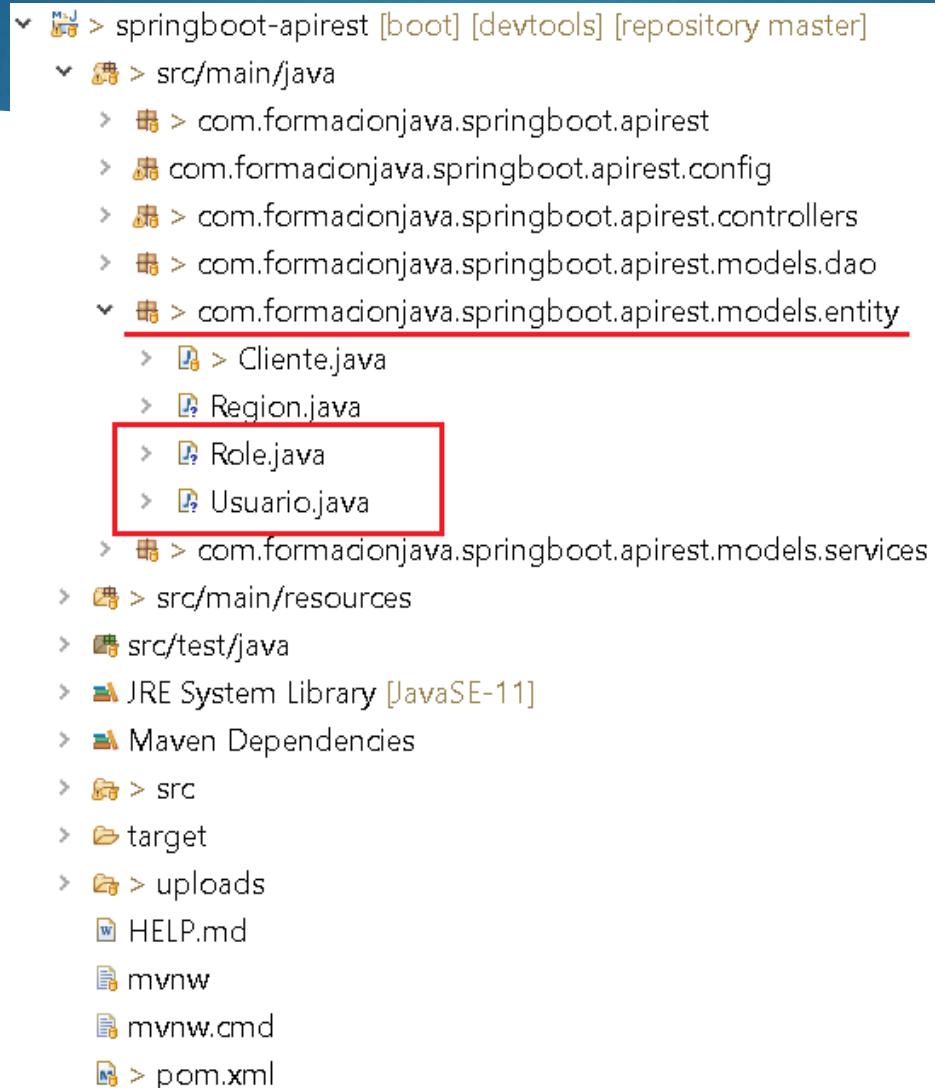
```
UsuarioService.java  springboot-apirest/pom.xml x
59      <!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-validation -->
60      <dependency>
61          <groupId>org.springframework.boot</groupId>
62          <artifactId>spring-boot-starter-validation</artifactId>
63      </dependency>
64
65      <!-- https://mvnrepository.com/artifact/org.springframework.security.oauth2/spring-security-oauth2 -->
66      <dependency>
67          <groupId>org.springframework.security.oauth2</groupId>
68          <artifactId>spring-security-oauth2</artifactId>
69          <version>2.5.1.RELEASE</version>
70      </dependency>
71      <!-- https://mvnrepository.com/artifact/org.springframework.security/spring-security-jwt -->
72      <dependency>
73          <groupId>org.springframework.security</groupId>
74          <artifactId>spring-security-jwt</artifactId>
75          <version>1.1.1.RELEASE</version>
76      </dependency>
77
78      <!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->
79      <dependency>
80          <groupId>javax.xml.bind</groupId>
81          <artifactId>jaxb-api</artifactId>
82      </dependency>
```

```
59      <!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-validation -->
60      <dependency>
61          <groupId>org.springframework.boot</groupId>
62          <artifactId>spring-boot-starter-validation</artifactId>
63      </dependency>
64
65      <!-- https://mvnrepository.com/artifact/org.springframework.security.oauth2/spring-security-oauth2 -->
66      <dependency>
67          <groupId>org.springframework.security.oauth2</groupId>
68          <artifactId>spring-security-oauth2</artifactId>
69          <version>2.3.4.RELEASE</version>
70      </dependency>
71      <dependency>
72          <groupId>org.springframework.security</groupId>
73          <artifactId>spring-security-jwt</artifactId>
74          <version>1.0.9.RELEASE</version>
75      </dependency>
76
77      <!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->
78      <dependency>
79          <groupId>javax.xml.bind</groupId>
80          <artifactId>jaxb-api</artifactId>
81      </dependency>
```

Debemos implementar nuevas entidades



Creamos la nuevas entidades



*Usuario.java

*Role.java

IUsuarioDao.java

```
3°import java.io.Serializable;
4
5 import javax.persistence.Entity;
6 import javax.persistence.GeneratedValue;
7 import javax.persistence.GenerationType;
8 import javax.persistence.Id;
9 import javax.persistence.Table;
10
11 @Entity
12 @Table(name = "roles")
13 public class Role implements Serializable {
14
15     @Id
16     @GeneratedValue(strategy = GenerationType.IDENTITY)
17     private Long id;
18
19     private String nombre;
20
21     public Long getId() {
22         return id;
23     }
24
25     public void setId(Long id) {
26         this.id = id;
27     }
28
29     public String getNombre() {
30         return nombre;
31     }
32
33     public void setNombre(String nombre) {
34         this.nombre = nombre;
35     }
36
37     private static final long serialVersionUID = 1L;
```

```
*Usuario.java x Role.java IUsuarioDao.java
1 package com.formacionjava.springboot.apirest.models.entity;
2
3 import java.io.Serializable;
18
19 @Entity
20 @Table(name = "usuarios")
21 public class Usuario implements Serializable{
22
23     @Id
24     @GeneratedValue(strategy = GenerationType.IDENTITY)
25     private Long id;
26
27     @Column(unique = true, length=20)
28     private String username;
29
30     @Column(length = 60)
31     private String password;
32     private Boolean enabled;
33
34     @ManyToMany(fetch = FetchType.LAZY, cascade = CascadeType.ALL)
35     @JoinTable(name="usuarios_roles",joinColumns= @JoinColumn(name="usuario_id"),
36     inverseJoinColumns =@JoinColumn(name="role_id"),
37     uniqueConstraints = {@UniqueConstraint(columnNames= {"usuario_id","role_id"})})
38     private List<Role> roles;
39
40
41
42     public Long getId() {
43         return id;
44     }
```

```
*Usuario.java x Role.java IUsuarioDao.java
 1 package com.formacionjava.springboot.apirest.models.entity;
 2
 3 import java.io.Serializable;
 4 import java.util.List;
 5
 6 import javax.persistence.CascadeType;
 7 import javax.persistence.Column;
 8 import javax.persistence.Entity;
 9 import javax.persistence.FetchType;
10 import javax.persistence.GeneratedValue;
11 import javax.persistence.GenerationType;
12 import javax.persistence.Id;
13 import javax.persistence.JoinColumn;
14 import javax.persistence.JoinTable;
15 import javax.persistence.ManyToMany;
16 import javax.persistence.Table;
17 import javax.persistence.UniqueConstraint;
18
19 @Entity
20 @Table(name = "usuarios")
21 public class Usuario implements Serializable{
22
23     @Id
24     @GeneratedValue(strategy = GenerationType.IDENTITY)
25     private Long id;
26
27     @Column(unique = true, length=20)
28     private String username;
29
30     @Column(length = 60)
```

```
41
42  public Long getId() {
43      return id;
44  }
45
46
47  public void setId(Long id) {
48      this.id = id;
49  }
50
51
52  public String getUsername() {
53      return username;
54  }
55
56
57  public void setUsername(String username) {
58      this.username = username;
59  }
60
61
62  public String getPassword() {
63      return password;
64  }
65
66
67  public void setPassword(String password) {
68      this.password = password;
69  }
70
71
72  public Boolean getEnabled() {
73      return enabled;
74  }
75
76
77  public void setEnabled(Boolean enabled) {
78      this.enabled = enabled;
79  }
80
81  private static final long serialVersionUID = 1L;
82
83
84 }
```

Compilamos y verificamos que nos genere las 3 tablas relacionadas

The screenshot shows the MySQL Workbench interface. The top bar displays the title "MySQL Workbench" and the connection information "Local instance MySQL80". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The left side features the "Navigator" pane, which is currently active and titled "SCHEMAS". It lists several databases: "bd_api_spring", "db_spring" (which is selected and highlighted with a blue border), "prueba_db", and "sys". Under the "db_spring" schema, there are categories for Tables, Views, Stored Procedures, and Functions. The "Tables" category is expanded, showing four tables: "clientes", "regiones", "roles", "usuarios", and "usuarios_roles". The "usuarios_roles" table is also highlighted with a red rectangular box. The right side of the interface contains the "Query 1" and "Administration - Startup / Shutdo..." panes. The "Administration" pane is titled "Local instance MySQL80 Startup / Shutdown MySQL Server". It displays a message stating that the database server is started and ready for client connections. It shows the server status as "running" and provides "Stop Server" and "Bring Offline" buttons. A note below states that stopping the server will prevent both the user and applications from using the database. At the bottom of the administration pane is a "Startup Message Log" section with two entries: "2021-11-29 03:23:37 - Workbench will use cmd shell commands to start/stop this instance" and "2021-11-29 03:23:37 - Server is running".

Creamos un nuevo DAO

edipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/models/dao/IUsuarioDao.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer X Usuario.java Role.java IUsuarioDao.java X application.properties

```
1 package com.formacionjava.springboot.apirest.models.dao;
2
3 import org.springframework.data.jpa.repository.Query;
4 import org.springframework.data.repository.CrudRepository;
5
6 import com.formacionjava.springboot.apirest.models.entity.Usuario;
7
8 public interface IUsuarioDao extends CrudRepository<Usuario, Long>{
9
10     public Usuario findByUsername(String username);
11
12     @Query("select u from Usuario u where u.username=?1")
13     public Usuario findByUsername2(String username);
14
15
16
17 }
18
```

Ahora creamos el servicio para Usuario

The screenshot shows the Eclipse IDE interface with a Java project named "springboot-apirest". The Project Explorer view on the left lists various packages and files under "src/main/java/com.formacionjava.springboot.apirest.models.services". A red underline highlights the package name "com.formacionjava.springboot.apirest.models.services". The code editor on the right displays the "UsuarioService.java" file, which contains the following code:

```
1 package com.formacionjava.springboot.apirest.models.services;
2
3 public class UsuarioService {
4
5 }
6
```

```
*UsuarioService.java x
1 import org.springframework.beans.factory.annotation.Autowired,
2 import org.springframework.security.core.userdetails.UserDetails;
3 import org.springframework.security.core.userdetails.UserDetailsService;
4 import org.springframework.security.core.userdetails.UsernameNotFoundException;
5 import org.springframework.stereotype.Service;
6 import org.springframework.transaction.annotation.Transactional;
7
8
9
10 import com.formacionjava.springboot.apirest.models.dao.IUsuarioDao;
11 import com.formacionjava.springboot.apirest.models.entity.Usuario;
12
13 @Service
14 public class UsuarioService implements UserDetailsService{
15
16     @Autowired
17     private IUsuarioDao usuarioDao;
18
19
20     @Override
21     @Transactional(readOnly=true)
22     public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
23         Usuario usuario = usuarioDao.findByUsername(username);
24         return null;
25     }
26
27 }
28
```

*UsuarioService.java x

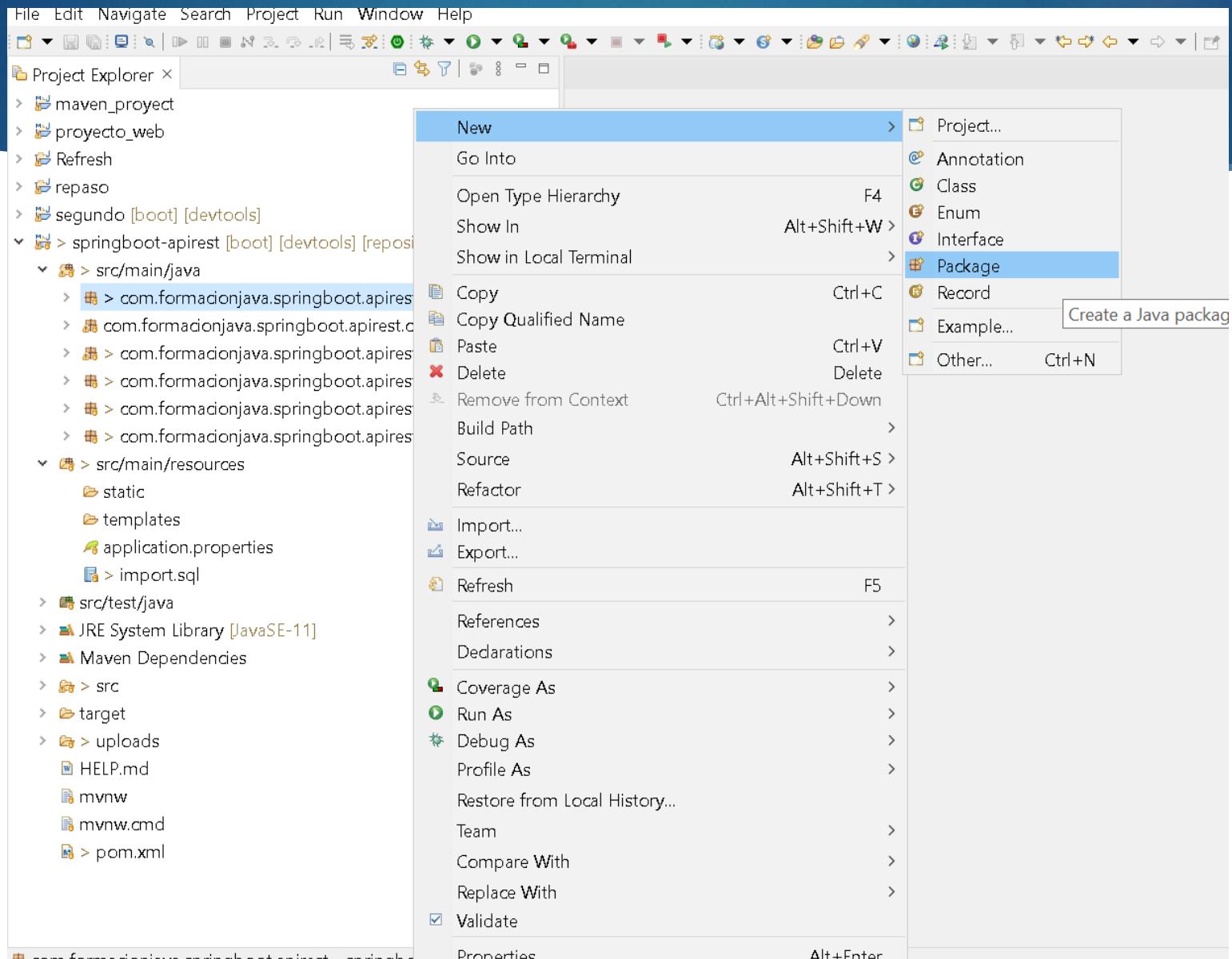
```
20
21 @Service
22 public class UsuarioService implements UserDetailsService{
23
24     private Logger logger = LoggerFactory.getLogger(UsuarioService.class);
25
26     @Autowired
27     private IUsuarioDao usuarioDao;
28
29     @Override
30     @Transactional(readOnly=true)
31     public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
32         Usuario usuario = usuarioDao.findByUsername(username);
33
34
35         if(usuario == null) {
36             logger.error("Error en el login: no existe el usuario '"+username+"' en el sistema");
37             throw new UsernameNotFoundException("Error en el login: no existe el usuario '"+username+"' en el sistema");
38         }
39         List<GrantedAuthority> authorities = usuario.getRoles()
40             .stream()
41             .map(role -> new SimpleGrantedAuthority(role.getNombre()))
42             .collect(Collectors.toList());
43
44
45         return new User(usuario.getUsername(), usuario.getPassword(), usuario.getEnabled(), true, true, true, authorities);
46     }
47
48 }
```

```
*UsuarioService.java x
1 package com.formacionjava.springboot.apirest.models.services;
2
3 import java.util.List;
4 import java.util.stream.Collectors;
5
6 import org.slf4j.Logger;
7 import org.slf4j.LoggerFactory;
8 import org.springframework.beans.factory.annotation.Autowired;
9 import org.springframework.security.core.GrantedAuthority;
10 import org.springframework.security.core.authority.SimpleGrantedAuthority;
11 import org.springframework.security.core.userdetails.User;
12 import org.springframework.security.core.userdetails.UserDetails;
13 import org.springframework.security.core.userdetails.UserDetailsService;
14 import org.springframework.security.core.userdetails.UsernameNotFoundException;
15 import org.springframework.stereotype.Service;
16 import org.springframework.transaction.annotation.Transactional;
17
18 import com.formacionjava.springboot.apirest.models.dao.IUsuarioDao;
19 import com.formacionjava.springboot.apirest.models.entity.Usuario;
20
21 @Service
22 public class UsuarioService implements UserDetailsService{
23
24     private Logger logger = LoggerFactory.getLogger(UsuarioService.class);
25
26     @Autowired
27     private IUsuarioDao usuarioDao;
28
29     @Override
30     @Transactional(readOnly=true)
```

UserService.java ×

```
21 @Service
22 public class UserService implements UserDetailsService{
23
24     private Logger logger = LoggerFactory.getLogger(UserService.class);
25
26     @Autowired
27     private IUsuarioDao usuarioDao;
28
29     @Override
30     @Transactional(readOnly=true)
31     public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
32         Usuario usuario = usuarioDao.findByUsername(username);
33
34
35         if(usuario == null) {
36             logger.error("Error en el login: no existe el usuario '" +username+ "' en el sistema");
37             throw new UsernameNotFoundException("Error en el login: no existe el usuario '" +username+ "' en el sistema");
38         }
39         List<GrantedAuthority> authorities = usuario.getRoles()
40             .stream()
41             .map(role -> new SimpleGrantedAuthority(role.getNombre()))
42             .peek(authority->logger.info("Role: " +authority.getAuthority()))
43             .collect(Collectors.toList());
44
45
46         return new User(usuario.getUsername(), usuario.getPassword(), usuario.getEnabled(), true, true, true, authorities);
47     }
48
49 }
```

Creamos un nuevo paquete



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - Eclipse IDE" and menu bar "File Edit Navigate Search Project Run Window Help". The toolbar has various icons for file operations like Open, Save, Find, and Run.

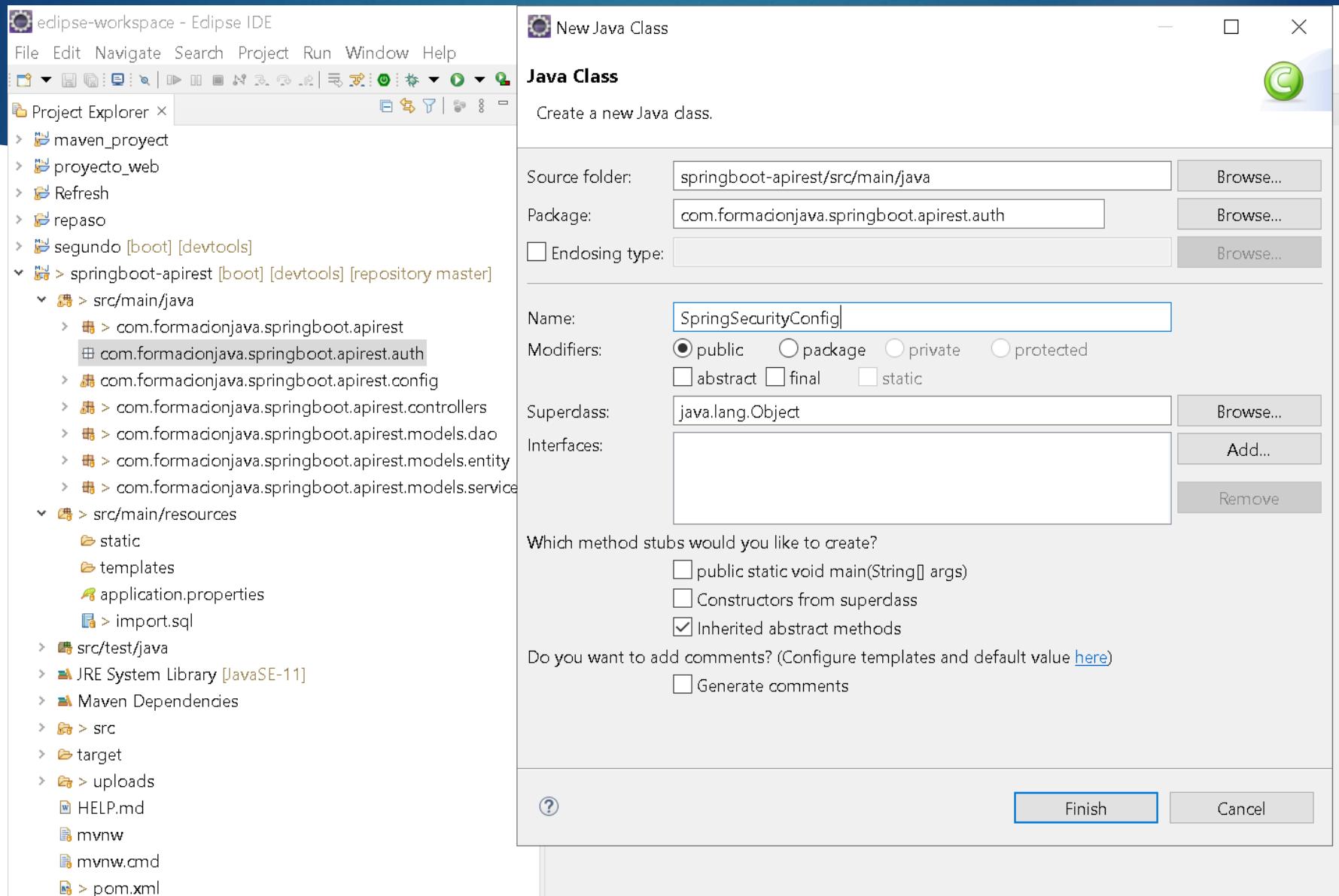
The left side features the "Project Explorer" view, which lists several projects and their contents:

- maven_proyect
- projeto_web
- Refresh
- repaso
- segundo [boot] [devtools]
- springboot-apirest [boot] [devtools] [repository master]
 - src/main/java
 - com.formacionjava.springboot.apirest
 - com.formacionjava.springboot.apirest.config
 - com.formacionjava.springboot.apirest.controllers
 - com.formacionjava.springboot.apirest.models.dao
 - com.formacionjava.springboot.apirest.models.entity
 - com.formacionjava.springboot.apirest.models.service
 - src/main/resources
 - static
 - templates
 - application.properties
 - import.sql
 - src/test/java
- JRE System Library [JavaSE-11]
- Maven Dependencies
- src
- target

A "New Java Package" dialog box is open in the center-right area. It contains the following fields and options:

- Java Package**: A title for the package.
- Create a new Java package.**: A descriptive text.
- Creates folders corresponding to packages.**: A note about folder creation.
- Source folder:** A dropdown set to "springboot-apirest/src/main/java" with a "Browse..." button.
- Name:** A text input field containing "com.formacionjava.springboot.apirest.auth".
- Create package-info.java**: A checkbox that is unchecked.
- Generate comments (configure templates and default value [here](#))**: A checkbox that is unchecked.
- Finish**: A blue-bordered button at the bottom right.
- Cancel**: A grey button at the bottom right.

Creamos una clase





Project Explorer ×

- > maven_proyect
- > proyecto_web
- > Refresh
- > repaso
- > segundo [boot] [devtools]
- > springboot-apirest [boot] [devtools] [repository master]
 - > src/main/java
 - > com.formacionjava.springboot.apirest
 - > com.formacionjava.springboot.apirest.auth
 - > SpringSecurityConfig.java
 - > com.formacionjava.springboot.apirest.config
 - > com.formacionjava.springboot.apirest.controllers
 - > com.formacionjava.springboot.apirest.models.dao
 - > com.formacionjava.springboot.apirest.models.entity
 - > com.formacionjava.springboot.apirest.models.services
 - > src/main/resources
 - > static
 - > templates
 - > application.properties
 - > import.sql
 - > src/test/java
 - > JRE System Library [JavaSE-11]
 - > Maven Dependencies
 - > src
 - > target
 - > uploads
 - > HELP.md
 - > mvnw
 - > mvnw.cmd
 - > pom.xml

SpringSecurityConfig.java ×

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 public class SpringSecurityConfig {
4
5 }
6
```

eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/auth/SpringSecurityConfig.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

*SpringSecurityConfig.java x

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.context.annotation.Configuration;
5 import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
6 import org.springframework.security.core.userdetails.UserDetailsService;
7
8 @Configuration
9 public class SpringSecurityConfig extends WebSecurityConfigurerAdapter{
10
11     @Autowired
12     private UserDetailsService usuarioService;
13
14 }
15
```

rest.auth;

.annotation.Autowired;

ation.Configuration;

ig.annotation.web.configuration.WebSecurityConfigurerAdapter;

.userdetails.UserDetailsService;

s WebSecurityConfigurerAdapter{

Toggle Comment Ctrl+7

Remove Block Comment Ctrl+Shift+\

Generate Element Comment Alt+Shift+J

Correct Indentation Ctrl+I

Format Ctrl+Shift+F

Format Element

Add Import Ctrl+Shift+M

Organize Imports Ctrl+Shift+O

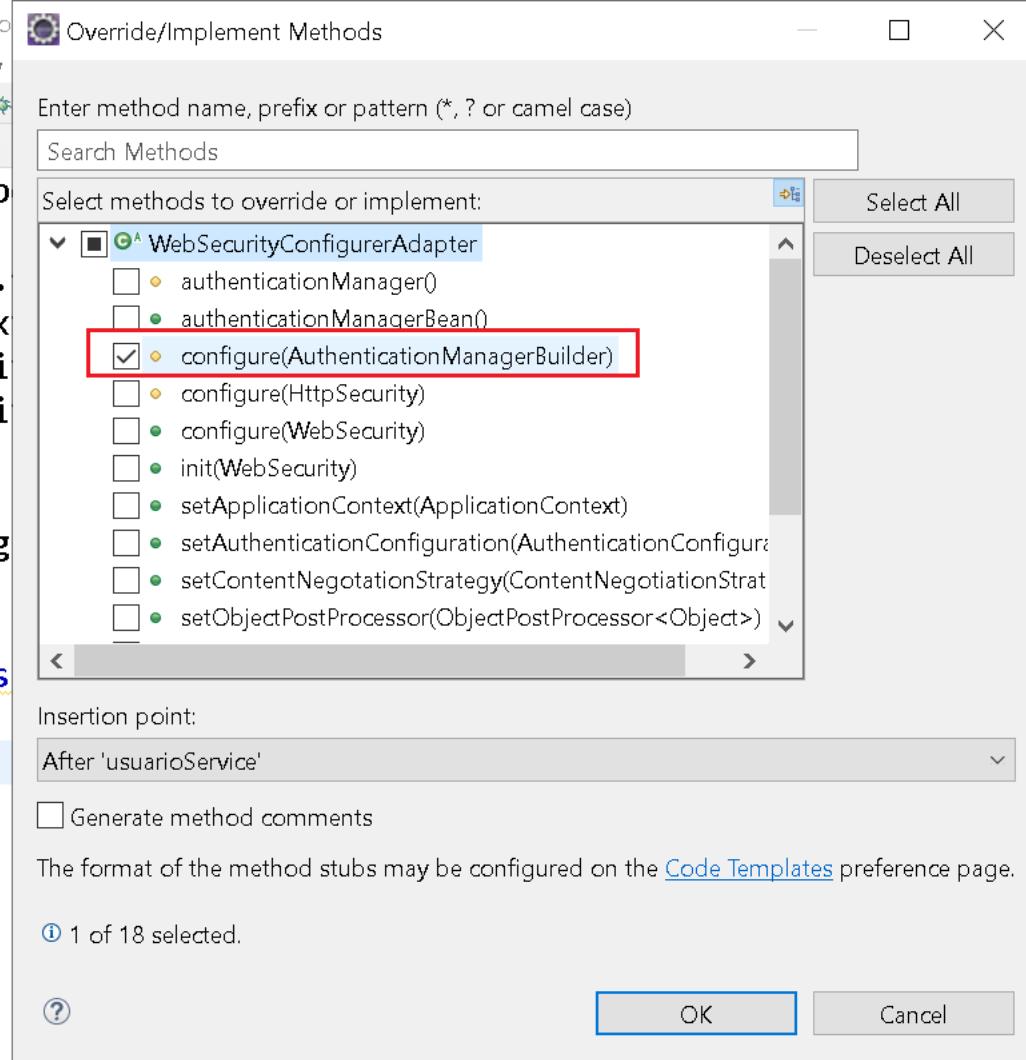
Sort Members...

Clean Up...

Override/Implement Methods...

Override/Implement Methods

```
eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/config  
File Edit Source Refactor Navigate Search Project Run Window  
*SpringSecurityConfig.java x  
1 package com.formacionjava.springboot  
2  
3 import org.springframework.beans.  
4 import org.springframework.context.  
5 import org.springframework.security.  
6 import org.springframework.security.  
7  
8 @Configuration  
9 public class SpringSecurityConfig  
10  
11     @Autowired  
12     private UserDetailsService us  
13  
14 }  
15 }  
16 }
```





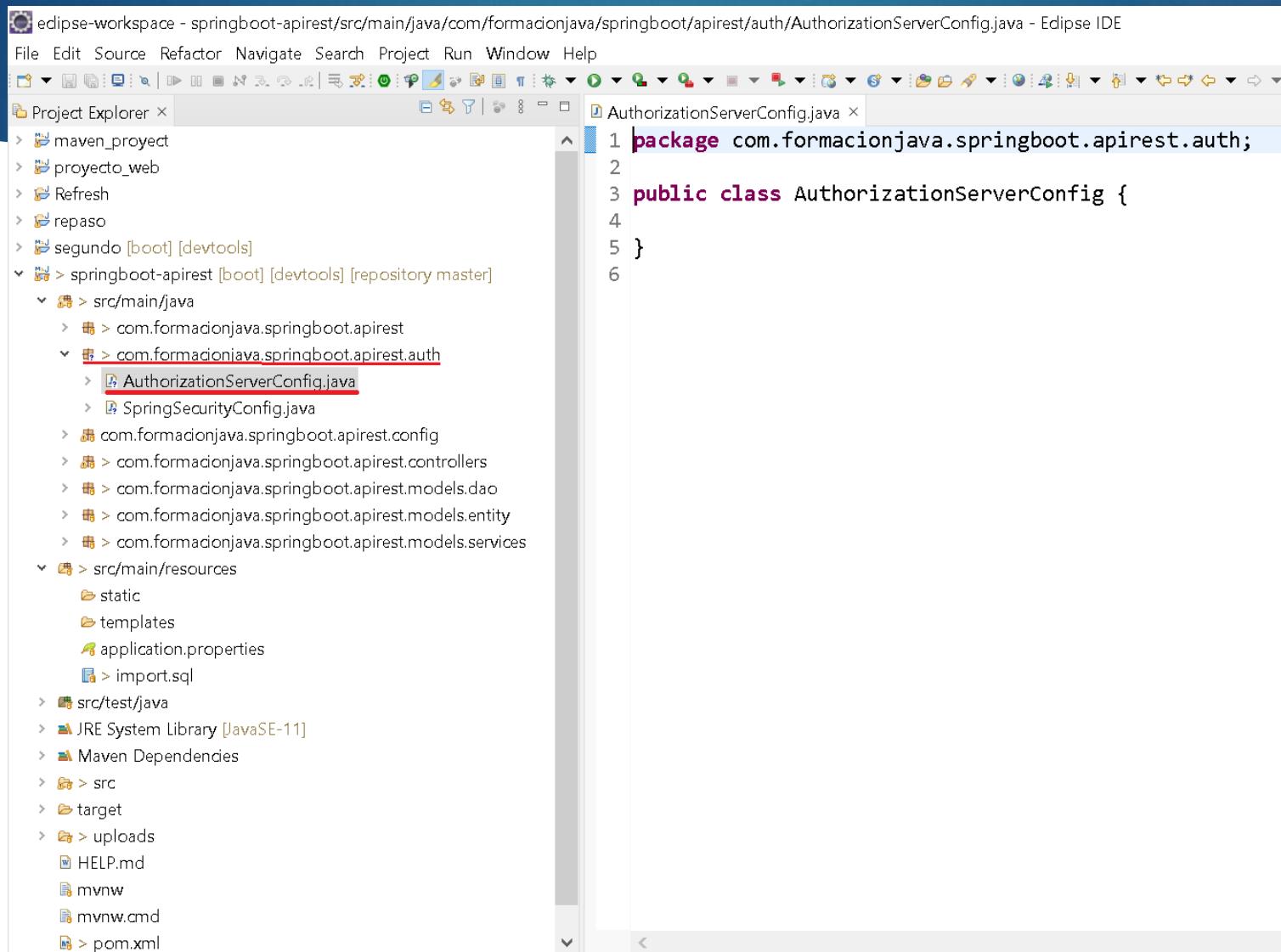
*SpringSecurityConfig.java x

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.context.annotation.Configuration;
5 import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
6 import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
7 import org.springframework.security.core.userdetails.UserDetailsService;
8
9 @Configuration
10 public class SpringSecurityConfig extends WebSecurityConfigurerAdapter{
11
12     @Autowired
13     private UserDetailsService usuarioService;
14
15     @Override
16     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
17         // TODO Auto-generated method stub
18         super.configure(auth);
19     }
20
21     |
22 }
23
```

*SpringSecurityConfig.java ×

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.context.annotation.Bean;
5 import org.springframework.context.annotation.Configuration;
6 import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
7 import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
8 import org.springframework.security.core.userdetails.UserDetailsService;
9 import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
10
11 @Configuration
12 public class SpringSecurityConfig extends WebSecurityConfigurerAdapter{
13
14     @Autowired
15     private UserDetailsService usuarioService;
16
17     @Bean
18     public BCryptPasswordEncoder passwordEncoder() {
19         return new BCryptPasswordEncoder();
20     }
21
22     @Override
23     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
24
25         auth.userDetailsService(this.usuarioService).passwordEncoder(passwordEncoder());
26     }
27
28 }
29 }
```

Configuración para el servidor de autorización



The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/auth/AuthorizationServerConfig.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbars:** Standard toolbar and various icons for file operations.
- Project Explorer:** Shows the project structure:
 - maven_proyect
 - projeto_web
 - Refresh
 - repaso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repository master]
 - src/main/java
 - com.formacionjava.springboot.apirest
 - com.formacionjava.springboot.apirest.auth
 - AuthorizationServerConfig.java
 - SpringSecurityConfig.java
 - com.formacionjava.springboot.apirest.config
 - com.formacionjava.springboot.apirest.controllers
 - com.formacionjava.springboot.apirest.models.dao
 - com.formacionjava.springboot.apirest.models.entity
 - com.formacionjava.springboot.apirest.models.services
 - src/main/resources
 - static
 - templates
 - application.properties
 - import.sql
 - src/test/java
 - JRE System Library [JavaSE-11]
 - Maven Dependencies
 - src
 - target
 - uploads
 - HELP.md
 - mvnw
 - mvnw.cmd
 - pom.xml
- Code Editor:** Shows the content of AuthorizationServerConfig.java:

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 public class AuthorizationServerConfig {
4
5 }
```



*AuthorizationServerConfig.java

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.context.annotation.Configuration;
5 import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
6 import org.springframework.security.oauth2.config.annotation.web.configuration.AuthorizationServerConfigurerAdapter;
7 import org.springframework.security.oauth2.config.annotation.web.configuration.EnableAuthorizationServer;
8
9 @Configuration
10 @EnableAuthorizationServer
11 public class AuthorizationServerConfig extends AuthorizationServerConfigurerAdapter {
12
13     @Autowired
14     private BCryptPasswordEncoder passwordEncoder;
15
16     @Autowired
17     private AuthenticationManager authenticationManager;
18
19
20
21 }
22
```

eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/auth/SpringSecurityConfig

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer X *AuthorizationServerConfig.java Spring

> maven_project
> proyecto_web
> Refresh
> repaso
> segundo [boot] [devtools]
> > springboot-apirest [boot] [devtools] [repository master]
> > src/main/java
> > com.formacionjava.springboot.apirest
> > com.formacionjava.springboot.apirest.auth
> > AuthorizationServerConfig.java
SpringSecurityConfig.java
> > com.formacionjava.springboot.apirest.config
> > com.formacionjava.springboot.apirest.controllers
> > com.formacionjava.springboot.apirest.models
> > com.formacionjava.springboot.apirest.models
> > com.formacionjava.springboot.apirest.models
> > src/main/resources
> static
> templates
> application.properties
> import.sql
> src/test/java
> JRE System Library [JavaSE-11]
> Maven Dependencies
> > src
> target
> uploads
> HELP.md
> mvnw
> mvnw.cmd
> pom.xml

1 package com.formacionjava
2
3+import org.springframework
10
11 @Configuration
12 public class SpringSecurityConfig {
13
14+ @Autowired
15 private UserDetailsService userDetailsService;
16
17+ @Bean
18 public BCryptPasswordEncoder passwordEncoder() {
19 return new BCryptPasswordEncoder();
 }
}

Undo Ctrl+Z
Revert File
Save Ctrl+S
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Show in Breadcrumb Alt+Shift+B
Quick Outline Ctrl+O
Quick Type Hierarchy Ctrl+T
Open With >
Show In Alt+Shift+W
Cut Ctrl+X
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Raw Paste
Quick Fix Ctrl+1
Source Alt+Shift+S
Refactor Alt+Shift+T
Local History >
References >
Declarations >
Add to Snippets...
Coverage As >
Run As >
Debug As >
Profile As >
Team >
Override/Implement Methods...
Generate Getters and Setters...
Generate Delegate Methods...
Generate hashCode() and equals()...
Generate toString()...
Replace With
 Validate
Preferences

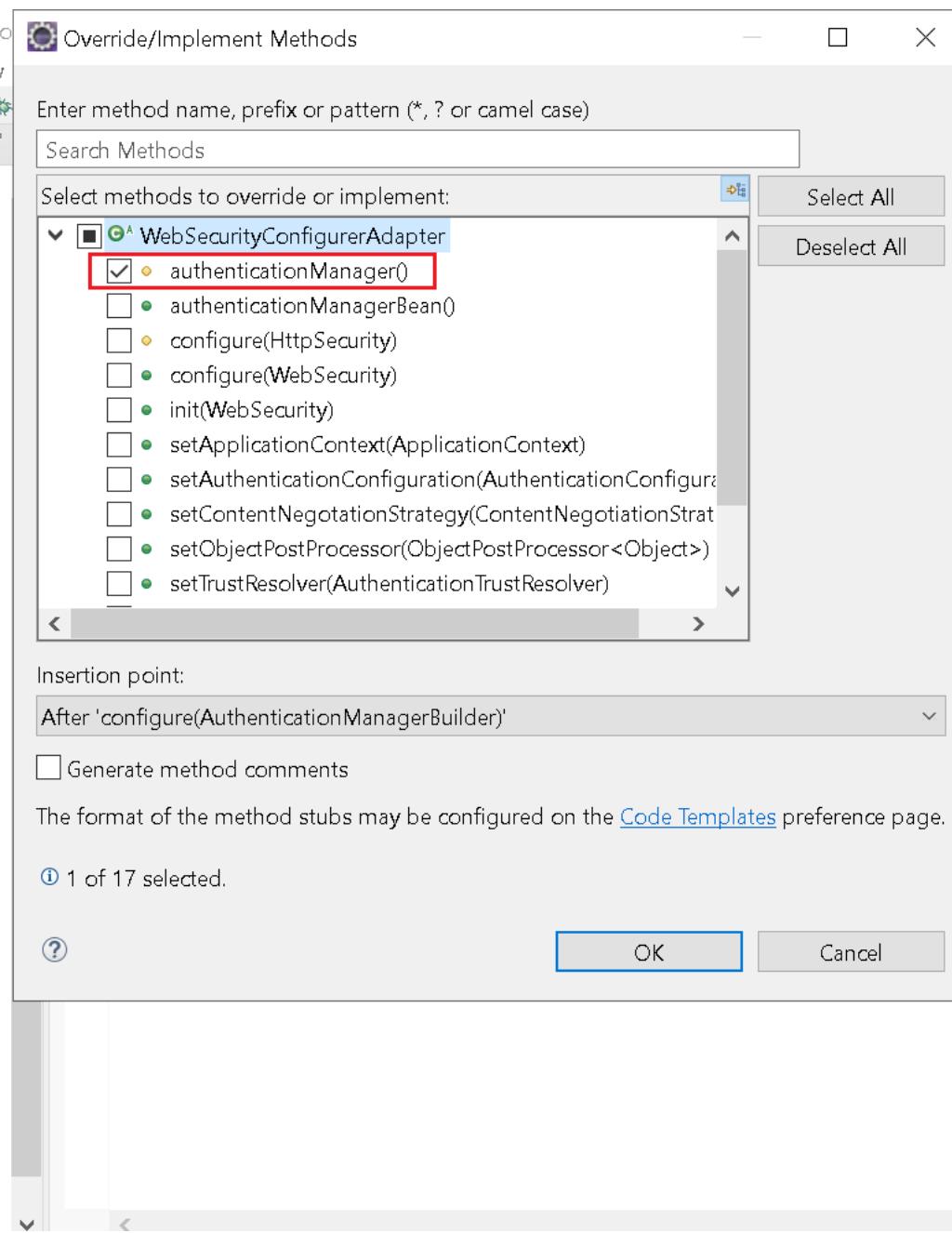
d; rerAdapter{
th) throws Exception {
dEncoder(passwordEncoder());

eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava

File Edit Source Refactor Navigate Search Project Run Window

Project Explorer x

- > maven_proyect
- > proyecto_web
- > Refresh
- > repaso
- > segundo [boot] [devtools]
- > > springboot-apirest [boot] [devtools] [repository master]
 - > > src/main/java
 - > > com.formacionjava.springboot.apirest
 - > > com.formacionjava.springboot.apirest.auth
 - > AuthorizationServerConfig.java
 - > SpringSecurityConfig.java
 - > com.formacionjava.springboot.apirest.config
 - > com.formacionjava.springboot.apirest.controllers
 - > com.formacionjava.springboot.apirest.models.dao
 - > com.formacionjava.springboot.apirest.models.entity
 - > com.formacionjava.springboot.apirest.models.services
 - > > src/main/resources
 - static
 - templates
 - application.properties
 - > import.sql
 - > src/test/java
 - > JRE System Library [JavaSE-11]
 - > Maven Dependencies
 - > > src
 - > > target
 - > > uploads
 - > HELP.md
 - > mvnw
 - > mvnw.cmd
 - > pom.xml



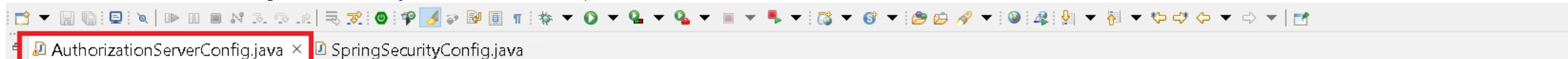
```
utowired;□  
  
ConfigurerAdapter{  
  
    @Override  
    public void configure(AuthenticationManagerBuilder auth) throws Exception {  
        auth.  
        passwordEncoder(passwordEncoder());  
    }  
}
```

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer View:** On the left, it lists several projects and their contents. The 'springboot-apirest' project is expanded, showing its structure under 'src/main/java' and 'src/main/resources'.
- Editor View:** The main window displays the code for `SpringSecurityConfig.java`. The file content is as follows:

```
2 import org.springframework.beans.factory.annotation.Autowired;
11
12 @Configuration
13 public class SpringSecurityConfig extends WebSecurityConfigurerAdapter{
14
15     @Autowired
16     private UserDetailsService usuarioService;
17
18     @Bean
19     public BCryptPasswordEncoder passwordEncoder() {
20         return new BCryptPasswordEncoder();
21     }
22
23     @Override
24     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
25
26         auth.userDetailsService(this.usuarioService).passwordEncoder(passwordEncoder());
27     }
28
29     @Bean
30     @Override
31     protected AuthenticationManager authenticationManager() throws Exception {
32         // TODO Auto-generated method stub
33         return super.authenticationManager();
34     }
35
36 }
```

The code implements a `SpringSecurityConfig` class that extends `WebSecurityConfigurerAdapter`. It injects a `UserDetailsService` and provides a `BCryptPasswordEncoder` for password hashing. It also overrides the `configure` method to set up the authentication manager.



```
1 package com.formacionjava.springboot.apirest.auth;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.beans.factory.annotation.Qualifier;
5 import org.springframework.context.annotation.Configuration;
6 import org.springframework.security.authentication.AuthenticationManager;
7 import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
8 import org.springframework.security.oauth2.config.annotation.web.configuration.AuthorizationServerConfigurerAdapter;
9 import org.springframework.security.oauth2.config.annotation.web.configuration.EnableAuthorizationServer;
10
11 @Configuration
12 @EnableAuthorizationServer
13 public class AuthorizationServerConfig extends AuthorizationServerConfigurerAdapter {
14
15     @Autowired
16     private BCryptPasswordEncoder passwordEncoder;
17
18     @Autowired
19     @Qualifier("authenticationManager")
20     private AuthenticationManager authenticationManager;
21
22
23
24 }
25
```

Implementamos métodos de configuración

The screenshot shows the Eclipse IDE interface with the following details:

- File Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window.
- Toolbar:** Standard Eclipse toolbar icons.
- Left Panel:** Package Explorer showing two files: AuthorizationServerConfig.java (selected) and SpringSecurityConfig.java.
- Right Panel:** Content Editor showing the code for AuthorizationServerConfig.java.
- Context Menu (Open with red box):** The menu is open over the first line of the code. It includes options like Undo Typing, Save, Open Declaration, Quick Fix, Source, and Override/Implement Methods... (highlighted with a blue box).
- Code Snippet:**

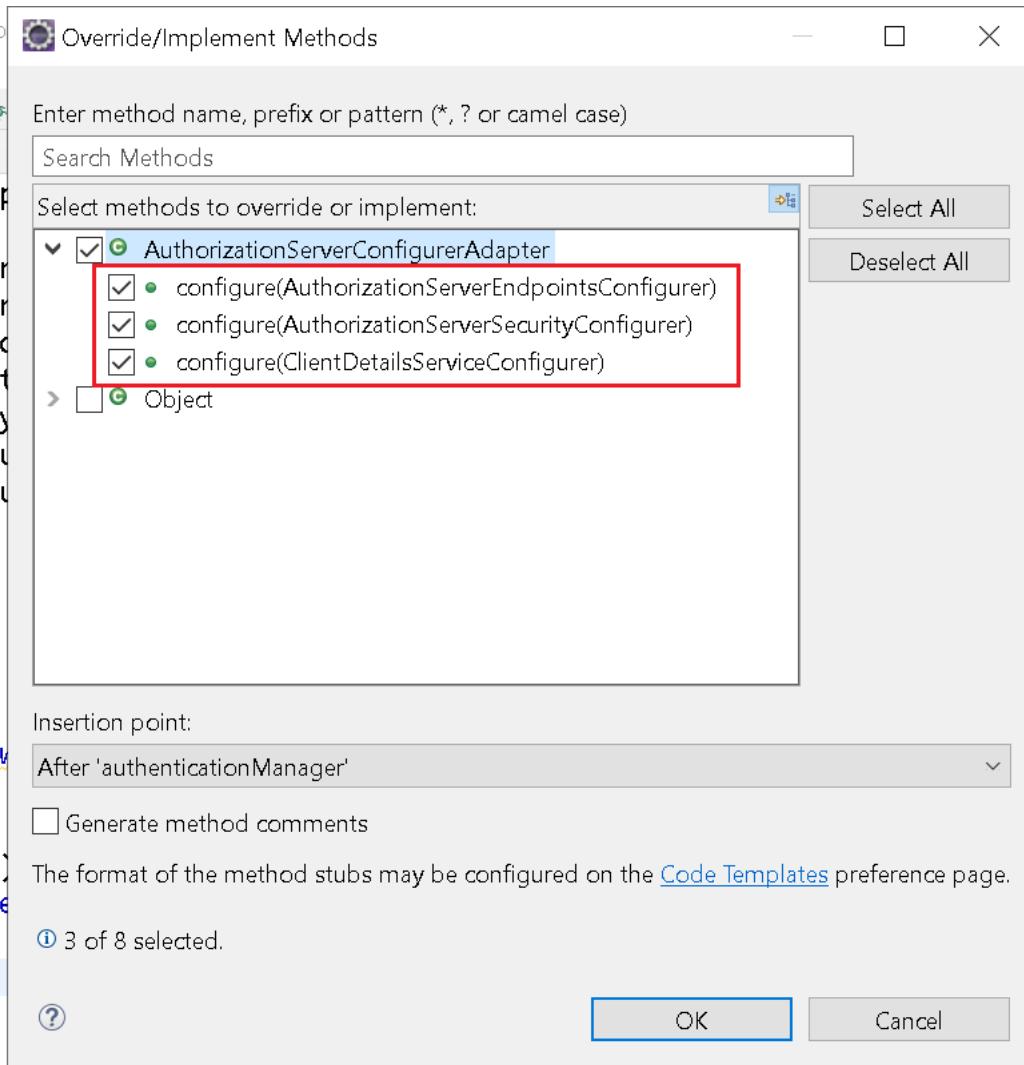
```
1 package com.formacionjava.springboot;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.beans.factory.annotation.Qualifier;
5 import org.springframework.context.annotation.Configuration;
6 import org.springframework.security.config.annotation.web.builders.HttpSecurity;
7 import org.springframework.security.config.annotation.web.configuration.EnableWebMvc;
8 import org.springframework.security.config.annotation.web.configuration.WebMvcConfigurerAdapter;
9 import org.springframework.security.core.userdetails.UserDetailsService;
10
11 @Configuration
12 @EnableWebMvc
13 public class AuthorizationServerConfig extends WebMvcConfigurerAdapter {
14
15     @Autowired
16     private BCryptPasswordEncoder passwordEncoder;
17
18     @Autowired
19     @Qualifier("authenticationManager")
20     private AuthenticationManager authenticationManager;
21
22 }
23
24 }
```

eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/ap

File Edit Source Refactor Navigate Search Project Run Window

AuthorizationServerConfig.java x SpringSecurityConfig.java

```
1 package com.formacionjava.springboot.ap
2
3 import org.springframework.beans.factory.
4 import org.springframework.beans.factory.
5 import org.springframework.context.anno
6 import org.springframework.security.aut
7 import org.springframework.security.cry
8 import org.springframework.security.oau
9 import org.springframework.security.oau
10
11 @Configuration
12 @EnableAuthorizationServer
13 public class AuthorizationServerConfig
14
15     @Autowired
16     private BCryptPasswordEncoder passw
17
18     @Autowired
19     @Qualifier("authenticationManager")
20     private AuthenticationManager auth
21
22
23 }
24 }
```



```
*AuthorizationServerConfig.java × SpringSecurityConfig.java
```

```
17  
18  @Autowired  
19  private BCryptPasswordEncoder passwordEncoder;  
20  
21  @Autowired  
22  @Qualifier("authenticationManager")  
23  private AuthenticationManager authenticationManager;  
24  
25  @Override  
26  public void configure(AuthorizationServerSecurityConfigurer security) throws Exception {  
27      // TODO Auto-generated method stub  
28      super.configure(security);  
29  }  
30  
31  @Override  
32  public void configure(ClientDetailsServiceConfigurer clients) throws Exception {  
33      // TODO Auto-generated method stub  
34      super.configure(clients);  
35  }  
36  
37  @Override  
38  public void configure(AuthorizationServerEndpointsConfigurer endpoints) throws Exception {  
39      // TODO Auto-generated method stub  
40      super.configure(endpoints);  
41  }  
42  
43  |  
44  
45 }  
46 }
```

```
17  
18  @Autowired  
19  private BCryptPasswordEncoder passwordEncoder;  
20  
21  @Autowired  
22  @Qualifier("authenticationManager")  
23  private AuthenticationManager authenticationManager;  
24  
25  @Override  
26  public void configure(AuthorizationServerSecurityConfigurer security) throws Exception {  
27      // TODO Auto-generated method stub  
28      super.configure(security);  
29  }  
30  
31  @Override  
32  public void configure(ClientDetailsServiceConfigurer clients) throws Exception {  
33      // TODO Auto-generated method stub  
34      super.configure(clients);  
35  }  
36  
37  @Override  
38  public void configure(AuthorizationServerEndpointsConfigurer endpoints) throws Exception {  
39  
40      endpoints.authenticationManager(authenticationManager)  
41      .accessTokenConverter(accessTokenConverter());  
42  }  
43  
44  
45  
46 }
```

The method accessTokenConverter() is undefined for the type AuthorizationServerConfig

1 quick fix available:

Create method 'accessTokenConverter()'

Press 'F2' for focus



*AuthorizationServerConfig.java x SpringSecurityConfig.java

```
24     private AuthenticationManager authenticationManager;
25
26     @Override
27     public void configure(AuthorizationServerSecurityConfigurer security) throws Exception {
28         // TODO Auto-generated method stub
29         super.configure(security);
30     }
31
32     @Override
33     public void configure(ClientDetailsServiceConfigurer clients) throws Exception {
34         // TODO Auto-generated method stub
35         super.configure(clients);
36     }
37
38     @Override
39     public void configure(AuthorizationServerEndpointsConfigurer endpoints) throws Exception {
40
41         endpoints.authenticationManager(authenticationManager)
42             .accessTokenConverter(accessTokenConverter());
43     }
44
45     public AccessTokenConverter accessTokenConverter() {
46         // TODO Auto-generated method stub
47         return null;
48     }
49
50
51
```

eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/auth/AuthorizationServerConfig.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

AuthorizationServerConfig.java × SpringSecurityConfig.java

```
26     private AuthenticationManager authenticationManager;
27
28     @Override
29     public void configure(AuthorizationServerSecurityConfigurer security) throws Exception {
30         // TODO Auto-generated method stub
31         super.configure(security);
32     }
33
34     @Override
35     public void configure(ClientDetailsServiceConfigurer clients) throws Exception {
36         // TODO Auto-generated method stub
37         super.configure(clients);
38     }
39
40     @Override
41     public void configure(AuthorizationServerEndpointsConfigurer endpoints) throws Exception {
42
43         endpoints.authenticationManager(authenticationManager)
44             .accessTokenConverter(accessTokenConverter());
45     }
46
47     @Bean
48     public JwtAccessTokenConverter accessTokenConverter() {
49
50         JwtAccessTokenConverter jwtAccessTokenConverter = new JwtAccessTokenConverter();
51
52         return jwtAccessTokenConverter;
53     }
54 }
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/auth/AuthorizationServerConfig.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Find, and Build. The left sidebar shows the package structure with "AuthorizationServerConfig.java" and "SpringSecurityConfig.java" selected. The main editor area displays the Java code for "AuthorizationServerConfig.java".

```
32     super.configure(security);
33 }
34
35* @Override
36 public void configure(ClientDetailsServiceConfigurer clients) throws Exception {
37     // TODO Auto-generated method stub
38     super.configure(clients);
39 }
40
41* @Override
42 public void configure(AuthorizationServerEndpointsConfigurer endpoints) throws Exception {
43
44     endpoints.authenticationManager(authenticationManager)
45         .tokenStore(tokenStore())
46         .accessTokenConverter(accessTokenConverter());
47 }
48
49* @Bean
50 public JwtTokenStore tokenStore() {
51     return new JwtTokenStore(accessTokenConverter());
52 }
53
54* @Bean
55 public JwtAccessTokenConverter accessTokenConverter() {
56
57     JwtAccessTokenConverter jwtAccessTokenConverter = new JwtAccessTokenConverter();
58
59     return jwtAccessTokenConverter;
60 }
61
```

The code implements the `AuthorizationServerConfig` interface. It overrides the `configure` methods for `ClientDetailsServiceConfigurer` and `AuthorizationServerEndpointsConfigurer`. The `clientDetailsConfigurer` method calls `super.configure`. The `endpointsConfigurer` method sets the authentication manager, token store, and access token converter. The `tokenStore` and `accessTokenConverter` methods are annotated with `@Bean` and implemented as `JwtTokenStore` and `JwtAccessTokenConverter` respectively.

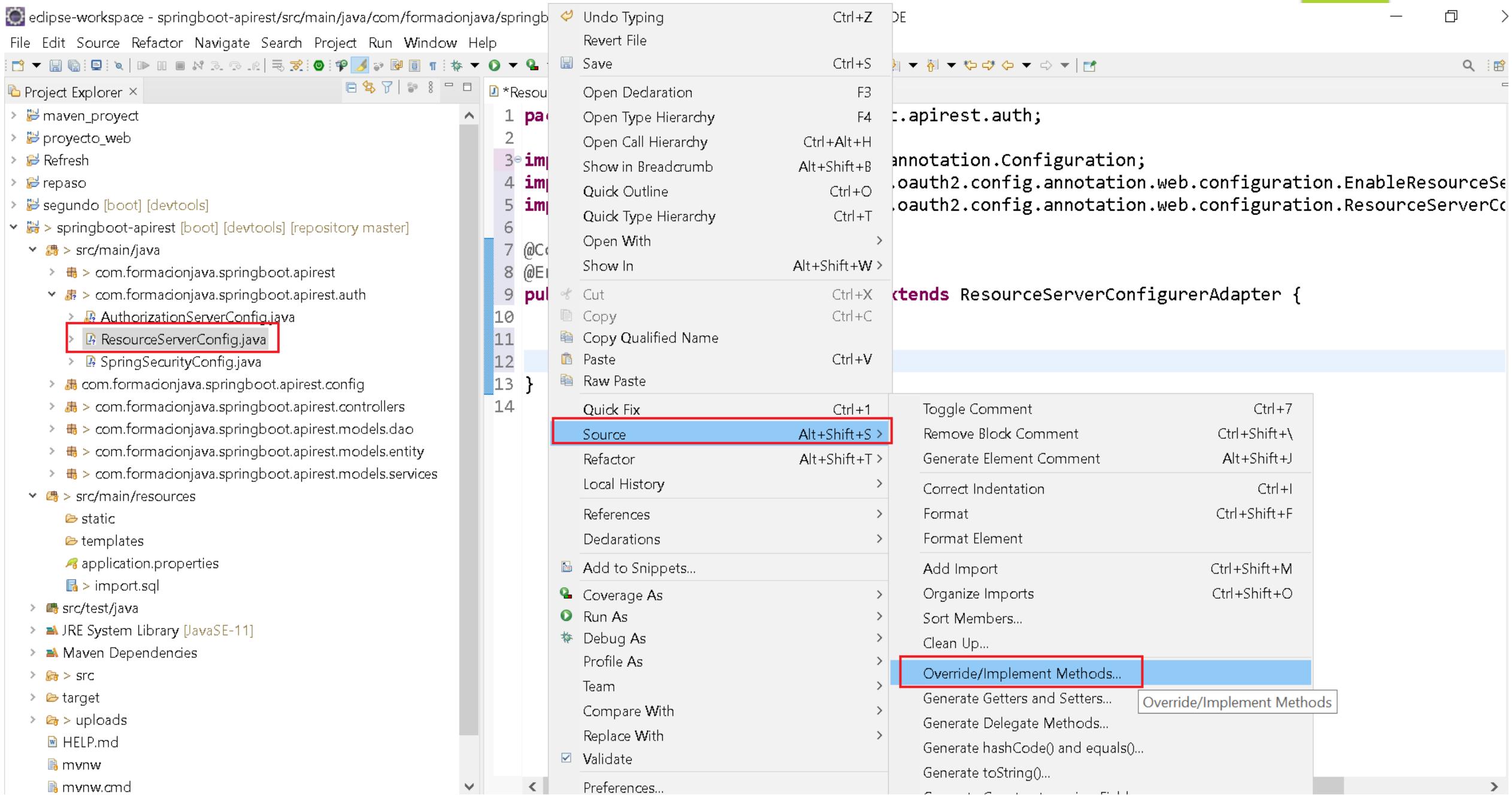
```
20 public class AuthorizationServerConfig extends AuthorizationServerConfigurerAdapter {
21
22     @Autowired
23     private BCryptPasswordEncoder passwordEncoder;
24
25     @Autowired
26     @Qualifier("authenticationManager")
27     private AuthenticationManager authenticationManager;
28
29     @Override
30     public void configure(AuthorizationServerSecurityConfigurer security) throws Exception {
31         security.tokenKeyAccess("permitAll()")
32             .checkTokenAccess("isAuthenticated()");
33     }
34
35     @Override
36     public void configure(ClientDetailsServiceConfigurer clients) throws Exception {
37
38         clients.inMemory().withClient("angularapp")
39             .secret(passwordEncoder.encode("12345"))
40             .scopes("read", "write")
41             .authorizedGrantTypes("password", "refresh_token")
42             .accessTokenValiditySeconds(3600)
43             .refreshTokenValiditySeconds(3600);
44     }
45
46     @Override
47     public void configure(AuthorizationServerEndpointsConfigurer endpoints) throws Exception {
48
49         endpoints.authenticationManager(authenticationManager)
```

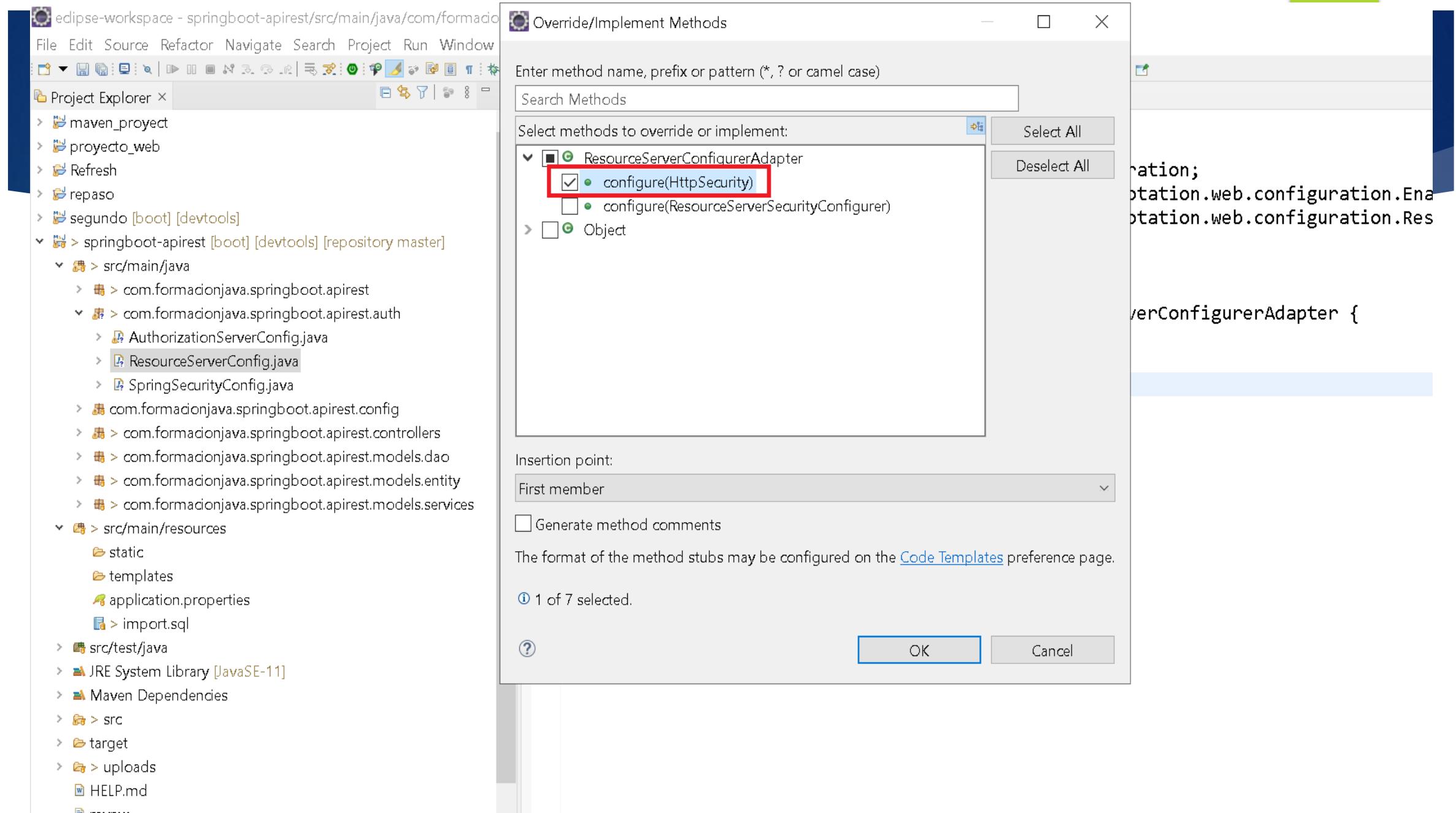
edipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/auth/ResourceServerConfig.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer × *ResourceServerConfig.java ×

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 import org.springframework.context.annotation.Configuration;
4 import org.springframework.security.oauth2.config.annotation.web.configuration.EnableResourceServer;
5 import org.springframework.security.oauth2.config.annotation.web.configuration.ResourceServerConfigurerAdapter;
6
7 @Configuration
8 @EnableResourceServer
9 public class ResourceServerConfig extends ResourceServerConfigurerAdapter {
10
11 }
12
```





The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** On the left, it lists the project structure. The `src/main/java` folder contains:
 - `com.formacionjava.springboot.apirest`
 - `com.formacionjava.springboot.apirest.auth` (selected)
 - `AuthorizationServerConfig.java`
 - `ResourceServerConfig.java` (highlighted in grey)
 - `SpringSecurityConfig.java`
 - `com.formacionjava.springboot.apirest.config`
 - `com.formacionjava.springboot.apirest.controllers`
 - `com.formacionjava.springboot.apirest.models.dao`
 - `com.formacionjava.springboot.apirest.models.entity`
 - `com.formacionjava.springboot.apirest.models.services`
- ResourceServerConfig.java Editor:** The main editor window displays the Java code for `ResourceServerConfig`. The code configures Spring Security to allow all `GET` requests to the `/api/clients` endpoint without authentication.

```
1 package com.formacionjava.springboot.apirest.auth;
2
3 import org.springframework.context.annotation.Configuration;
4 import org.springframework.http.HttpMethod;
5 import org.springframework.security.config.annotation.web.builders.HttpSecurity;
6 import org.springframework.security.oauth2.config.annotation.web.configuration.EnableResourceServer;
7 import org.springframework.security.oauth2.config.annotation.web.configuration.ResourceServerConfigurerAdapter;
8
9 @Configuration
10 @EnableResourceServer
11 public class ResourceServerConfig extends ResourceServerConfigurerAdapter {
12
13     @Override
14     public void configure(HttpSecurity http) throws Exception {
15         http.authorizeRequests().antMatchers(HttpMethod.GET, "/api/clients").permitAll()
16             .anyRequest().authenticated();
17     }
18
19
20
21 }
22
```

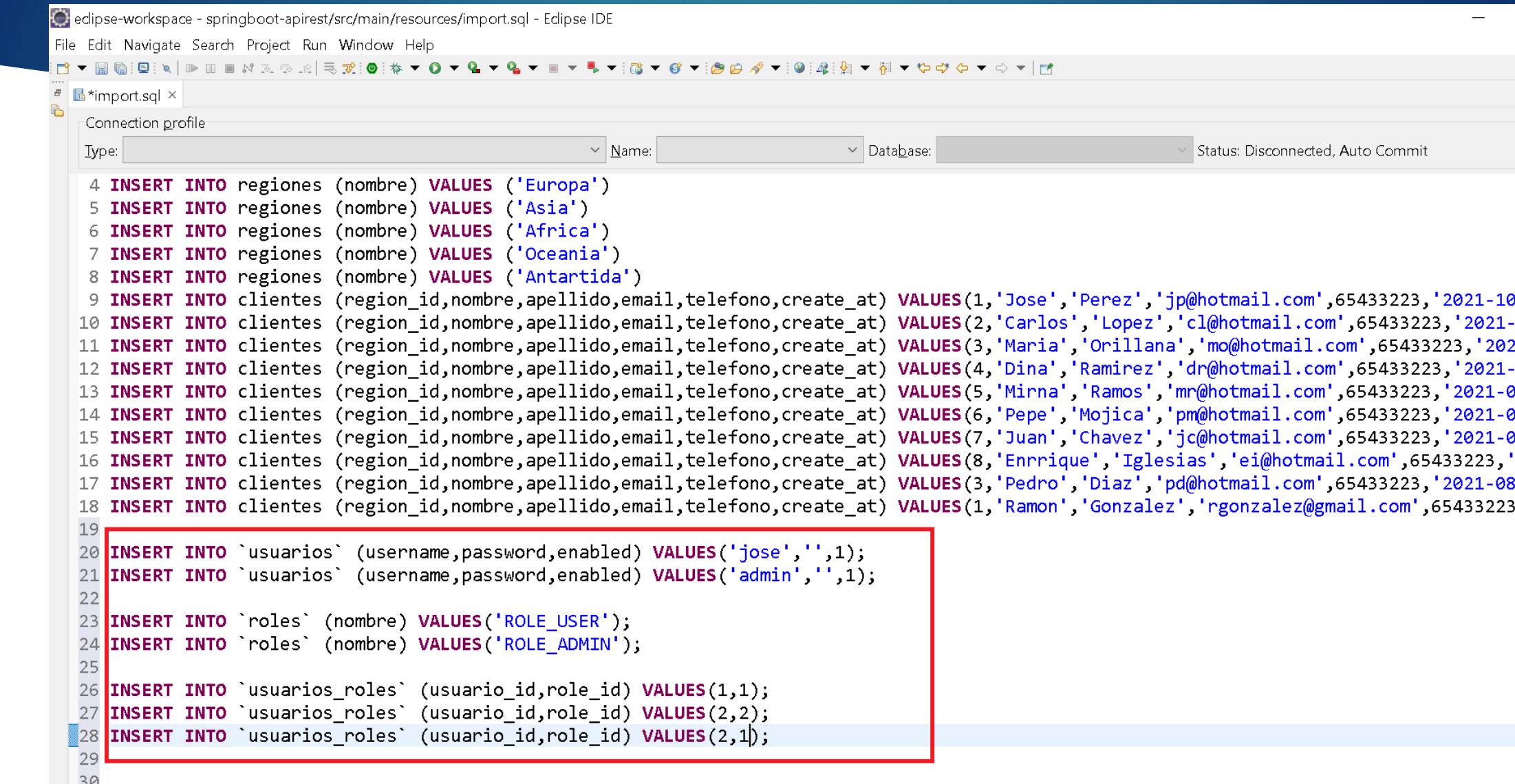
```
16  
17  @Autowired  
18  private UserDetailsService usuarioService;  
19  
20  @Bean  
21  public BCryptPasswordEncoder passwordEncoder() {  
22      return new BCryptPasswordEncoder();  
23  }  
24  
25  @Override  
26  protected void configure(AuthenticationManagerBuilder auth) throws Exception {  
27      auth.userDetailsService(this.usuarioService).passwordEncoder(passwordEncoder());  
28  }  
29  
30  @Bean  
31  @Override  
32  protected AuthenticationManager authenticationManager() throws Exception {  
33      // TODO Auto-generated method stub  
34      return super.authenticationManager();  
35  }  
36  
37  
38  @Override  
39  public void configure(HttpSecurity http) throws Exception {  
40      http.authorizeRequests().antMatchers(HttpMethod.GET, "/api/clientes").permitAll()  
41          .anyRequest().authenticated();  
42  }  
43  
44  
45 }
```

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** On the left, it lists several projects and their components. The "springboot-apirest" project is expanded, showing its structure under "src/main/java".
- Code Editor:** The main window displays the content of the file `SpringSecurityConfig.java`. The code implements the `HttpSecurity` interface, defining security configurations.
- Annotations:** Several annotations are used throughout the code:
 - `@Override`: Used to indicate that methods are overriding inherited ones.
 - `@Bean`: Used to mark methods that return beans for dependency injection.
 - `throws Exception`: Used as the exception type for method signatures.
- Red Box Selection:** A red rectangular box highlights the following block of code, which configures the `HttpSecurity` object:

```
    @Override
    public void configure(HttpSecurity http) throws Exception {
        http.authorizeRequests()
            .anyRequest().authenticated()
            .and()
            .csrf()
            .disable()
            .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);
    }
```

Creamos datos de pruebas



The screenshot shows the Eclipse IDE interface with a SQL script named "import.sql" open. The script contains several INSERT statements used to seed a database with test data for regions, clients, users, and user roles.

```
4 INSERT INTO regiones (nombre) VALUES ('Europa')
5 INSERT INTO regiones (nombre) VALUES ('Asia')
6 INSERT INTO regiones (nombre) VALUES ('Africa')
7 INSERT INTO regiones (nombre) VALUES ('Oceania')
8 INSERT INTO regiones (nombre) VALUES ('Antartida')
9 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(1,'Jose','Perez','jp@hotmail.com',65433223,'2021-10-01')
10 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(2,'Carlos','Lopez','cl@hotmail.com',65433223,'2021-09-20')
11 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(3,'Maria','Orillana','mo@hotmail.com',65433223,'2021-08-15')
12 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(4,'Dina','Ramirez','dr@hotmail.com',65433223,'2021-07-05')
13 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(5,'Mirna','Ramos','mr@hotmail.com',65433223,'2021-04-20')
14 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(6,'Pepe','Mojica','pm@hotmail.com',65433223,'2021-05-10')
15 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(7,'Juan','Chavez','jc@hotmail.com',65433223,'2021-06-01')
16 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(8,'Enrique','Iglesias','ei@hotmail.com',65433223,'2021-07-15')
17 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(3,'Pedro','Diaz','pd@hotmail.com',65433223,'2021-08-05')
18 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(1,'Ramon','Gonzalez','rgonzalez@gmail.com',65433223,
19
20 INSERT INTO `usuarios` (username,password(enabled)) VALUES('jose','','1');
21 INSERT INTO `usuarios` (username,password(enabled)) VALUES('admin','','1');
22
23 INSERT INTO `roles` (nombre) VALUES('ROLE_USER');
24 INSERT INTO `roles` (nombre) VALUES('ROLE_ADMIN');
25
26 INSERT INTO `usuarios_roles` (usuario_id,role_id) VALUES(1,1);
27 INSERT INTO `usuarios_roles` (usuario_id,role_id) VALUES(2,2);
28 INSERT INTO `usuarios_roles` (usuario_id,role_id) VALUES(2,1);
29
30
```

Generamos los password encriptados

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/SpringbootApirestApplication.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Project Explorer:** Shows the project structure:
 - maven_proyect
 - proyecto_web
 - Refresh
 - repaso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repository master]
 - src/main/java
 - com.formacionjava.springboot.apirest
 - SpringbootApirestApplication.java
 - com.formacionjava.springboot.apirest.auth
 - com.formacionjava.springboot.apirest.config
 - com.formacionjava.springboot.apirest.controllers
 - com.formacionjava.springboot.apirest.models.dao
 - com.formacionjava.springboot.apirest.models.entity
 - com.formacionjava.springboot.apirest.models.services
 - src/main/resources
 - static
 - templates
 - application.properties
 - import.sql
 - src/test/java
 - JRE System Library [JavaSE-11]
 - Maven Dependencies
 - src
 - target
 - uploads
 - HELP.md
 - mvnw
- Code Editor:** The file *SpringbootApirestApplication.java* is open. The code implements the `CommandLineRunner` interface and overrides the `run` method to demonstrate password hashing using BCrypt. The `passwordEncoder` field and the `run` method body are highlighted with red boxes.

```
1 package com.formacionjava.springboot.apirest;
2
3+import org.springframework.beans.factory.annotation.Autowired;[]
4
5 @SpringBootApplication
6 public class SpringbootApirestApplication implements CommandLineRunner {
7
8     @Autowired
9     private BCryptPasswordEncoder passwordEncoder;
10
11
12     public static void main(String[] args) {
13         SpringApplication.run(SpringbootApirestApplication.class, args);
14     }
15
16
17     @Override
18     public void run(String... args) throws Exception {
19
20         String password = "12345";
21
22         for(int i = 0;i<4;i++) {
23             String passwordBcrypt = passwordEncoder.encode(password);
24             System.out.println(passwordBcrypt);
25         }
26
27     }
28
29 }
30
31 }
32 }
```

Project Explorer × SpringbootAp... × Markers × Properties × Servers × Data Source Explorer × Snippets × Console × Git Staging

```
1 package com.formacionjava.sprin
2
3 import org.springframework.boot.
4
5 @SpringBootApplication
6 public class SpringbootApirestApplication {
7
8     @Autowired
9     private BCryptPasswordEncoder bCryptPasswordEncoder;
10
11    public static void main(String[] args) {
12        SpringApplication.run(SpringbootApirestApplication.class, args);
13    }
14
15    @Override
16    public void run(String[] args) {
17        String password = "123456";
18
19        for(int i = 0; i < 10; i++) {
20            String hashedPassword = bCryptPasswordEncoder.encode(password);
21
22            System.out.println(hashedPassword);
23        }
24    }
25}
26
27}
28
29}
30
31}
32}
```

springboot-apirest - SpringbootApirestApplication [Spring Boot App] C:\Users\dell\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jdt.compiler_4.12.0.v20210915-1200\jre\bin\java

Date	Level	Thread ID	Message
2021-11-29 06:30:51.024	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.034	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.044	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.054	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.062	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.072	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.082	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.091	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.099	DEBUG	1768	[restartedMain] org.hibernate.SQL
2021-11-29 06:30:51.110	INFO	1768	[restartedMain] o.h.e.t.j.p.i.JtaPlat
2021-11-29 06:30:51.123	INFO	1768	[restartedMain] j.LocalContainerEntity
2021-11-29 06:30:51.182	WARN	1768	[restartedMain] JpaBaseConfiguration\$
2021-11-29 06:30:51.991	WARN	1768	[restartedMain] o.s.s.o.p.t.s.JwtAcces
2021-11-29 06:30:52.327	INFO	1768	[restartedMain] pertySourcedRequestMap
2021-11-29 06:30:52.493	INFO	1768	[restartedMain] o.s.b.d.a.OptionalLive
2021-11-29 06:30:53.001	INFO	1768	[restartedMain] o.s.s.web.DefaultSecur
2021-11-29 06:30:53.015	INFO	1768	[restartedMain] o.s.s.web.DefaultSecur
2021-11-29 06:30:53.020	INFO	1768	[restartedMain] o.s.s.web.DefaultSecur
2021-11-29 06:30:53.132	INFO	1768	[restartedMain] o.s.b.w.embedded.tomca
2021-11-29 06:30:53.133	INFO	1768	[restartedMain] d.s.w.p.Documentation
2021-11-29 06:30:53.157	INFO	1768	[restartedMain] d.s.w.p.Documentation
2021-11-29 06:30:53.207	INFO	1768	[restartedMain] s.d.s.w.s.ApiListingRe
2021-11-29 06:30:53.442	INFO	1768	[restartedMain] c.f.s.a.SpringbootAp
\$2a\$10\$Jf1B1DvYy3spSruEe8kf4OXx1jeyPaOgTHPgXiUaUQQ/s/O.PWhbu			
\$2a\$10\$8t2e9DE1.ZSajFHzwu/JKexkpmgoIpH6JQsK.rWlseVjAxCCzuf/K			
\$2a\$10\$4juaknnD9oRuRrRVJVHSme7d4yVwsHMWLp4fVoTqRkTfuNCgA3w.2			
\$2a\$10\$GyLqNMlTj64457F98..CbutV4NAFrOOXNIIsCfDu8RTZTtx9nVs8Ki			

```
18 INSERT INTO `clientes` (region_id,nombre,apellido,email,telefono,create_at) VALUES(1,'Ramon','Gonzalez','rgonzalez@gmail.com',65433223,'2022-01-01');
19
20 INSERT INTO `usuarios` (username,password(enabled)) VALUES('jose','$2a$10$Jf1B1DvYy3spSruEe8kf4OXX1jeyPaOgTHPgXiUaUQQ/s/O.PWhbu',1);
21 INSERT INTO `usuarios` (username,password(enabled)) VALUES('admin','$2a$10$8t2e9DEl.ZSajFHzwu/JKexkpmgoIpH6JQsK.rWlseVjAxCCzuf/K',1);
22
23 INSERT INTO `roles` (nombre) VALUES('ROLE_USER');
24 INSERT INTO `roles` (nombre) VALUES('ROLE_ADMIN');
25
26 INSERT INTO `usuarios_roles` (usuario_id,role_id) VALUES(1,1);
27 INSERT INTO `usuarios_roles` (usuario_id,role_id) VALUES(2,2);
28 INSERT INTO `usuarios_roles` (usuario_id,role_id) VALUES(2,1);
29
30
```

Compilamos y probarlo, deberíamos verlo en nuestra base de datos

The image displays three separate instances of MySQL Workbench, each showing a different table from a database schema.

- Left Window:** Shows the `usuarios` table from the `db_spring` schema. The table has columns `id`, `enabled`, `password`, and `username`. Two rows are present: one for user `jose` and one for user `admin`.
- Middle Window:** Shows the `usuarios_roles` table from the `db_spring` schema. The table has columns `usuario_id` and `role_id`. Three rows are present: two for user `jose` (role `1`) and one for user `admin` (role `2`).
- Right Window:** Shows the `roles` table from the `db_spring` schema. The table has columns `id` and `nombre`. Two rows are present: `ROLE_USER` and `ROLE_ADMIN`.

Probamos con Postman

- El único api habilitado es el listado de todos los clientes los demás estarían deshabilitados

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'File', 'Edit', 'View', 'Help', and several icons. Below the bar, a search field says 'Search Postman'. To the right are buttons for 'Invite', 'Settings', 'Bell', and 'Upgrade'. The main workspace shows a list of API endpoints with status icons. A specific endpoint for 'GET /api/clientes' is highlighted with a red box. The 'Headers' tab is selected in the request configuration panel, which lists six headers. The 'Body' tab is selected in the response panel, showing a JSON response with 12 numbered lines. The response body is as follows:

```
1 {  
2   "id": 1,  
3   "nombre": "Jose",  
4   "apellido": "Perez",  
5   "email": "jp@hotmail.com",  
6   "telefono": 65433223,  
7   "createdAt": "2021-10-01",  
8   "imagen": null,  
9   "region": {  
10     "id": 1,  
11     "nombre": "Sudamerica"  
12   }  
}
```

At the bottom, there are tabs for 'Find and Replace', 'Console', 'Bootcamp', 'Runner', 'Trash', and other application controls.

Postman

File Edit View Help

Home Workspaces API Network Reports Explore Search Postman Invite Gear Bell Upgrade

No Environment

http://localhost:8087/api/clientes/1

GET http://localhost:8087/api/clientes/1

Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
--	-----	-------	-------------	-----	-----------

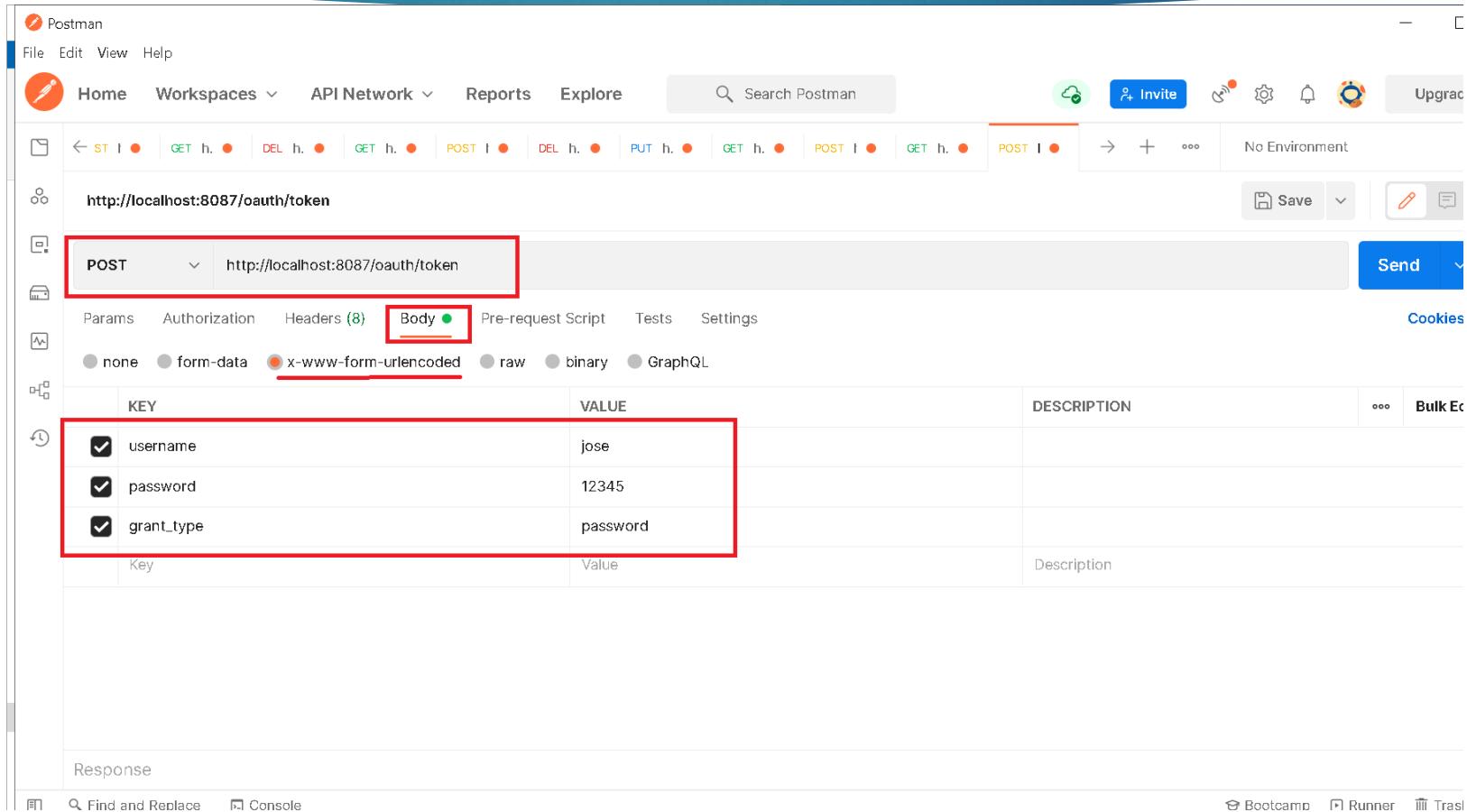
Body Cookies Headers (11) Test Results

Status: 401 Unauthorized Time: 43 ms Size: 557 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {  
2   "error": "unauthorized",  
3   "error_description": "Full authentication is required to access this resource"  
4 }
```

Con Spring Security realizamos la petición de un Token para poder utilizar las demás APIs



The screenshot shows the Postman application interface. A red box highlights the 'Body' tab under the request settings, and another red box highlights the 'x-www-form-urlencoded' option under the body type dropdown. The 'Body' table contains three key-value pairs:

KEY	VALUE
username	jose
password	12345
grant_type	password

Postman

File Edit View Help

Home Workspaces API Network Reports Explore Search Postman Invite Settings Upgrade

No Environment

http://localhost:8087/oauth/token

POST http://localhost:8087/oauth/token Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

Type Basic Auth Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. Learn more about variables

The authorization header will be automatically generated when you send the request. Learn more about authorization

Username angularapp
Password 12345 Show Password

ST | GET | DEL | GET | POST | DEL | PUT | GET | POST | GET | POST | → | + | No Environment | Save | Edit | Delete



Postman

File Edit View Help



Home

Workspaces

API Network

Reports

Explore

Search Postman



+ Invite



Upgrade



ST



GET h.



DEL h.



GET h.



POST h.



PUT h.



GET h.



POST h.



GET h.



POST h.



No Environment



http://localhost:8087/oauth/token

Save



POST

http://localhost:8087/oauth/token

Send



Params

Authorization



Headers

(9)

Body



Pre-request Script

Tests

Settings

Cookies

Body Cookies Headers (10) Test Results

Status: 200 OK Time: 586 ms Size: 1.08 KB

Save Response



Pretty

Raw

Preview

Visualize

JSON



```
1 {  
2   "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJleHAi0jE2MzgyNDU3NDcsInVZZXJfbmFtZSI6Impvc2UiLCJhdXRob3JpdGllcyI6WyJST0xFX1VTRVIiXSwianRpIjoiNzI2NTIzMmMTZGRh0S00YWIZLTLmZjItN2Q5ZTJjYmZkZmFkIiwiY2xpZW5  
ox2lkIjoiYW5ndWxcmFwcCIsInNjb3BlIjpbinJlyWQilCJ3cm10ZSJdfQ.Kybr4bUaafg_EmgIYP3VH2sMUuEtTQEEmbE_Ke5oc1rs",  
3   "token_type": "bearer",  
4   "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJ1c2VyX25hbWUi0iJqb3NlIiwc2NvcGUIolsicmVhZCIsIndyaXRlIl0sImF0aSI6IjcyNjUyMzzjLWRkYTktNGFiMy05ZmYyLTdkOWUyY2JmZGzhZCIsImV4cCI6MTYzODI0NTc0NywiYXV0aG9yaXR  
pZXMiolsiUk9MRV9VU0VSIL0sImp0aSI6ImFkZTkzYjFilWJkMGItNGRmMS05YzQ0LTg1ZjFl0TU3ZDI3MyIsImNsawudF9pZCI6ImFuZ3VsYXJhcHAifQ.  
BwqlFNw7df-90mgKXMQiO5WwixdSoqz-mpHdKqUTA1U",  
5   "expires_in": 3599,  
6   "scope": "read write",  
7   "jti": "7265236c-dda9-4ab3-9ff2-7d9e2cbfdfad"  
8 }
```



Find and Replace



Console

Bootcamp

Runner

Trash



Con el token podemos utilizar las todas las apis

The screenshot shows the Postman application interface. At the top, there's a navigation bar with 'File', 'Edit', 'View', 'Help', 'Home', 'Workspaces', 'API Network', 'Reports', 'Explore', and a search bar. To the right of the search bar are icons for cloud storage, invite, settings, notifications, and upgrade.

The main workspace shows a list of recent requests at the top, followed by a single request for 'http://localhost:8087/api/clientes/1'. The method is set to 'GET'.

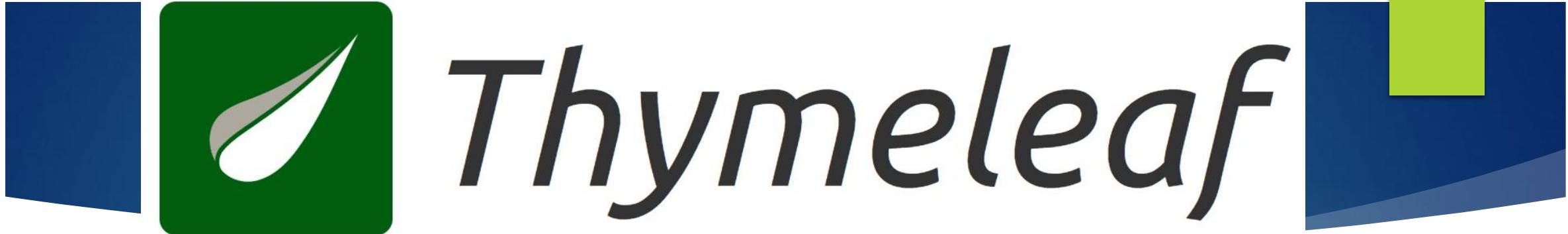
In the 'Authorization' tab of the request details, the 'Type' dropdown is set to 'Bearer Token'. A tooltip message is displayed: 'Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables.' Below this, it says 'The authorization header will be automatically generated when you send the request.' and provides a link to 'Learn more about authorization'.

The 'Token' input field contains the value 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ...'. This entire section is highlighted with a red box.

At the bottom, the response body is displayed in JSON format:

```
1 {  
2   "id": 1,  
3   "nombre": "Jose",  
4   "apellido": "Perez",  
5   "email": "jp@hotmail.com",  
6   "telefono": 65433223,  
7   "createdAt": "2021-10-01",  
}
```

The status bar at the bottom shows 'Status: 200 OK' and other details like time and size.

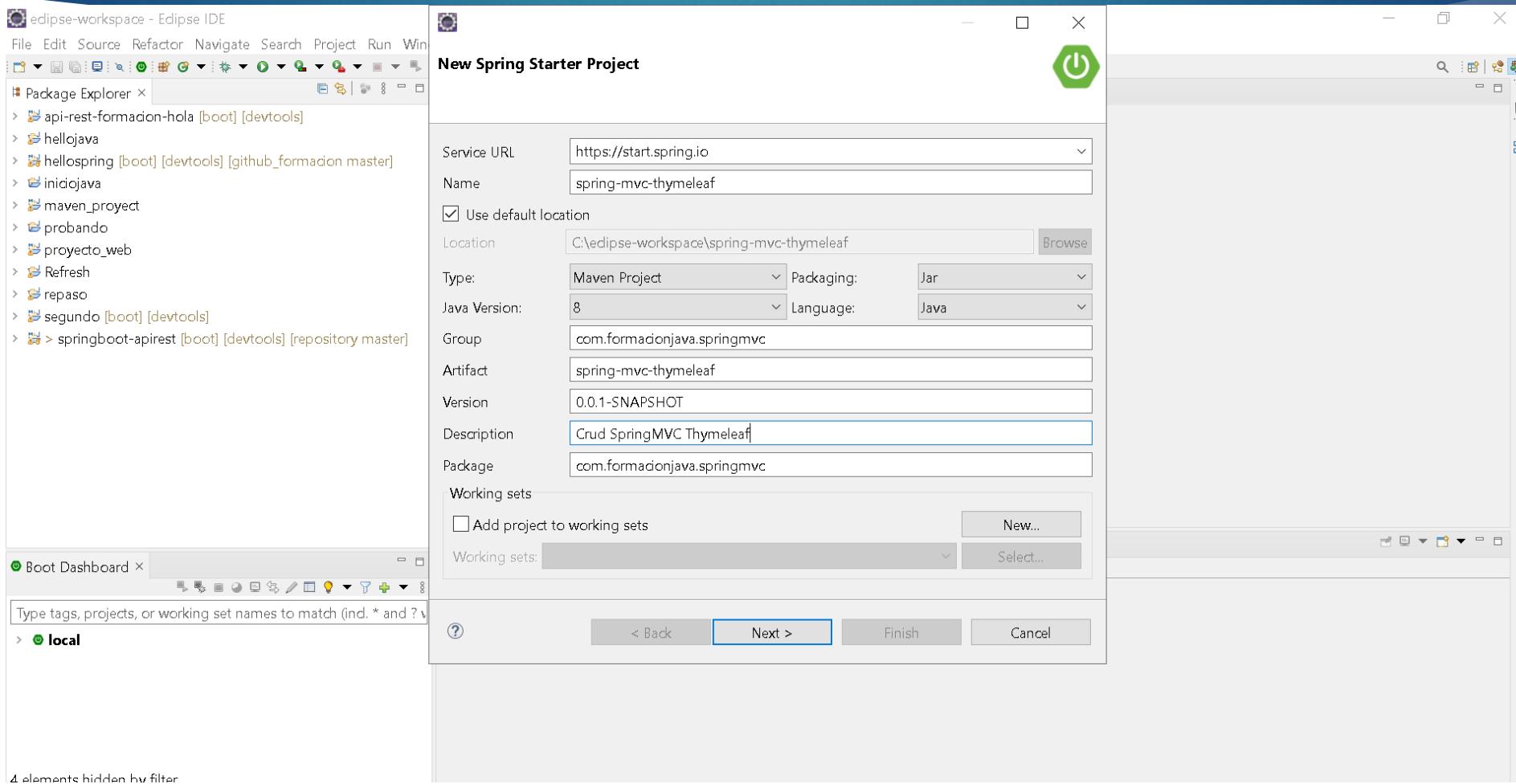


Thymeleaf

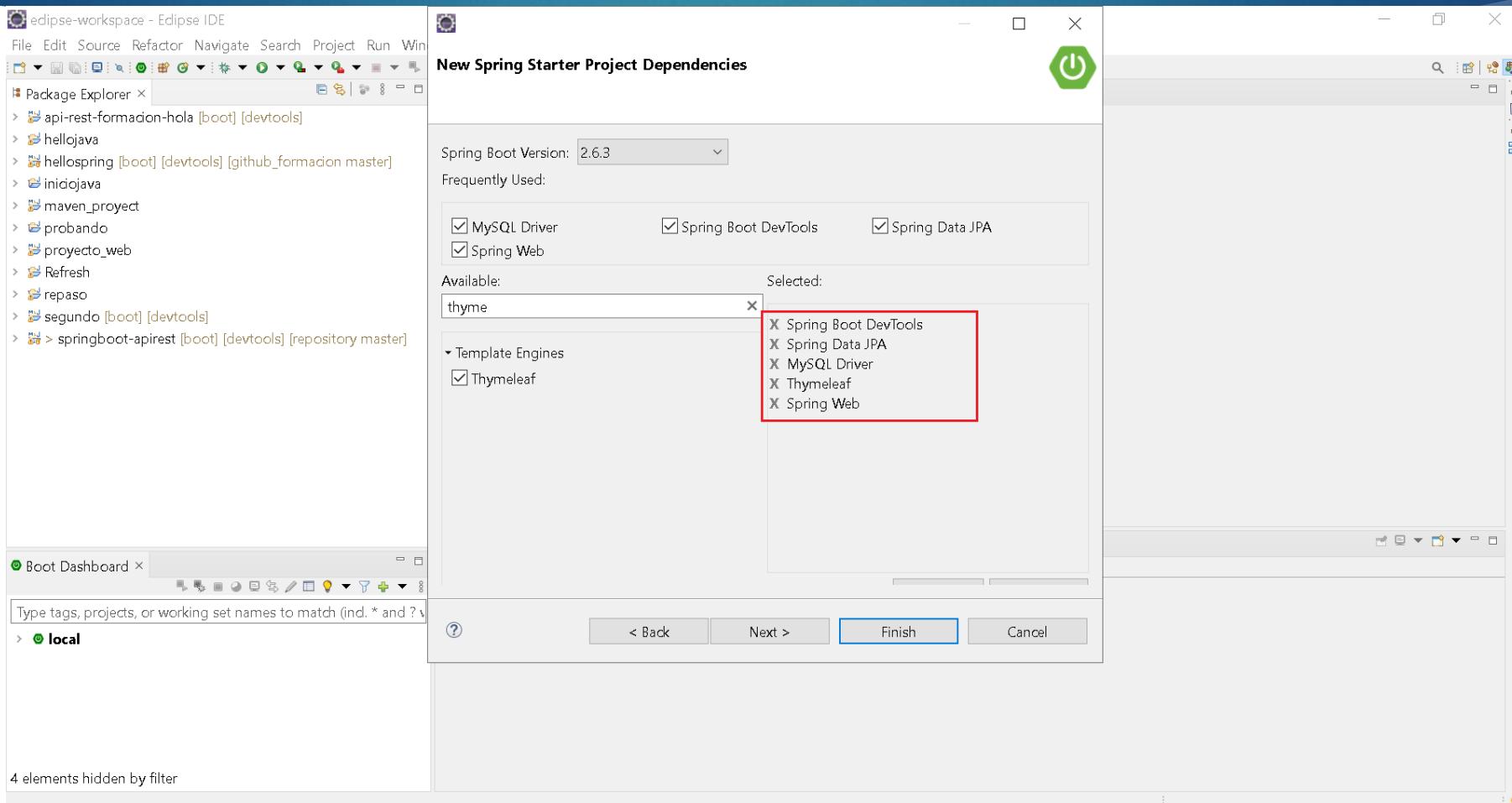
- ▶ **Thymeleaf** es un motor de plantillas de Java para procesar y crear HTML, XML, JavaScript, CSS y texto.

- ▶ <https://www.thymeleaf.org/>

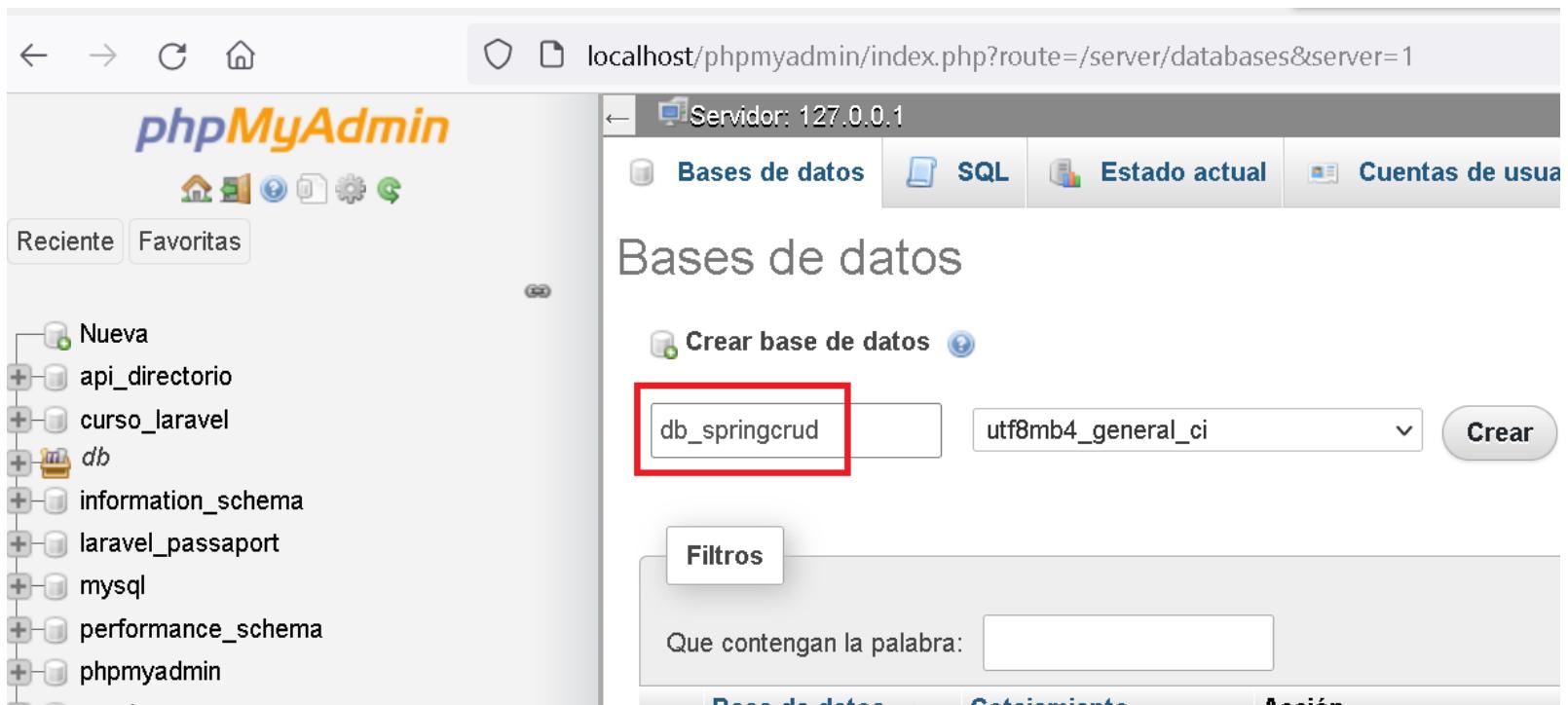
Creamos un nuevo proyecto



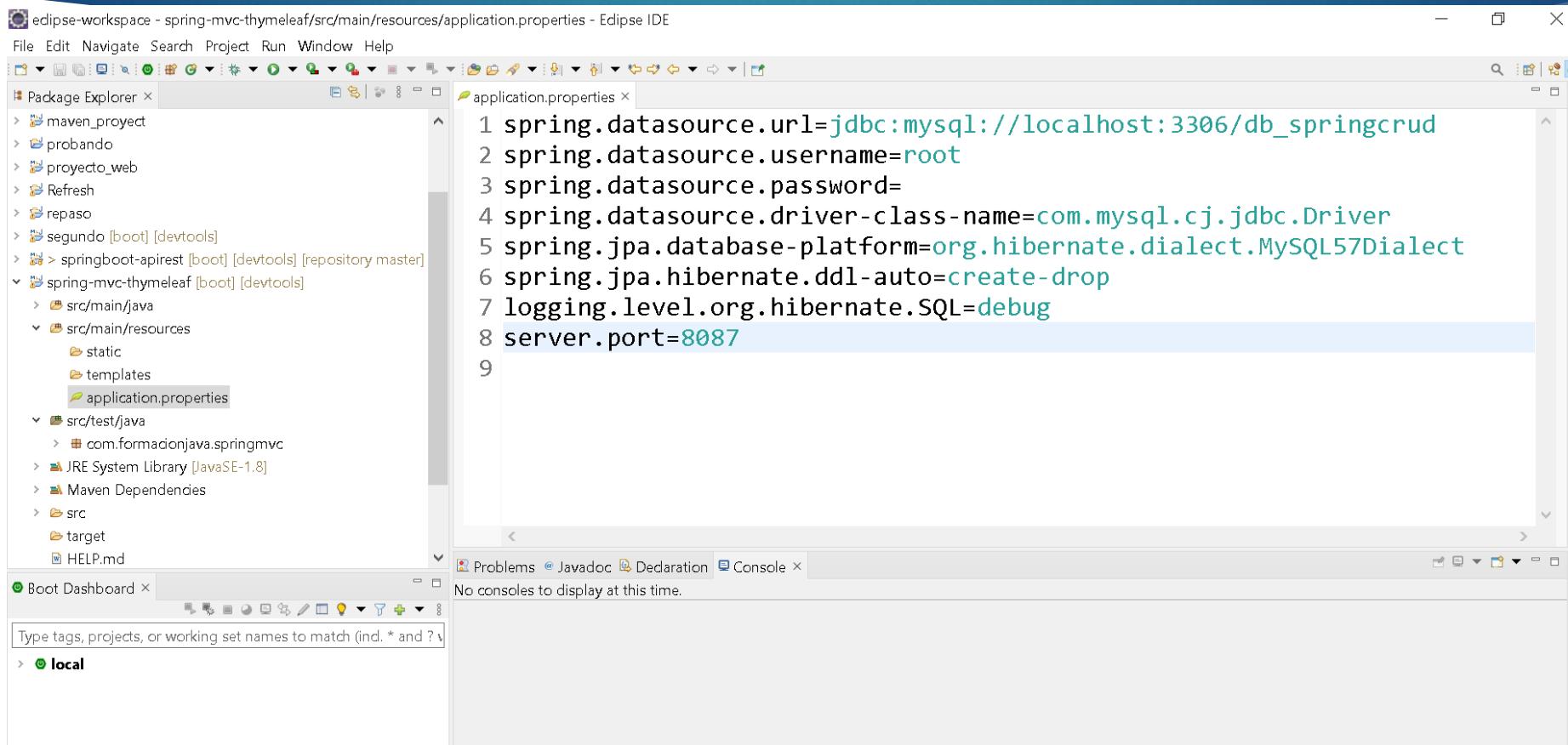
Agregamos las siguientes dependencias



Creamos la base de datos



Configuramos el application.properties



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - spring-mvc-thymeleaf/src/main/resources/application.properties - Edipse IDE". The menu bar includes File, Edit, Navigate, Search, Project, Run, Window, Help. The left sidebar is the Package Explorer showing projects like maven_proyect, probando, proyecto_web, Refresh, repaso, segundo [boot] [devtools], springboot-apirest [boot] [devtools] [repository master], and spring-mvc-thymeleaf [boot] [devtools]. The main editor window displays the contents of application.properties:

```
1 spring.datasource.url=jdbc:mysql://localhost:3306/db_springcrud
2 spring.datasource.username=root
3 spring.datasource.password=
4 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
5 spring.jpa.database-platform=org.hibernate.dialect.MySQL57Dialect
6 spring.jpa.hibernate.ddl-auto=create-drop
7 logging.level.org.hibernate.SQL=debug
8 server.port=8087
9
```

The bottom status bar says "No consoles to display at this time." and there is a search bar at the bottom left.

Creamos nuestra entidad

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** edipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/entity/Trabajador.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar with various icons for file operations.
- Package Explorer:** Shows the project structure:
 - mvnw
 - mvnw.cmd
 - pom.xml
 - spring-mvc-thymeleaf [boot] [devtools]
 - src/main/java
 - com.formacionjava.springmvc
 - com.formacionjava.springmvc.dao
 - com.formacionjava.springmvc.entity
 - Trabajador.java
 - src/main/resources
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - src
 - target
 - HELP.md
 - mvnw
 - mvnw.cmd
 - pom.xml
- Editor:** The main editor window displays the code for `Trabajador.java`. The code defines a class `Trabajador` that implements `Serializable`. It includes annotations for `@Entity`, `@Table(name="empleado")`, `@Id`, `@GeneratedValue(strategy = GenerationType.IDENTITY)`, and three `@Column` annotations for `nombre`, `apellido`, and `email`. A private attribute `telefono` is also declared.

```
2
3+import java.io.Serializable;■
11
12 @Entity
13 @Table(name="empleado")
14 public class Trabajador implements Serializable {
15
16@ Id
17 @GeneratedValue(strategy = GenerationType.IDENTITY)
18 private Long id;
19
20@ Column(nullable= false, length= 50)
21 private String nombre;
22
23@ Column(nullable= false, length= 50)
24 private String apellido;
25
26@ Column(nullable= false, length= 50,unique = true)
27 private String email;
28
29 private int telefono;
30
31
32
```
- Bottom Bar:** Shows tabs for Problems, Javadoc, Declaration, and Console. The Console tab is selected, displaying the message "No consoles to display at this time."

eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/entity/Trabajador.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer × Trabajador.java ×

```
44
45  public Long getId() {
46      return id;
47  }
48  public void setId(Long id) {
49      this.id = id;
50  }
51  public String getNombre() {
52      return nombre;
53  }
54  public void setNombre(String nombre) {
55      this.nombre = nombre;
56  }
57  public String getApellido() {
58      return apellido;
59  }
60  public void setApellido(String apellido) {
61      this.apellido = apellido;
62  }
63  public String getEmail() {
64      return email;
65  }
66  public void setEmail(String email) {
67      this.email = email;
```

Boot Dashboard ×

Type tags, projects, or working set names to match (ind)

> local

Problems Javadoc Declaration Console ×

No consoles to display at this time.

4 elements hidden by filter

edipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/entity/Trabajador.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help



Package Explorer ×

```
mvnw  
mvnw.cmd  
pom.xml  
spring-mvc-thymeleaf [boot] [devtools]  
src/main/java  
com.formacionjava.springmvc  
com.formacionjava.springmvc.dao  
com.formacionjava.springmvcentity  
Trabajador.java  
src/main/resources  
src/test/java  
JRE System Library [JavaSE-1.8]  
Maven Dependencies  
src  
target  
HELP.md  
mvnw  
mvnw.cmd  
pom.xml
```

Boot Dashboard ×

Type tags, projects, or working set names to match (in)

> local

```
*Trabajador.java ×  
60  public void setApellido(String apellido) {  
61      this.apellido = apellido;  
62  }  
63  public String getEmail() {  
64      return email;  
65  }  
66  public void setEmail(String email) {  
67      this.email = email;  
68  }  
69  public int getTelefono() {  
70      return telefono;  
71  }  
72  public void setTelefono(int telefono) {  
73      this.telefono = telefono;  
74  }  
75  
76  /**  
80  *  
81  */  
82  private static final long serialVersionUID = 1L;  
83
```

Problems @ Javadoc Declaration Console ×

No consoles to display at this time.

Creamos un repositorio dao

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/dao/TrabajadorRepositorio.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Package Explorer:** Shows the project structure:
 - repaso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repository]
 - spring-mvc-thymeleaf [boot] [devtools]
 - src/main/java
 - com.formacionjava.springmvc
 - com.formacionjava.springmvc.dao
 - TrabajadorRepositorio.java
 - com.formacionjava.springmvc.entity
 - src/main/resources
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - src
 - target
 - HELP.md
 - mvnw
 - mvnw.cmd
 - pom.xml
- Editor:** The file TrabajadorRepositorio.java is open, showing the following code:

```
1 package com.formacionjava.springmvc.dao;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4 import org.springframework.stereotype.Repository;
5
6 import com.formacionjava.springmvc.entity.Trabajador;
7
8 @Repository
9 public interface TrabajadorRepositorio extends JpaRepository<Trabajador, Long> {
10
11 }
```
- Bottom Bar:** Problems, Javadoc, Declaration, Console. The Console tab shows: "No consoles to display at this time."

Ahora creamos los servicios

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/service/TrabajadorServices.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Package Explorer:** Shows the project structure:
 - iniciojava
 - maven_proyect
 - probando
 - projeto_web
 - Refresh
 - repaso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repository mas]
 - spring-mvc-thymeleaf [boot] [devtools]
 - src/main/java
 - com.formacionjava.springmvc
 - com.formacionjava.springmvc.controllers
 - com.formacionjava.springmvc.dao
 - com.formacionjava.springmvc.entity
 - com.formacionjava.springmvc.service
 - TrabajadorServices.java (highlighted with a red box)
 - TrabajadorServiceImpl.java
 - src/main/resources
 - static
 - templates
 - application.properties
 - import.sql
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - Editor:** Displays the code for `TrabajadorServices.java`:

```
1 package com.formacionjava.springmvc.service;
2
3 import java.util.List;
4
5 public interface TrabajadorServices {
6
7     public List<Trabajador> listarTodosLosTrabajadores();
8
9 }
10
11
12 }
```

eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/service/TrabajadorServicesImpl.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer TrabajadorServicesImpl.java

```
1 package com.formacionjava.springmvc.service;
2
3 import java.util.List;
4
5 @Service
6 public class TrabajadorServicesImpl implements TrabajadorServices {
7
8     @Autowired
9     private TrabajadorRepository repositorio;
10
11     @Override
12     public List<Trabajador> listarTodosLosTrabajadores() {
13         return repositorio.findAll();
14     }
15 }
```

Problems Javadoc Declaration Console

<terminated> spring-mvc-thymeleaf - SpringMvcThymeleafApplication [Spring Boot App] C:\Users\dell\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/service/TrabajadorServicesImpl.java - Eclipse IDE
- Menu Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help
- Toolbar:** Standard Eclipse toolbar icons.
- Package Explorer:** Shows the project structure. A red box highlights the package `com.formacionjava.springmvc.service` and its file `TrabajadorServicesImpl.java`.
- Editor:** Displays the Java code for `TrabajadorServicesImpl.java`. The code implements the `TrabajadorServices` interface, uses `@Service` and `@Autowired` annotations, and overrides the `listarTodosLosTrabajadores` method to return all `Trabajador` objects from the `repositorio` repository.
- Bottom Status Bar:** Shows the status message: <terminated> spring-mvc-thymeleaf - SpringMvcThymeleafApplication [Spring Boot App] C:\Users\dell\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64

Creamos el controlador

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/controllers/TrabajadorController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations. The left side features the Package Explorer view, which lists several projects and their contents. A red box highlights the "src/main/java/com.formacionjava.springmvc.controllers.TrabajadorController.java" entry. The right side displays the code editor with the following Java code:

```
1 package com.formacionjava.springmvc.controllers;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Controller;
5 import org.springframework.ui.Model;
6 import org.springframework.web.bind.annotation.GetMapping;
7
8 import com.formacionjava.springmvc.service.TrabajadorServices;
9
10
11 @Controller
12 public class TrabajadorController {
13
14     @Autowired
15     private TrabajadorServices servicio;
16
17     @GetMapping={"/trabajadores","/"})
18     public String listarTrabajadores(Model modelo) {
19         modelo.addAttribute("trabajador",servicio.listarTodosLosTrabajadores());
20         return "trabajador";
21     }
22
23
24 }
```

Modelo

- ▶ El modelo puede proporcionar atributos utilizados para representar vistas

Creamos los datos de pruebas

The screenshot shows a Java IDE interface with the following components:

- Package Explorer:** On the left, it lists several projects and their contents:
 - iniciojava
 - maven_proyect
 - probando
 - proyecto_web
 - Refresh
 - repaso
 - segundo [boot] [devtools]
 - > springboot-apirest [boot] [devtools] [repository mas]
 - > spring-mvc-thymeleaf [boot] [devtools]
 - src/main/java
 - > com.formacionjava.springmvc
 - > com.formacionjava.springmvc.controllers
 - > TrabajadorController.java
 - > com.formacionjava.springmvc.dao
 - > com.formacionjava.springmvc.entity
 - > com.formacionjava.springmvc.service
 - src/main/resources
 - static
 - templates
 - application.properties
 - import.sql
 - > src/test/java
 - > JRE System Library [JavaSE-1.8]
- Connection profile:** A panel titled "Connection profile" with fields for Type, Name, Database, and Status (Disconnected, Auto).
- import.sql:** An open SQL file containing four `INSERT INTO empleado` statements, each inserting data for a different employee (Jose, Carlos, Maria, Gloria) with their respective names, surnames, emails, and phone numbers.

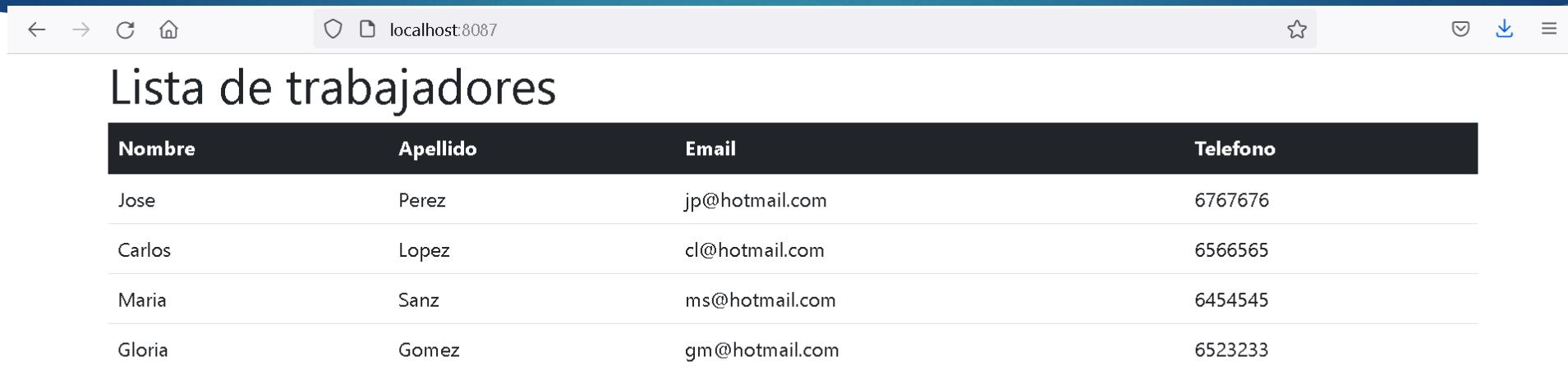
```
1 INSERT INTO empleado (nombre,apellido,email,telefono) VALUES('Jose','Perez','jp@hotmail.com',6767676);
2 INSERT INTO empleado (nombre,apellido,email,telefono) VALUES('Carlos','Lopez','cl@hotmail.com',6566565);
3 INSERT INTO empleado (nombre,apellido,email,telefono) VALUES('Maria','Sanz','ms@hotmail.com',6454545);
4 INSERT INTO empleado (nombre,apellido,email,telefono) VALUES('Gloria','Gomez','gm@hotmail.com',6523233);
```

Ahora agregamos la plantilla

The screenshot shows the Eclipse IDE interface. The title bar reads "eclipse-workspace - spring-mvc-thymeleaf/src/main/resources/templates/trabajador.html - Eclipse IDE". The menu bar includes File, Edit, Source, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations. The left side features the Package Explorer with a tree view of project files. The right side is the code editor with the file "trabajador.html" open. The code is written in Thymeleaf syntax, displaying HTML, CSS links, and Java beans. A red box highlights the "templates" folder in the package explorer and the head section of the HTML file.

```
<!DOCTYPE html>
<html xmlns:th="https://www.thymeleaf.org">
<head>
    <meta charset="utf-8" th:inline="never">
    <title>Listado de trabajadores</title>
    <!-- CSS only -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">
</head>
<body>
    <div class="container">
        <div class="row">
            <h1>Lista de trabajadores</h1>
        </div>
        <table class="table">
            <thead class="table-dark">
                <tr>
                    <th>Nombre</th>
                    <th>Apellido</th>
                    <th>Email</th>
                    <th>Telefono</th>
                </tr>
            </thead>
            <tbody>
                <tr th:each="trabajador: ${trabajador}">
                    <td th:text="${trabajador.nombre}*>Nombre</td>
                    <td th:text="${trabajador.apellido}*>Apellido</td>
                    <td th:text="${trabajador.email}*>Email</td>
                    <td th:text="${trabajador.telefono}*>Telefono</td>
                </tr>
            </tbody>
        </table>
    </div>
</body>
</html>
```

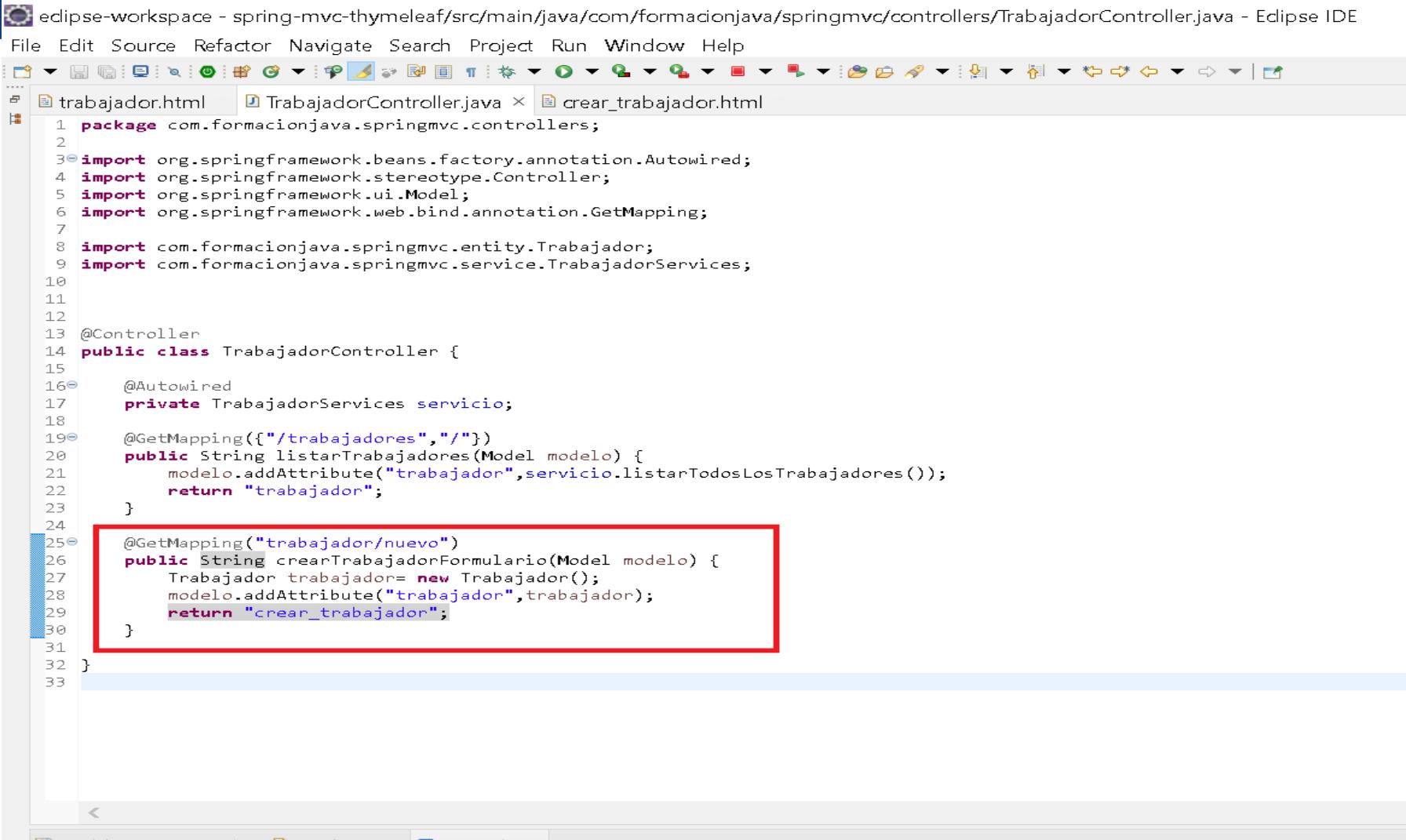
Ejecutamos y probamos todo hasta aquí



A screenshot of a web browser window displaying a table of workers. The browser's address bar shows "localhost:8087". The table has four columns: Nombre, Apellido, Email, and Telefono. The data is as follows:

Nombre	Apellido	Email	Telefono
Jose	Perez	jp@hotmail.com	6767676
Carlos	Lopez	cl@hotmail.com	6566565
Maria	Sanz	ms@hotmail.com	6454545
Gloria	Gomez	gm@hotmail.com	6523233

Ahora me creo un nuevo controlador



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/controllers/TrabajadorController.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Run. The left sidebar shows project files: "trabajador.html", "TrabajadorController.java", and "crear_trabajador.html". The main editor area contains Java code for a controller:

```
1 package com.formacionjava.springmvc.controllers;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.stereotype.Controller;
5 import org.springframework.ui.Model;
6 import org.springframework.web.bind.annotation.GetMapping;
7
8 import com.formacionjava.springmvc.entity.Trabajador;
9 import com.formacionjava.springmvc.service.TrabajadorServices;
10
11
12
13 @Controller
14 public class TrabajadorController {
15
16     @Autowired
17     private TrabajadorServices servicio;
18
19     @GetMapping={"/trabajadores","/"})
20     public String listarTrabajadores(Model modelo) {
21         modelo.addAttribute("trabajador",servicio.listarTodosLosTrabajadores());
22         return "trabajador";
23     }
24
25     @GetMapping("trabajador/nuevo")
26     public String crearTrabajadorFormulario(Model modelo) {
27         Trabajador trabajador= new Trabajador();
28         modelo.addAttribute("trabajador",trabajador);
29         return "crear_trabajador";
30     }
31
32 }
```

A red rectangular box highlights the code block from line 25 to 30, which defines a new method for creating a new employee form.

Agrego un navbar a mi html

edipse-workspace - spring-mvc-thymeleaf/src/main/resources/templates/trabajador.html - Eclipse IDE

File Edit Source Navigate Project Run Window Help

trabajador.html x TrabajadorController.java crear_trabajador.html

```
1 <!DOCTYPE html>
2 @html xmlns:th="https://www.thymeleaf.org/">
3 <head>
4   <meta charset="utf-8">
5   <title>Listado de trabajadores</title>
6   <!-- CSS only -->
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EqzqF4YVdCiI4qQz0WEfXKjwDKEZIuEc9pgwdqO6gWsfwpDfGzIbVUJG4/S4c" crossorigin="anonymous">
8
9
10 </head>
11 <body>
12
13   <nav class="navbar navbar-expand-md navbar-light bg-light">
14     <div class="container-fluid">
15       <a class="navbar-brand">Trabajadores</a>
16       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
17         <span class="navbar-toggler-icon"></span>
18       </button>
19       <div class="collapse navbar-collapse" id="navbarNav">
20         <ul class="navbar-nav">
21           <li class="nav-item">
22             <a class="nav-link active" aria-current="page" th:href="@{/trabajadores}">Trabajadores</a>
23           </li>
24           <li class="nav-item">
25             <a class="nav-link" th:href="@{/trabajador/nuevo}">Nuevo Trabajador</a>
26           </li>
27
28         </ul>
29       </div>
30     </div>
31   </nav>
32
33
34
35   <div class="container">
36     <div class="row">
37       <h1>Lista de trabajadores</h1>
38     </div>
39     <table class="table table-striped table-bordered">
40       <thead>
41         <tr>
42           <th>Nombre</th>
43           <th>Apellido</th>
44           <th>Edad</th>
45           <th>Sexo</th>
46           <th>Acciones</th>
47         </tr>
48       </thead>
49       <tbody>
50       </tbody>
51     </table>
52   </div>
```

Creo un nuevo html crear_trabajador

The screenshot shows the Eclipse IDE interface with the title bar "edipse-workspace - spring-mvc-thymeleaf/src/main/resources/templates/crear_trabajador.html - Edipse IDE". The menu bar includes File, Edit, Source, Navigate, Search, Project, Run, Window, and Help. The toolbar has various icons for file operations. The left sidebar shows project files: "trabajador.html", "TrabajadorController.java", and "crear_trabajador.html" (which is selected and highlighted with a red border). The main editor area displays the content of "crear_trabajador.html":

```
1 <!DOCTYPE html >
2 <html xmlns:th="https://www.thymeleaf.org/">
3 <head>
4 <meta charset="utf-8">
5 <title>Listado de trabajadores</title>
6 <!-- CSS only -->
7 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
8 rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFLdvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
9 crossorigin="anonymous">
10 </head>
11 <body>
12
13
14 <nav class="navbar navbar-expand-md navbar-light bg-light">
15   <div class="container-fluid">
16     <a class="navbar-brand" href="#">Trabajadores</a>
17     <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
18       <span class="navbar-toggler-icon"></span>
19     </button>
20     <div class="collapse navbar-collapse" id="navbarNav">
21       <ul class="navbar-nav">
22         <li class="nav-item">
23           <a class="nav-link active" href="#" th:href="@{/trabajadores}">Trabajadores</a>
24         </li>
25         <li class="nav-item">
26           <a class="nav-link" href="#">Nuevo Trabajador</a>
27         </li>
28       </ul>
29     </div>
30   </div>
31 </nav>
32
33
34
35
36 <div class="container">
37   <div class="row">
38     <div class="col-lg-6 col-md-6 col-sm-6 container justify-content-center card">
```

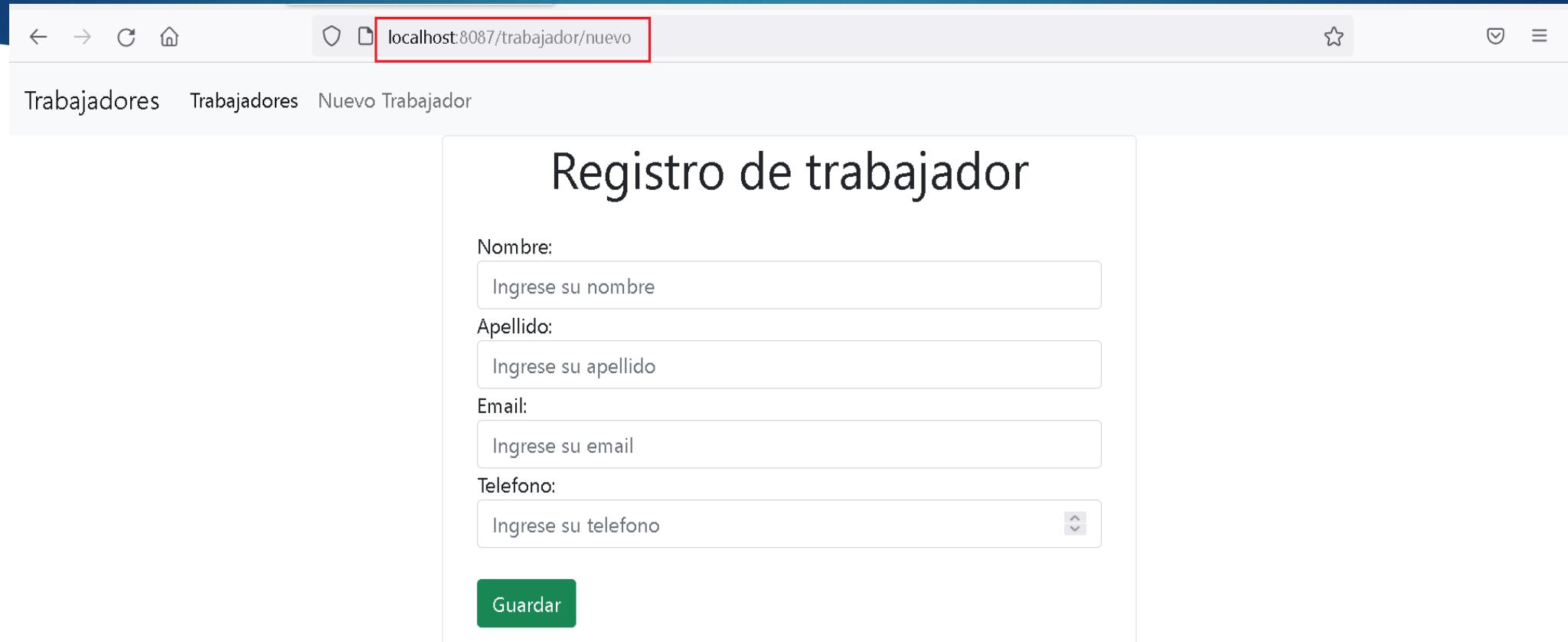
The status bar at the bottom shows "Problems @ Javadoc Declaration Console" and "spring-mvc-thymeleaf - SpringMvcThymeleafApplication [Spring Boot App] C:\Users\dell\p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe (2 feb 2022 7:15:21)".

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** On the left, under the package `spring-mvc-thymeleaf`, the `src/main/resources/templates` folder is expanded, showing files `crear_trabajador.html` and `trabajador.html`. The file `crear_trabajador.html` is currently selected and highlighted with a red box.
- Code Editor:** The main window displays the content of the `crear_trabajador.html` file. The code is a Thymeleaf template for creating a new employee. It includes a container div, a form for inputting employee details (Nombre, Apellido, Email, Telefono), and a footer with a "Guardar" button. The code uses Thymeleaf's `th:action`, `th:object`, and `th:field` attributes to bind data to the model.

```
<div class="container">
<div class="row">
<div class="col-lg-6 col-md-6 col-sm-6 container justify-content-center card">
    <h1 class="text-center" >Registro de trabajador</h1>
    <div class="card-body">
        <form th:action="@{/trabajador}" th:object="${trabajador}" method="POST" >
            <div class="form-group">
                <label>Nombre:</label>
                <input type="text" name="nombre" th:field="*{nombre}" class="form-control" placeholder="Ingrese su nombre" required>
            </div>
            <div class="form-group">
                <label>Apellido:</label>
                <input type="text" name="apellido" th:field="*{apellido}" class="form-control" placeholder="Ingrese su apellido" required>
            </div>
            <div class="form-group">
                <label>Email:</label>
                <input type="email" name="email" th:field="*{email}" class="form-control" placeholder="Ingrese su email" required>
            </div>
            <div class="form-group">
                <label>Telefono:</label>
                <input type="number" name="telefono" th:field="*{telefono}" class="form-control" placeholder="Ingrese su telefono" required>
            </div>
            <br>
            <div class="box-footer">
                <button class="btn btn-success">Guardar</button>
            </div>
        </form>
    </div>
</div>
</div>
</div>
```

Probamos la url y la vista



The screenshot shows a web browser window with the following details:

- Address Bar:** localhost:8087/trabajador/nuevo
- Page Title:** Registro de trabajador
- Form Fields:**
 - Nombre: Ingrese su nombre
 - Apellido: Ingrese su apellido
 - Email: Ingrese su email
 - Telefono: Ingrese su telefono
- Buttons:** A green "Guardar" button at the bottom left.

Agregamos al servicio el método para realizar registro

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** edipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formationjava/springmvc/service/TrabajadorServices.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Package Explorer:** Shows the project structure:
 - api-rest-formacion-hola [boot] [devtools]
 - hellojava
 - hellospring [boot] [devtools] [github_formacion]
 - iniciojava
 - maven_proyect
 - probando
 - projeto_web
 - Refresh
 - repasso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repository]
 - spring-mvc-thymeleaf [boot] [devtools]**
 - src/main/java
 - com.formationjava.springmvc
 - com.formationjava.springmvc.controllers
 - com.formationjava.springmvc.dao
 - com.formationjava.springmvc.entity
 - com.formationjava.springmvc.service**
 - TrabajadorServices.java**
 - TrabajadorServicesImpl.java
 - src/main/resources
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - src
 - target
 - HELP.md
 - mvnw
 - mvnw.cmd
 - pom.xml
- Editor:** Displays the content of TrabajadorServices.java:

```
1 package com.formationjava.springmvc.service;
2
3 import java.util.List;
4
5 public interface TrabajadorServices {
6
7     public List<Trabajador> listarTodosLosTrabajadores();
8
9     public Trabajador guardarTrabajador(Trabajador trabajador);
10
11 }
12
13
```

The method `public Trabajador guardarTrabajador(Trabajador trabajador);` is highlighted with a red box.
- Bottom Status Bar:** Problems, Javadoc, Declaration, Console.
- Bottom Footer:** spring-mvc-thymeleaf - SpringMvcThymeleafApplication [Spring Boot App] C:\Users\dell\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full

edipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/service/TrabajadorServicesImpl.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer TrabajadorServices.java TrabajadorServicesImpl.java

```
1 package com.formacionjava.springmvc.service;
2
3 import java.util.List;
4
5 @Service
6 public class TrabajadorServicesImpl implements TrabajadorServices {
7
8     @Autowired
9     private TrabajadorRepositorio repositorio;
10
11    @Override
12    public List<Trabajador> listarTodosLosTrabajadores() {
13        return repositorio.findAll();
14    }
15
16    @Override
17    public Trabajador guardarTrabajador(Trabajador trabajador) {
18        return repositorio.save(trabajador);
19    }
20
21}
22
23
24
25
26
27}
28
```

Problems Javadoc Declaration Console

spring-mvc-thymeleaf - SpringMvcThymeleafApplication [Spring Boot App] C:\Users\dell\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149

Writable Smart Insert 21 : 1 : 556

Agregamos el método de registro al controlador

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/controllers/TrabajadorController.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Package Explorer:** Shows the project structure:
 - iniciojava
 - maven_proyect
 - probando
 - projeto_web
 - Refresh
 - repaso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repo]
 - spring-mvc-thymeleaf [boot] [devtools]
 - src/main/java
 - com.formacionjava.springmvc
 - com.formacionjava.springmvc.controllers
 - TrabajadorController.java
 - com.formacionjava.springmvc.dao
 - com.formacionjava.springmvc.entity
 - com.formacionjava.springmvc.service
 - src/main/resources
 - static
 - templates
 - crear_trabajador.html
 - trabajador.html
 - application.properties
 - import.sql
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - src
- TrabajadorController.java Tab:** The current file being edited.
- Code Editor:** The code for the TrabajadorController class. A red box highlights the last section of the code, which contains the new registration method:

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.ModelAndView;
import com.formacionjava.springmvc.TrabajadorServices;
import com.formacionjava.springmvc.entity.Trabajador;

@Controller
public class TrabajadorController {

    @Autowired
    private TrabajadorServices servicio;

    @GetMapping({"/trabajadores", "/"})
    public String listarTrabajadores(Model modelo) {
        modelo.addAttribute("trabajador", servicio.listarTodosLosTrabajadores());
        return "trabajador";
    }

    @GetMapping("trabajador/nuevo")
    public String crearTrabajadorFormulario(Model modelo) {
        Trabajador trabajador= new Trabajador();
        modelo.addAttribute("trabajador", trabajador);
        return "crear_trabajador";
    }

    @PostMapping("/trabajador")
    public String guardarTrabajador(@ModelAttribute("trabajador") Trabajador trabajador) {
        servicio.guardarTrabajador(trabajador);
        return "redirect:/trabajadores";
    }
}
```

@ModelAttribute

- ▶ Es una anotación que vincula un parámetro de método o un valor de retorno de método a un atributo de modelo con nombre y luego lo expone a una vista web

Probamos el registro

The screenshot shows a web browser window with the following details:

- Address Bar:** localhost:8087/trabajador/nuevo
- Navigation:** Back, Forward, Stop, Home, Refresh, Star, Lock, and More.
- Breadcrumbs:** Trabajadores > Trabajadores > Nuevo Trabajador
- Section Title:** Registro de trabajador
- Fields:**
 - Nombre: Manuel
 - Apellido: Molas
 - Email: manuel@email.com
 - Telefono: 323232
- Buttons:** Guardar (Save)

Trabajadores Trabajadores Nuevo Trabajador

Lista de trabajadores

Nombre	Apellido	Email	Telefono
Jose	Perez	jp@hotmail.com	6767676
Carlos	Lopez	cl@hotmail.com	6566565
Maria	Sanz	ms@hotmail.com	6454545
Gloria	Gomez	gm@hotmail.com	6523233
Leonel	Messi	leo@messi.com	322323
Manuel	Molas	manuel@email.com	323232

Agrego a mi servicio los métodos faltantes para el crud completo

```
1 package com.formacionjava.springmvc.service;
2
3 import java.util.List;
4
5 public interface TrabajadorServices {
6
7     public List<Trabajador> listarTodosLosTrabajadores();
8
9     public Trabajador guardarTrabajador(Trabajador trabajador);
10
11    public Trabajador obtenerTrabajadorPorId(Long id);
12
13    public Trabajador actualizarTrabajador(Trabajador trabajador);
14
15    public void eliminarTrabajador(Long id);
16
17 }
18
19 }
```

```
17  @Override
18  public List<Trabajador> listarTodosLosTrabajadores() {
19      return repositorio.findAll();
20  }
21
22  @Override
23  public Trabajador guardarTrabajador(Trabajador trabajador) {
24      return repositorio.save(trabajador);
25  }
26
27  @Override
28  public Trabajador obtenerTrabajadorPorId(Long id) {
29      return repositorio.findById(id).get();
30  }
31
32  @Override
33  public Trabajador actualizarTrabajador(Trabajador trabajador) {
34      return repositorio.save(trabajador);
35  }
36
37  @Override
38  public void eliminarTrabajador(Long id) {
39      repositorio.deleteById(id);
40  }
41
42
43 }
```

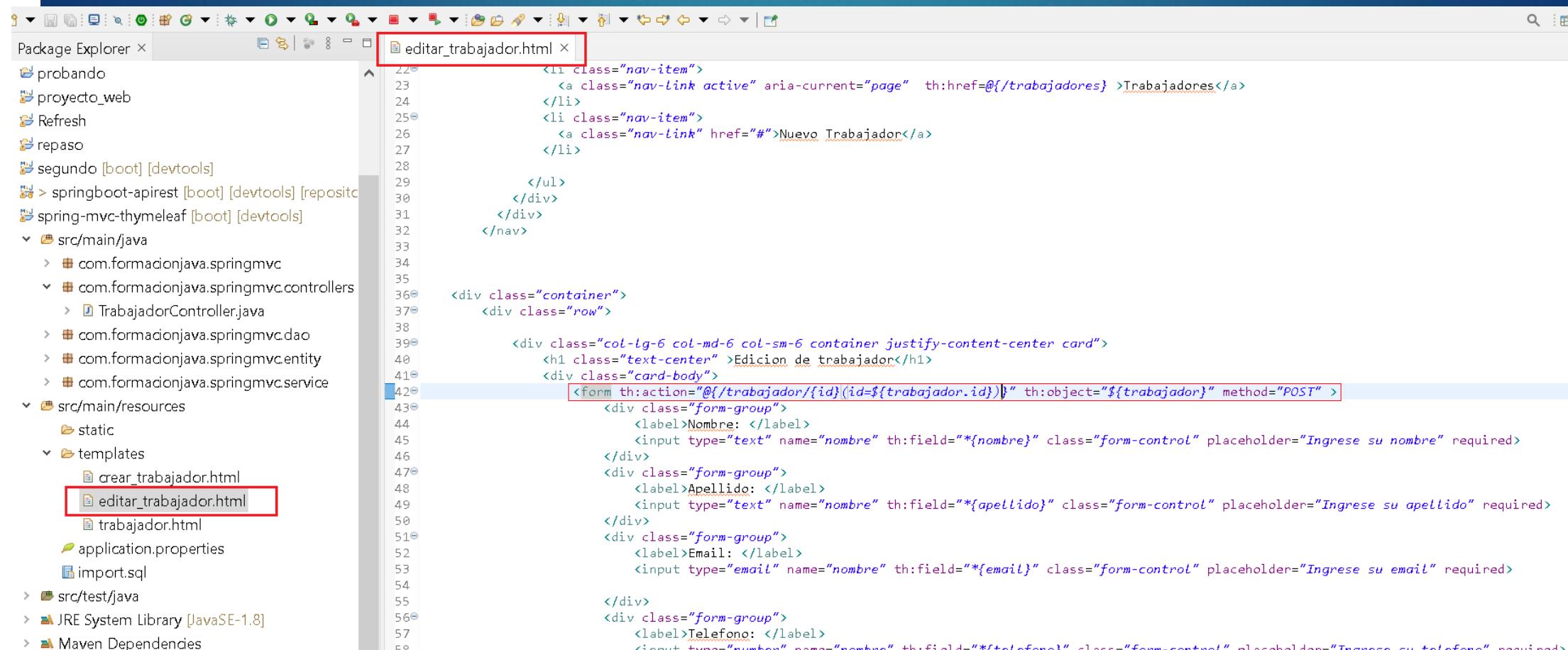
Agregamos los métodos al controlador

The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/controllers/TrabajadorController.java - Eclipse IDE
- Menu Bar:** File Edit Source Refactor Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Package Explorer:** Shows the project structure:
 - probando
 - projeto_web
 - Refresh
 - repaso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repository]
 - spring-mvc-thymeleaf [boot] [devtools]
 - src/main/java
 - com.formacionjava.springmvc
 - com.formacionjava.springmvc.controllers
 - TrabajadorController.java (highlighted with a red box)
 - com.formacionjava.springmvc.dao
 - com.formacionjava.springmvc.entity
 - com.formacionjava.springmvc.service
 - src/main/resources
 - static
 - templates
 - crear_trabajador.html
 - editar_trabajador.html
 - trabajador.html
 - application.properties
 - import.sql
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - src
 - target
 - HELP.md
 - mvnw
 - Editor:** Displays the code for TrabajadorController.java:

```
40
41     @GetMapping("/trabajador/editar/{id}")
42     public String mostrarFormularioEditar(@PathVariable Long id, Model modelo) {
43         modelo.addAttribute("trabajador", servicio.obtenerTrabajadorPorId(id));
44         return "editar_trabajador";
45     }
46
47     @PostMapping("/trabajador/{id}")
48     public String actualizarTrabajador(@PathVariable Long id, @ModelAttribute("trabajador") Trabajador trabajador,
49                                         Model modelo) {
50
51         Trabajador trabajadorExistente = servicio.obtenerTrabajadorPorId(id);
52
53         trabajadorExistente.setId(id);
54         trabajadorExistente.setNombre(trabajador.getNombre());
55         trabajadorExistente.setApellido(trabajador.getApellido());
56         trabajadorExistente.setEmail(trabajador.getEmail());
57         trabajadorExistente.setTelefono(trabajador.getTelefono());
58
59         servicio.actualizarTrabajador(trabajadorExistente);
60
61         return "redirect:/trabajadores";
62     }
63
64     @GetMapping("/trabajador/{id}")
65     public String eliminarTrabajador(@PathVariable Long id) {
66         servicio.eliminarTrabajador(id);
67         return "redirect:/trabajadores";
68     }
69
70
71 }
72 }
```
 - Bottom Bar:** Problems, Javadoc, Declaration, Console.

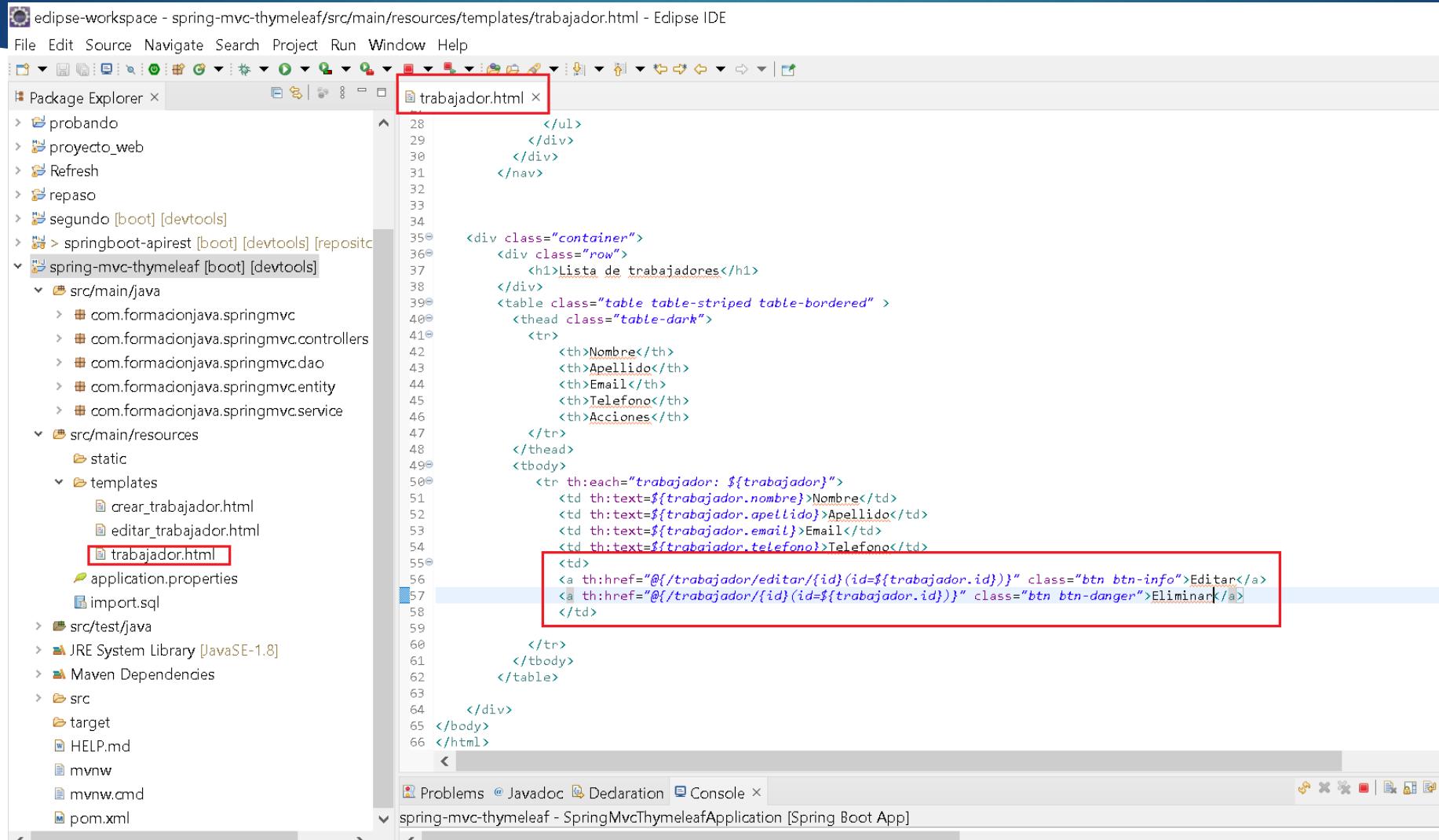
Creamos una copia de crear_trabajador y la llamamos editar_trabajador y solo editamos la línea marcada aquí



```
22<li class="nav-item">
23    <a class="nav-link active" aria-current="page" th:href="@{/trabajadores}">Trabajadores</a>
24</li>
25<li class="nav-item">
26    <a class="nav-link" href="#">Nuevo Trabajador</a>
27</li>
28
29</ul>
30</div>
31</div>
32</nav>
33
34
35
36<div class="container">
37    <div class="row">
38
39        <div class="col-lg-6 col-md-6 col-sm-6 container justify-content-center card">
40            <h1 class="text-center">Edición de trabajador</h1>
41            <div class="card-body">
42                <form th:action="@{/trabajador/{id}(id=${trabajador.id})}" th:object="${trabajador}" method="POST">
43                    <div class="form-group">
44                        <label>Nombre:</label>
45                        <input type="text" name="nombre" th:field="*{nombre}" class="form-control" placeholder="Ingrese su nombre" required>
46                    </div>
47                    <div class="form-group">
48                        <label>Apellido:</label>
49                        <input type="text" name="apellido" th:field="*{apellido}" class="form-control" placeholder="Ingrese su apellido" required>
50                    </div>
51                    <div class="form-group">
52                        <label>Email:</label>
53                        <input type="email" name="email" th:field="*{email}" class="form-control" placeholder="Ingrese su email" required>
54                    </div>
55                    <div class="form-group">
56                        <label>Teléfono:</label>
57                        <input type="number" name="telefono" th:field="*{telefono}" class="form-control" placeholder="Ingrese su teléfono" required>
58                </form>

```

En la plantilla trabajador.html agrego lo siguiente



The screenshot shows the Eclipse IDE interface with the following details:

- Title Bar:** eclipse-workspace - spring-mvc-thymeleaf/src/main/resources/templates/trabajador.html - Eclipse IDE
- Menu Bar:** File Edit Source Navigate Search Project Run Window Help
- Toolbar:** Standard Eclipse toolbar icons.
- Package Explorer:** Shows the project structure:
 - probando
 - projeto_web
 - Refresh
 - repasso
 - segundo [boot] [devtools]
 - springboot-apirest [boot] [devtools] [repository]
 - spring-mvc-thymeleaf [boot] [devtools]
 - src/main/java
 - com.formacionjava.springmvc
 - com.formacionjava.springmvc.controllers
 - com.formacionjava.springmvc.dao
 - com.formacionjava.springmvc.entity
 - com.formacionjava.springmvc.service
 - src/main/resources
 - static
 - templates
 - crear_trabajador.html
 - editar_trabajador.html
 - trabajador.html
 - application.properties
 - import.sql
 - src/test/java
 - JRE System Library [JavaSE-1.8]
 - Maven Dependencies
 - src
 - target
 - HELP.md
 - mvnw
 - mvnw.cmd
 - pom.xml
- Editor Area:** Displays the content of the 'trabajador.html' template file. A red box highlights the section of code being added:

```
<td>
<a th:href="@{/trabajador/editar/{id}(id=${trabajador.id})}" class="btn btn-info">Editar</a>
<a th:href="@{/trabajador/{id}(id=${trabajador.id})}" class="btn btn-danger">Eliminar</a>
</td>
```
- Bottom Status Bar:** Shows the tabs Problems, Javadoc, Declaration, Console, and the current project name: spring-mvc-thymeleaf - SpringMvcThymeleafApplication [Spring Boot App].

Probamos el CRUD completo

The screenshot shows a web browser window with the title "Listado de trabajadores". The address bar displays "localhost:8087/trabajadores". The page content is titled "Lista de trabajadores" and contains a table with four rows of employee data. Each row includes edit and delete buttons.

Nombre	Apellido	Email	Telefono	Acciones
Jose	Perez	jp@hotmail.com	6767676	<button>Editar</button> <button>Eliminar</button>
Carlos	Lopez	cl@hotmail.com	6566565	<button>Editar</button> <button>Eliminar</button>
Maria	Sanz	ms@hotmail.com	6454545	<button>Editar</button> <button>Eliminar</button>
Leonel	Messi	leo@messi.com	54578	<button>Editar</button> <button>Eliminar</button>