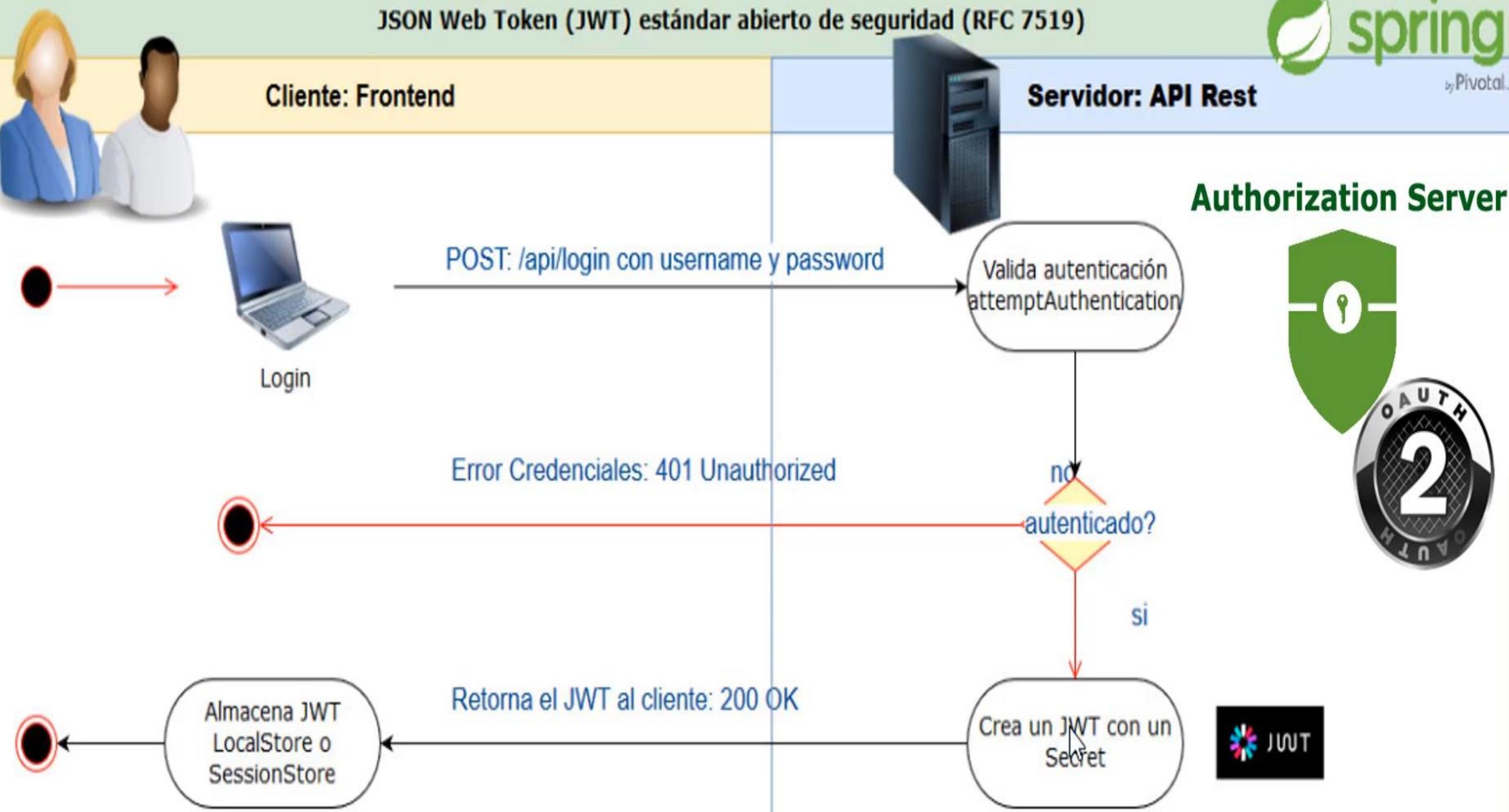
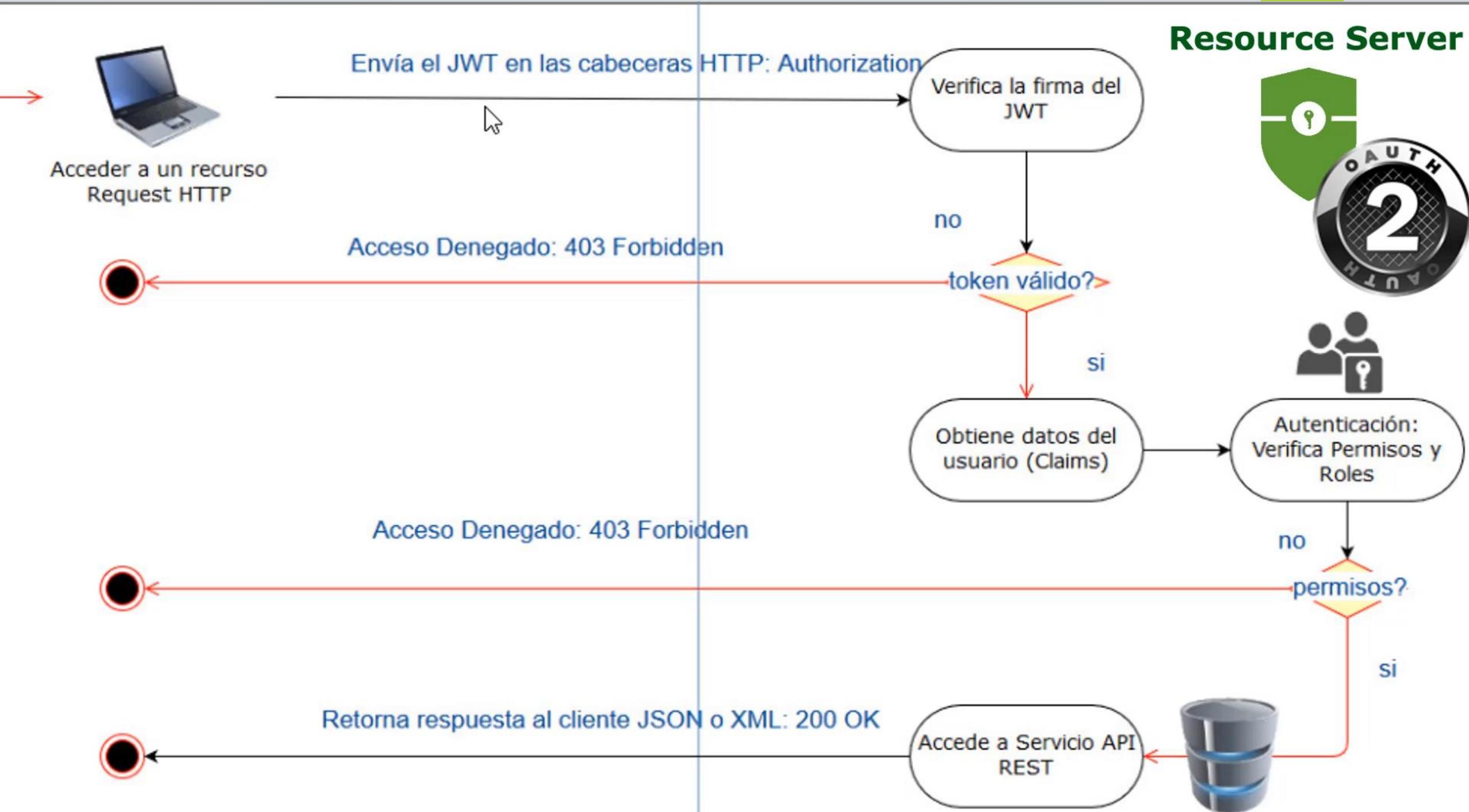


# Que es JWT

- ▶ 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/>



# 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)

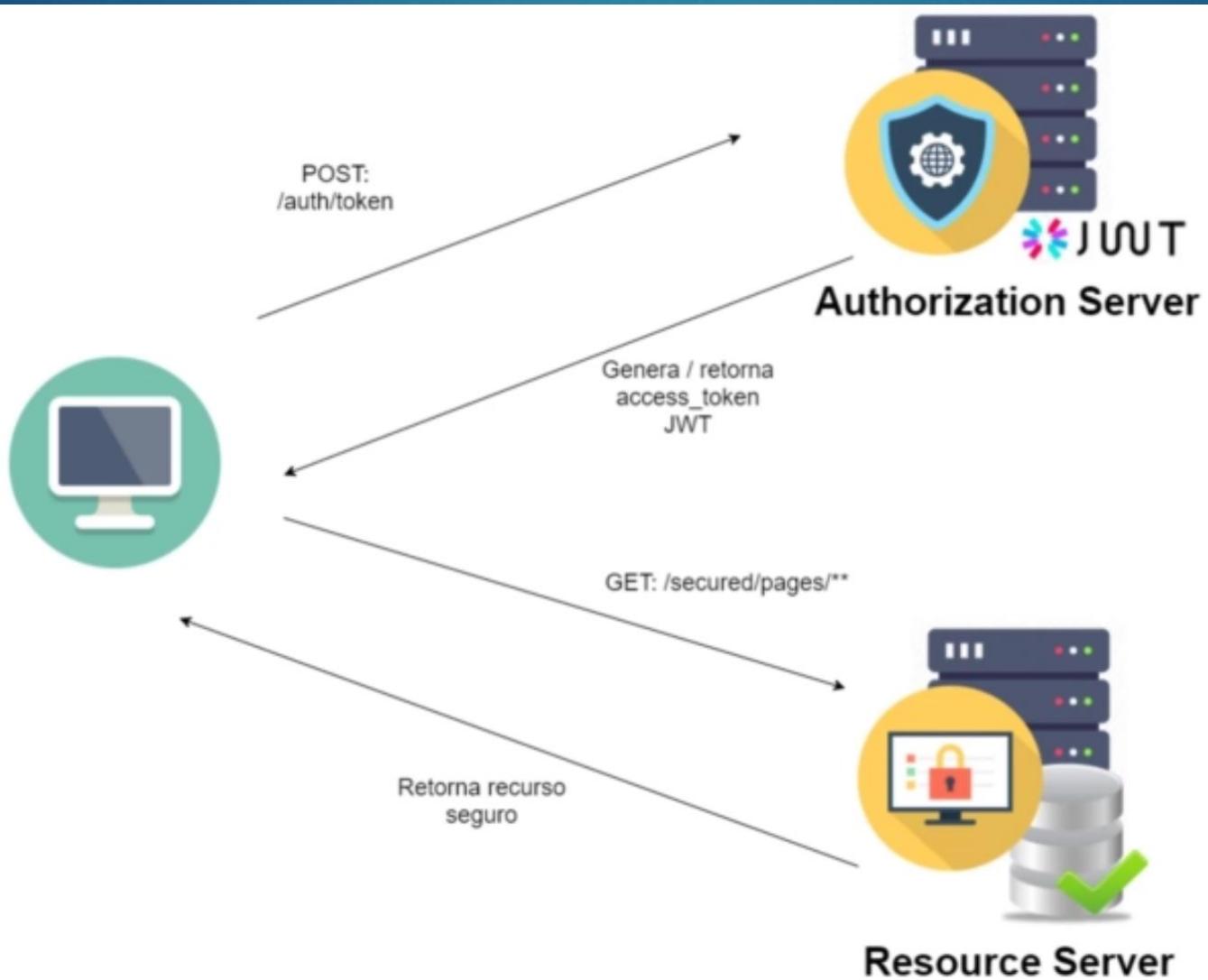
# Autenticación

- ▶ 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

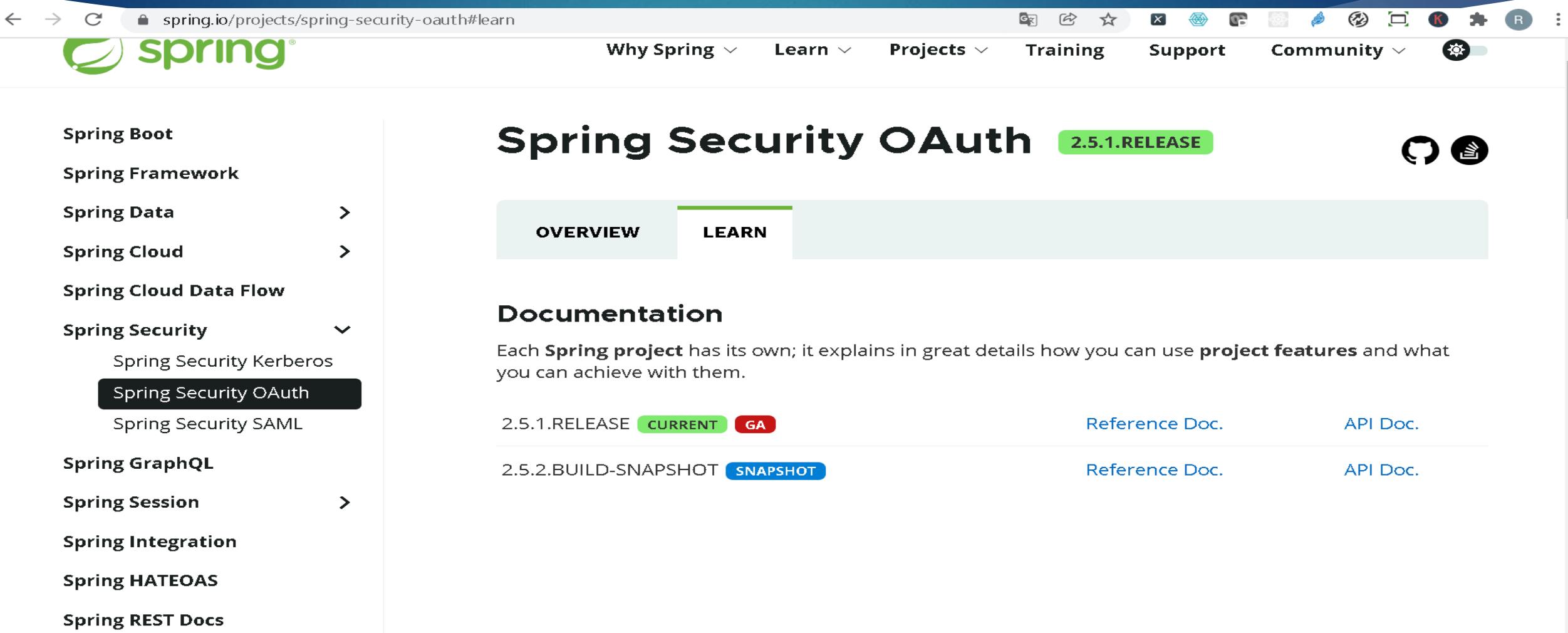


```
[  
  {  
    "id": 1,  
    "nombre": "Andrés",  
    "apellido": "Guzmán",  
    "email": "a@gmail.com"  
  
  },  
  {  
    "id": 2,  
    "nombre": "Mr. John",  
    "apellido": "Doe",  
    "email": "doe@gmail.com"  
  }  
]
```



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.

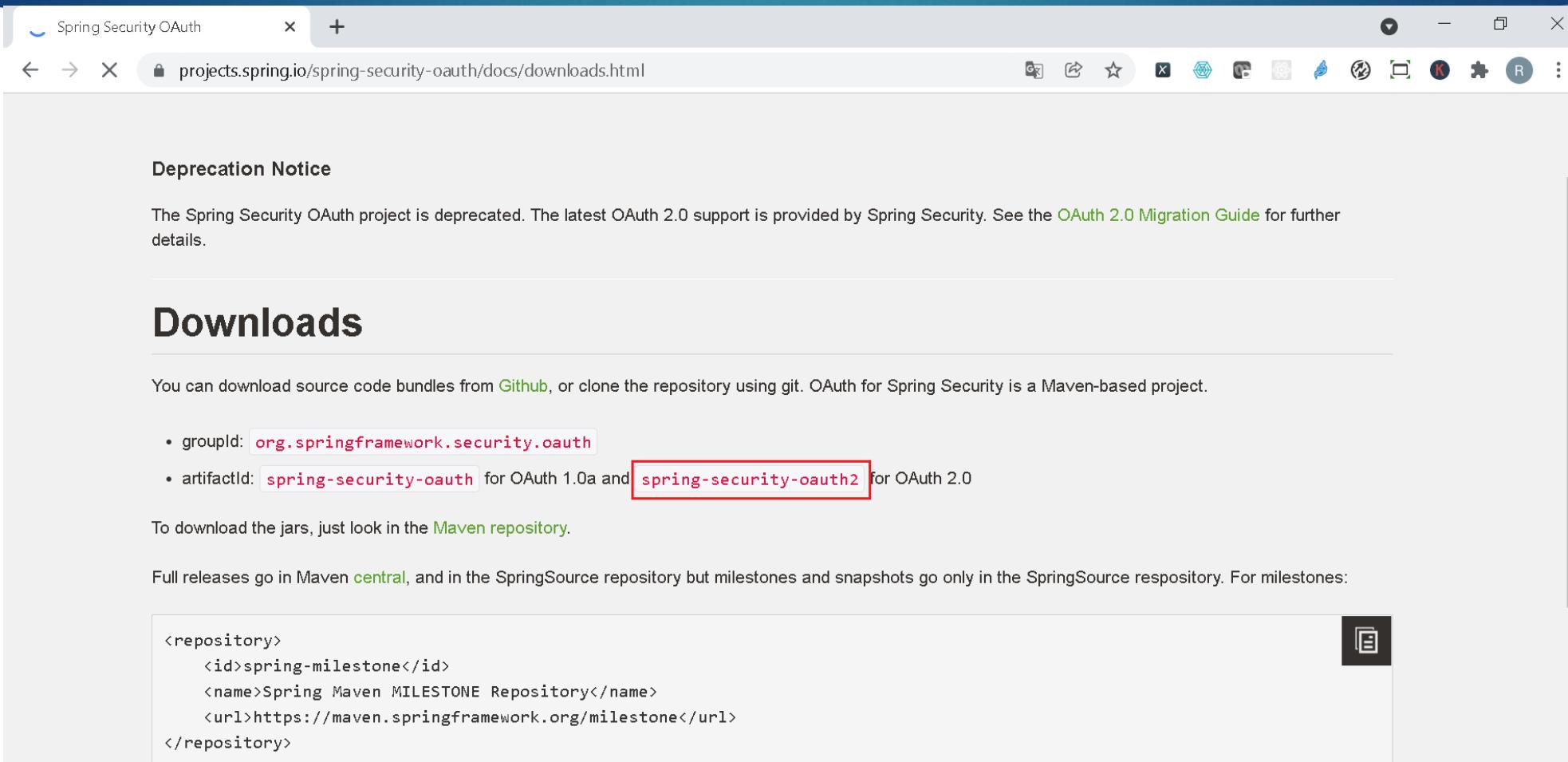
You're going to want to be quite familiar with both [OAuth](#) (and/or [OAuth2](#)) and [Spring Security](#), to maximize the effectiveness of this developers guide. OAuth for Spring Security is tightly tied to both technologies, so the more familiar you are with them, the more likely you'll be to recognize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

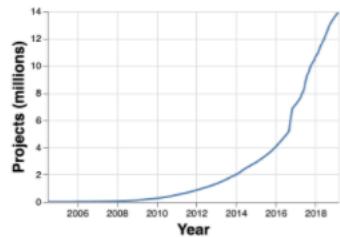


Search for groups, artifacts, categories

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	<a href="#">402 artifacts</a>

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

	Version	Repository	Usages	Date
2.5.x	<a href="#">2.5.1.RELEASE</a>	Central	13	Apr, 2021
	<a href="#">2.5.0.RELEASE</a>	Central	30	May, 2020
2.4.x	<a href="#">2.4.2.RELEASE</a>	Central	0	Oct, 2021
	<a href="#">2.4.1.RELEASE</a>	Central	11	Apr, 2020
	<a href="#">2.4.0.RELEASE</a>	Central	28	Nov, 2019
	<a href="#">2.3.8.RELEASE</a>	Central	25	Nov, 2019
	<a href="#">2.3.7.RELEASE</a>	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

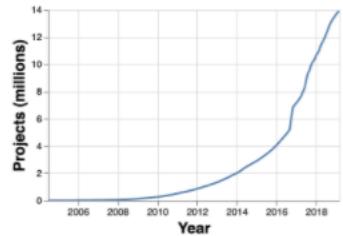


Search for groups, artifacts, categories

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 results**Sort: [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

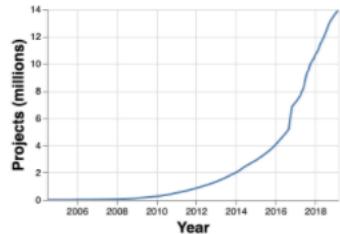


Search for groups, artifacts, categories

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

Search for groups, artifacts, categories

Search

Categories | Popular | Contact Us

**Indexed Artifacts (24.5M)**

Year	Projects (millions)
2006	0
2008	0.5
2010	1.0
2012	2.0
2014	4.0
2016	6.0
2018	12.0

**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

**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.

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

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>
```

Categories | Popular | Contact Us

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

**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
- janstey.sources 17

**Category****Found 943 results**Sort: [relevance](#) | [popular](#) | [newest](#)**1. JAXB API**

javax.xml.bind » jaxb-api

4,741 usages

CDDL

## JAXB API

Last Release on Aug 30, 2018

**2. Old JAXB Runtime**

com.sun.xml.bind » jaxb-impl

2,746 usages

EDL

Old JAXB Runtime module. Contains sources required for runtime processing.

Last Release on Aug 3, 2021

**3. JAXB Runtime**

org.glassfish.jaxb » jaxb-runtime

1,549 usages

EDL

JAXB (JSR 222) Reference Implementation

Last Release on Aug 3, 2021

**4. Old JAXB Core**

sun.xml.bind » jaxb-core

1,149 usages

EDL

**Search that just works**[Get started](#)

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

[Learn More!](#)

ADS VIA CARBON

**Indexed Repositories (1351)**

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

# 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

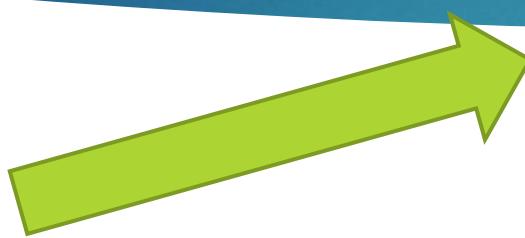
Project Explorer \*springboot-apirest/pom.xml

```
<!-- 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>
```

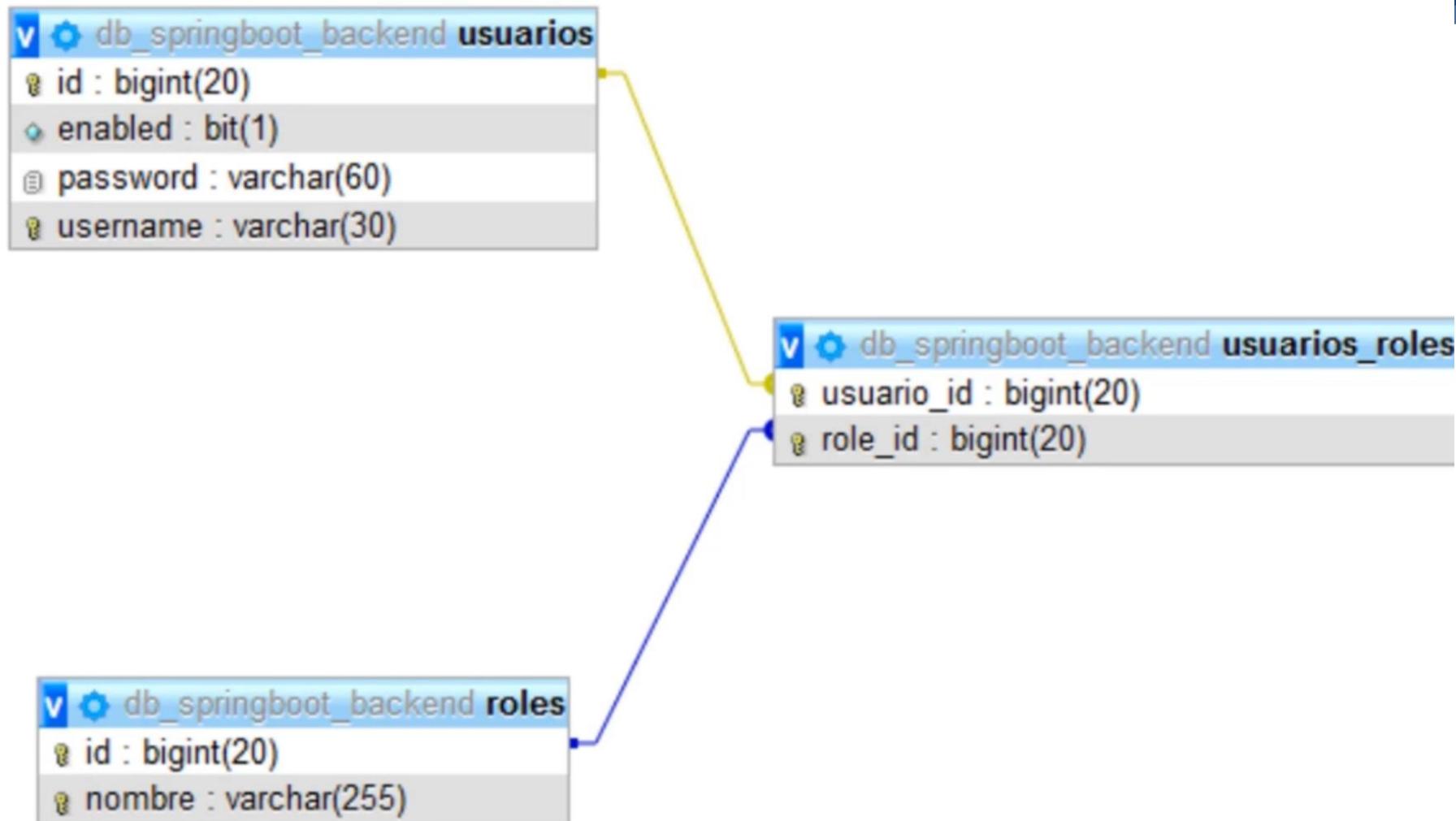
# Actualizamos las dependencias



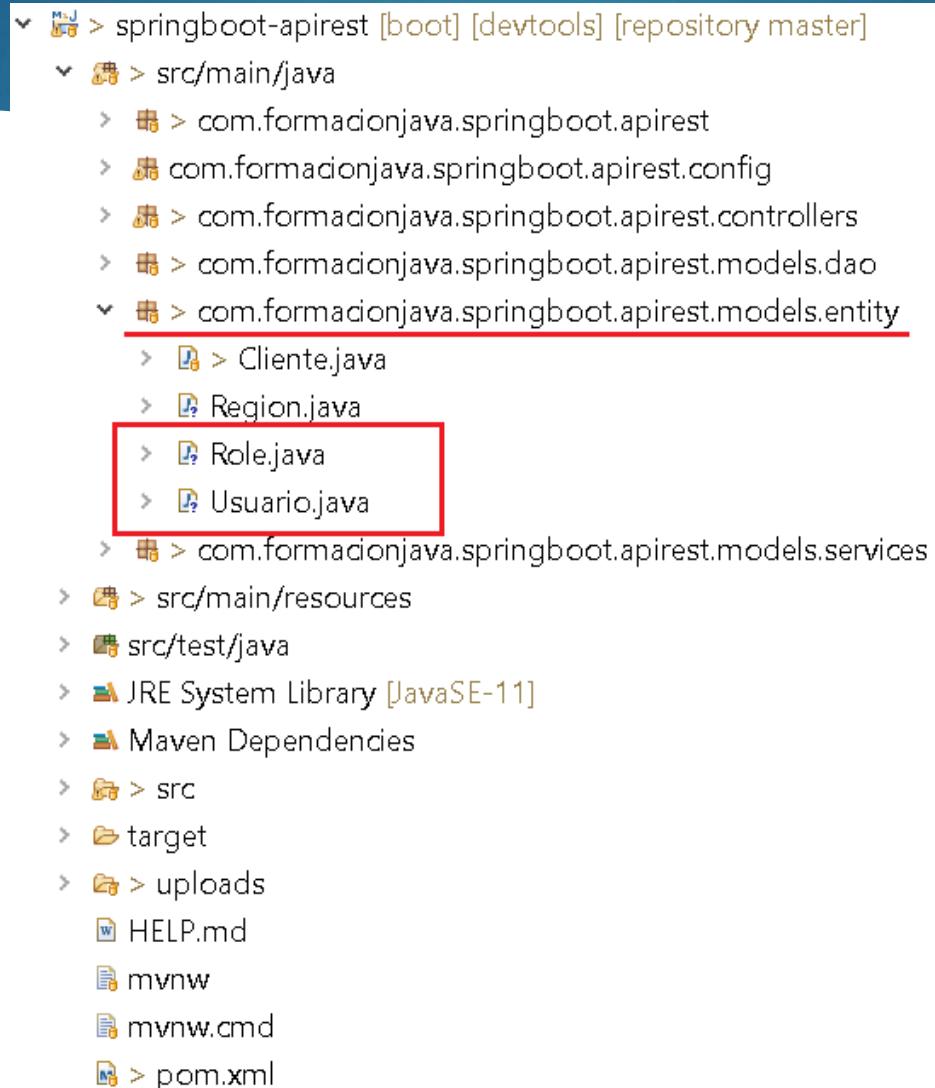
```
UsuarioService.java  springboot-apirest/pom.xml x
59      <!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-validation -->
60      <dependency>
61          <groupId>org.springframework.boot</groupId>
62          <artifactId>spring-boot-starter-validation</artifactId>
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



```
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 blue header bar. The main area displays a Java code editor for a file named `UsuarioService.java`. The code is as follows:

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

To the left of the code editor is the Project Explorer view, which lists several projects and their contents. The `springboot-apirest` project is expanded, showing its `src/main/java` directory. Inside this directory, the `com.formacionjava.springboot.apirest.models.services` package is highlighted with a red underline. Within this package, the `ClienteService.java` and `UsuarioService.java` files are listed. The `UsuarioService.java` file is currently selected and highlighted in the code editor.

```
*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

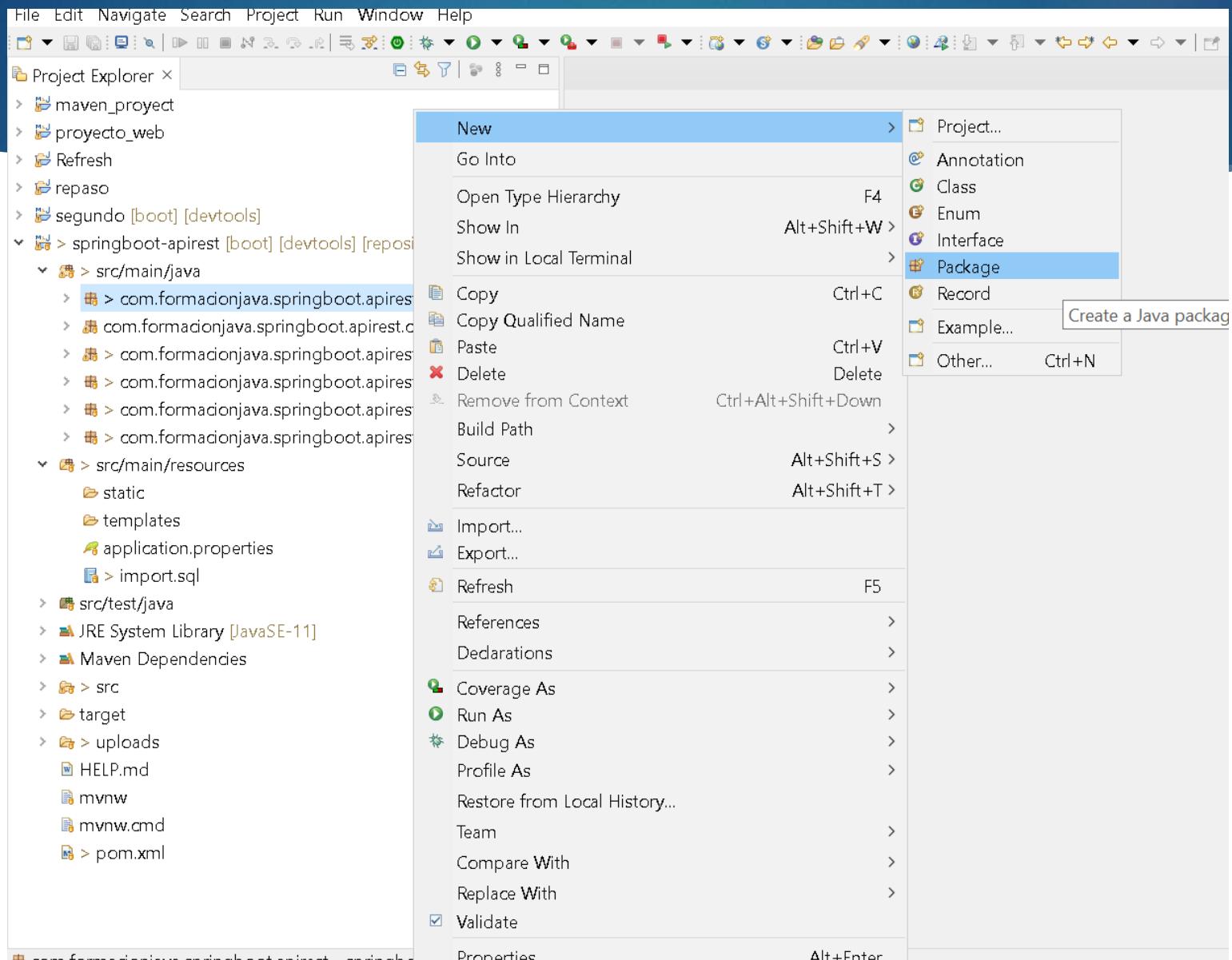
```
20
21 @Service
22 public class UsuarioService implements UserDetailsService{
23
24     private Logger logger = LoggerFactory.getLogger(UsuarioService.class);
25
26     @Autowired
27     private IUsuarioDao usuarioDao;
28
29     @Override
30     @Transactional(readOnly=true)
31     public UserDetails 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



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.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

## New Java Package

### Java Package

Create a new Java package.

Creates folders corresponding to packages.

Source folder:

Name:

Create package-info.java

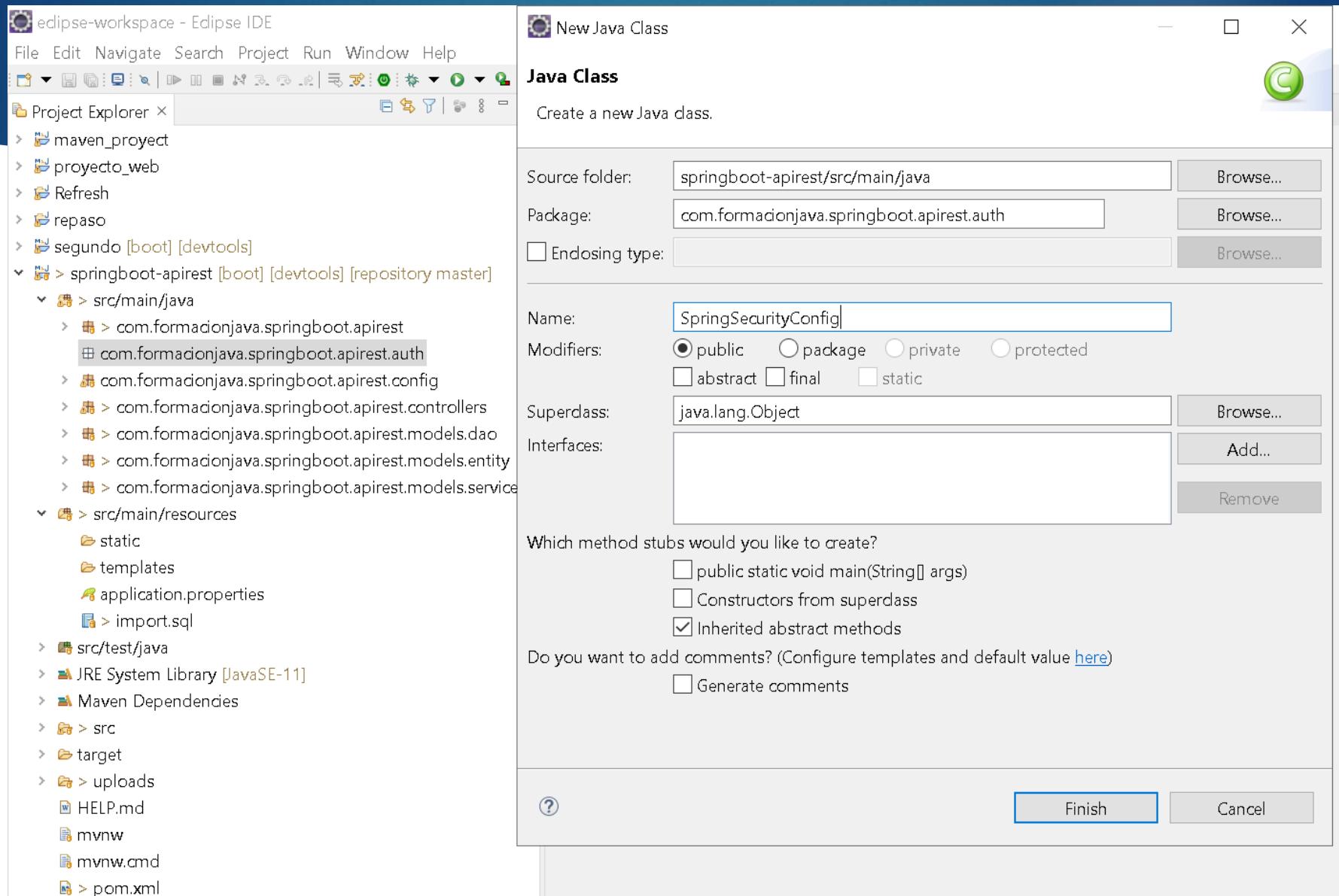
Generate comments (configure templates and default value [here](#))



Finish

Cancel

# Creamos una clase





Project Explorer ×

- > maven\_proyect
- > proyecto\_web
- > Refresh
- > repaso
- > segundo [boot] [devtools]
- > <b>springboot-apirest [boot] [devtools] [repository master]</b>
  - > <b>src/main/java</b>
    - > 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
  - > <b>src/main/resources</b>
    - > static
    - > templates
    - > application.properties
    - > import.sql
  - > <b>src/test/java</b>
  - > JRE System Library [JavaSE-11]
  - > Maven Dependencies
  - > <b>src</b>
  - > target
  - > <b>uploads</b>
    - > 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
```

```
*SpringSecurityConfig.java
1 package com
2
3 import org.
4 import org.
5 import org.
6 import org.
7
8 @Configurat
9 public class
10
11 @Autowi
12 private
13
14 }
15
16 }
```

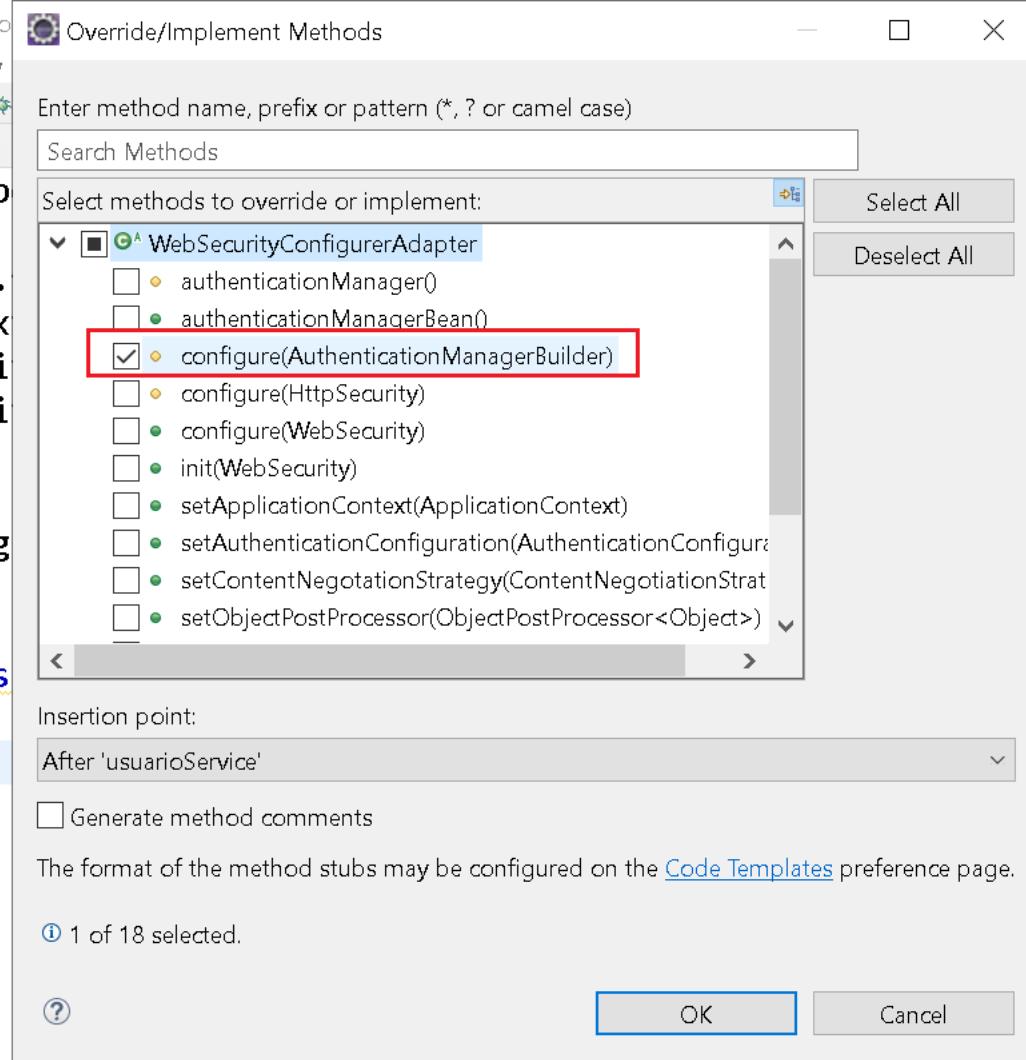
- Open Declaration F3
- Open Type Hierarchy F4
- Open Call Hierarchy Ctrl+Alt+H
- Show in Breadcrumb Alt+Shift+B
- Quick Outline Ctrl+O
- Quick Type Hierarchy Ctrl+T
- Open With >
- Show In Alt+Shift+W >
- Cut Ctrl+X
- Copy Ctrl+C
- Copy Qualified Name
- Paste Ctrl+V
- Raw Paste
- Quick Fix Ctrl+1
- Source Alt+Shift+S >
- Refactor Alt+Shift+T >
- Local History >
- References >
- Dedarations >
- Add to Snippets...
- Coverage As >
- Run As >
- Debug As >
- Profile As >
- Team >
- Compare With >
- Replace With >
- Validate >
- Preferences...

```
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
- Generate Getters and Setters...
- Generate Delegate Methods...
- Generate hashCode() and equals()...
- Generate toString()...
- Generate Constructor using Fields...

```
eclipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/config/SpringSecurityConfig.java
File Edit Source Refactor Navigate Search Project Run Window
*SpringSecurityConfig.java x
1 package com.formacionjava.springboot.config;
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
8 @Configuration
9 public class SpringSecurityConfig {
10
11     @Autowired
12     private UserDetailsService userDetailsService;
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 x

```
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
- Toolbar:** Standard Eclipse toolbar icons.
- 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
- Editor:** Shows the code for `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

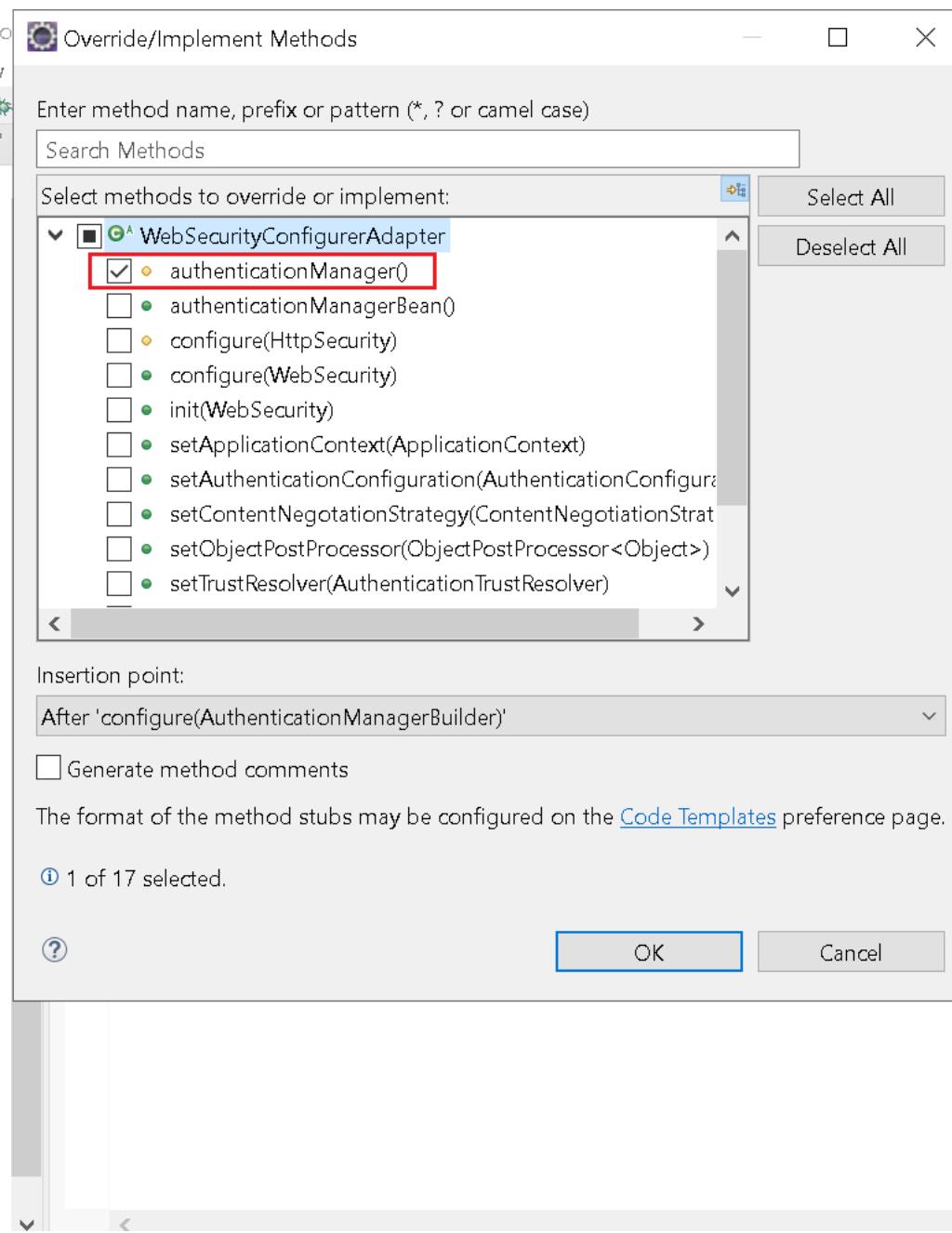
d; rerAdapter{  
th) throws Exception {  
dEncoder(passwordEncoder());

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



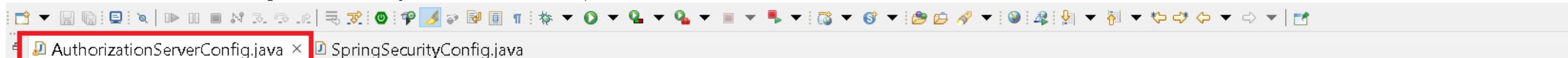
```
utowired;□  
  
ConfigurerAdapter{  
  
    @Override  
    public void configure(AuthenticationManagerBuilder auth) throws Exception {  
        auth.  
        passwordEncoder(passwordEncoder());  
    }  
}
```

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
3+ 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 defines 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 files: AuthorizationServerConfig.java (selected) and SpringSecurityConfig.java.
- Right Panel:** Content Editor showing Java code for AuthorizationServerConfig.java.
- Context Menu (Open with red box):** The menu is open over the first line of code. It includes options like Undo Typing, Save, Open Declaration, Quick Fix, Source, and Override/Implement Methods... (highlighted with a blue box).
- Bottom Status Bar:** Shows the current file path: config.java - Eclipse IDE.

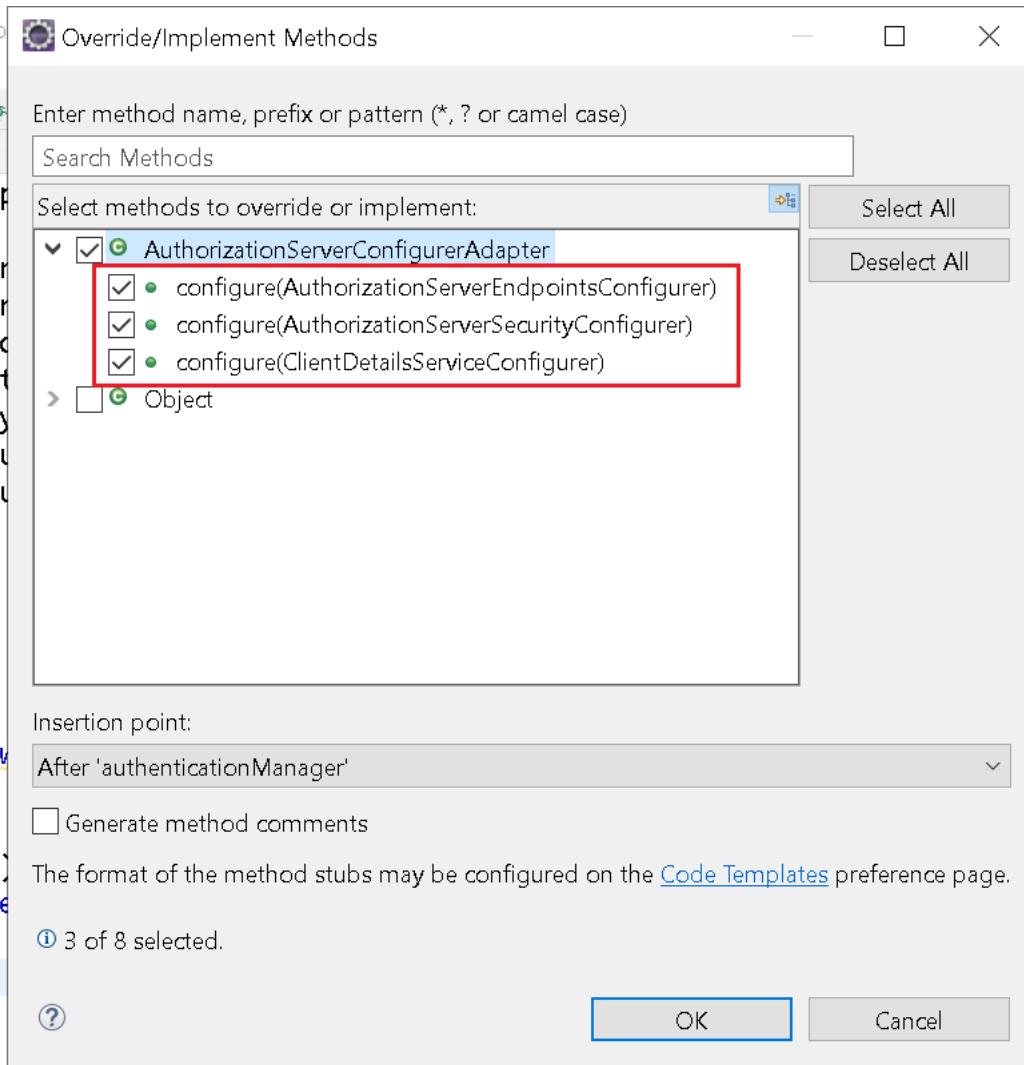
```
1 package com.formacionjava.springboot;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.beans.factory.annotation.Qualifier;
5 import org.springframework.context.annotation.Configuration;
6 import org.springframework.security.config.annotation.web.configuration.EnableAuthorizationServer;
7 import org.springframework.security.config.annotation.web.configuration.AuthorizationServerConfigurerAdapter;
8 import org.springframework.security.core.userdetails.UserDetailsService;
9 import org.springframework.security.core.userdetails.UserDetailsService;
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 }
```

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 contains the following Java code:

```
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 }
```

The code defines two configuration methods: `configure` for `ClientDetailsServiceConfigurer` and `configure` for `AuthorizationServerEndpointsConfigurer`. It uses the `super.configure` method and sets up the authentication manager, token store, and access token converter. Two specific sections of the code are highlighted with red boxes: the `@Bean` annotated `tokenStore()` method and the `@Bean` annotated `accessTokenConverter()` method.

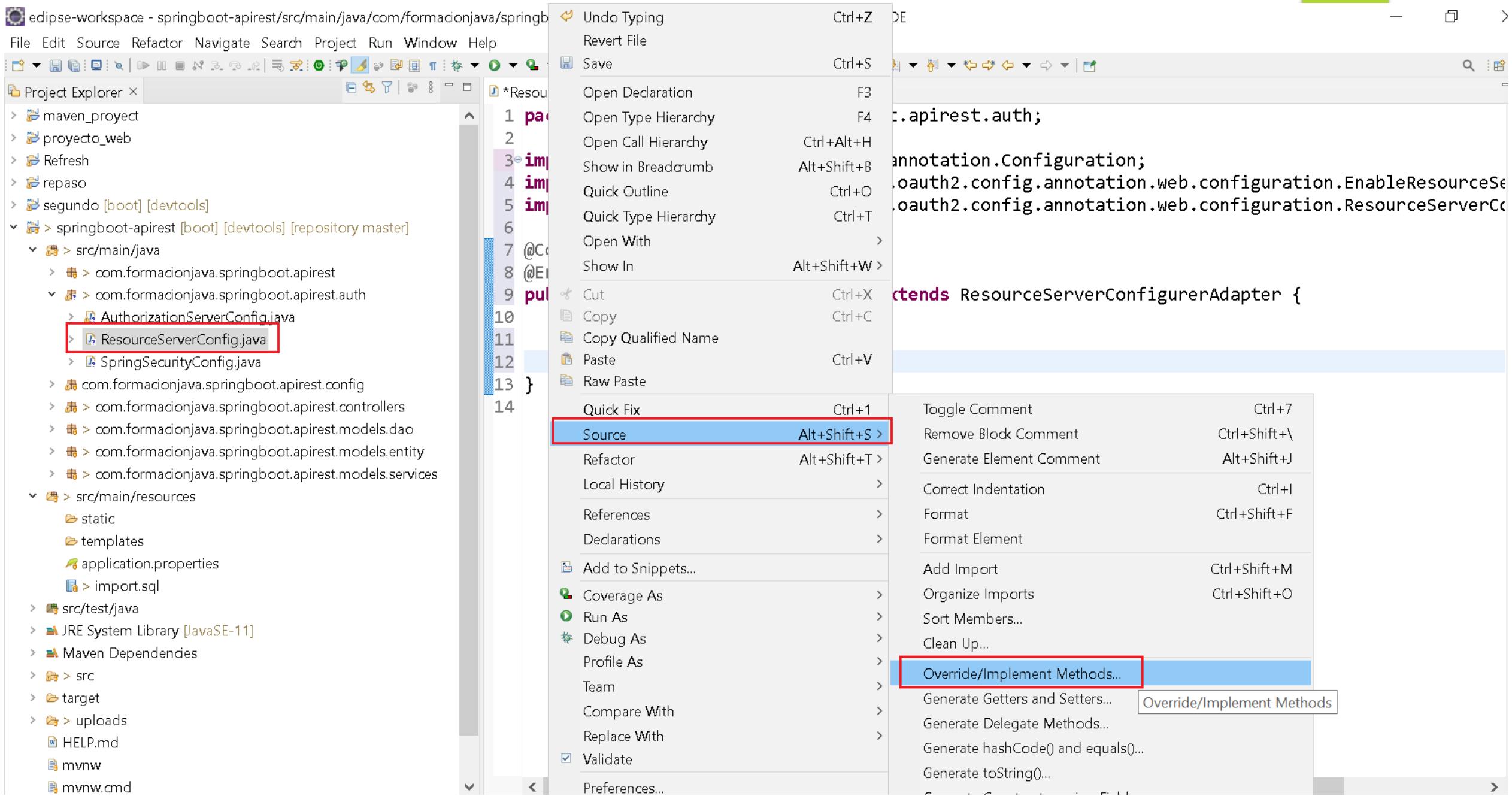
```
20 public class AuthorizationServerConfig extends AuthorizationServerConfigurerAdapter {
21
22     @Autowired
23     private BCryptPasswordEncoder passwordEncoder;
24
25     @Autowired
26     @Qualifier("authenticationManager")
27     private AuthenticationManager authenticationManager;
28
29     @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)
```

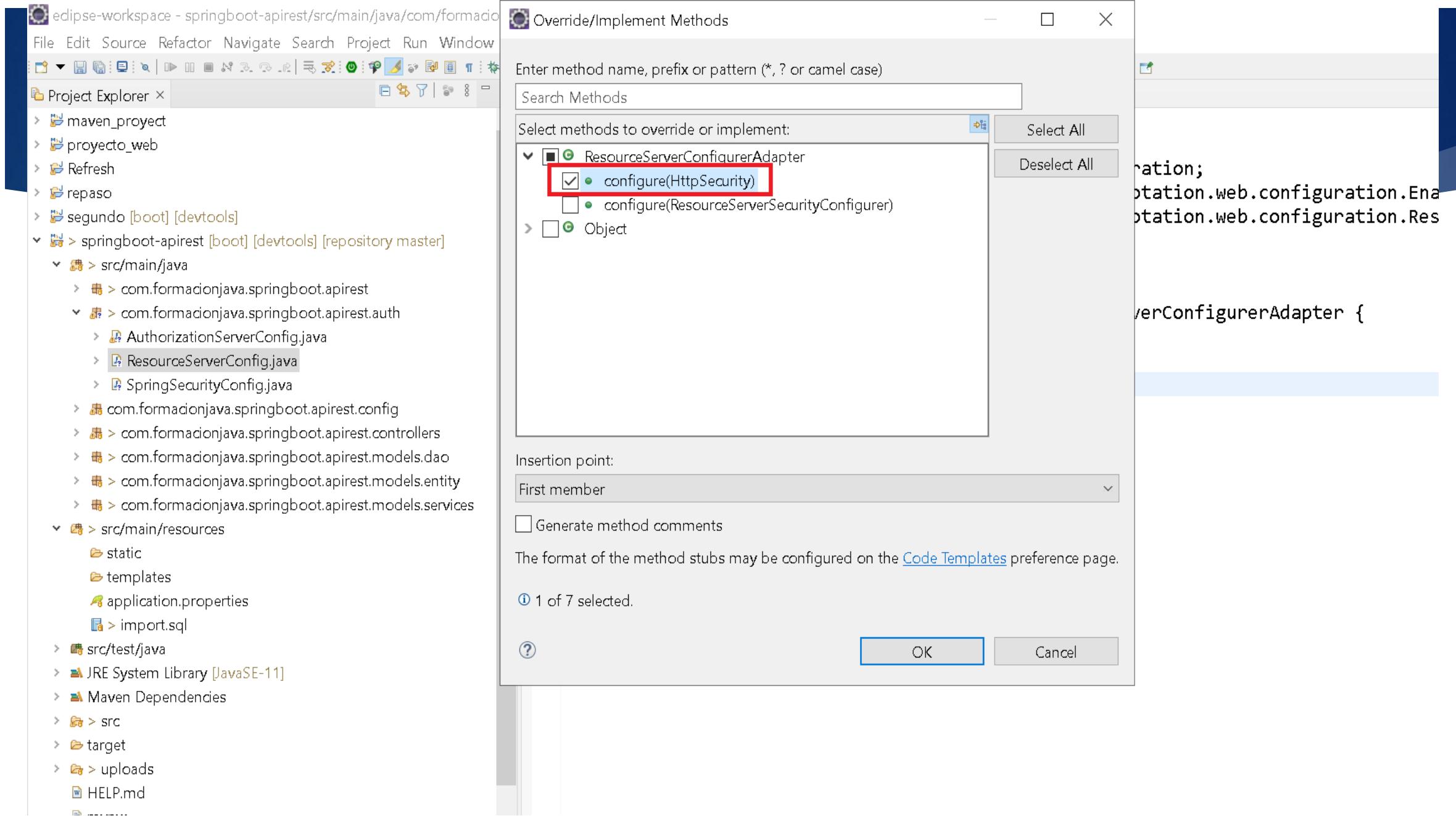
edipse-workspace - springboot-apirest/src/main/java/com/formacionjava/springboot/apirest/auth/ResourceServerConfig.java - Eclipse IDE

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 in the tree)
  - `AuthorizationServerConfig.java`
  - `ResourceServerConfig.java` (highlighted in the code editor)
  - `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:** On the right, the code for `ResourceServerConfig.java` is displayed. The file content is as follows:

```
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/clientes").permitAll()
16             .anyRequest().authenticated();
17     }
18
19
20
21 }
22
```

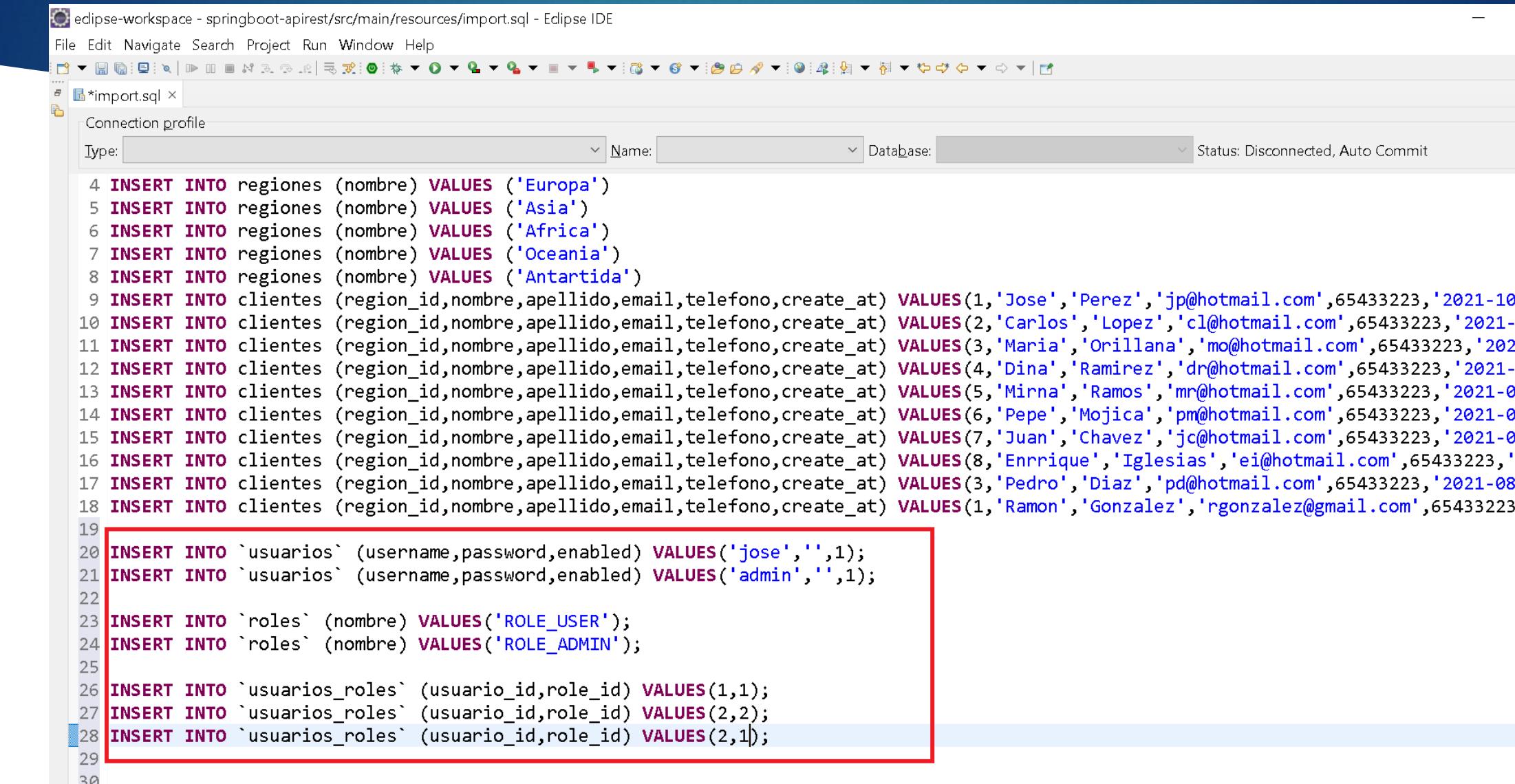
```
16  package com.formacionjava.springboot.apirest.auth;
17
18  import org.springframework.context.annotation.Configuration;
19  import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
20  import org.springframework.security.config.annotation.web.builders.HttpSecurity;
21  import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
22  import org.springframework.security.config.annotation.web.configuration.ResourceServerConfigurerAdapter;
23  import org.springframework.security.core.userdetails.UserDetailsService;
24  import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
25
26  @Configuration
27  @EnableWebSecurity
28  public class ResourceServerConfig extends ResourceServerConfigurerAdapter {
29
30     @Autowired
31     private UserDetailsService usuarioService;
32
33     @Override
34     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
35         auth.userDetailsService(this.usuarioService).passwordEncoder(passwordEncoder());
36     }
37
38     @Override
39     protected 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 `SpringSecurityConfig` class, which configures the security manager and HTTP security settings.
- Annotations:** The code uses various Java annotations such as `@Override`, `@Bean`, and `@Protected`.
- Red Box:** A red rectangular box highlights the `configure(HttpSecurity http)` method, specifically the line `http.authorizeRequests().anyRequest().authenticated()`.

```
ResourceServerConfig.java SpringSecurityConfig.java x
22     public BCryptPasswordEncoder passwordEncoder() {
23         return new BCryptPasswordEncoder();
24     }
25
26     @Override
27     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
28
29         auth.userDetailsService(this.usuarioService).passwordEncoder(passwordEncoder());
30     }
31
32     @Bean
33     @Override
34     protected AuthenticationManager authenticationManager() throws Exception {
35
36         // TODO Auto-generated method stub
37         return super.authenticationManager();
38     }
39
40     @Override
41     public void configure(HttpSecurity http) throws Exception {
42         http.authorizeRequests()
43             .anyRequest().authenticated()
44             .and()
45             .csrf()
46             .disable()
47             .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);
48
49
50 }
```

# 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,'2021-05-15')
11 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(3,'Maria','Orillana','mo@hotmail.com',65433223,'2021-03-20')
12 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(4,'Dina','Ramirez','dr@hotmail.com',65433223,'2021-06-10')
13 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(5,'Mirna','Ramos','mr@hotmail.com',65433223,'2021-04-05')
14 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(6,'Pepe','Mojica','pm@hotmail.com',65433223,'2021-05-20')
15 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(7,'Juan','Chavez','jc@hotmail.com',65433223,'2021-06-20')
16 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(8,'Enrique','Iglesias','ei@hotmail.com',65433223,'2021-07-01')
17 INSERT INTO clientes (region_id,nombre,apellido,email,telefono,create_at) VALUES(3,'Pedro','Diaz','pd@hotmail.com',65433223,'2021-08-15')
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
- Editor Area:** Displays the code for `SpringbootApirestApplication.java`. The code implements `CommandLineRunner` and uses `BCryptPasswordEncoder`. A red box highlights the implementation of `CommandLineRunner` and the declaration of `passwordEncoder`. Another red box highlights the `run` method where the password is hashed and printed.

```
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    public static void main(String[] args) {
12        SpringApplication.run(SpringbootApirestApplication.class, args);
13    }
14
15    @Override
16    public void run(String... args) throws Exception {
17
18        String password = "12345";
19
20        for(int i = 0;i<4;i++) {
21            String passwordBcrypt = passwordEncoder.encode(password);
22            System.out.println(passwordBcrypt);
23        }
24
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

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.SpringbootApire : \$2a\$10\$Jf1B1DvYy3spSruEe8kf4OXx1jeyPaOgTHPgXiUaUQQ/s/O.PWhbu : \$2a\$10\$8t2e9DE1.ZSajFHzwu/JKexkpmgoIpH6JQsK.rWlseVjAxCCzuf/K : \$2a\$10\$4juaknnD9oRuRrRVJVHSme7d4yVwsHMWLp4fVoTqRkTfuNCgA3w.2 : \$2a\$10\$GyLqNMlTj64457F98..CbutV4NAFrOOXNIscFdU8RTZTtx9nVs8Ki

```
18 INSERT INTO `clientes` (region_id,nombre,apellido,email,telefono,create_at) VALUES(1,'Ramon','Gonzalez','rgonzalez@gmail.com',65433223,'2022-01-01 10:00:00');
19
20 INSERT INTO `usuarios` (username,password(enabled)) VALUES('jose','$2a$10$Jf1B1DvYy3spSruEe8kf4OXXx1jeyPaOgTHPgXiUaUQQ/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 details panel shows the method 'GET' and the URL 'http://localhost:8087/api/clientes'. Below this, under 'Headers (6)', there are six header fields listed. The 'Body' tab is selected, showing a JSON response with the following data:

```
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     }
}
```

The status bar at the bottom indicates 'Status: 200 OK Time: 50 ms Size: 2 KB Save Response'.

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
--	-----	-------	-------------	-----	-----------

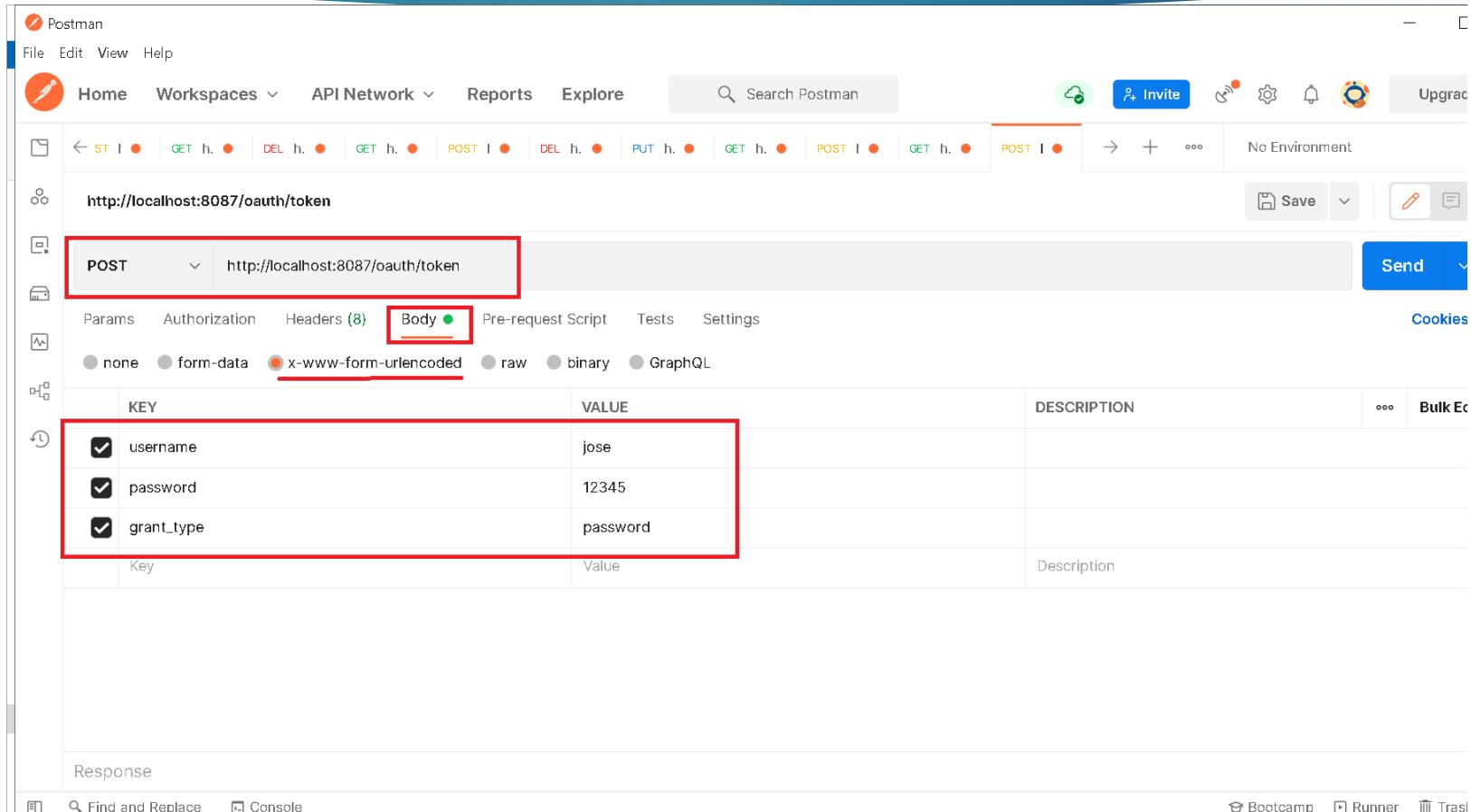
Body Cookies Headers (11) Test Results

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 rows with the following data:

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 h. ● DEL h. ● GET h. ● POST h. ● DEL h. ● PUT h. ● GET h. ● POST h. ● GET h. ● POST h. ● → + ooo No Environment



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

```
1 {  
2   "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJleHAiOiE2MzgyNDU3NDcsInVZZXJfbmFtZSI6Impvc2UiLCJhdXRob3JpdGllcyI6WyJST0xFX1VTRVIiXSwianRpIjoiNzI2NTIzMmMTZGRh0S00YWIZLTLmZjItN2Q5ZTJjYmZkZmFkIiwiY2xpZW5  
ox2lkIjoiYW5ndWxcmFwcCIsInNjb3BlIjpbinJlyWQilCJ3cm10ZSJdfQ.Kybr4bUaafg_EmgIYP3VH2sMUuEtTQEmbE_Ke5oc1rs",  
3   "token_type": "bearer",  
4   "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJ1c2VyX25hbWUiOiJqb3NlIiwick2NvcGUIolsicmVhZCIsIndyaXRlIl0sImF0aSI6IjcyNjUyMzzjLWRkYTktNGFiMy05ZmYyLTdkOWUyY2JmZGzhZCIsImV4cCI6MTYzODI0NTc0NywiYXV0aG9yaXR  
pZXMiolsiUk9MRV9VU0VSIL0sImp0aSI6ImFkZTkzYjF1LWJkMGItNGRmMS05YzQ0LTg1ZjFl0TU3ZDI3MyIsImNsawVudF9pZCI6ImFuZ3VsYXJhcHAifQ.  
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.