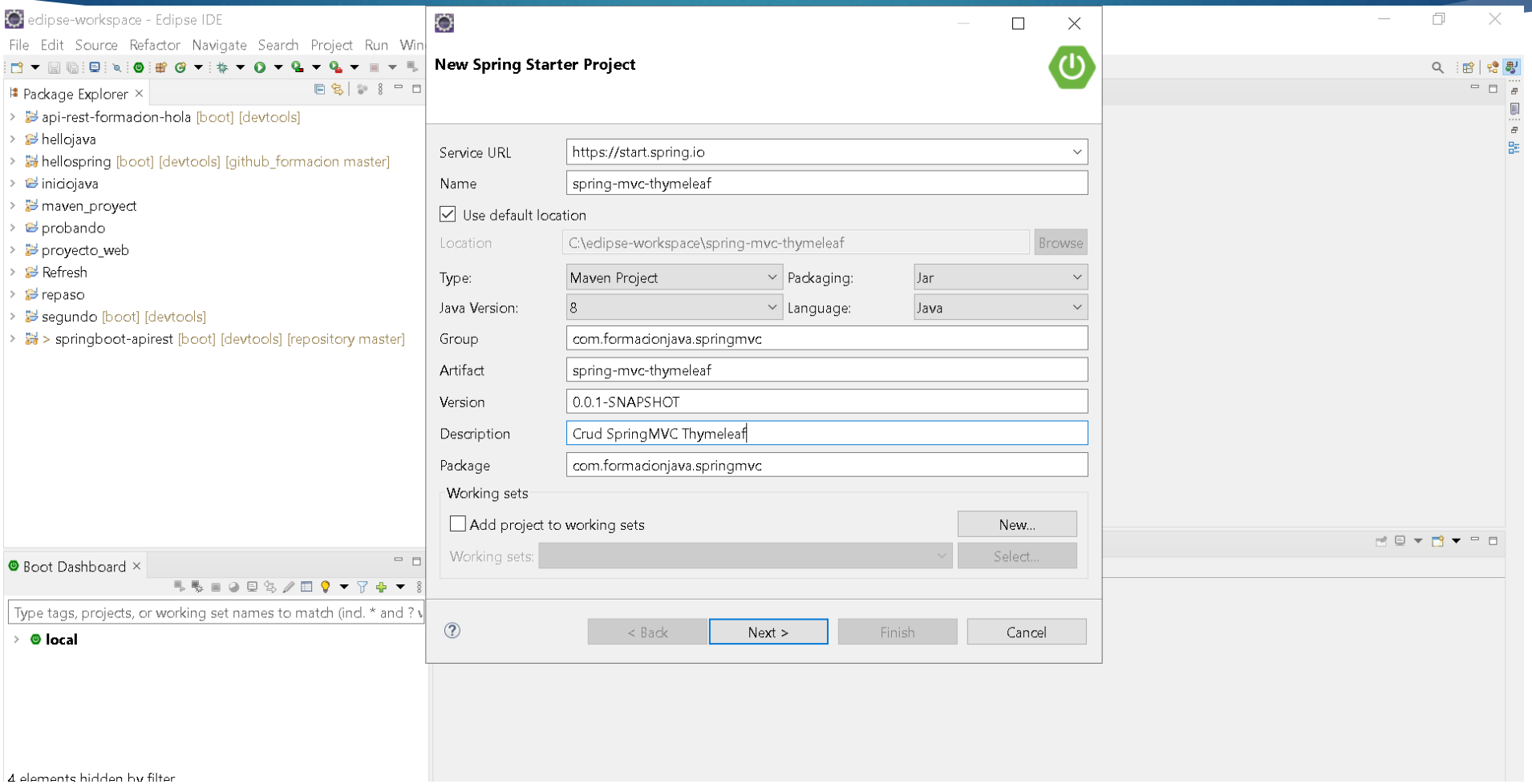
# Thymeleaf

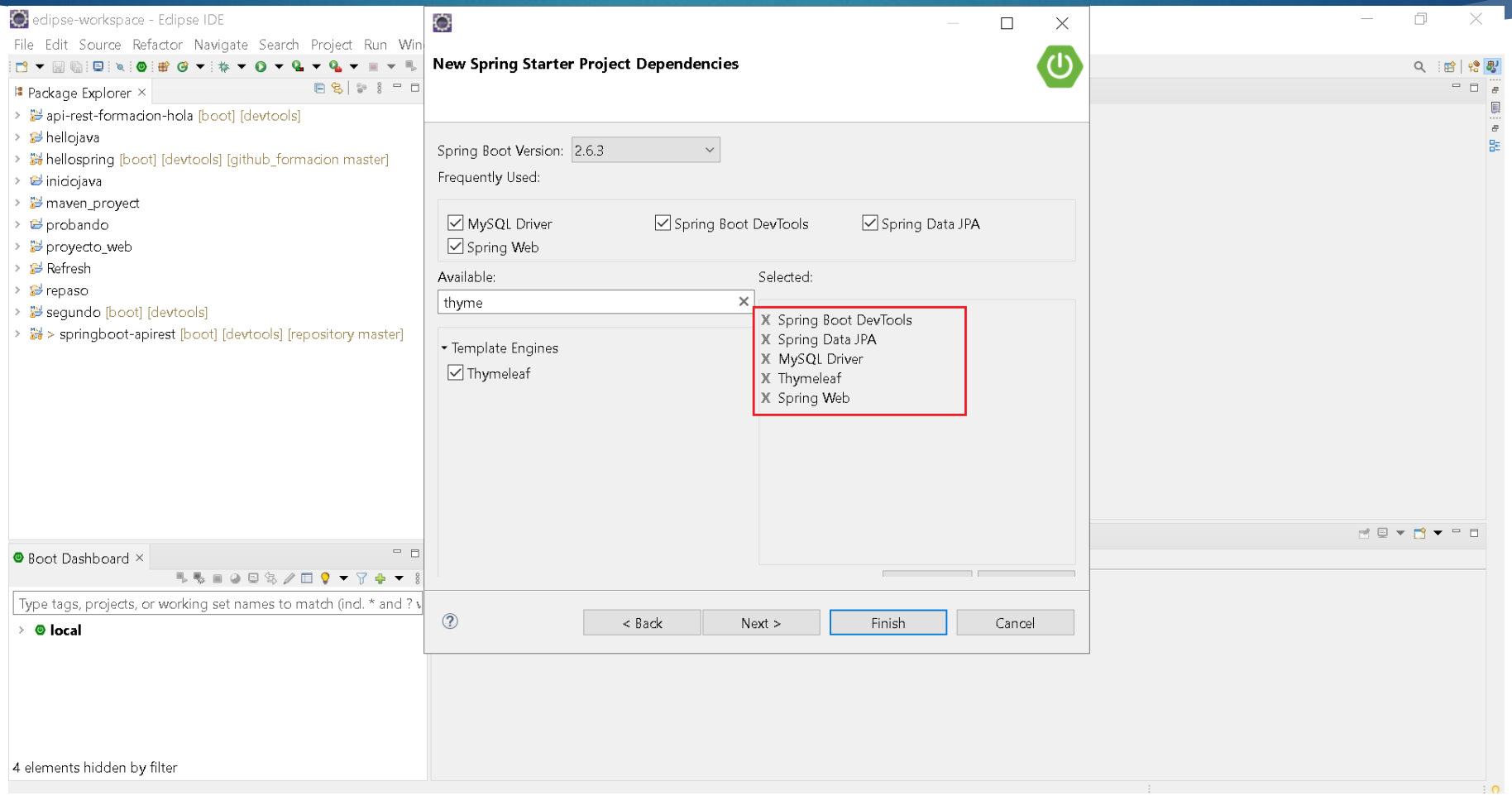
- **Thymeleaf** es un motor de plantillas de Java para procesar y crear HTML, XML, JavaScript, CSS y texto.

- https://www.thymeleaf.org/

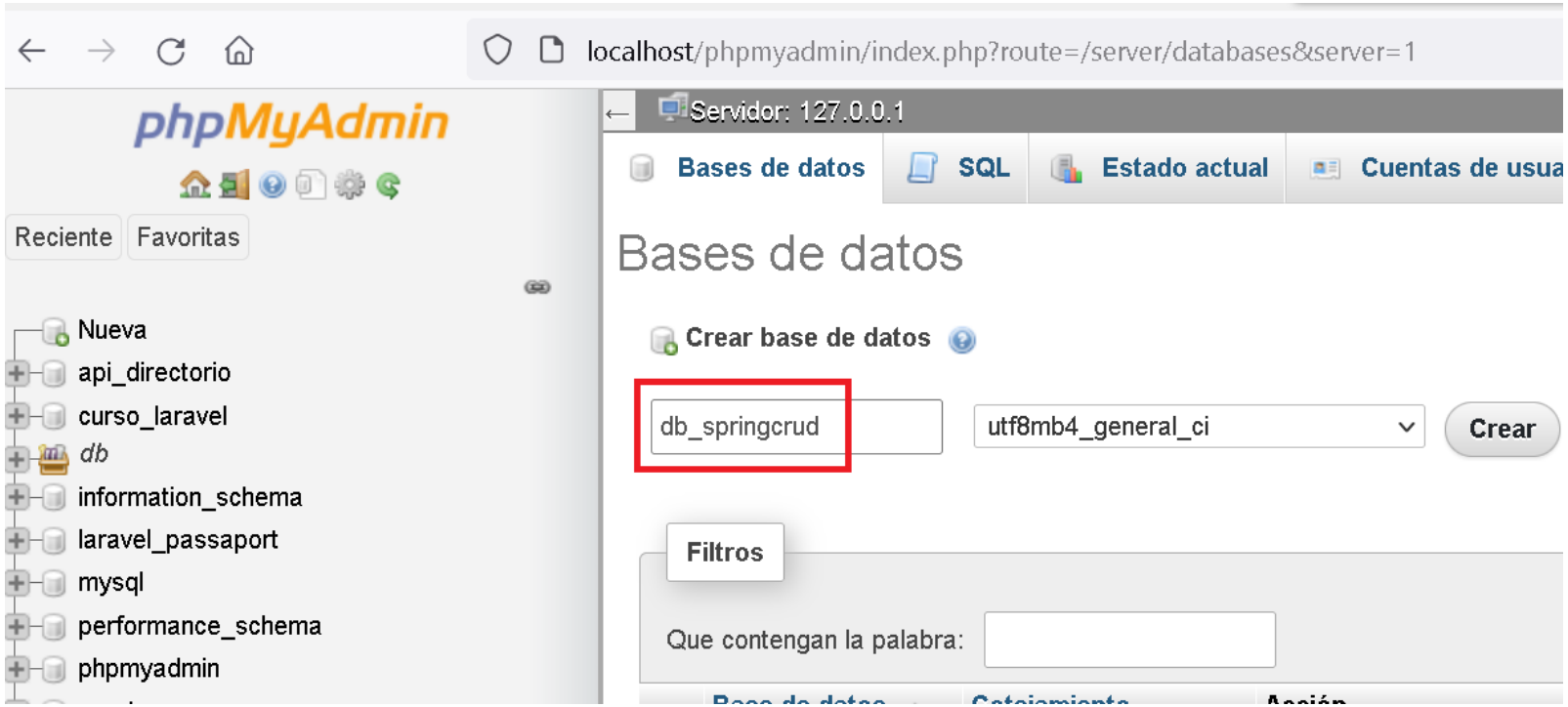
# Creamos un nuevo proyecto



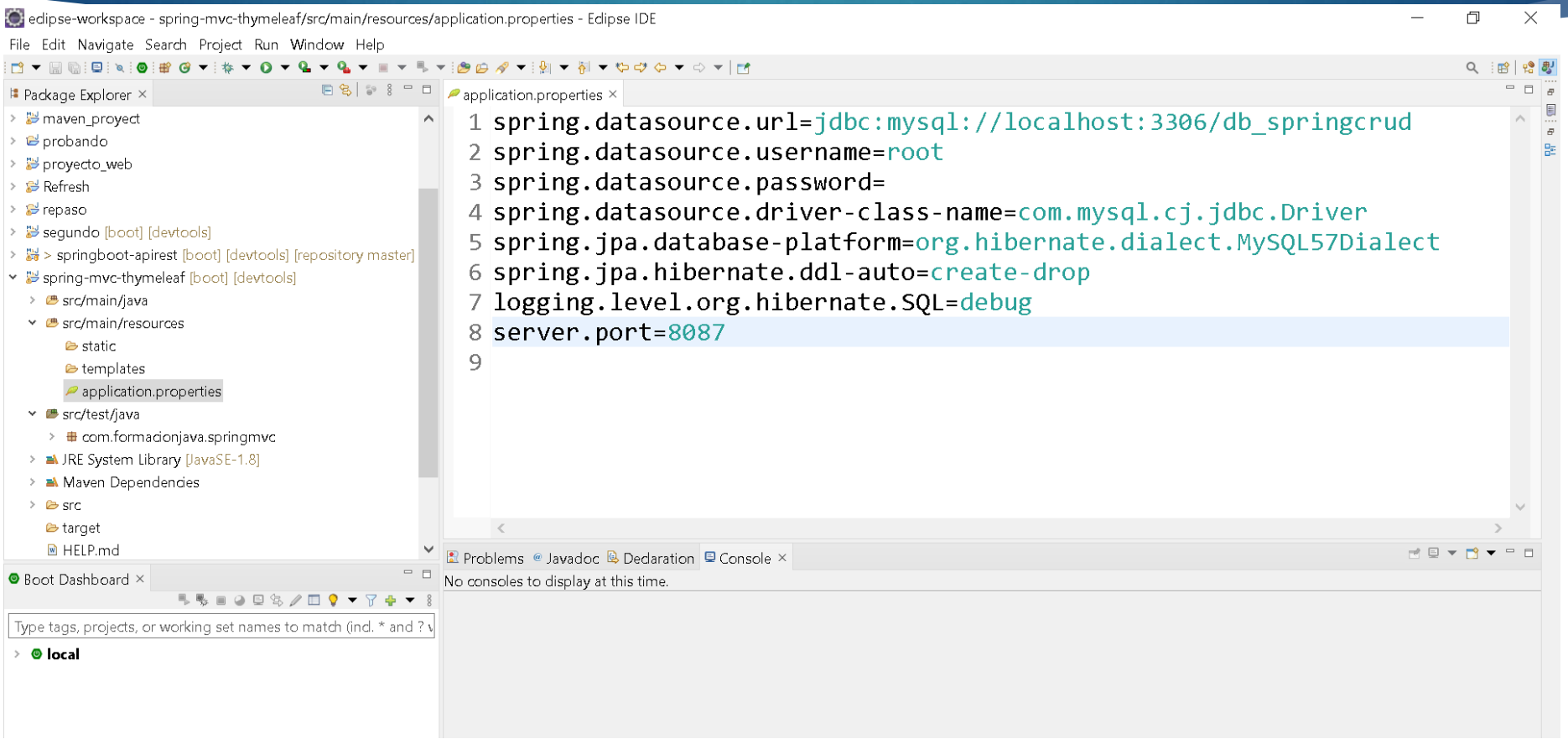
# Agregamos las siguientes dependencias



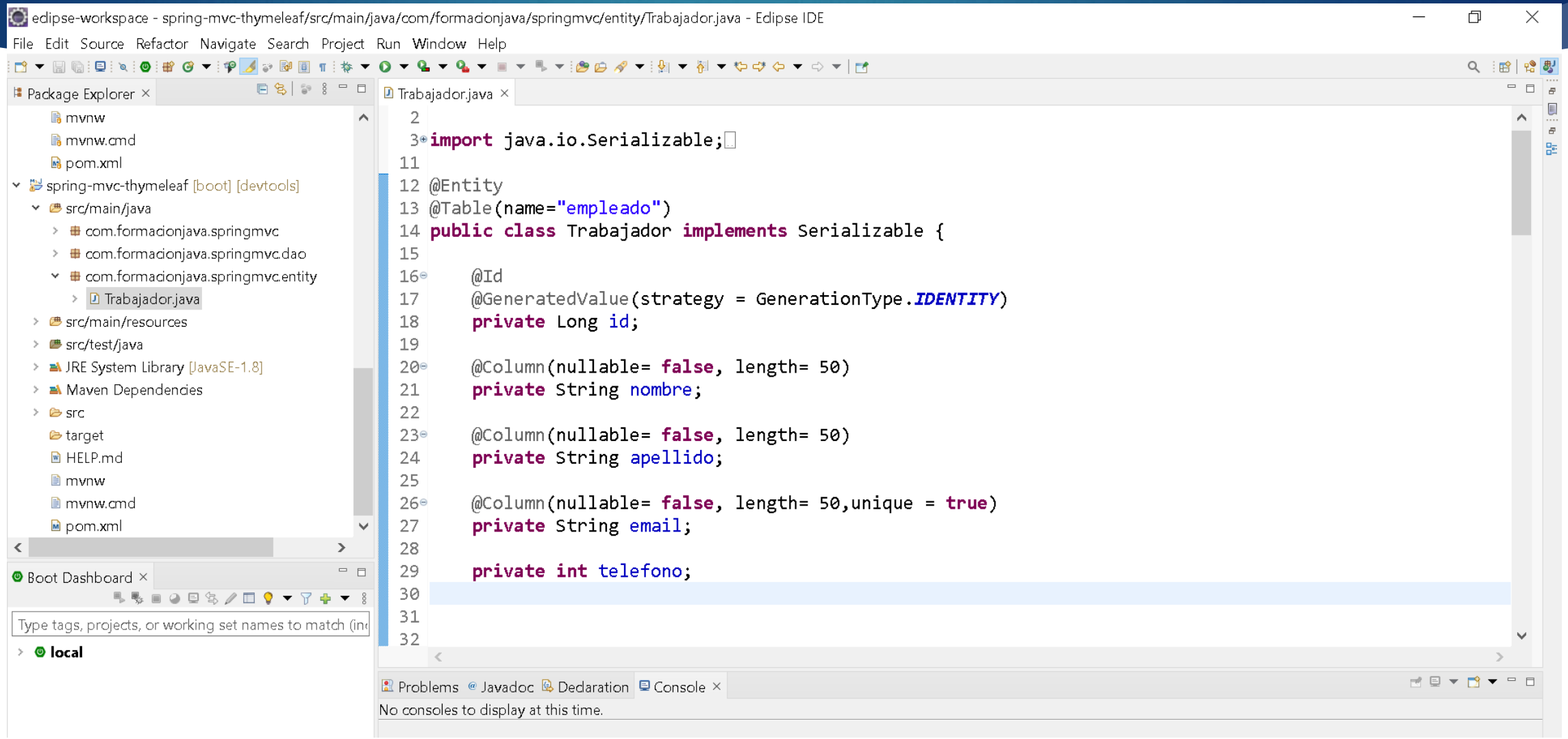
# Creamos la base de datos

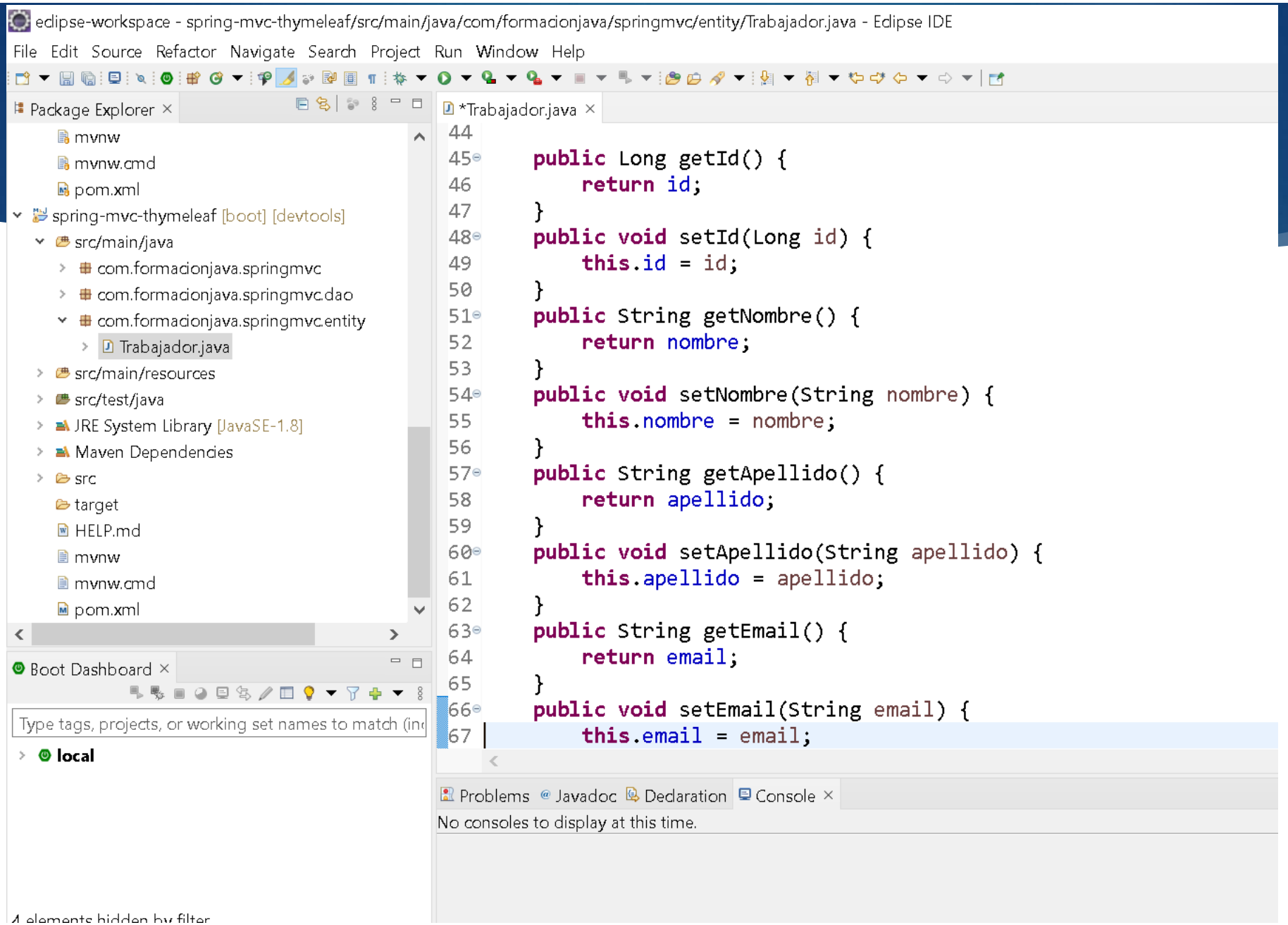


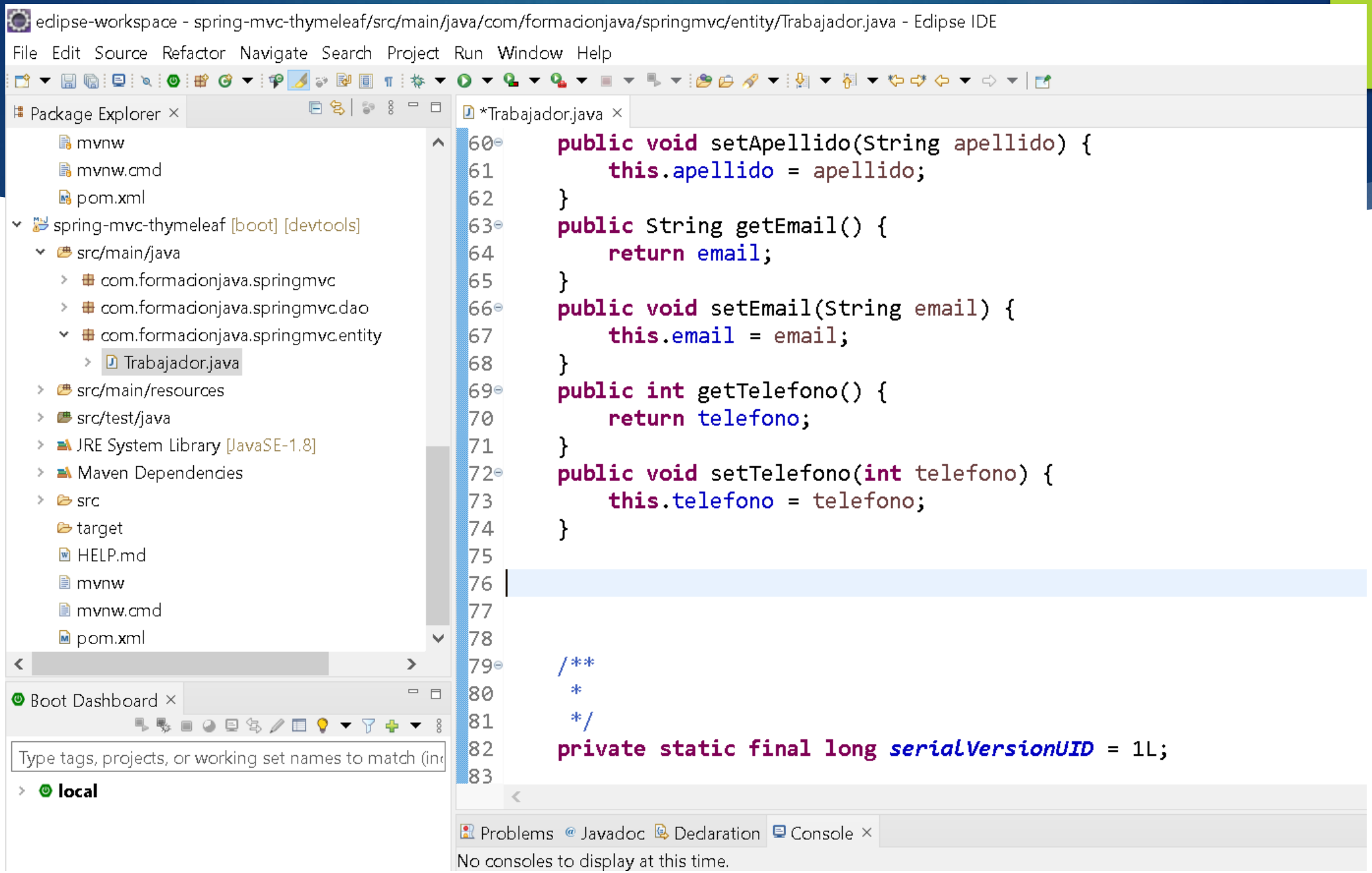
# Configuramos el application.prioperties



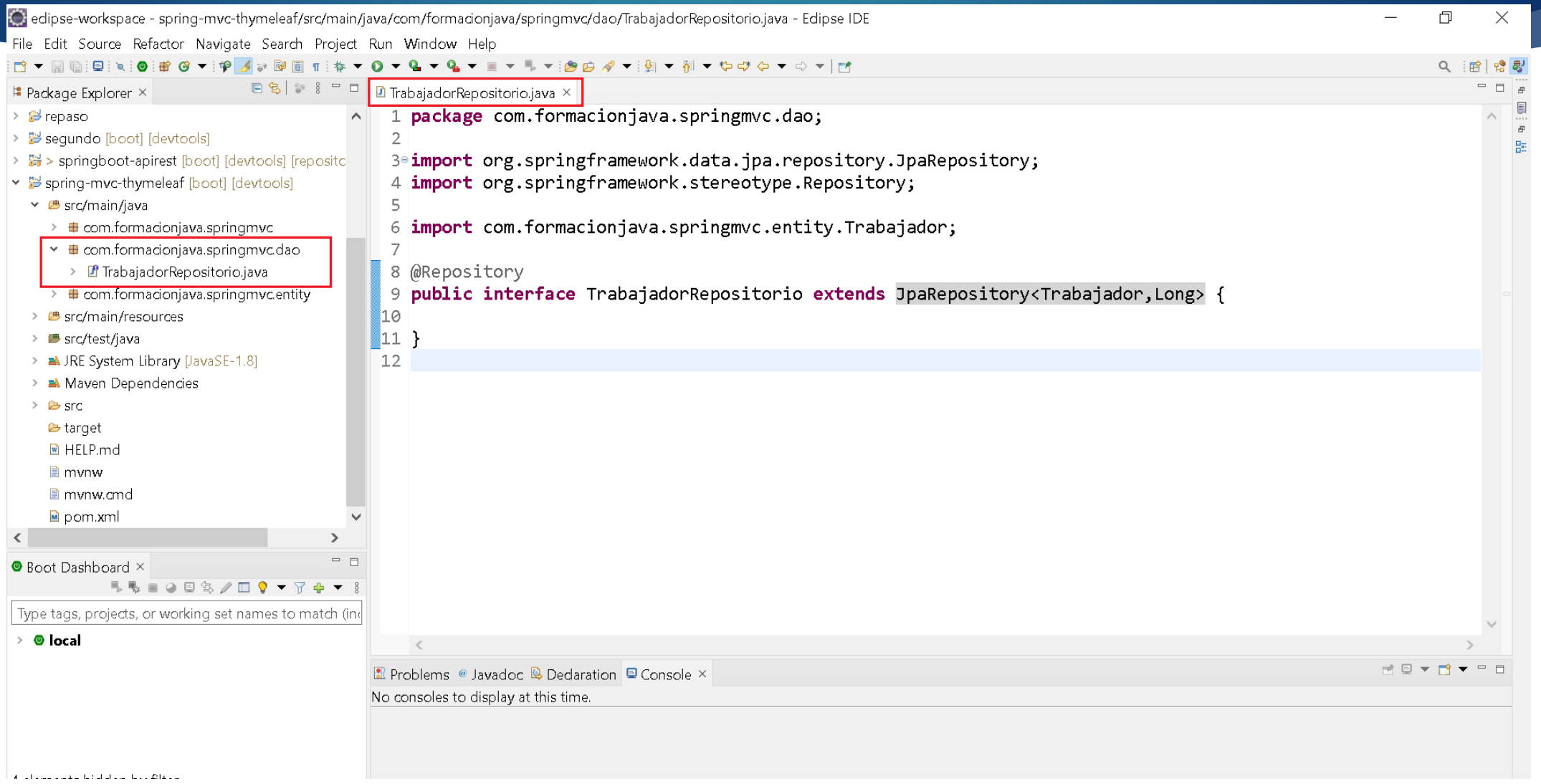
# Creamos nuestra entidad



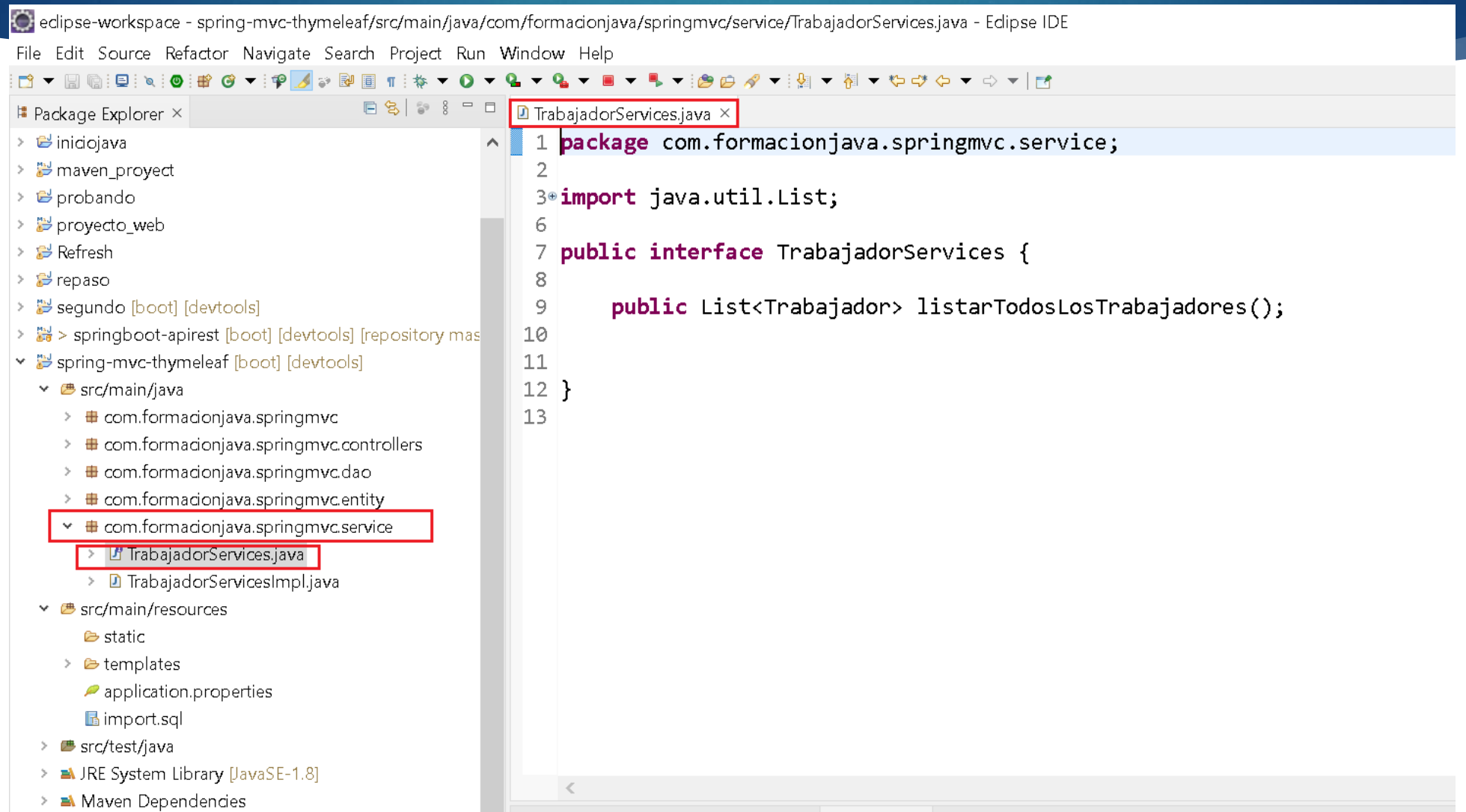

eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/entity/Trabajador.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
mvnw
mvnw.cmd
pom.xml
spring-mvc-thymeleaf [boot] [devtools]
src/main/java
com.formacionjava.springmvc
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
Boot Dashboard
Type tags, projects, or working set names to match (in
local
4 elements hidden by filter
*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;
Problems Javadoc Declaration Console
No consoles to display at this time.

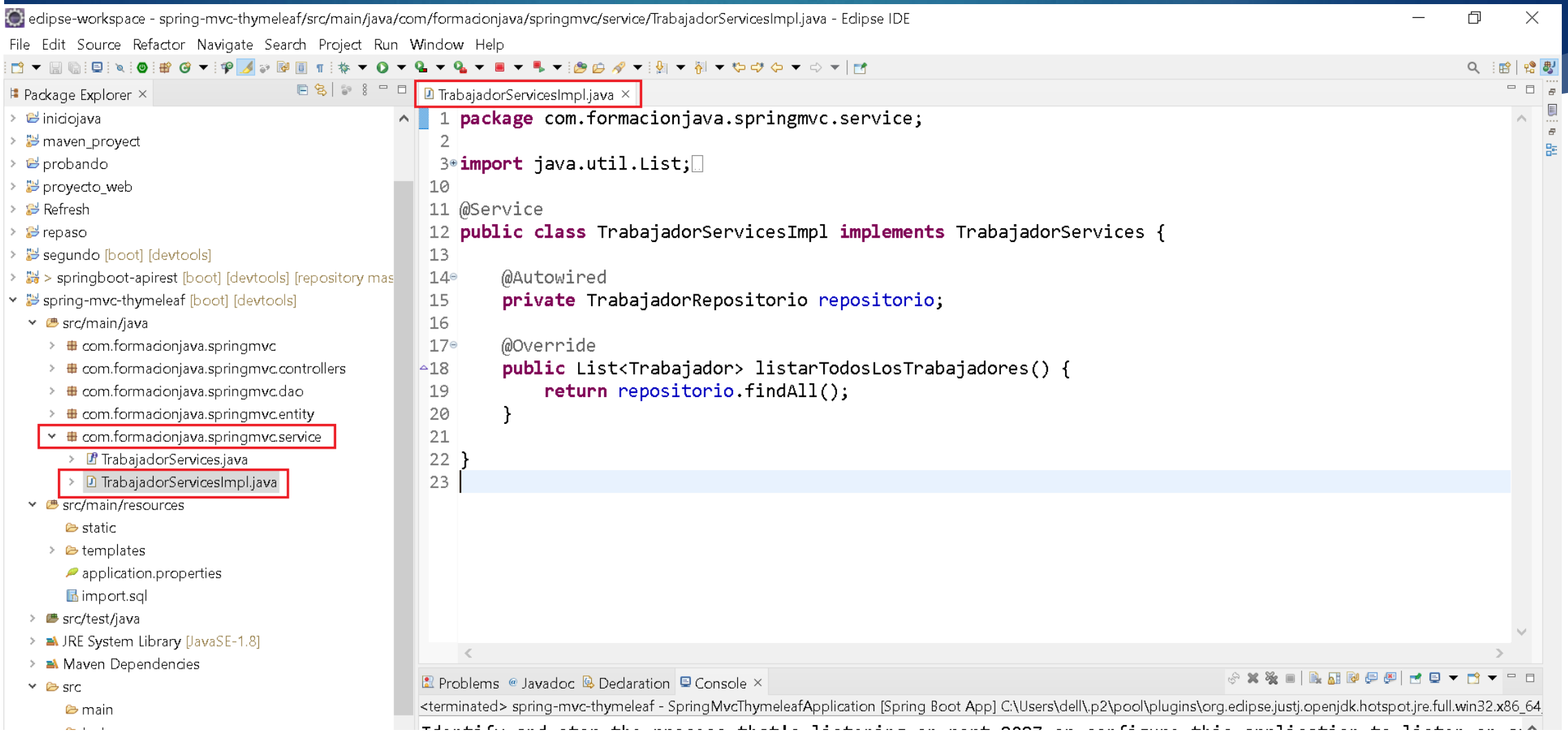

eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/entity/Trabajador.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
mvnw
mvnw.cmd
pom.xml
spring-mvc-thymeleaf [boot] [devtools]
src/main/java
com.formacionjava.springmvc
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
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
77
78
79 /**
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

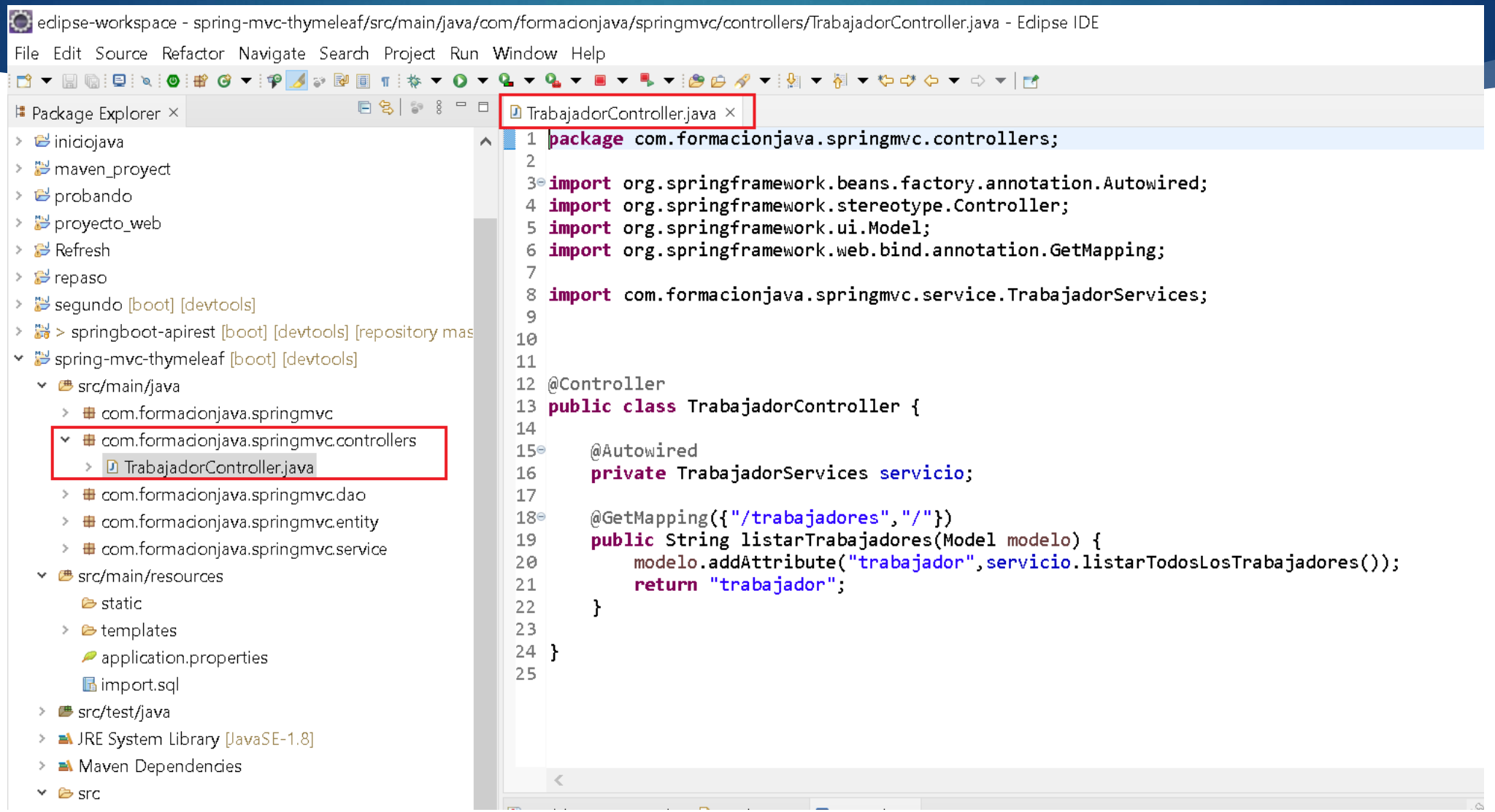


# Ahora creamos los servicios




edipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/service/TrabajadorServicesImpl.java - Edipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
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
com.formacionjava.springmvc.dao
com.formacionjava.springmvc.entity
com.formacionjava.springmvc.service
TrabajadorServices.java
TrabajadorServicesImpl.java
src/main/resources
static
templates
application.properties
import.sql
src/test/java
JRE System Library [JavaSE-1.8]
Maven Dependencies
src
main
TrabajadorServicesImpl.java
1 package com.formacionjava.springmvc.service;
2
3 import java.util.List;
10
11 @Service
12 public class TrabajadorServicesImpl implements TrabajadorServices {
13
14 @Autowired
15 private TrabajadorRepositorio repositorio;
16
17 @Override
18 public List<Trabajador> listarTodosLosTrabajadores() {
19 return repositorio.findAll();
20 }
21
22 }
23
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

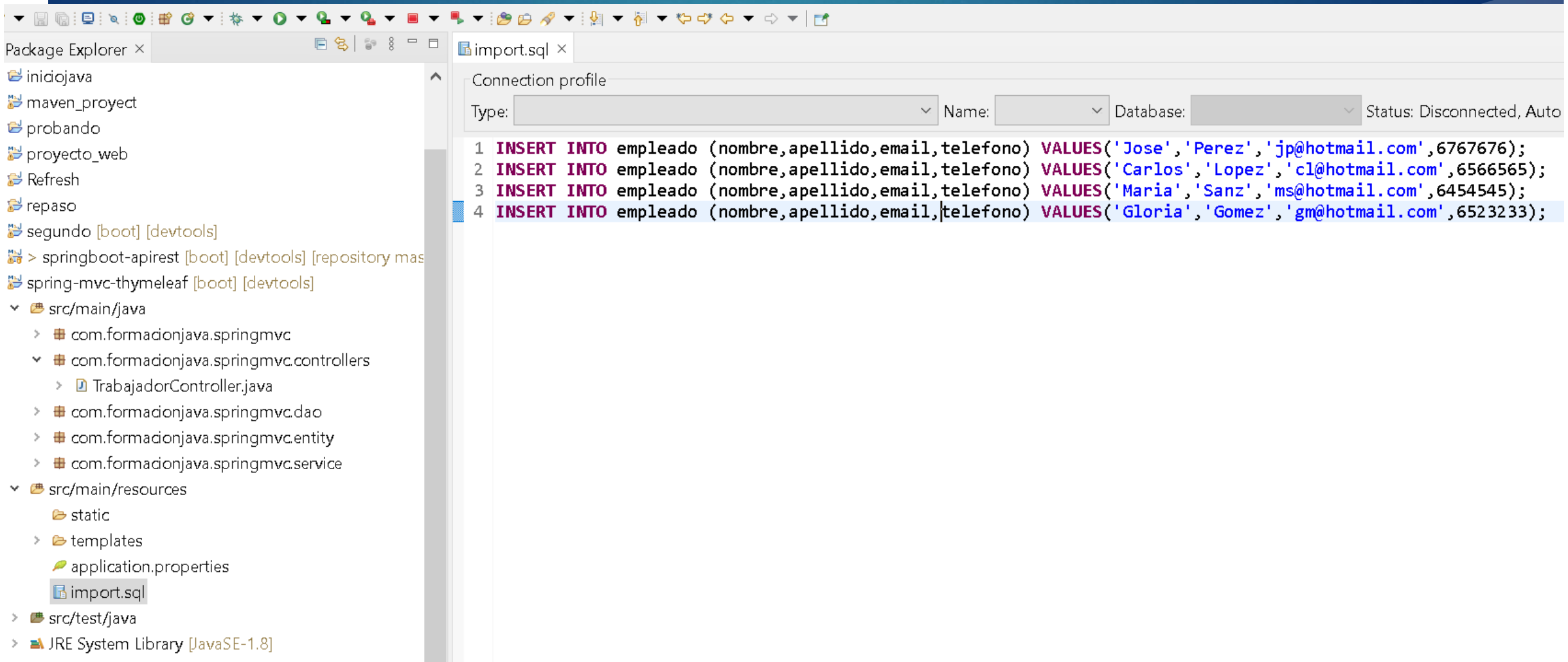
# Creamos el controlador



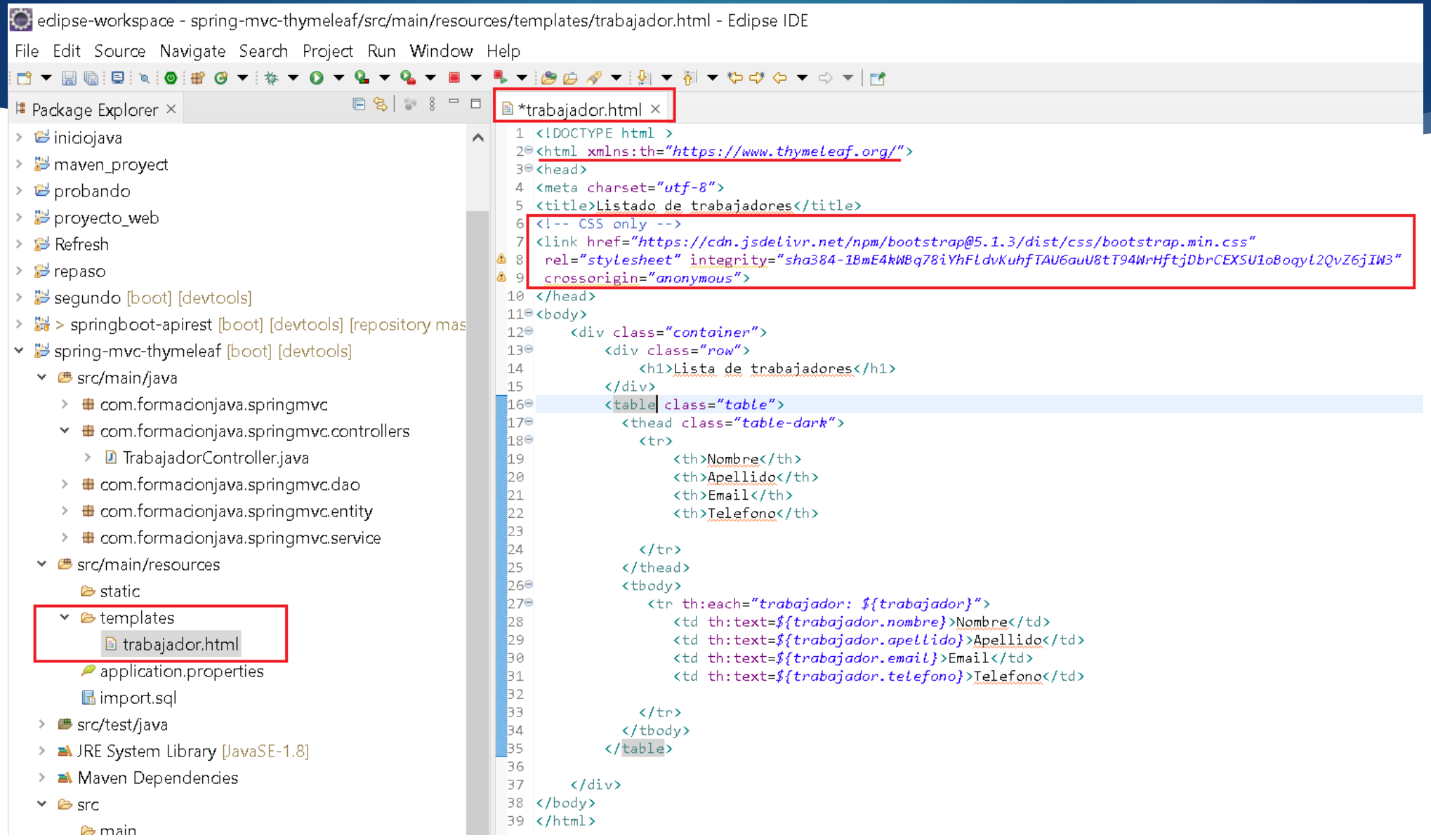
# Modelo

- El modelo puede proporcionar atributos utilizados para representar vistas

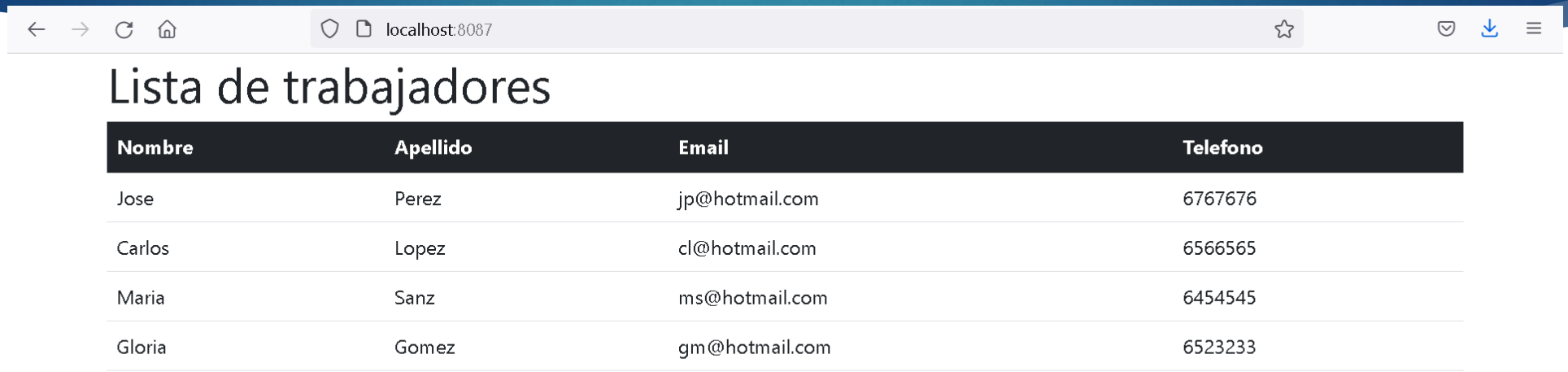
# Creamos los datos de pruebas



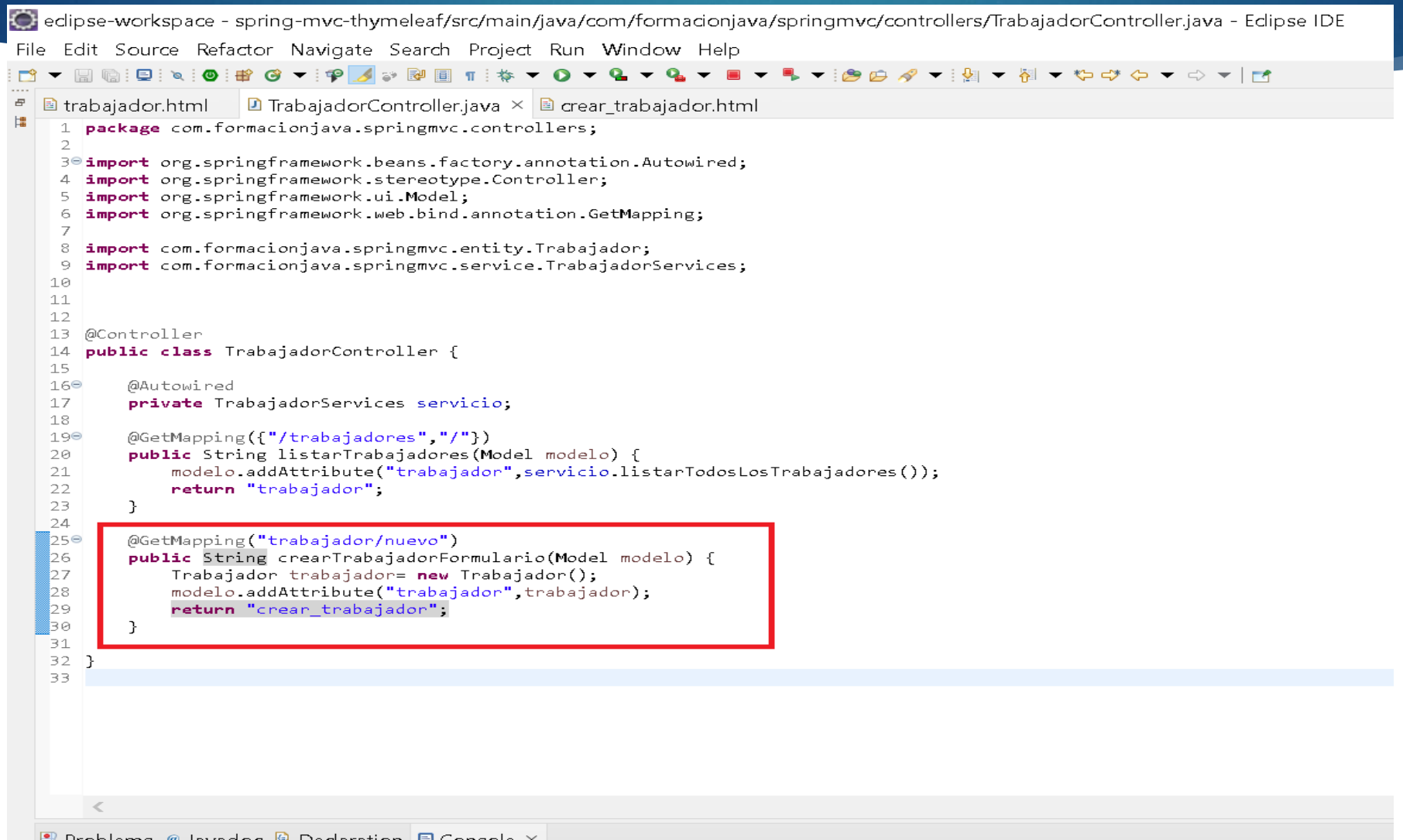
# Ahora agregamos la plantilla



# Ejecutamos y probamos todo hasta aquí



# Ahora me creo un nuevo controlador



# Agrego un navbar a mi html

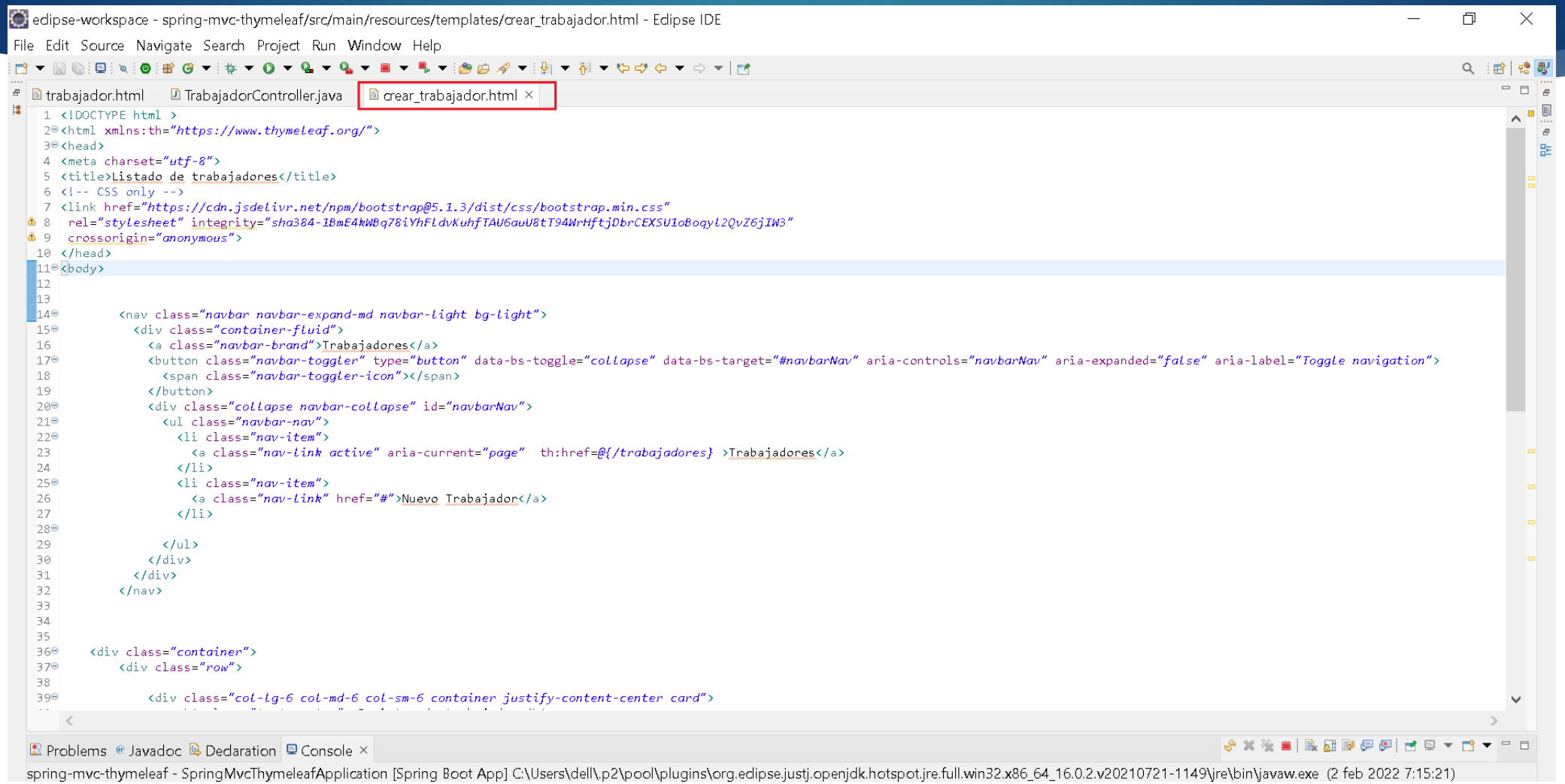
edipse-workspace - spring-mvc-thymeleaf/src/main/resources/templates/trabajador.html - Edipse IDE

File Edit Source Navigate Search Project Run Window Help

trabajador.html × TrabajadorController.java crear_trabajador.html

```
<!DOCTYPE html >
<html xmlns:th="https://www.thymeleaf.org/">
<head>
<meta charset="utf-8">
<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>

    <nav class="navbar navbar-expand-md navbar-light bg-light">
      <div class="container-fluid">
        <a class="navbar-brand">Trabajadores</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
          <ul class="navbar-nav">
            <li class="nav-item">
              <a class="nav-link active" aria-current="page"  th:href=@{/trabajadores} >Trabajadores</a>
            </li>
            <li class="nav-item">
              <a class="nav-link"th:href=@{/trabajador/nuevo}>Nuevo Trabajador</a>
            </li>

          </ul>
        </div>
      </div>
    </nav>


  <div class="container">
      <div class="row">
          <h1>Lista de trabajadores</h1>
      </div>
      <table class="table table-striped table-bordered" >
```

# Creo un nuevo html crear_trabajador



edipse-workspace - spring-mvc-thymeleaf/src/main/resources/templates/crear_trabajador.html - Edipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

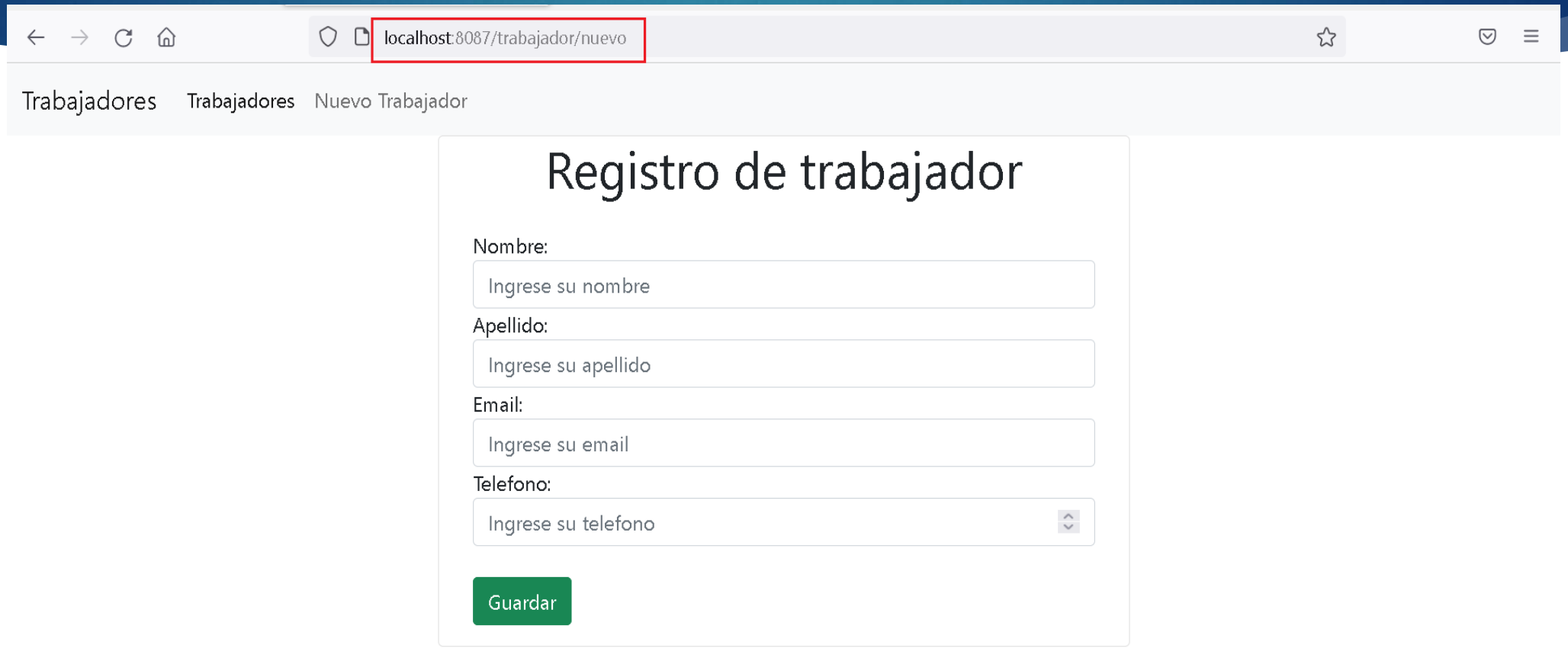
Package Explorer

- iniciojava
- maven_proyect
- probando
- proyecto_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.controll
      - TrabajadorController.java
    - com.formacionjava.springmvc.dao
    - com.formacionjava.springmvc.entity
      - Trabajador.java
    - com.formacionjava.springmvc.service
      - TrabajadorServices.java
      - TrabajadorServicesImpl.java
  - src/main/resources
    - static
    - templates
      - crear_trabajador.html
      - trabajador.html
    - application.properties
    - import.sql
  - src/test/java

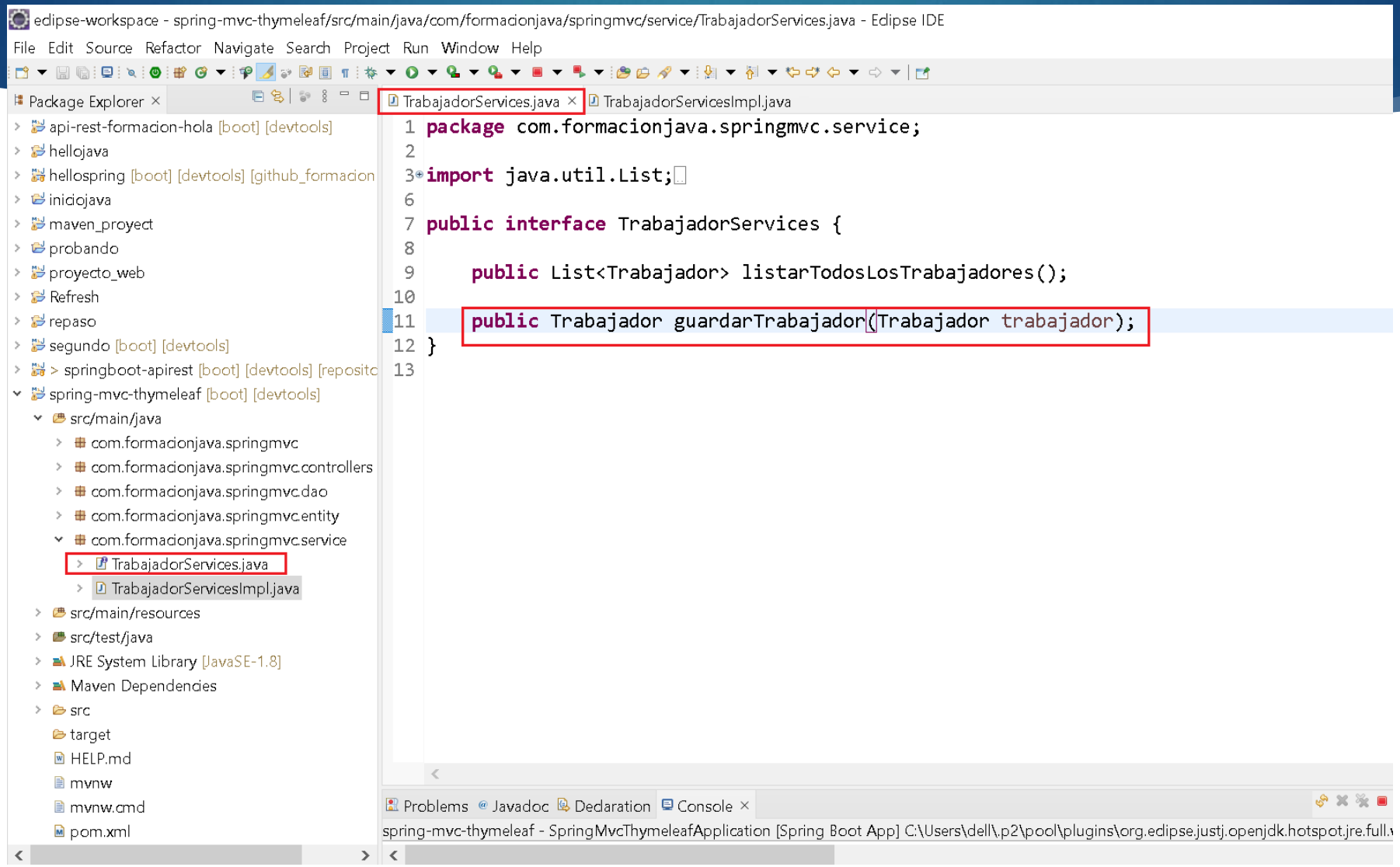
crear_trabajador.html

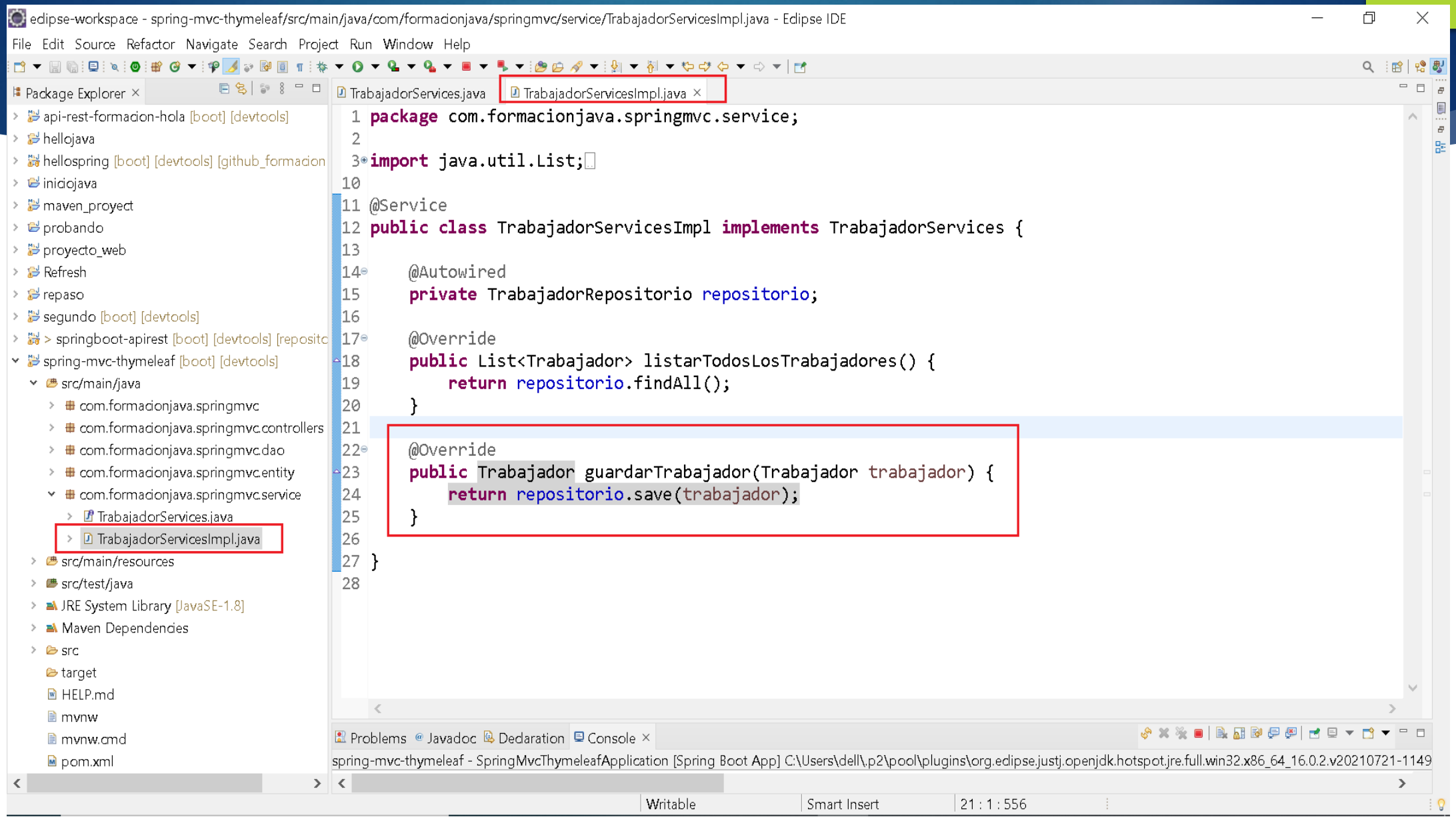
```
36  <div class="container">
37      <div class="row">
38
39          <div class="col-lg-6 col-md-6 col-sm-6 container justify-content-center card">
40              <h1 class="text-center" >Registro de trabajador</h1>
41              <div class="card-body">
42                  <form th:action="@{/trabajador}" 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="nombre" 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="nombre" th:field="*{email}" class="form-control" placeholder="Ingrese su email" required>
54
55                      </div>
56                      <div class="form-group">
57                          <label>Telefono: </label>
58                          <input type="number" name="nombre" th:field="*{telefono}" class="form-control" placeholder="Ingrese su telefono" required>
59                      </div>
60
61                      <br>
62                      <div class="box-footer">
63                          <button class="btn btn-success">Guardar</button>
64                      </div>
65
66                  </form>
67
68              </div>
69
70          </div>
71      </div>
72
73  </div>
74 body>
```

# Probamos la url y la vista

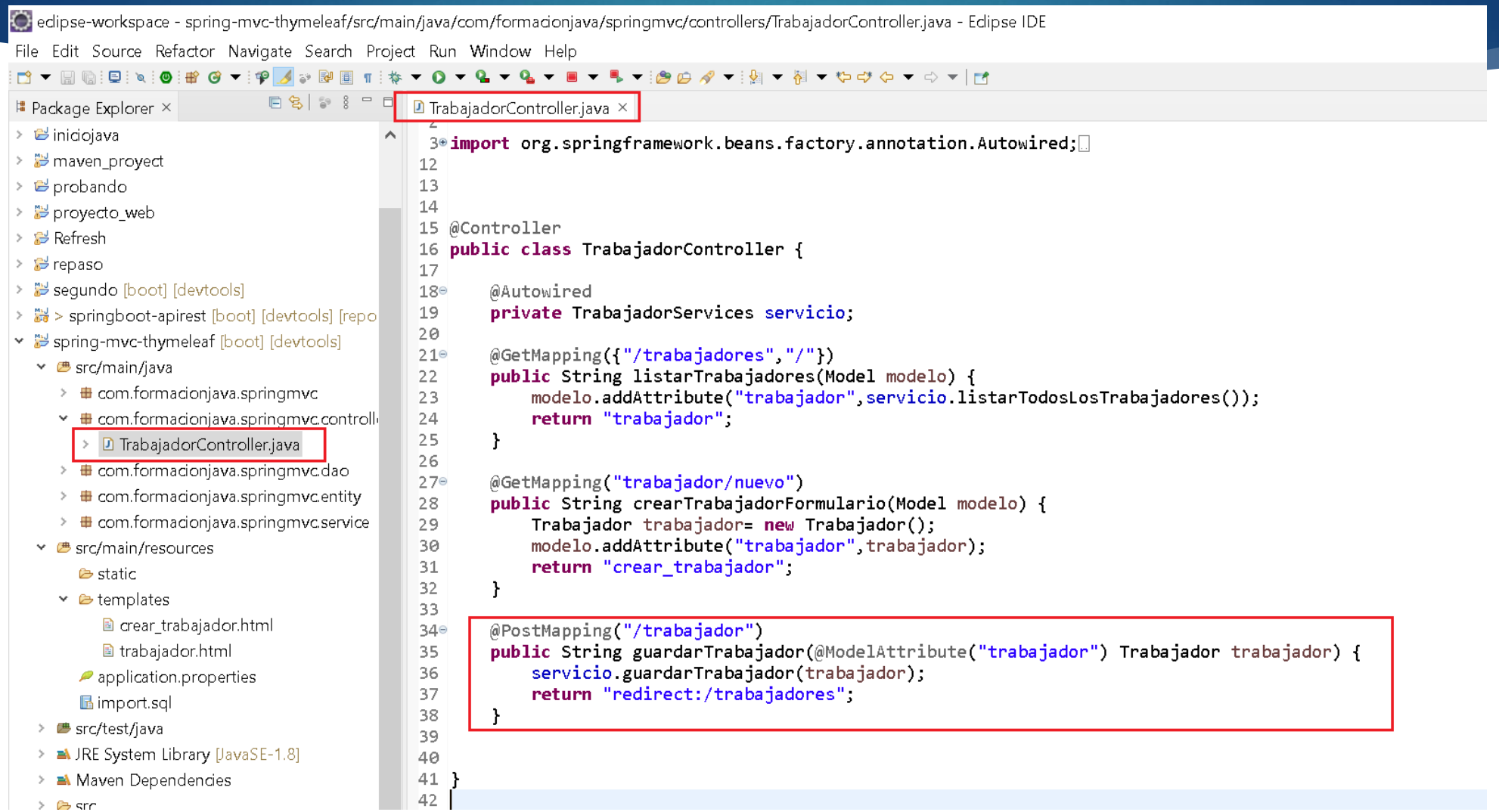


# Agregamos al servicio el método para realizar registro




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
api-rest-formacion-hola [boot] [devtools]
hellojava
hellospring [boot] [devtools] [github_formacion
iniciojava
maven_proyect
probando
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
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
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
TrabajadorServices.java
TrabajadorServicesImpl.java
1 package com.formacionjava.springmvc.service;
2
3 import java.util.List;
10
11 @Service
12 public class TrabajadorServicesImpl implements TrabajadorServices {
13
14 @Autowired
15 private TrabajadorRepositorio repositorio;
16
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 }
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

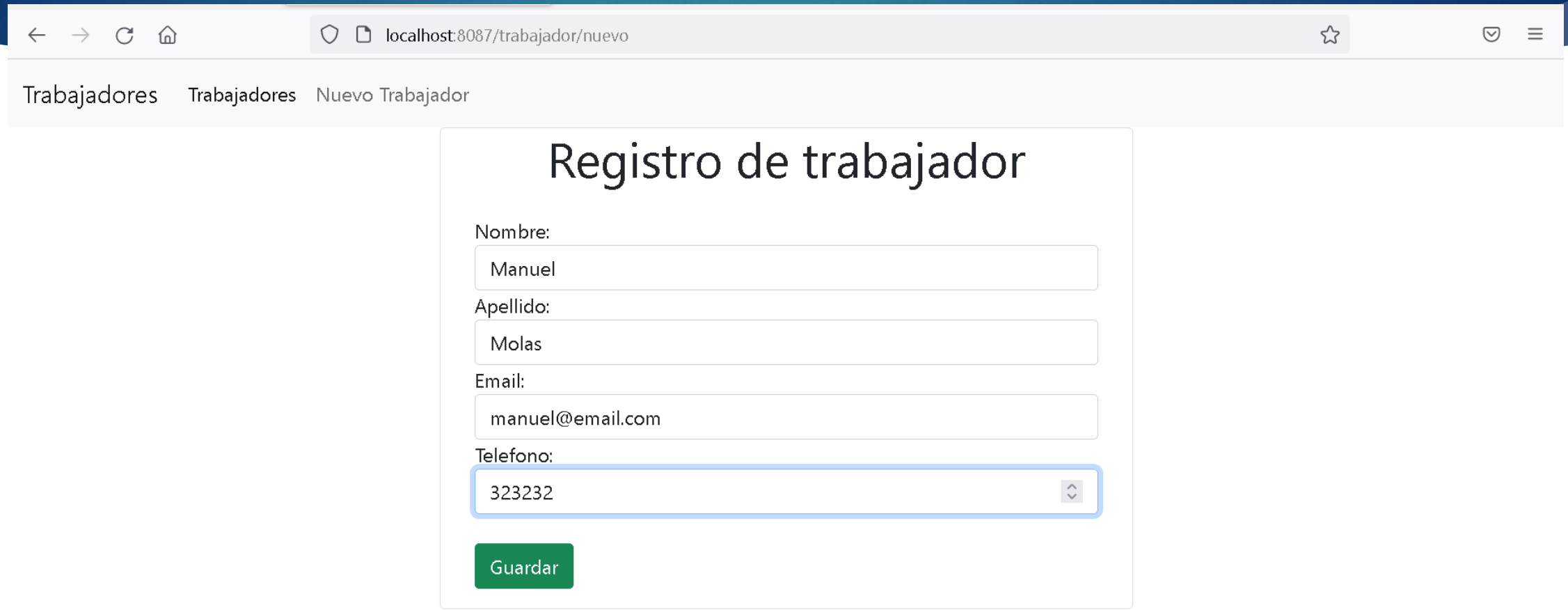
# Agregamos el método de registro al controlador

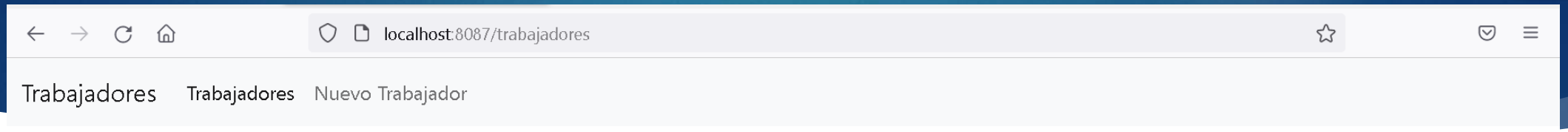


# @ModelAttribute

- 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

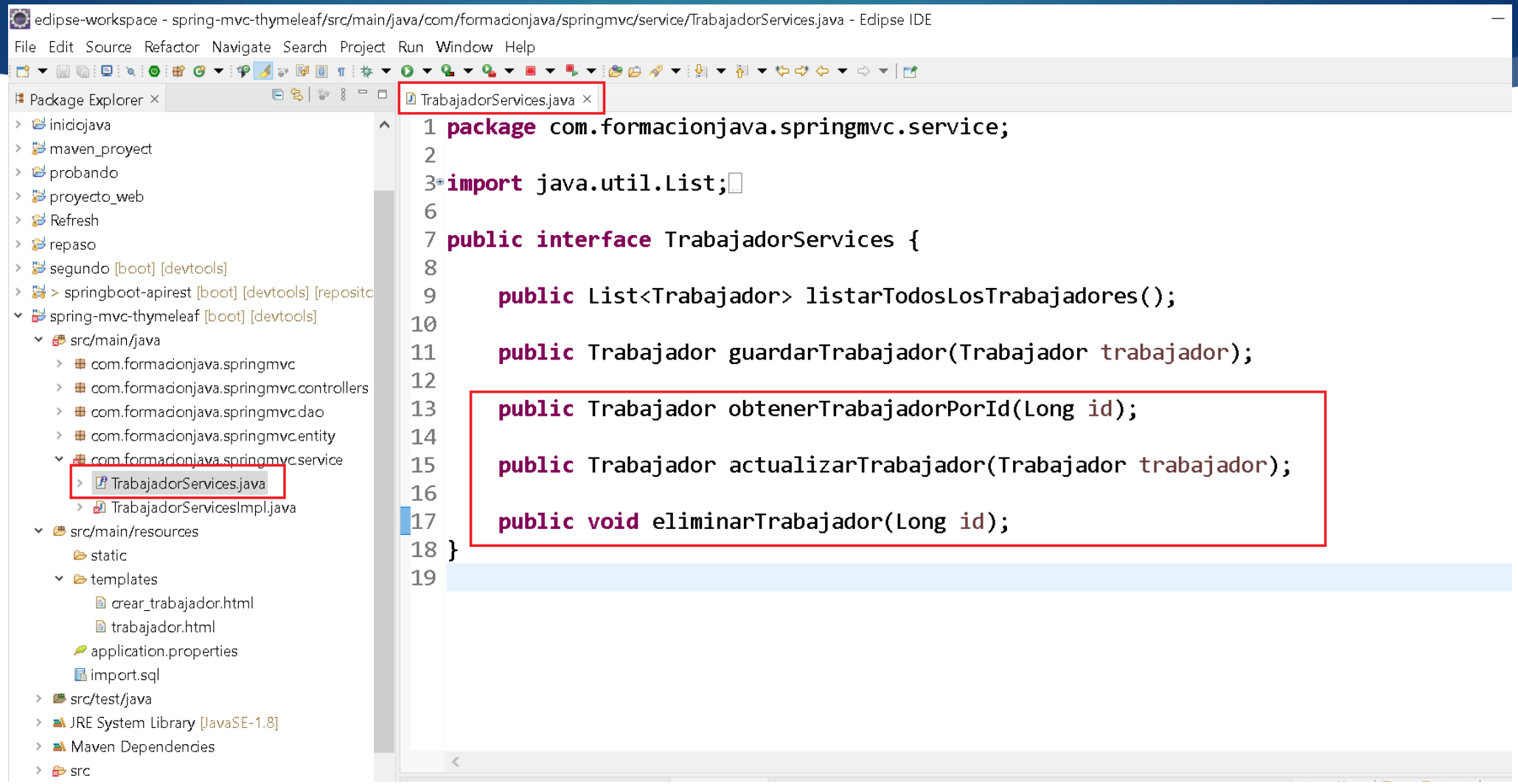


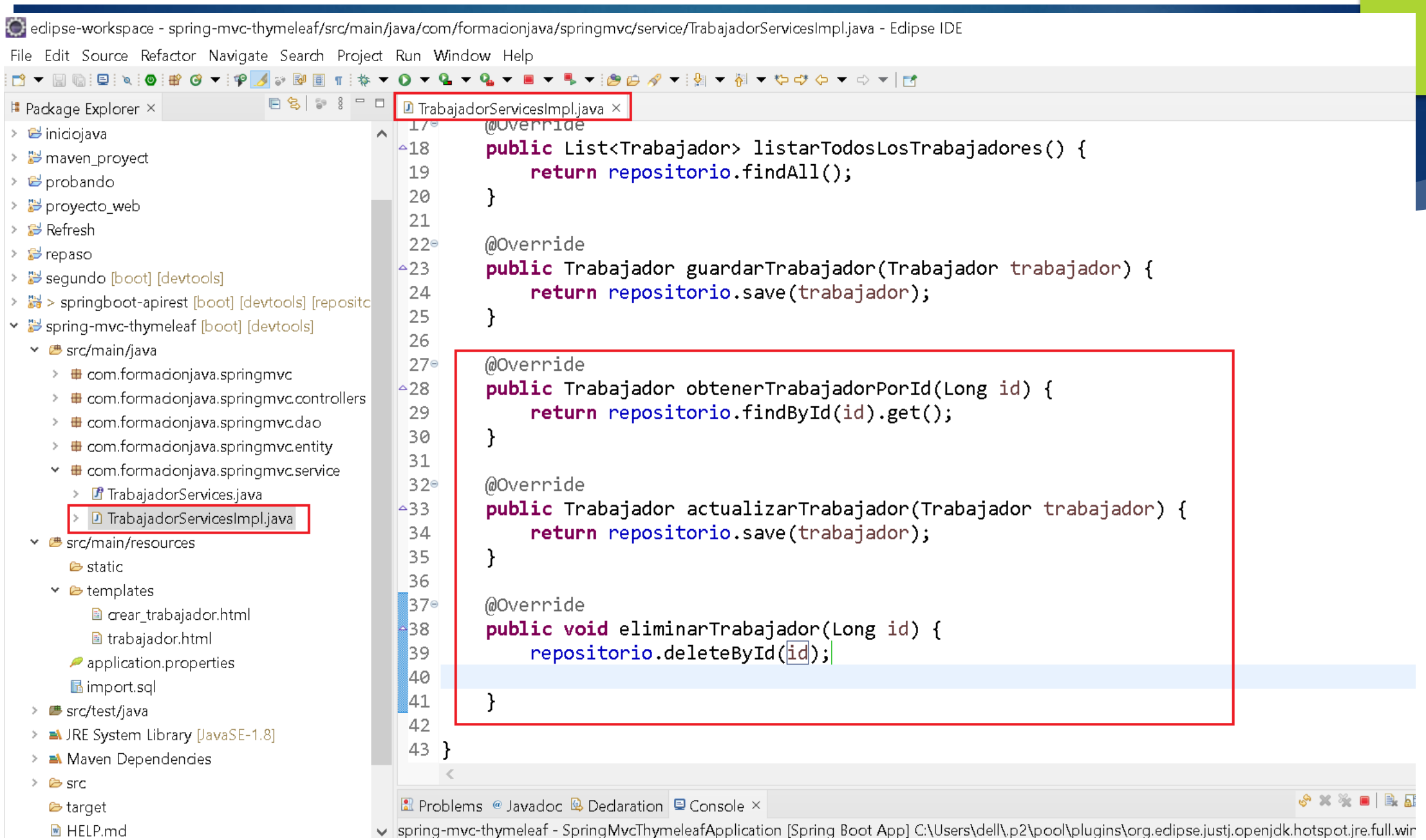



# Lista de trabajadores

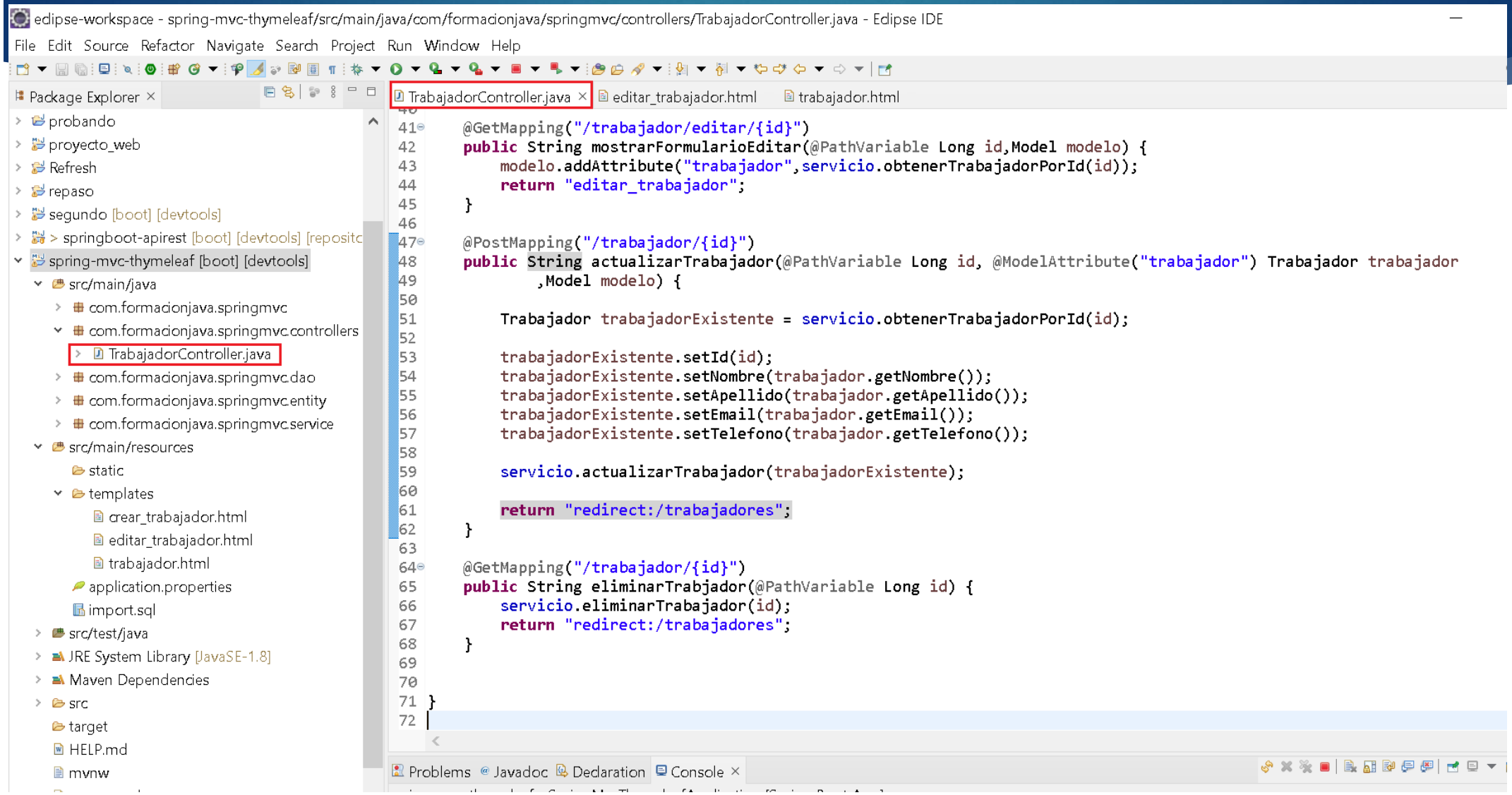
| Nombre | Apellido | Email | Telefono |
|---|---|---|---|
| Jose | Perez | jp@hotmail.com | 6767676 |
| Carlos | Lopez | cl@hotmail.com | 6566565 |
| Maria | Sanz | ms@hotmail.com | 6454545 |
| Gloria | Gomez | gm@hotmail.com | 6523233 |
| Leonel | Messi | leo@messi.com | 322323 |
| Manuel | Molas |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

# Agrego a mi servicio los métodos faltantes para el crud completo

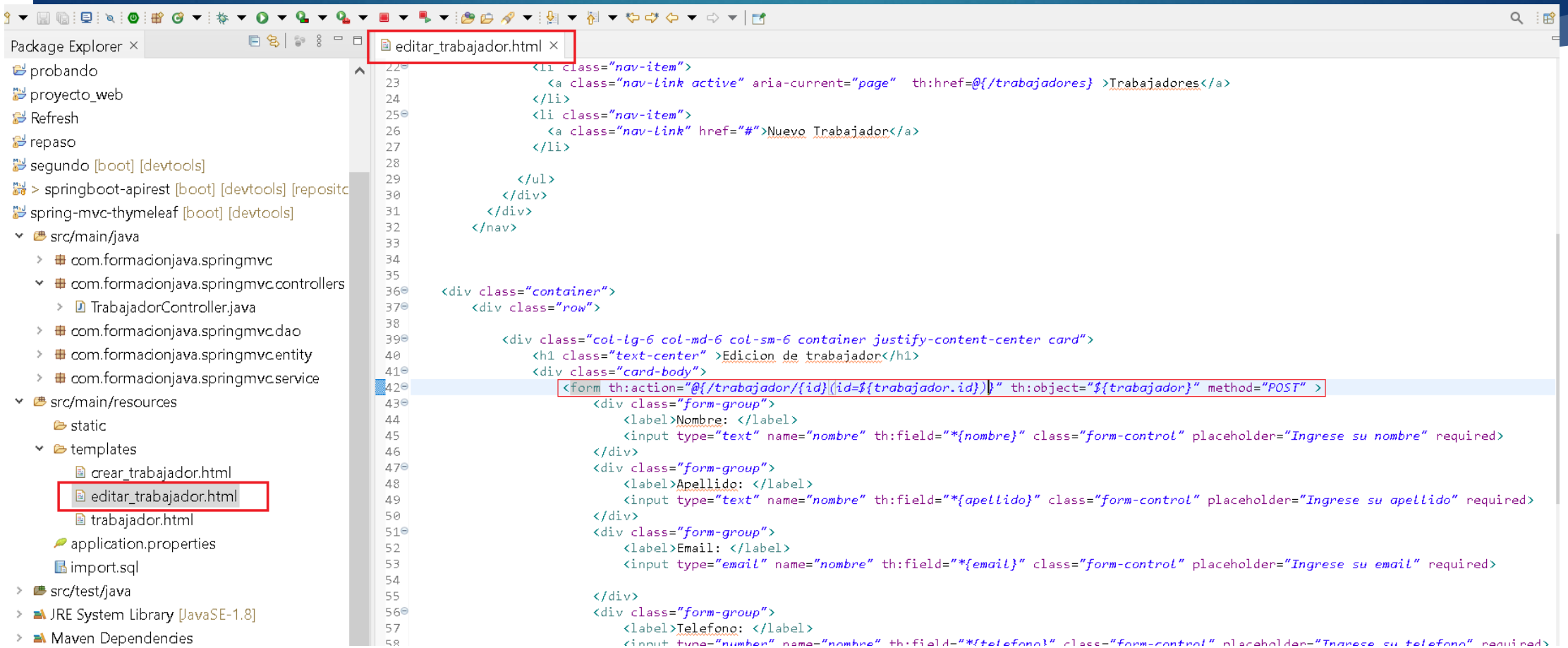



eclipse-workspace - spring-mvc-thymeleaf/src/main/java/com/formacionjava/springmvc/service/TrabajadorServicesImpl.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
iniciojava
maven_proyect
probando
proyecto_web
Refresh
repaso
segundo [boot] [devtools]
springboot-apirest [boot] [devtools]
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
TrabajadorServicesImpl.java
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
target
HELP.md
TrabajadorServicesImpl.java
@Override
public List<Trabajador> listarTodosLosTrabajadores() {
return repositorio.findAll();
}
@Override
public Trabajador guardarTrabajador(Trabajador trabajador) {
return repositorio.save(trabajador);
}
@Override
public Trabajador obtenerTrabajadorPorId(Long id) {
return repositorio.findById(id).get();
}
@Override
public Trabajador actualizarTrabajador(Trabajador trabajador) {
return repositorio.save(trabajador);
}
@Override
public void eliminarTrabajador(Long id) {
repositorio.deleteById(id);
}
}
Problems Javadoc Declaration Console
spring-mvc-thymeleaf - SpringMvcThymeleafApplication [Spring Boot App] C:\Users\dell\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.wir

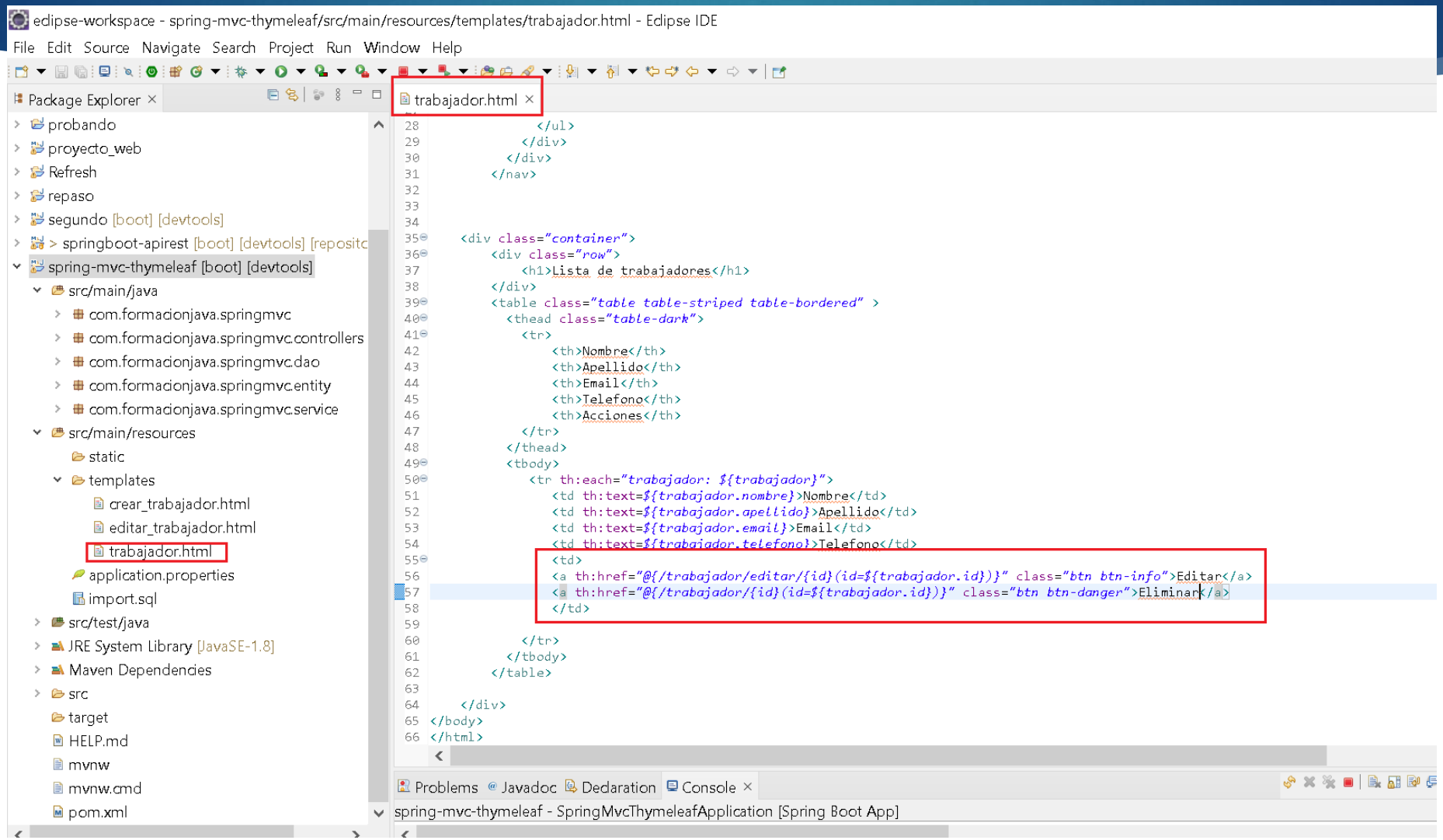
# Agregamos los métodos al controlador



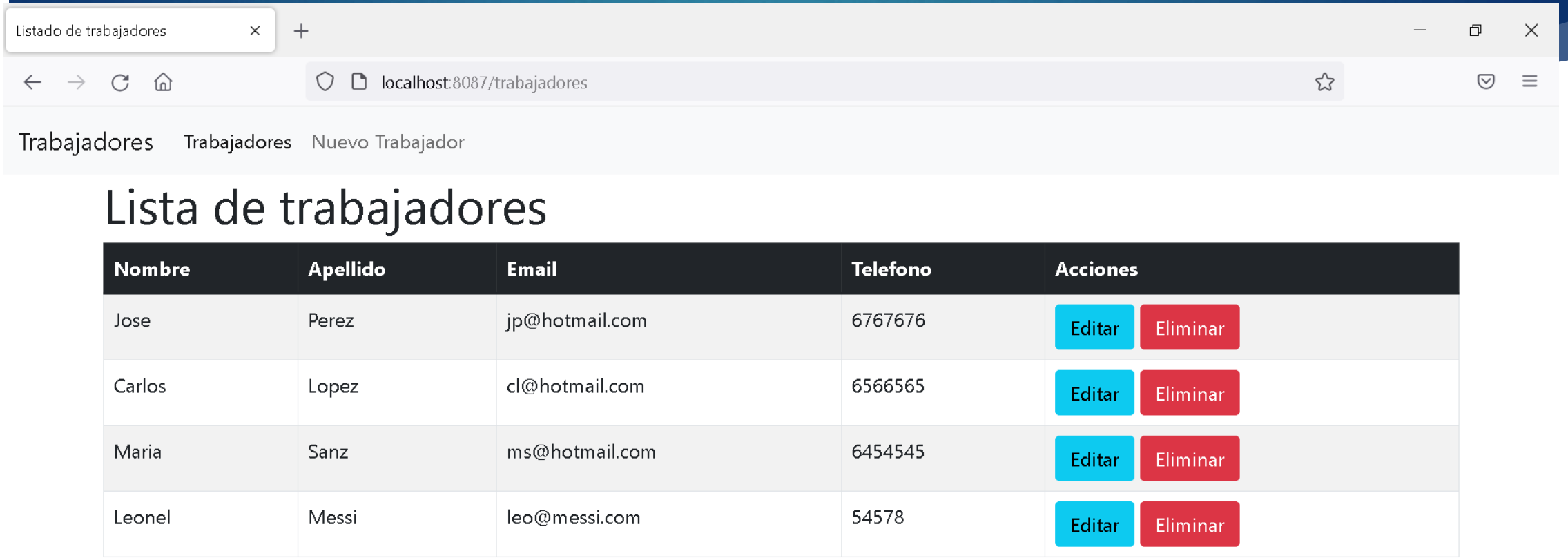
# Creamos una copia de crear_trabajador y la llamamos editar_trabajador y solo editamos la línea marcada aquí



# En la plantilla trabajador.html agrego lo siguiente



# Probamos el CRUD completo



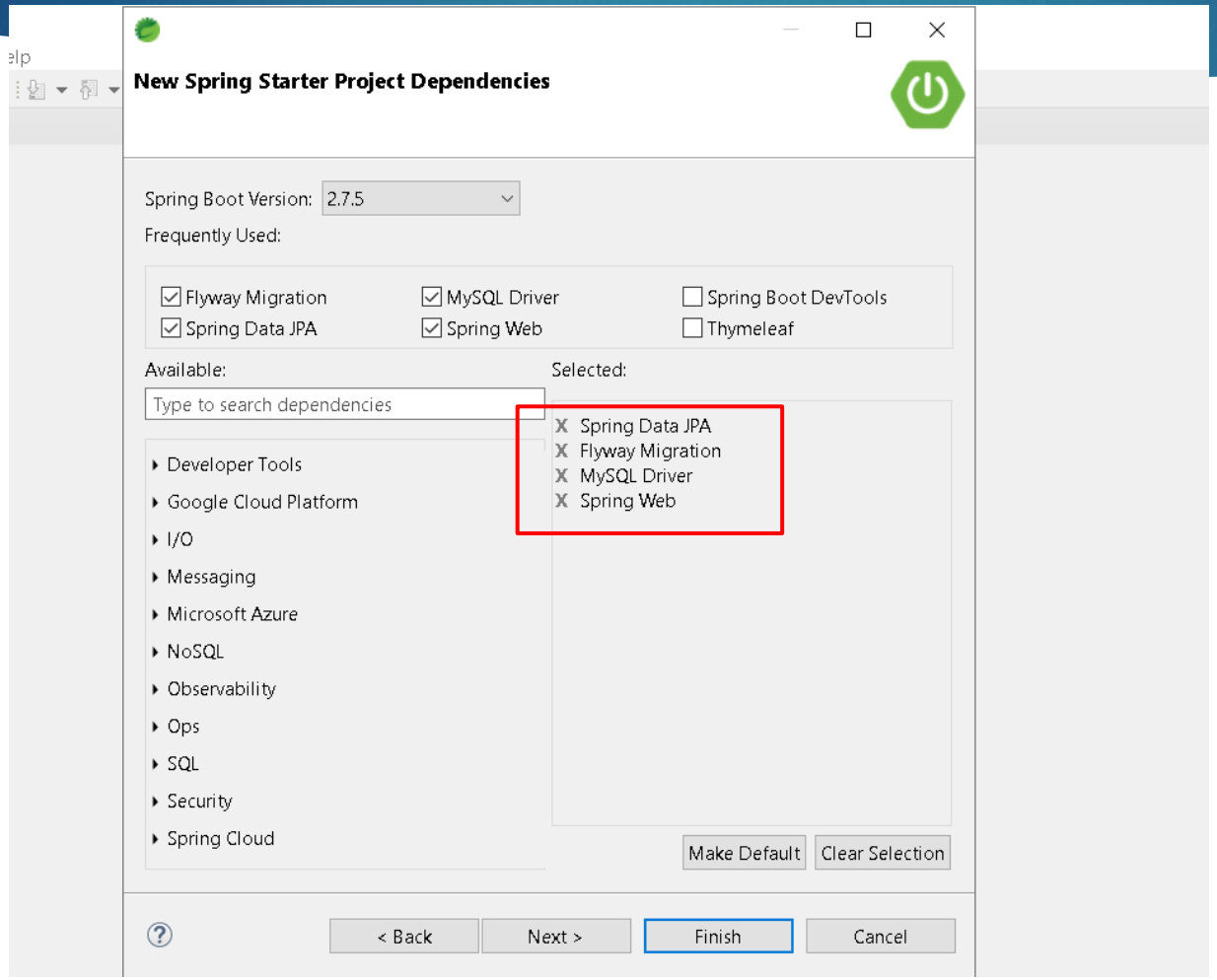
# ¿Qué es Flyway?

- Es un sistema robusto de **migración de base de datos**, que funciona con diferentes bases de datos y hace cualquier migración fácil y sencilla. Tiene compatibilidad con Java y también puede realizar migraciones a través de plugins de maven y gradle

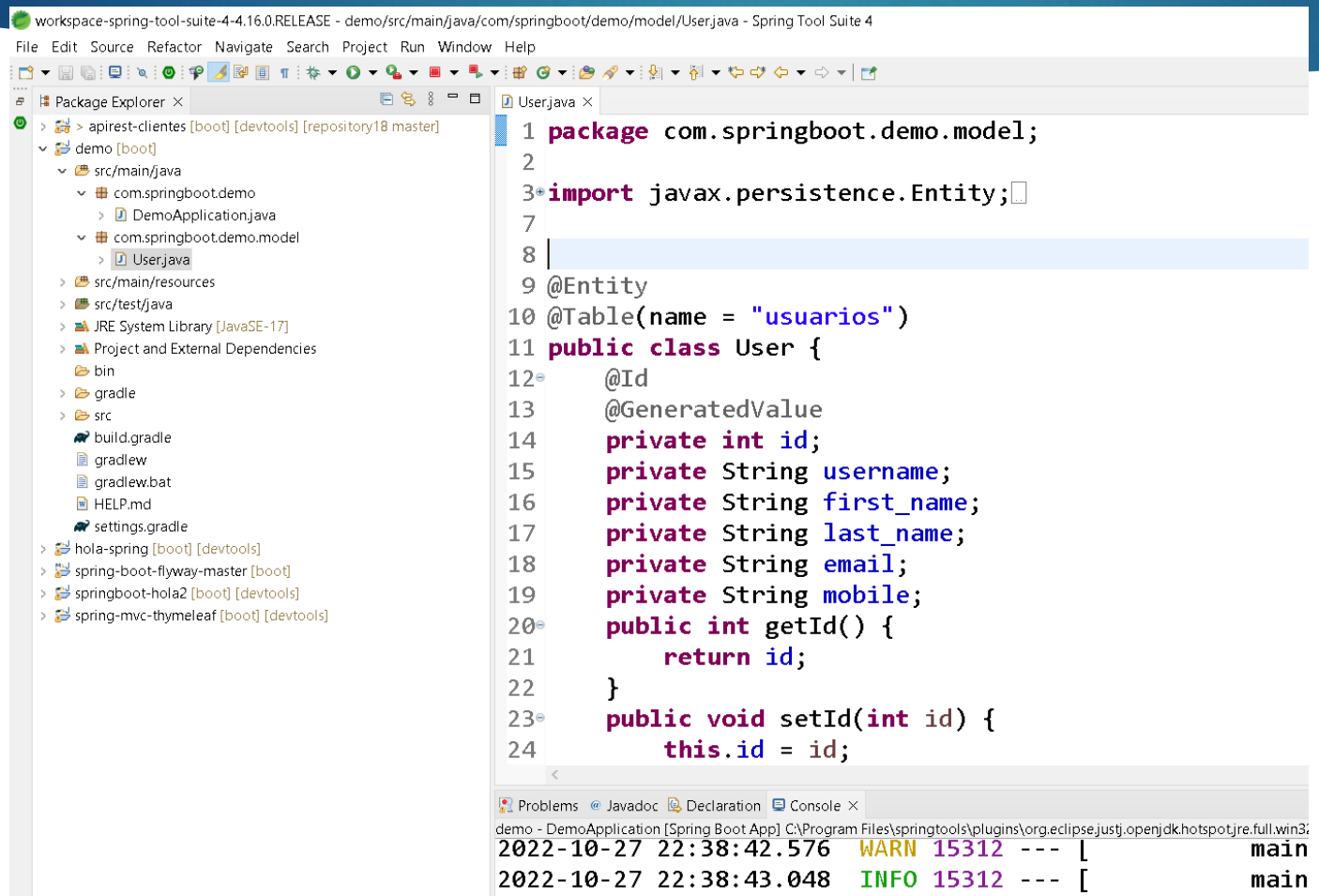
# Como funciona Flyway

- Flyway nos va a ayudar a seguir **una trazabilidad de que scripts que ya se han migrado, cuando y quién lo ha realizado**. Esta nueva tabla que Flyway crea en nuestro schema, va añade metada con un ***checksum*** en el que se indica si la migración tuvo éxito.
- Cuando se ejecuta flyway para realizar una migración de sql, se realiza las siguiente comprobaciones:
- Se crea la tabla en la que guardar la información.
- Se analiza los ficheros de la migración.
- Se comparan los ficheros con las migraciones pasadas si ha habido cambios en los ficheros de versiones anteriores fallará la migración.
- Los nuevos scripts de migración serán ejecutados y se añadirá la información de los nuevos scripts.

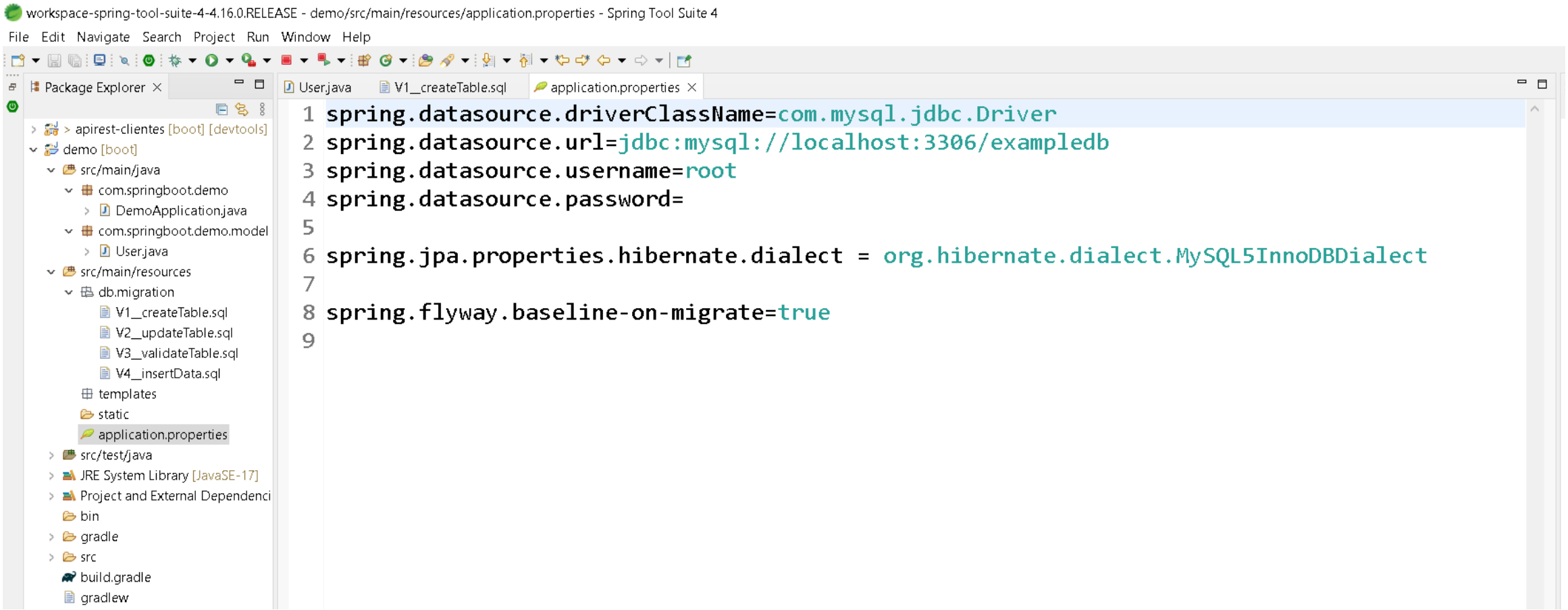
# Creamos un ejemplo

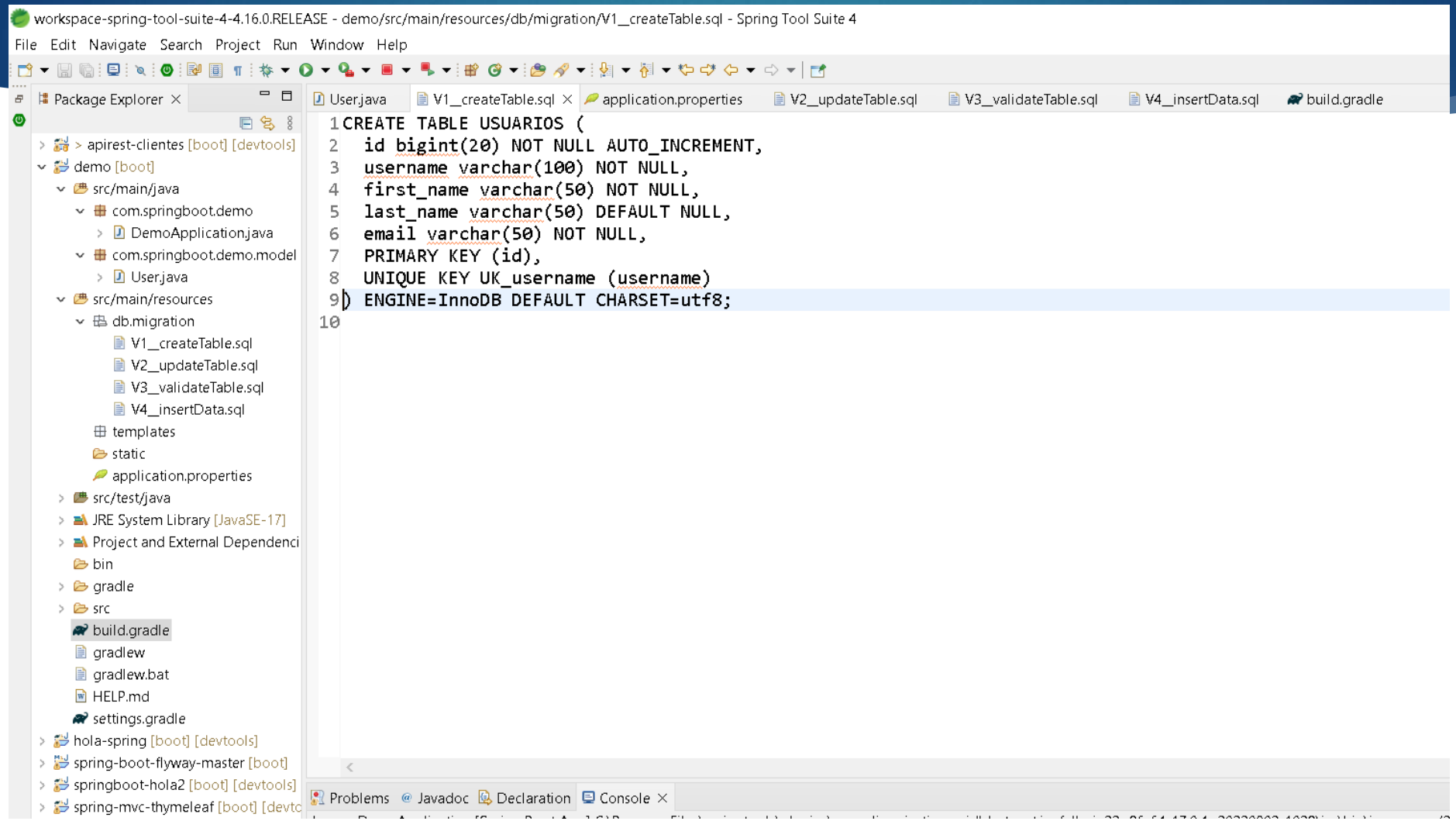


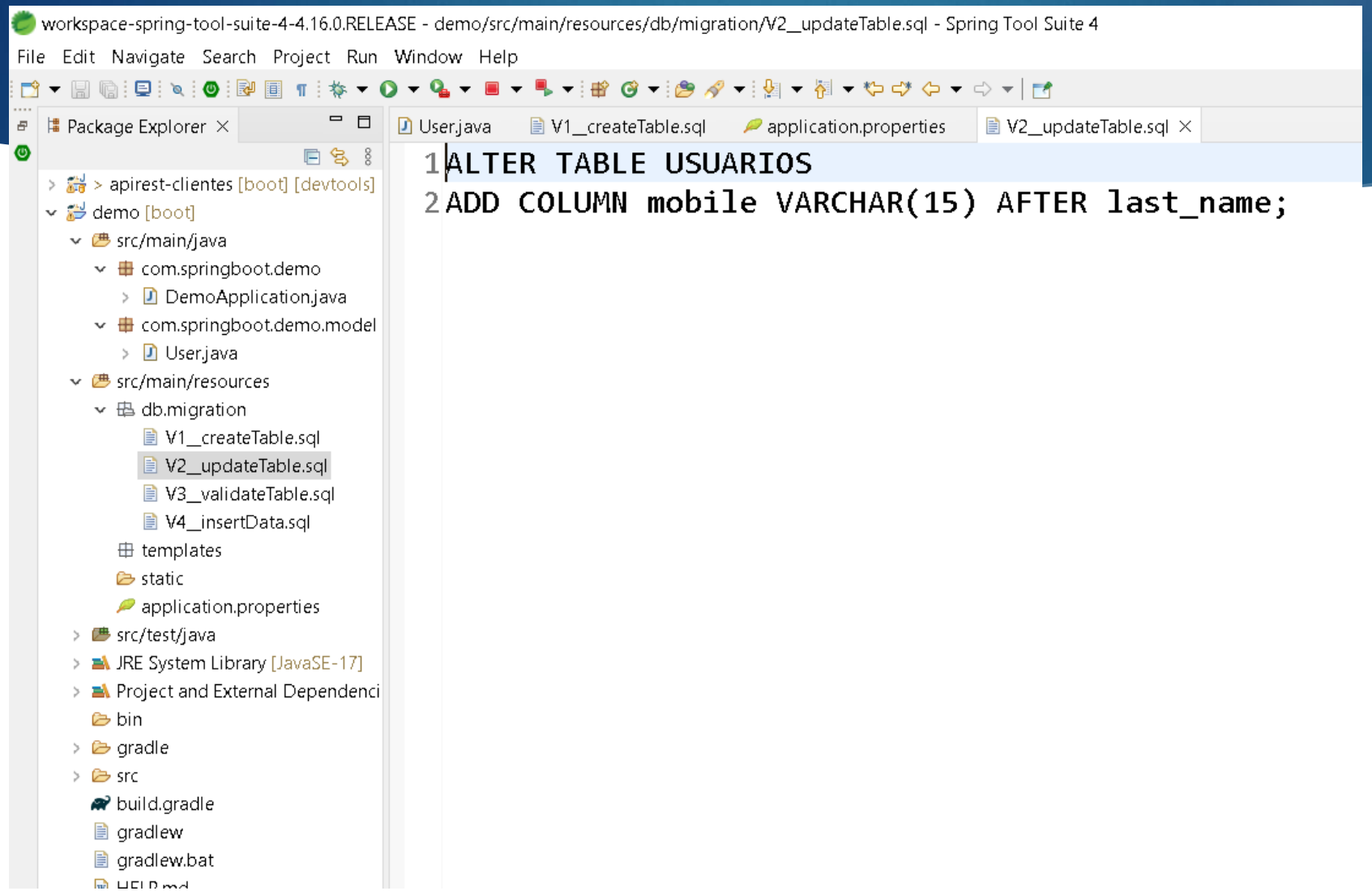
# Creamos una entidad user

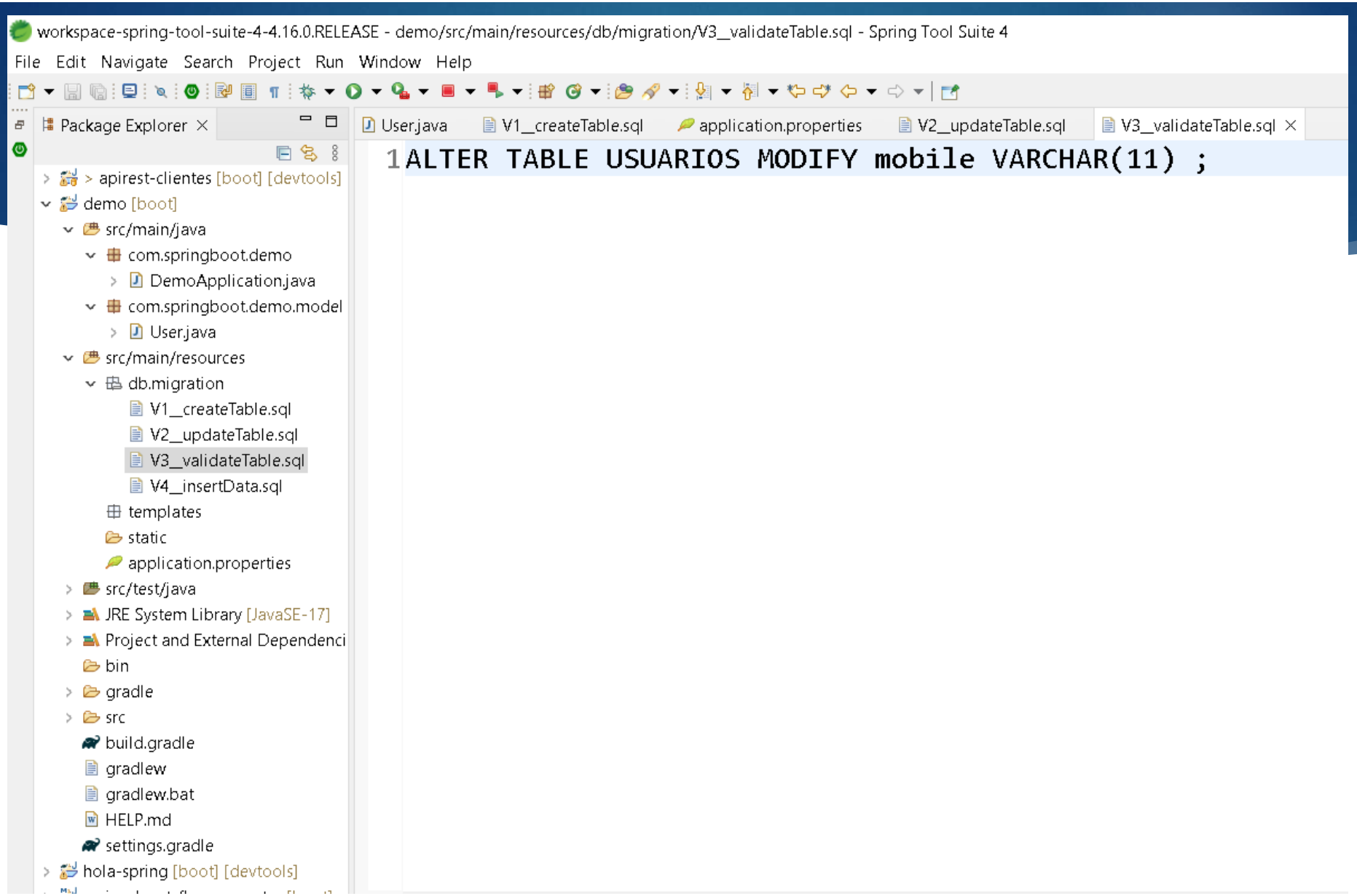


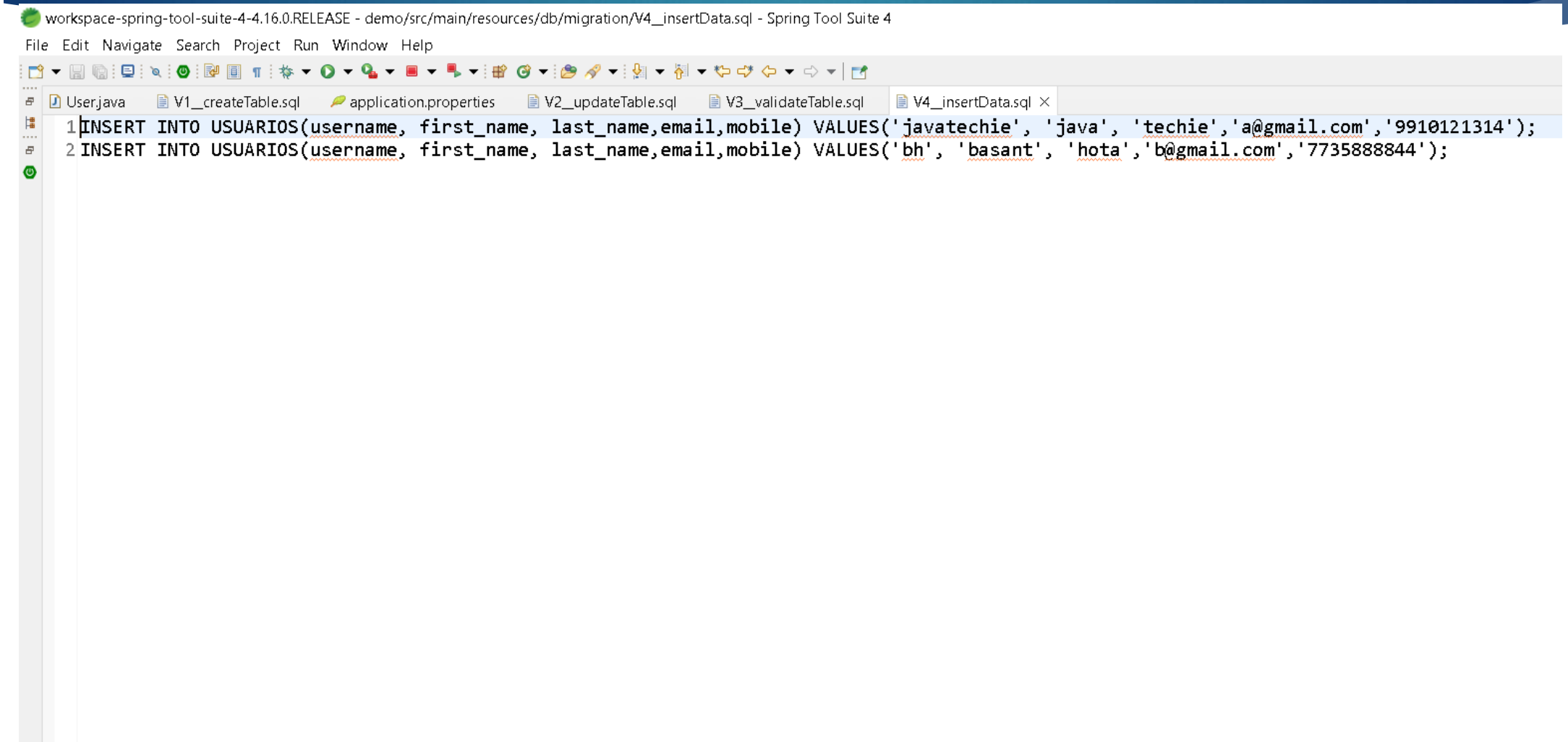
# Configuramos el application.properties




workspace-spring-tool-suite-4-4.16.0.RELEASE - demo/src/main/resources/db/migration/V1__createTable.sql - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
Package Explorer
apirest-clientes [boot] [devtools]
demo [boot]
src/main/java
com.springboot.demo
DemoApplication.java
com.springboot.demo.model
User.java
src/main/resources
db.migration
V1__createTable.sql
V2__updateTable.sql
V3__validateTable.sql
V4__insertData.sql
templates
static
application.properties
src/test/java
JRE System Library [JavaSE-17]
Project and External Dependenci
bin
gradle
src
build.gradle
gradlew
gradlew.bat
HELP.md
settings.gradle
hola-spring [boot] [devtools]
spring-boot-flyway-master [boot]
springboot-hola2 [boot] [devtools]
spring-mvc-thymeleaf [boot] [devtc
User.java
V1__createTable.sql
application.properties
V2__updateTable.sql
V3__validateTable.sql
V4__insertData.sql
build.gradle
1 CREATE TABLE USUARIOS (
2 id bigint(20) NOT NULL AUTO_INCREMENT,
3 username varchar(100) NOT NULL,
4 first_name varchar(50) NOT NULL,
5 last_name varchar(50) DEFAULT NULL,
6 email varchar(50) NOT NULL,
7 PRIMARY KEY (id),
8 UNIQUE KEY UK_username (username)
9 ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
10
Problems Javadoc Declaration Console


workspace-spring-tool-suite-4-4.16.0.RELEASE - demo/src/main/resources/db/migration/V2__updateTable.sql - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
Package Explorer
> apirest-clientes [boot] [devtools]
demo [boot]
src/main/java
com.springboot.demo
DemoApplication.java
com.springboot.demo.model
User.java
src/main/resources
db.migration
V1__createTable.sql
V2__updateTable.sql
V3__validateTable.sql
V4__insertData.sql
templates
static
application.properties
src/test/java
JRE System Library [JavaSE-17]
Project and External Dependenci
bin
gradle
src
build.gradle
gradlew
gradlew.bat
User.java
V1__createTable.sql
application.properties
V2__updateTable.sql
1 ALTER TABLE USUARIOS
2 ADD COLUMN mobile VARCHAR(15) AFTER last_name;


workspace-spring-tool-suite-4-4.16.0.RELEASE - demo/src/main/resources/db/migration/V3__validateTable.sql - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
Package Explorer
apirest-clientes [boot] [devtools]
demo [boot]
src/main/java
com.springboot.demo
DemoApplication.java
com.springboot.demo.model
User.java
src/main/resources
db.migration
V1__createTable.sql
V2__updateTable.sql
V3__validateTable.sql
V4__insertData.sql
templates
static
application.properties
src/test/java
JRE System Library [JavaSE-17]
Project and External Dependenci
bin
gradle
src
build.gradle
gradlew
gradlew.bat
HELP.md
settings.gradle
hola-spring [boot] [devtools]
User.java
V1__createTable.sql
application.properties
V2__updateTable.sql
V3__validateTable.sql
1 ALTER TABLE USUARIOS MODIFY mobile VARCHAR(11) ;


workspace-spring-tool-suite-4-4.16.0.RELEASE - demo/src/main/resources/db/migration/V4__insertData.sql - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
User.java
V1__createTable.sql
application.properties
V2__updateTable.sql
V3__validateTable.sql
V4__insertData.sql
1 INSERT INTO USUARIOS(username, first_name, last_name,email,mobile) VALUES('javatechie', 'java', 'techie','a@gmail.com','9910121314');
2 INSERT INTO USUARIOS(username, first_name, last_name,email,mobile) VALUES('bh', 'basant', 'hota','b@gmail.com','7735888844');

# Unit Test

- Son códigos diseñados para comprobar que el código principal este funcionando de la forma que esperamos.

# Tipos de pruebas

- Pruebas Unitarias (Unit Test)
- Pruebas de Integración
- Pruebas de funcionamiento del sistema
- Pruebas de Aceptación
- Pruebas de estress

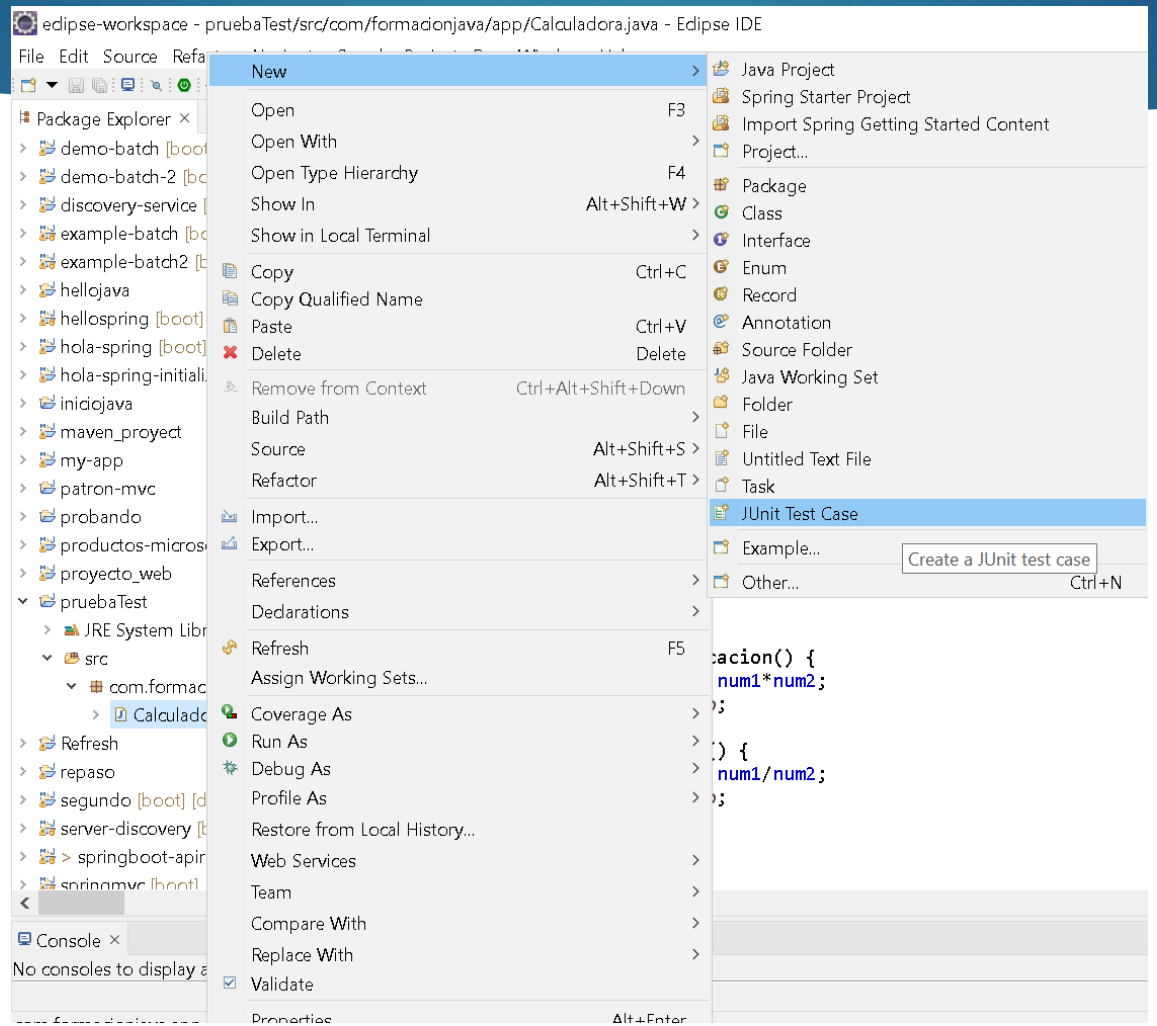
# JUnit

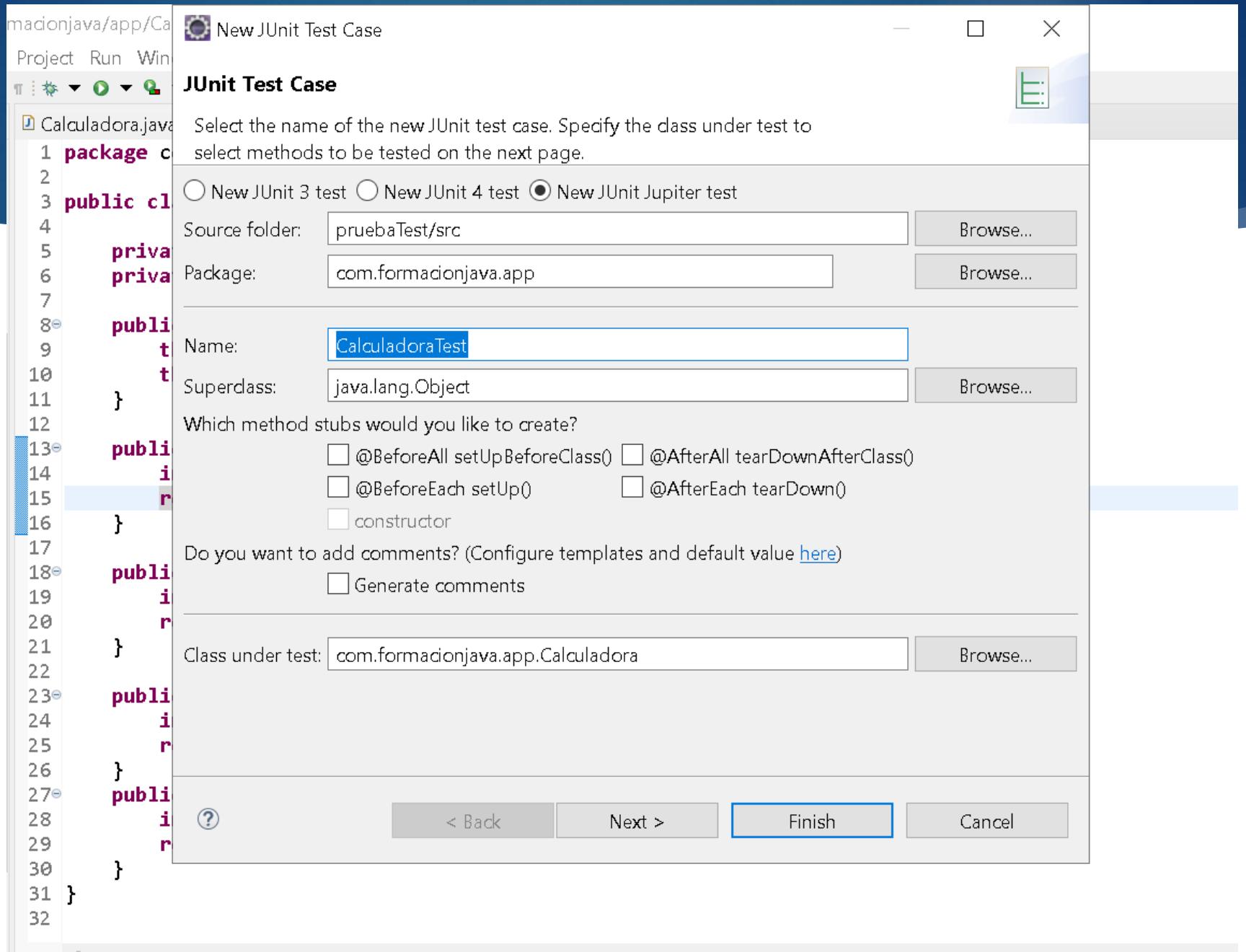
- Es una librería java para escribir y ejecutar pruebas unitarias

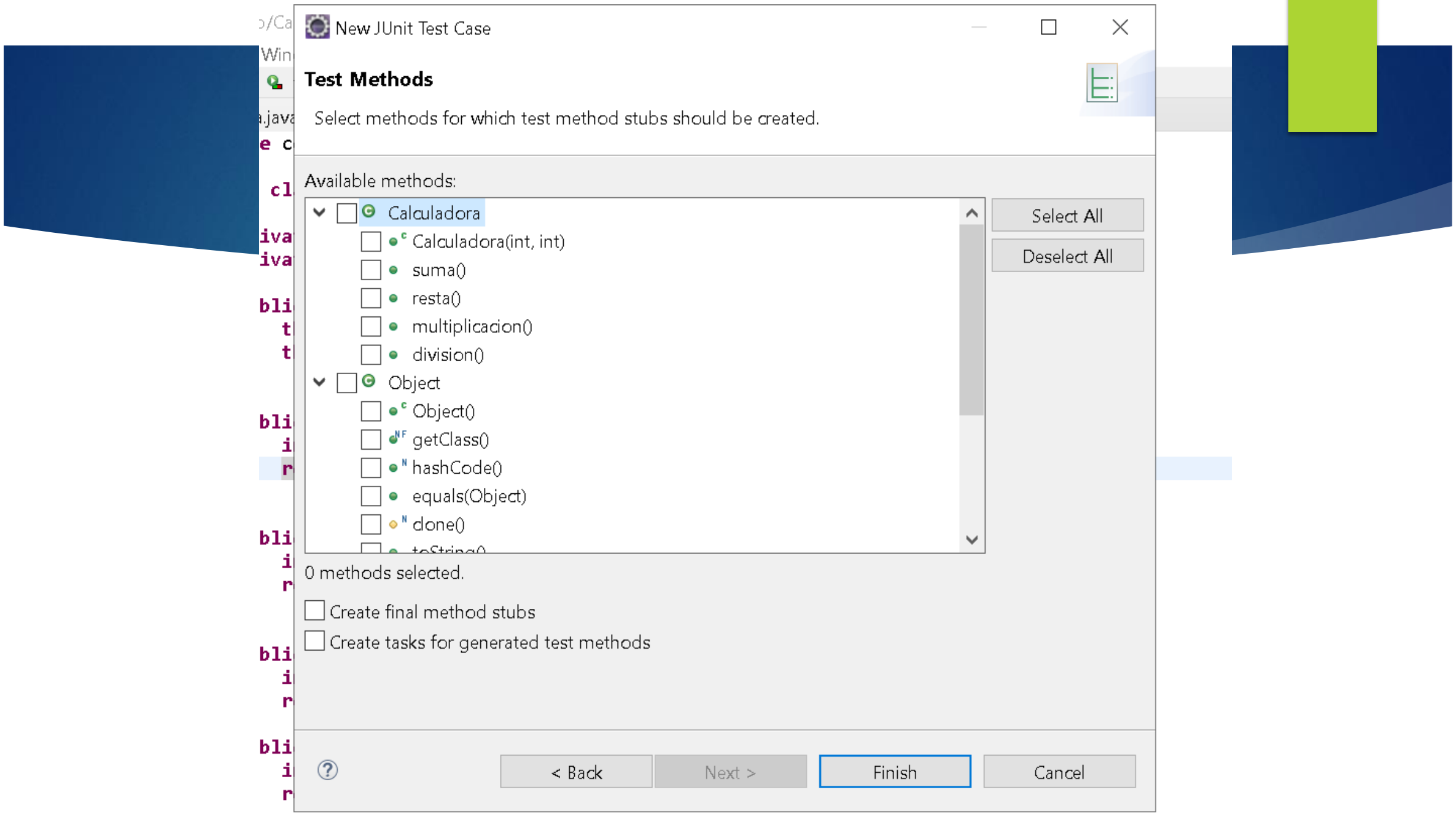
# Realizamos una prueba

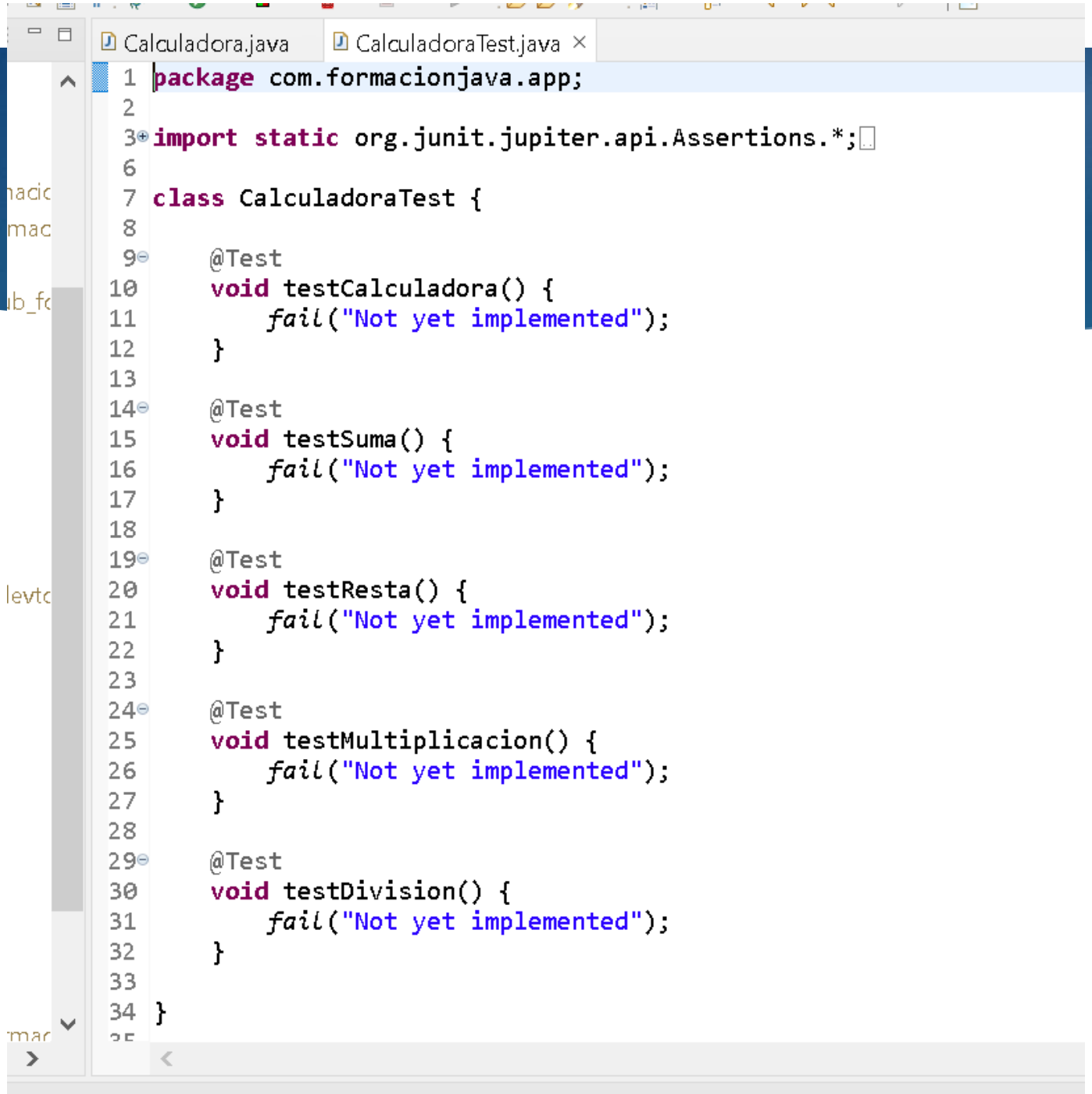
```
2
3  public class Calculadora {
4
5      private int num1;
6      private int num2;
7
8      public Calculadora(int a,int b) {
9          this.num1 = a;
10         this.num2 = b;
11     }
12
13     public int suma() {
14         int resultado = num1+num2;
15         return resultado;
16     }
17
18     public int resta() {
19         int resultado = num1-num2;
20         return resultado;
21     }
22
23     public int multiplicacion() {
24         int resultado = num1*num2;
25         return resultado;
26     }
27     public int division() {
28         int resultado = num1/num2;
29         return resultado;
30     }
31 }
32
```

# Creamos la clase de unitTest

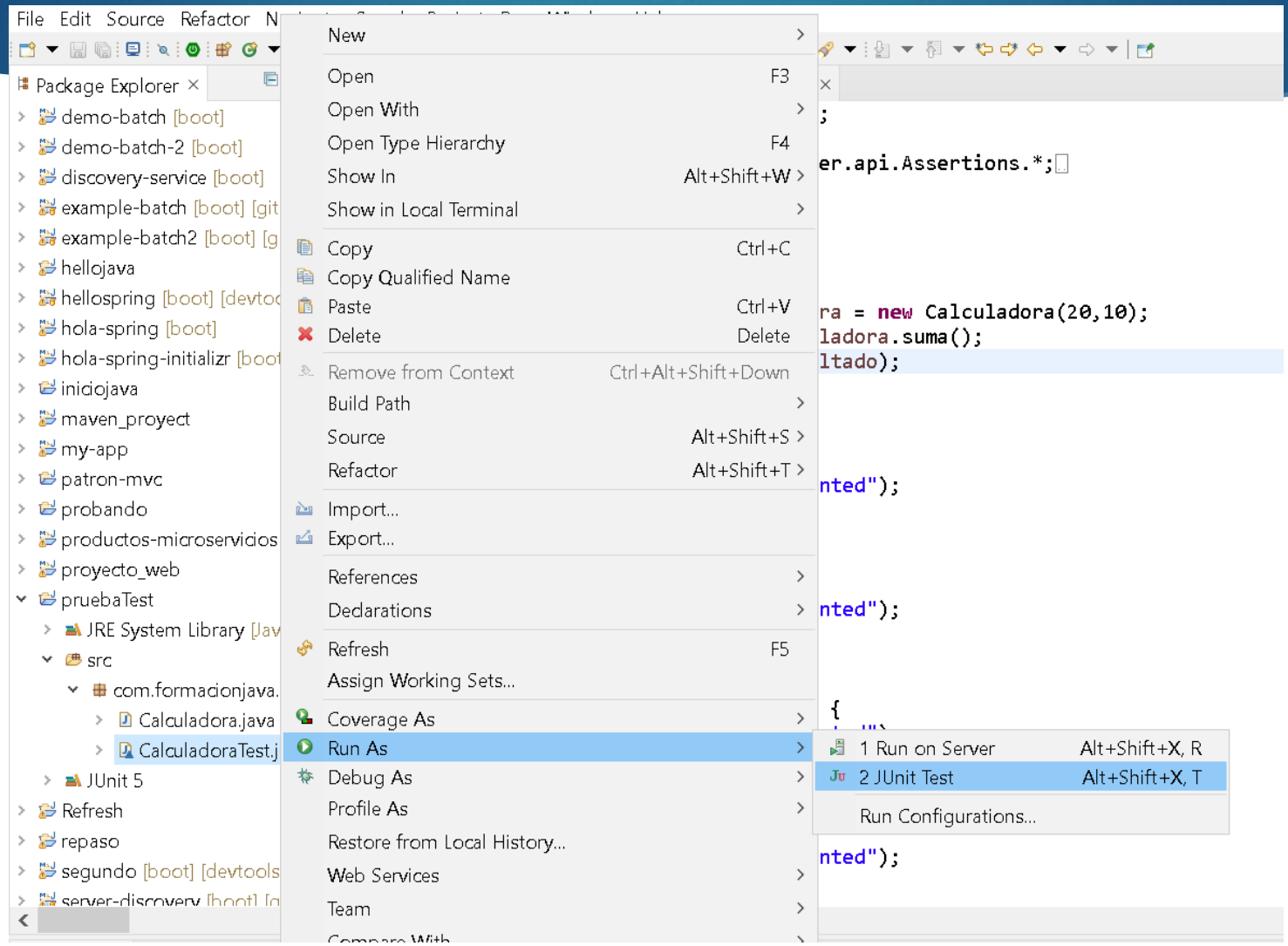


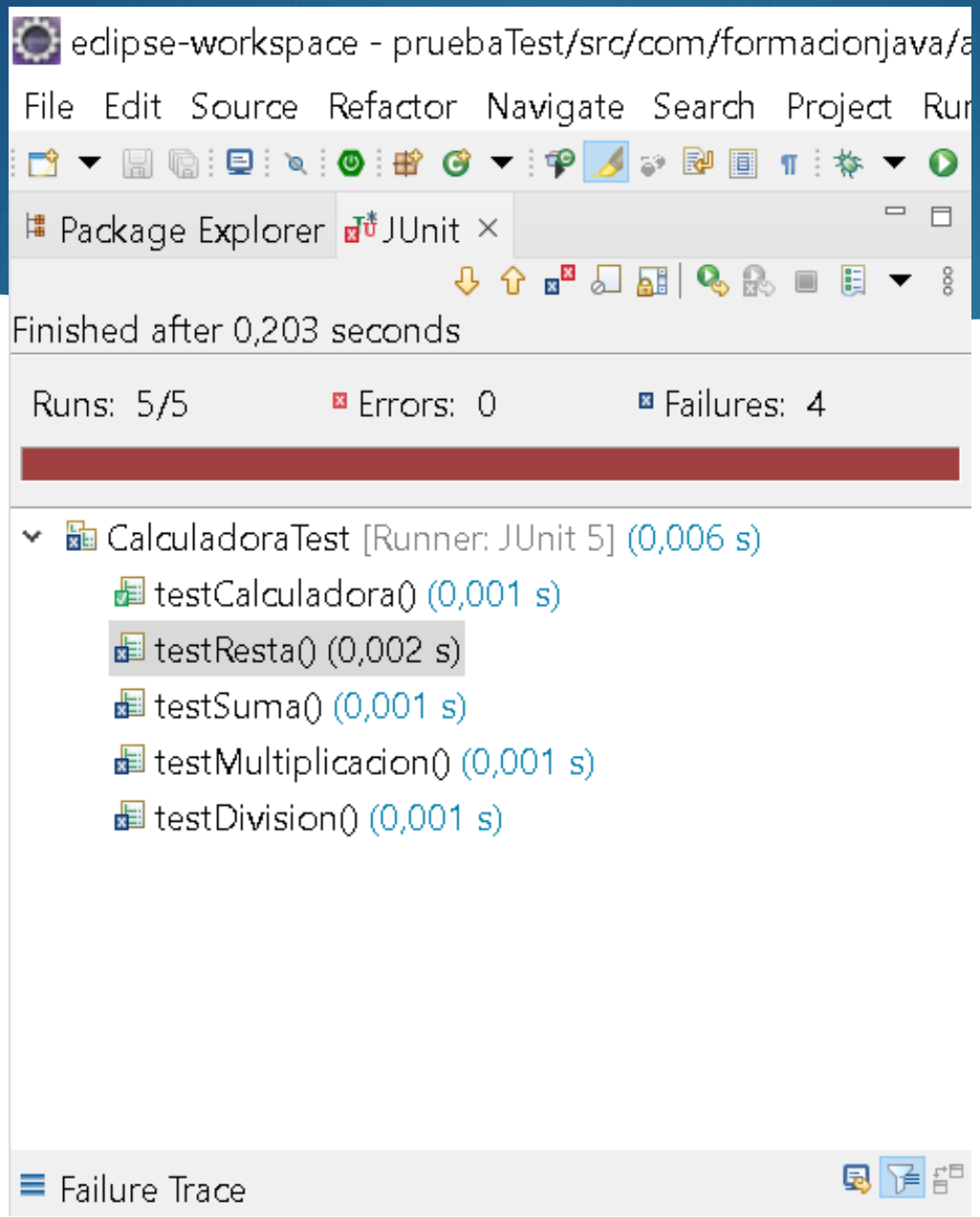

macionjava/app/Ca
Project Run Win
Calculadora.java
1 package c
2
3 public cl
4
5 priva
6 priva
7
8 publi
9 t
10 t
11 }
12
13 publi
14 i
15 r
16 }
17
18 publi
19 i
20 r
21 }
22
23 publi
24 i
25 r
26 }
27 publi
28 i
29 r
30 }
31 }
32
New JUnit Test Case
JUnit Test Case
Select the name of the new JUnit test case. Specify the class under test to select methods to be tested on the next page.
New JUnit 3 test
New JUnit 4 test
New JUnit Jupiter test
Source folder:
pruebaTest/src
Browse...
Package:
com.formacionjava.app
Browse...
Name:
CalculadoraTest
Superclass:
java.lang.Object
Browse...
Which method stubs would you like to create?
@BeforeAll setUpBeforeClass()
@AfterAll tearDownAfterClass()
@BeforeEach setUp()
@AfterEach tearDown()
constructor
Do you want to add comments? (Configure templates and default value here)
Generate comments
Class under test:
com.formacionjava.app.Calculadora
Browse...
< Back
Next >
Finish
Cancel

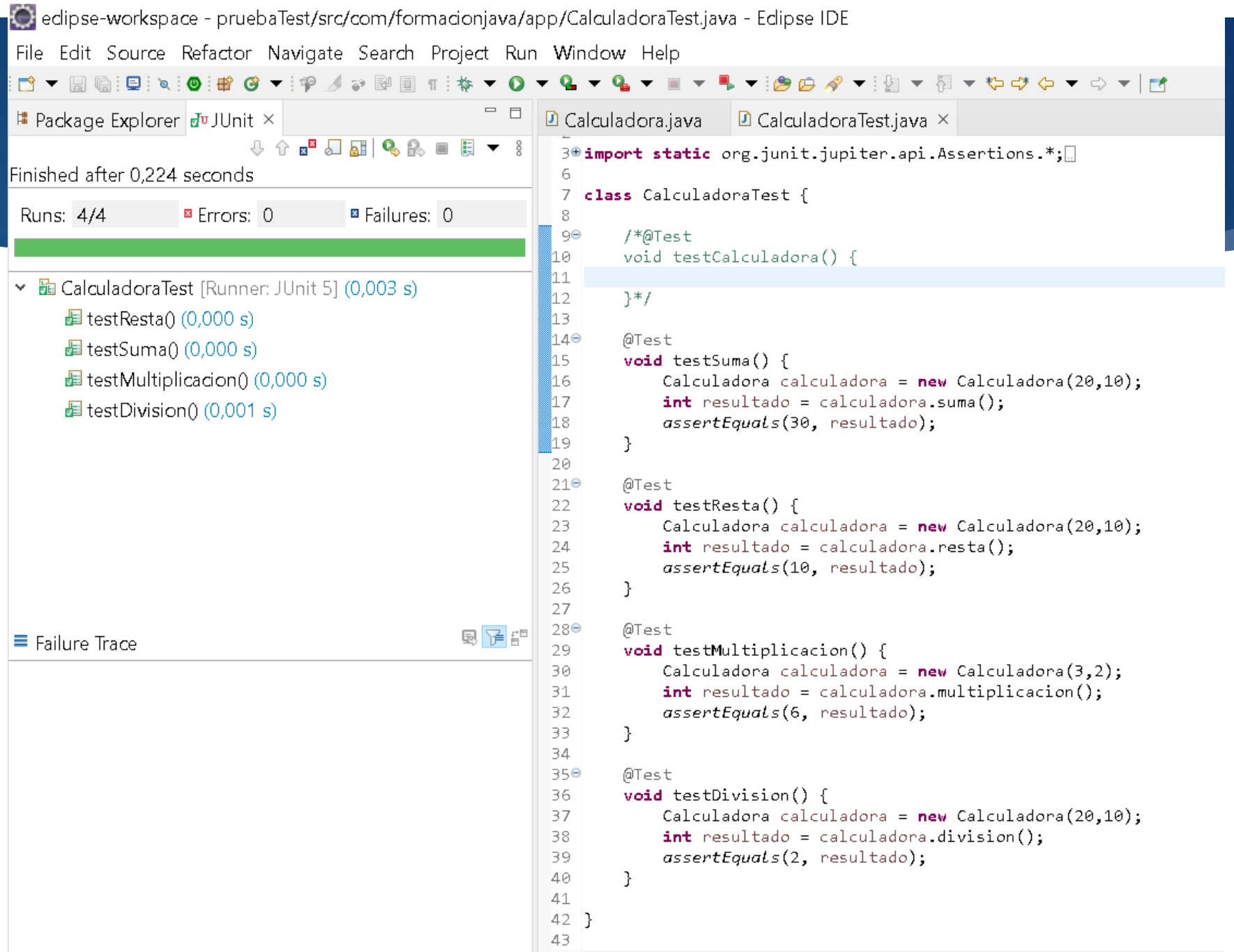

New JUnit Test Case
Test Methods
Select methods for which test method stubs should be created.
Available methods:
Calculadora
Calculadora(int, int)
suma()
resta()
multiplicacion()
division()
Object
Object()
getClass()
hashCode()
equals(Object)
clone()
Select All
Deselect All
0 methods selected.
Create final method stubs
Create tasks for generated test methods
< Back
Next >
Finish
Cancel


Calculadora.java
CalculadoraTest.java
1 package com.formacionjava.app;
2
3 import static org.junit.jupiter.api.Assertions.*;
6
7 class CalculadoraTest {
8
9 @Test
10 void testCalculadora() {
11 fail("Not yet implemented");
12 }
13
14 @Test
15 void testSuma() {
16 fail("Not yet implemented");
17 }
18
19 @Test
20 void testResta() {
21 fail("Not yet implemented");
22 }
23
24 @Test
25 void testMultiplicacion() {
26 fail("Not yet implemented");
27 }
28
29 @Test
30 void testDivision() {
31 fail("Not yet implemented");
32 }
33
34 }

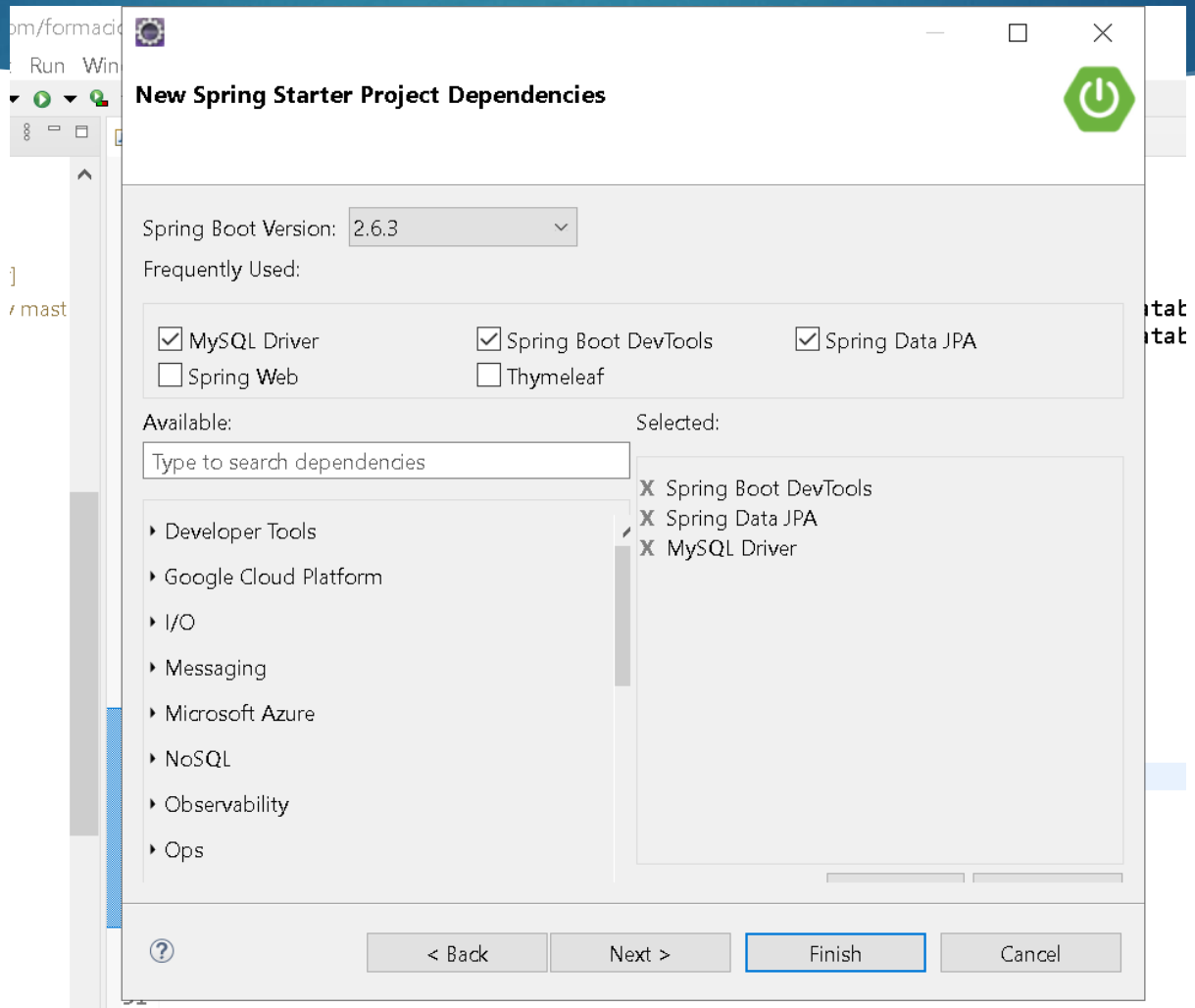
# Ejecutamos el UnitTest



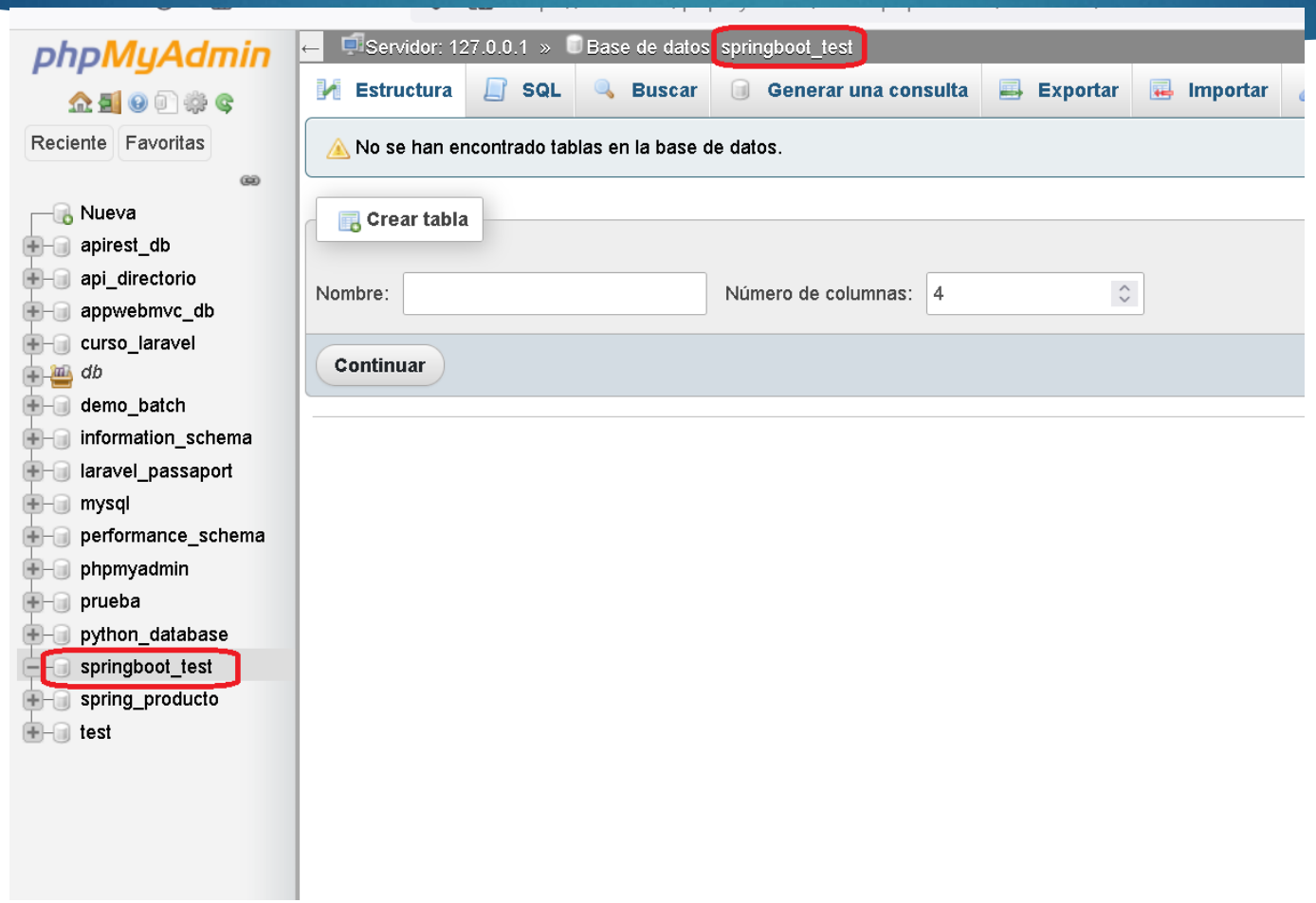

eclipse-workspace - pruebaTest/src/com/formacionjava/a
File Edit Source Refactor Navigate Search Project Ru
Package Explorer
JUnit
Finished after 0,203 seconds
Runs: 5/5
Errors: 0
Failures: 4
CalculadoraTest [Runner: JUnit 5] (0,006 s)
testCalculadora() (0,001 s)
testResta() (0,002 s)
testSuma() (0,001 s)
testMultiplicacion() (0,001 s)
testDivision() (0,001 s)
Failure Trace


eclipse-workspace - pruebaTest/src/com/formacionjava/app/CalculadoraTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 0,224 seconds
Runs: 4/4
Errors: 0
Failures: 0
CalculadoraTest [Runner: JUnit 5] (0,003 s)
testResta() (0,000 s)
testSuma() (0,000 s)
testMultiplicacion() (0,000 s)
testDivision() (0,001 s)
Failure Trace
Calculadora.java
CalculadoraTest.java
3 import static org.junit.jupiter.api.Assertions.*;
6
7 class CalculadoraTest {
8
9 /*@Test
10 void testCalculadora() {
11
12 }*/
13
14 @Test
15 void testSuma() {
16 Calculadora calculadora = new Calculadora(20,10);
17 int resultado = calculadora.suma();
18 assertEquals(30, resultado);
19 }
20
21 @Test
22 void testResta() {
23 Calculadora calculadora = new Calculadora(20,10);
24 int resultado = calculadora.resta();
25 assertEquals(10, resultado);
26 }
27
28 @Test
29 void testMultiplicacion() {
30 Calculadora calculadora = new Calculadora(3,2);
31 int resultado = calculadora.multiplicacion();
32 assertEquals(6, resultado);
33 }
34
35 @Test
36 void testDivision() {
37 Calculadora calculadora = new Calculadora(20,10);
38 int resultado = calculadora.division();
39 assertEquals(2, resultado);
40 }
41
42 }
43

# Utilizando pruebas unitarias en SpringBoot, creamos un proyecto con las siguientes dependencias



# Creamos una base de datos



# En nuestro application.prioperties

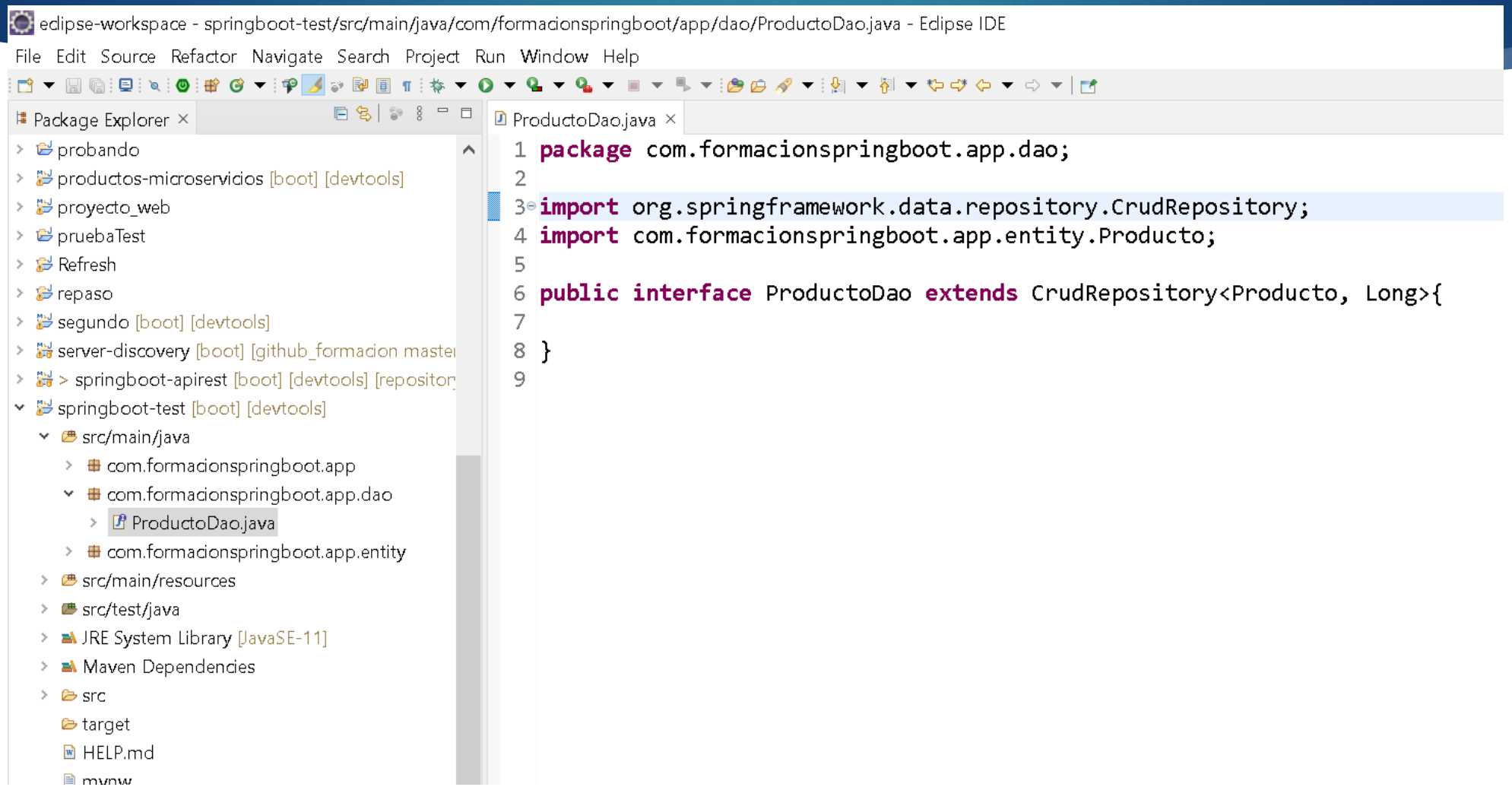
Producto.java | application.properties

```
1 spring.datasource.url= jdbc:mysql://localhost/springboot_test
2 spring.datasource.username=root
3 spring.datasource.password=
4 spring.jpa.hibernate.ddl-auto=create
5 spring.jpa.show-sql=true
6 spring.jpa.properties.hibernate.format_sql=true
7 server.port=8087
```

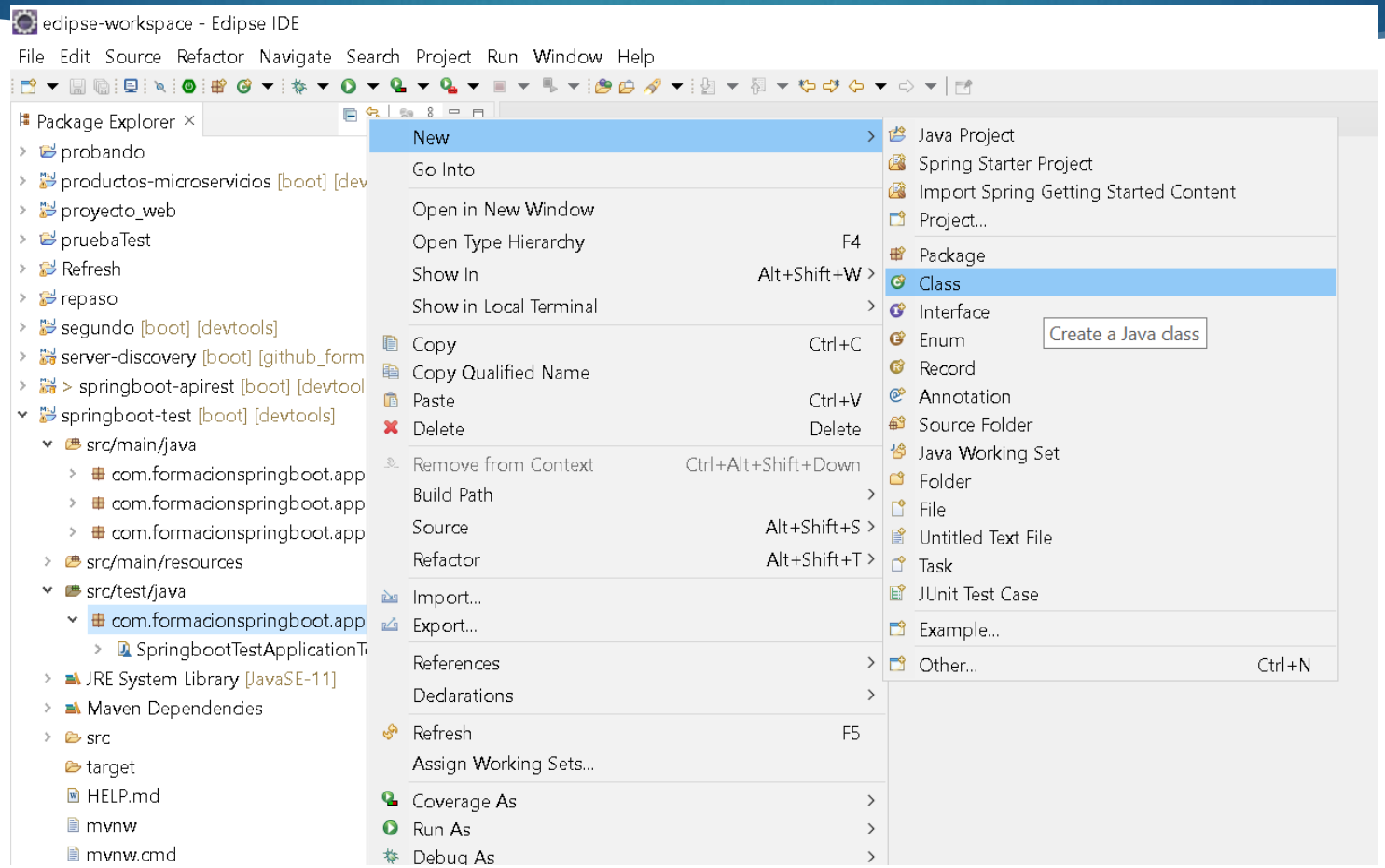
# Creamos la entidad Producto

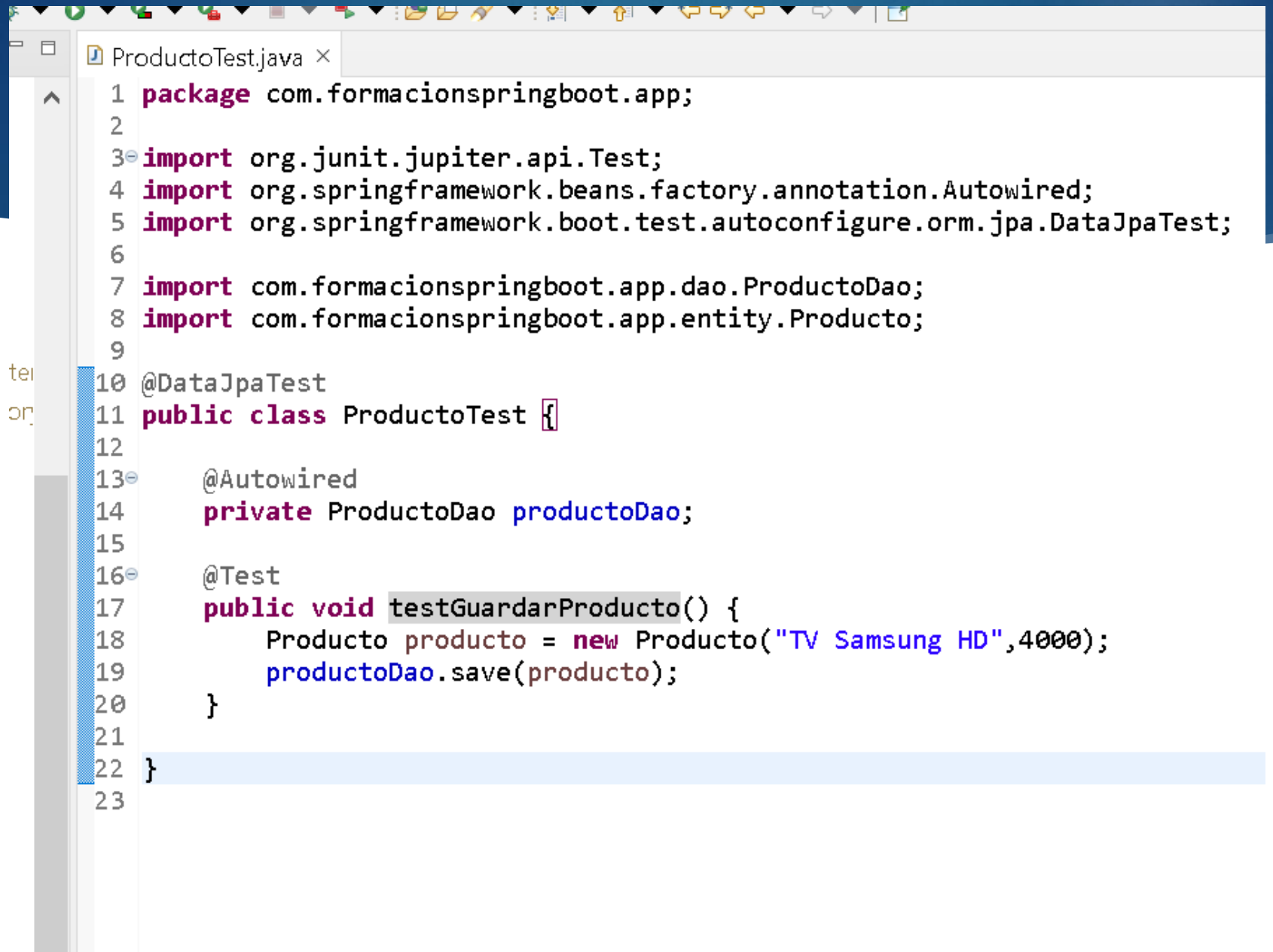
```java
package com.formacionspringboot.app.entity;

import javax.persistence.Entity;

@Entity
@Table(name="productos")
public class Producto {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nombre;
    private float precio;

    public Producto() {

    }
    public Producto(String nombre, float precio) {
        this.nombre = nombre;
        this.precio = precio;
    }
    public Long getId() {
        return id;
    }
    public void setId(Long id) {
        this.id = id;
    }
    public String getNombre() {
        return nombre;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
```

# Creamos un repositorio ProductoDao

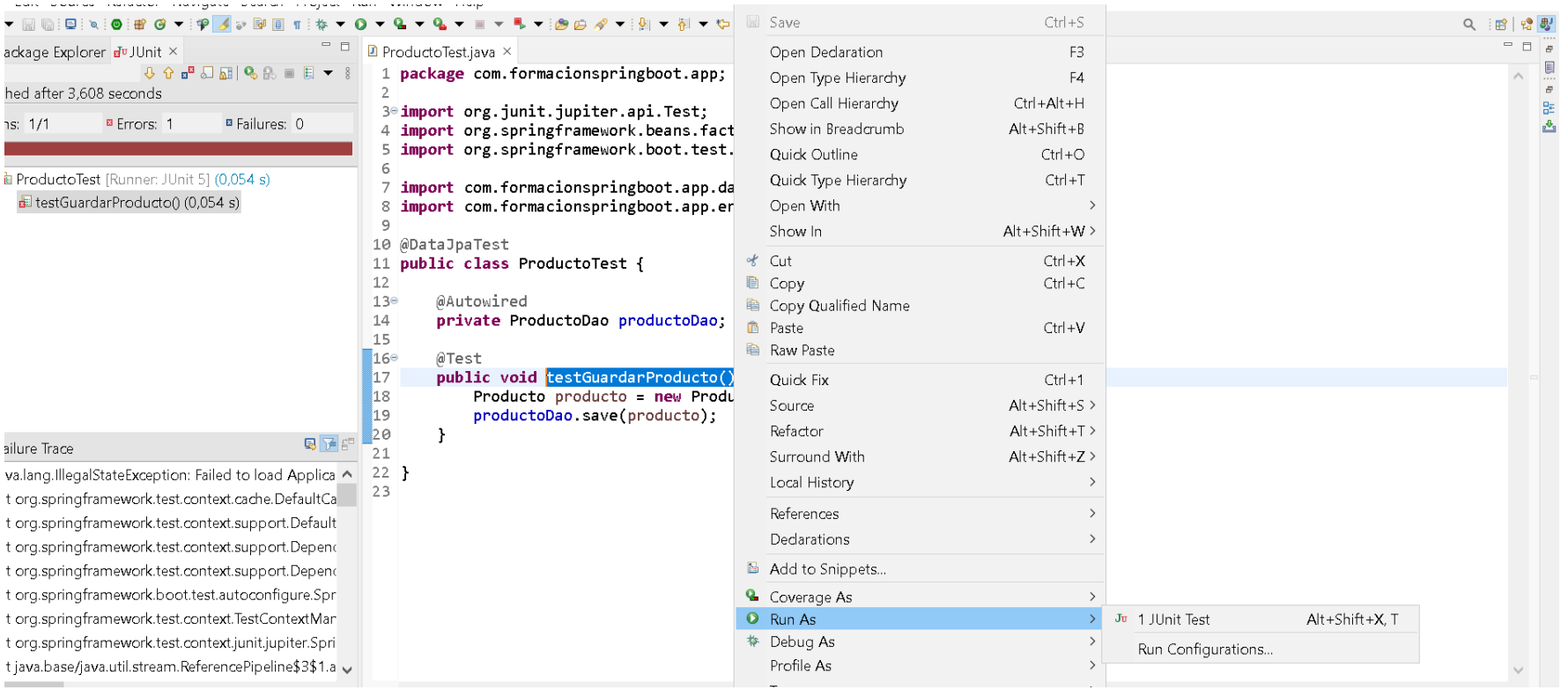


# Dentro del paquete principal de src/test/java creamos una clase ProductoTest




ProductoTest.java
1 package com.formacionspringboot.app;
2
3 import org.junit.jupiter.api.Test;
4 import org.springframework.beans.factory.annotation.Autowired;
5 import org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTest;
6
7 import com.formacionspringboot.app.dao.ProductoDao;
8 import com.formacionspringboot.app.entity.Producto;
9
10 @DataJpaTest
11 public class ProductoTest {
12
13 @Autowired
14 private ProductoDao productoDao;
15
16 @Test
17 public void testGuardarProducto() {
18 Producto producto = new Producto("TV Samsung HD",4000);
19 productoDao.save(producto);
20 }
21
22 }
23

# Lo ejecutamos



# Agregamos las notaciones y código para necesarias para lanzar los test y registrarlos en la base de datos

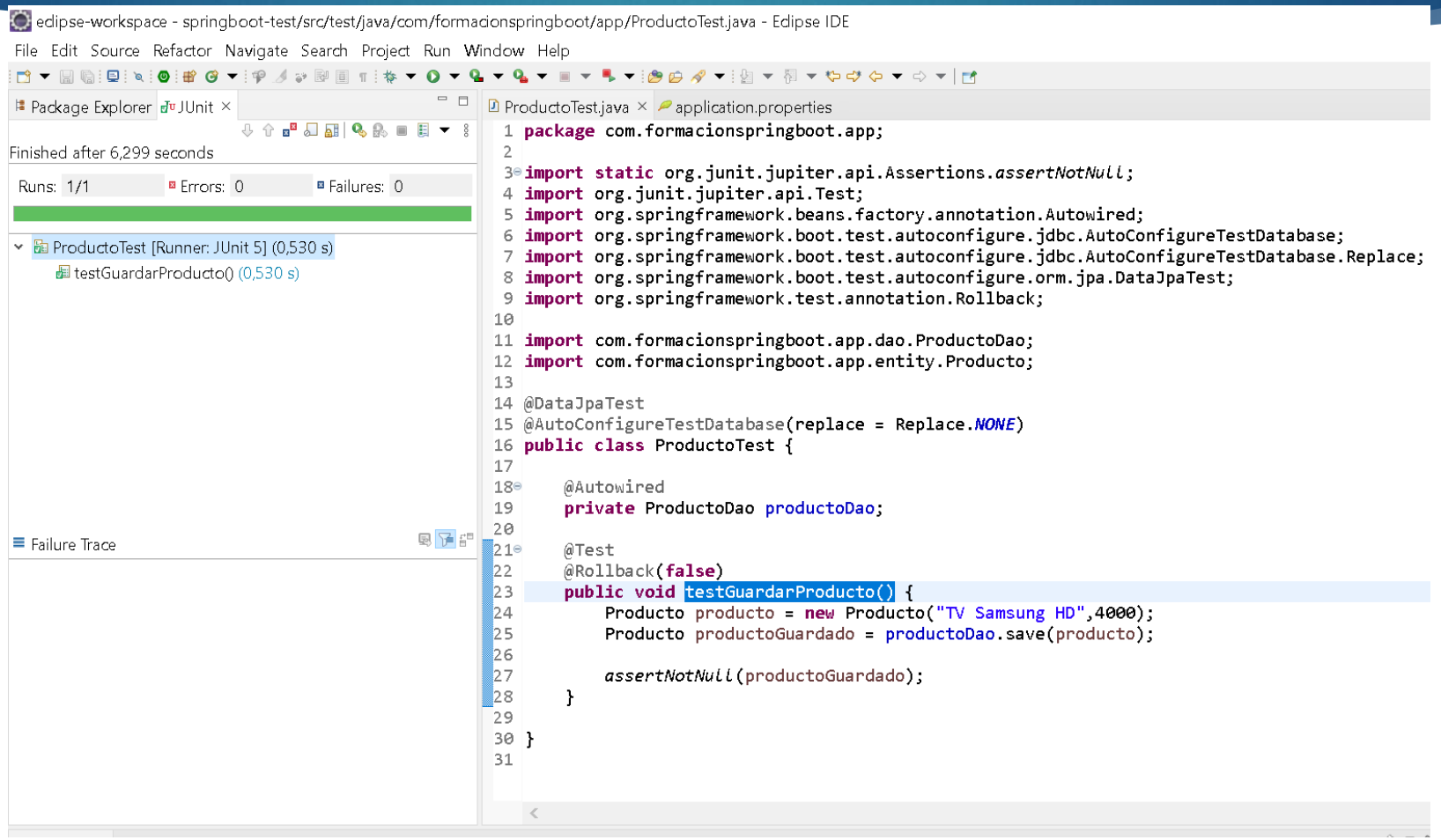
nacionspringboot/app/ProductoTest.java - Eclipse IDE

Window Help

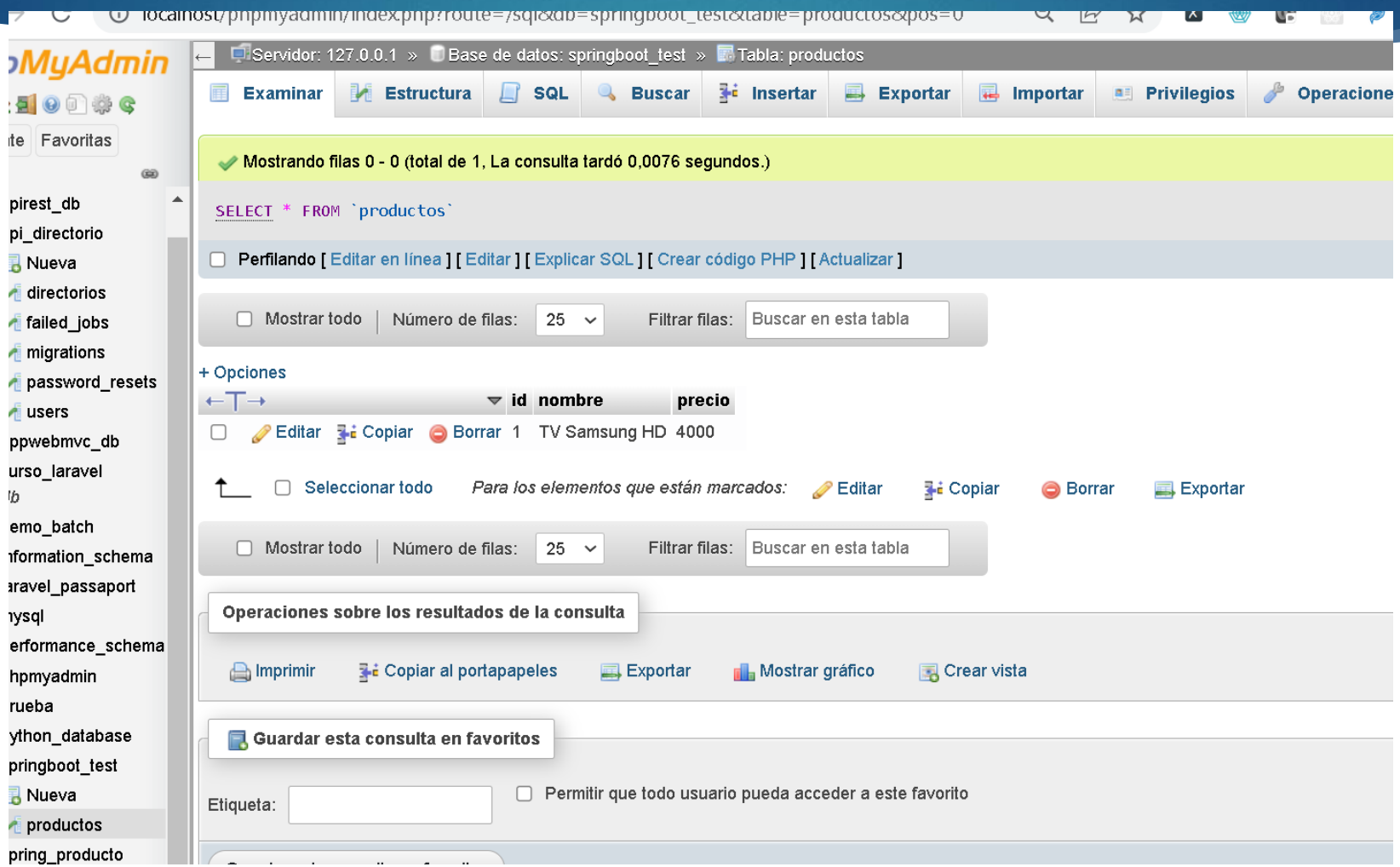
ProductoTest.java × application.properties

```java
package com.formacionspringboot.app;

import static org.junit.jupiter.api.Assertions.assertNotNull;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.jdbc.AutoConfigureTestDatabase;
import org.springframework.boot.test.autoconfigure.jdbc.AutoConfigureTestDatabase.Replace;
import org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTest;
import org.springframework.test.annotation.Rollback;

import com.formacionspringboot.app.dao.ProductoDao;
import com.formacionspringboot.app.entity.Producto;

@DataJpaTest
@AutoConfigureTestDatabase(replace = Replace.NONE)
public class ProductoTest {

    @Autowired
    private ProductoDao productoDao;

    @Test
    @Rollback(false)
    public void testGuardarProducto() {
        Producto producto = new Producto("TV Samsung HD",4000);
        Producto productoGuardado = productoDao.save(producto);

        assertNotNull(productoGuardado);
    }

}
```

# Ejecutamos el test



# Verificamos el registro en la base de datos



# Agregamos este método al repositorio

formacionspringboot/app/dao/ProductoDao.java - Eclipse IDE

n Window Help

ProductoTest.java | ProductoDao.java ×

```
1  package com.formacionspringboot.app.dao;
2
3  import org.springframework.data.repository.CrudRepository;
5
6  public interface ProductoDao extends CrudRepository<Producto, Long>{
7
8      public Producto findByNombre(String nombre);
9
10 }
11
```

# Agregamos el siguiente método para el test

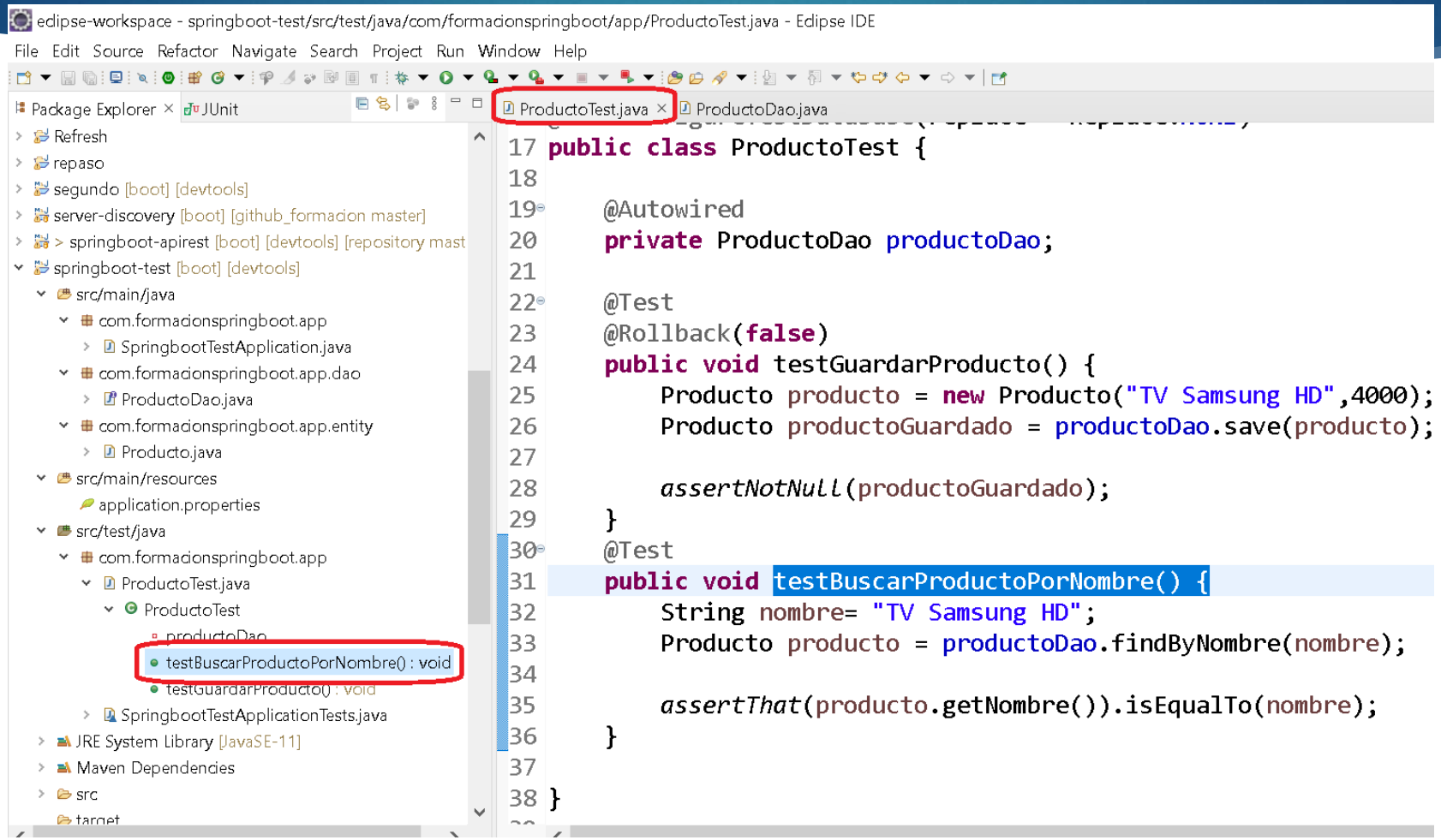
onspringboot/app/ProductoTest.java - Eclipse IDE

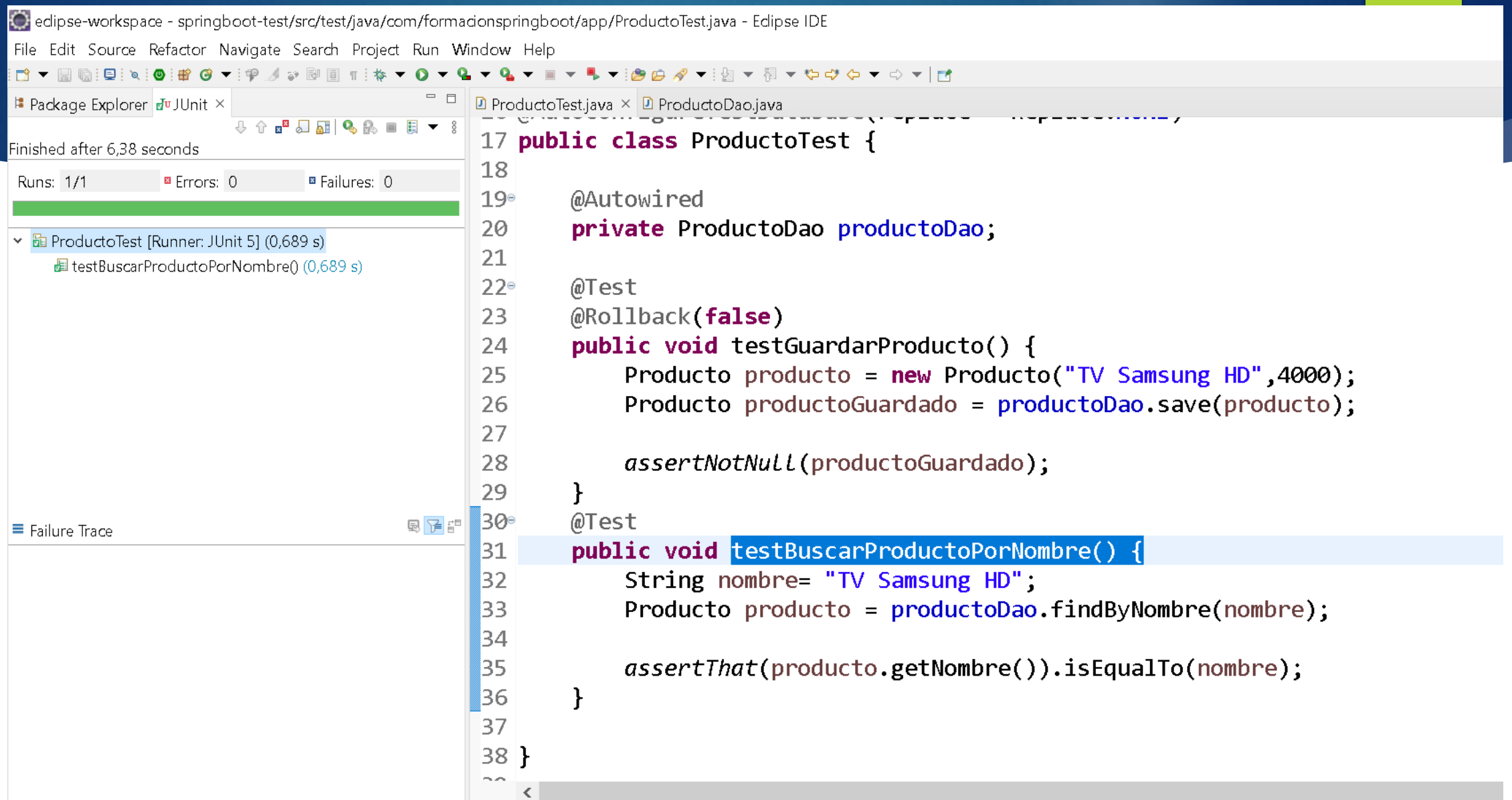
dow Help

ProductoTest.java × ProductoDao.java

```
17 public class ProductoTest {
18
19     @Autowired
20     private ProductoDao productoDao;
21
22     @Test
23     @Rollback(false)
24     public void testGuardarProducto() {
25         Producto producto = new Producto("TV Samsung HD",4000);
26         Producto productoGuardado = productoDao.save(producto);
27
28         assertNotNull(productoGuardado);
29     }
30     @Test
31     public void testBuscarProductoPorNombre() {
32         String nombre= "TV Samsung HD";
33         Producto producto = productoDao.findByNombre(nombre);
34
35         assertThat(producto.getNombre()).isEqualTo(nombre);
36     }
37
38 }
```

# Ejecutamos

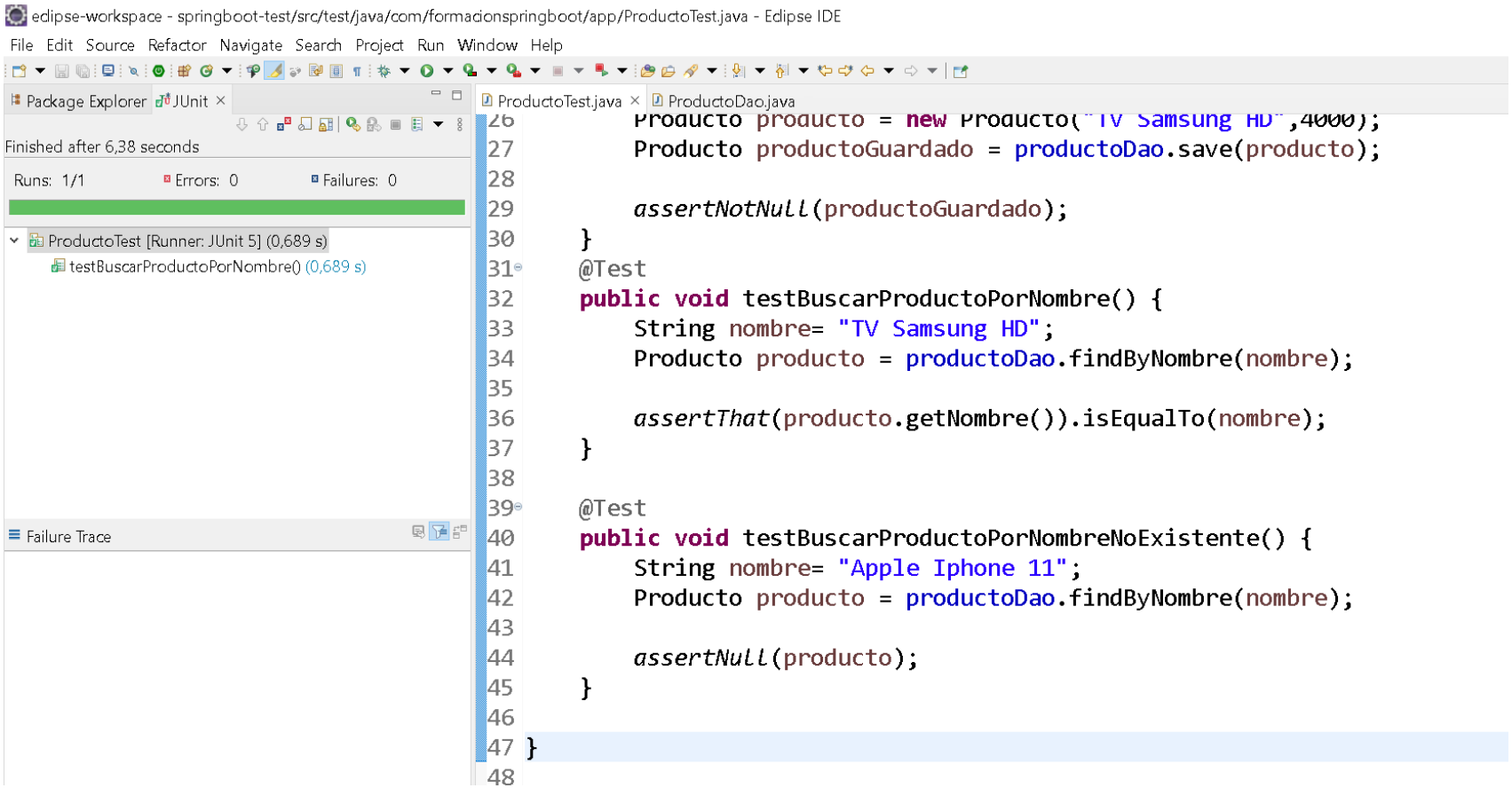

edipse-workspace - springboot-test/src/test/java/com/formacionspringboot/app/ProductoTest.java - Edipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
ProductoTest.java
ProductoDao.java
Refresh
repaso
segundo [boot] [devtools]
server-discovery [boot] [github_formacion master]
springboot-apirest [boot] [devtools] [repository mast
springboot-test [boot] [devtools]
src/main/java
com.formacionspringboot.app
SpringbootTestApplication.java
com.formacionspringboot.app.dao
ProductoDao.java
com.formacionspringboot.app.entity
Producto.java
src/main/resources
application.properties
src/test/java
com.formacionspringboot.app
ProductoTest.java
ProductoTest
testBuscarProductoPorNombre() : void
testGuardarProducto() : void
SpringbootTestApplicationTests.java
JRE System Library [JavaSE-11]
Maven Dependencies
src
target
17 public class ProductoTest {
18
19 @Autowired
20 private ProductoDao productoDao;
21
22 @Test
23 @Rollback(false)
24 public void testGuardarProducto() {
25 Producto producto = new Producto("TV Samsung HD",4000);
26 Producto productoGuardado = productoDao.save(producto);
27
28 assertNotNull(productoGuardado);
29 }
30 @Test
31 public void testBuscarProductoPorNombre() {
32 String nombre= "TV Samsung HD";
33 Producto producto = productoDao.findByNombre(nombre);
34
35 assertThat(producto.getNombre()).isEqualTo(nombre);
36 }
37
38 }


eclipse-workspace - springboot-test/src/test/java/com/formacionspringboot/app/ProductoTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 6,38 seconds
Runs: 1/1
Errors: 0
Failures: 0
ProductoTest [Runner: JUnit 5] (0,689 s)
testBuscarProductoPorNombre() (0,689 s)
Failure Trace
ProductoTest.java
ProductoDao.java
17 public class ProductoTest {
18
19 @Autowired
20 private ProductoDao productoDao;
21
22 @Test
23 @Rollback(false)
24 public void testGuardarProducto() {
25 Producto producto = new Producto("TV Samsung HD",4000);
26 Producto productoGuardado = productoDao.save(producto);
27
28 assertNotNull(productoGuardado);
29 }
30 @Test
31 public void testBuscarProductoPorNombre() {
32 String nombre= "TV Samsung HD";
33 Producto producto = productoDao.findByNombre(nombre);
34
35 assertThat(producto.getNombre()).isEqualTo(nombre);
36 }
37
38 }

ProductoTest.java × ProductoDao.java

```
26          Producto producto = new Producto("TV Samsung HD",4000);
27          Producto productoGuardado = productoDao.save(producto);
28
29          assertNotNull(productoGuardado);
30      }
31      @Test
32      public void testBuscarProductoPorNombre() {
33          String nombre= "TV Samsung HD";
34          Producto producto = productoDao.findByNombre(nombre);
35
36          assertThat(producto.getNombre()).isEqualTo(nombre);
37      }
38
39      @Test
40      public void testBuscarProductoPorNombreNoExistente() {
41          String nombre= "Apple Iphone 11";
42          Producto producto = productoDao.findByNombre(nombre);
43
44          assertNull(producto);
45      }
46
47  }
48
```

# ejecutamos

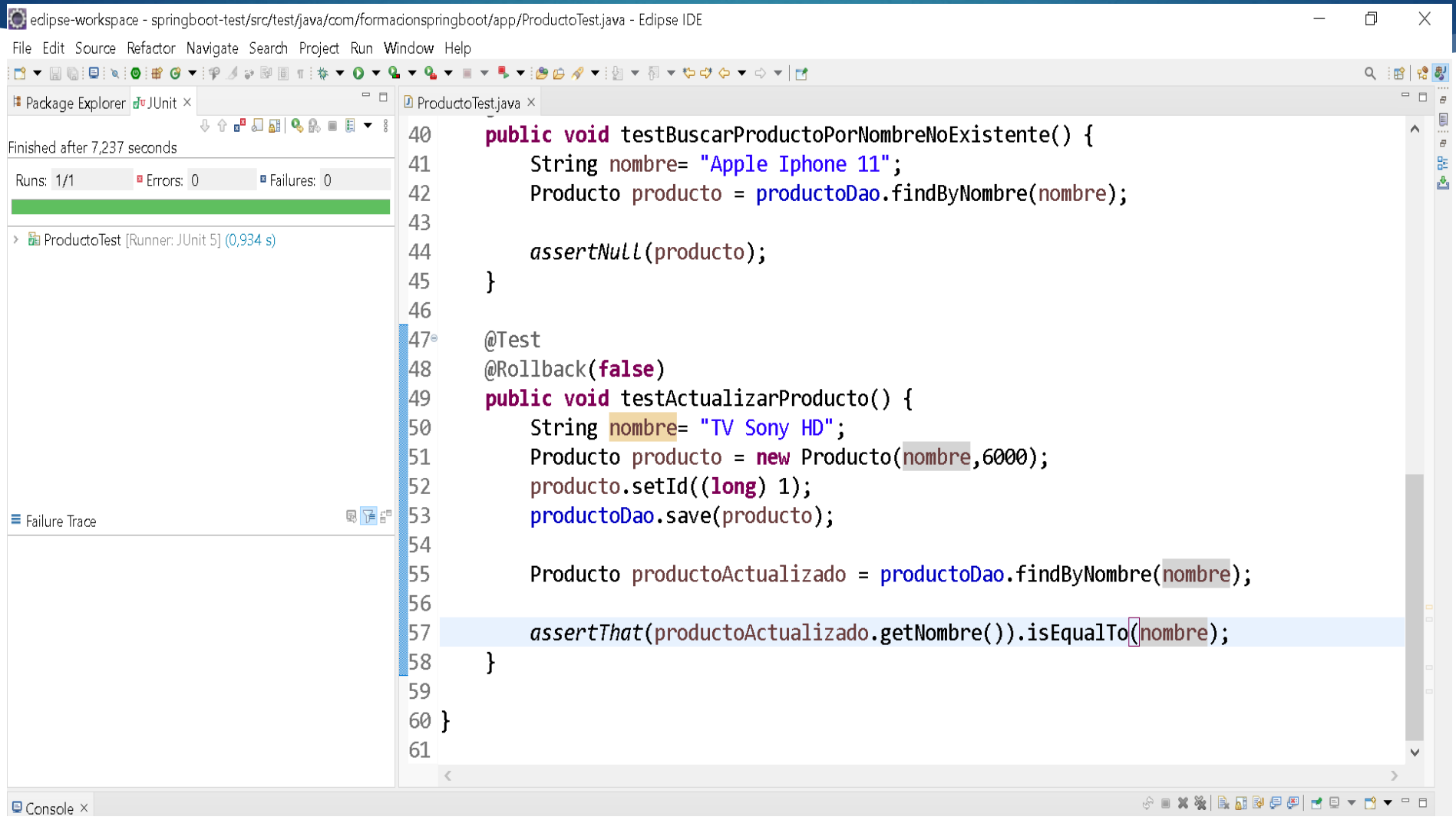

eclipse-workspace - springboot-test/src/test/java/com/formacionspringboot/app/ProductoTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 6,38 seconds
Runs: 1/1
Errors: 0
Failures: 0
ProductoTest [Runner: JUnit 5] (0,689 s)
testBuscarProductoPorNombre() (0,689 s)
Failure Trace
ProductoTest.java
ProductoDao.java
Producto producto = new Producto("TV Samsung HD",4000);
Producto productoGuardado = productoDao.save(producto);
assertNotNull(productoGuardado);
}
@Test
public void testBuscarProductoPorNombre() {
String nombre= "TV Samsung HD";
Producto producto = productoDao.findByNombre(nombre);
assertThat(producto.getNombre()).isEqualTo(nombre);
}
@Test
public void testBuscarProductoPorNombreNoExistente() {
String nombre= "Apple Iphone 11";
Producto producto = productoDao.findByNombre(nombre);
assertNull(producto);
}
}

# Método actualizar

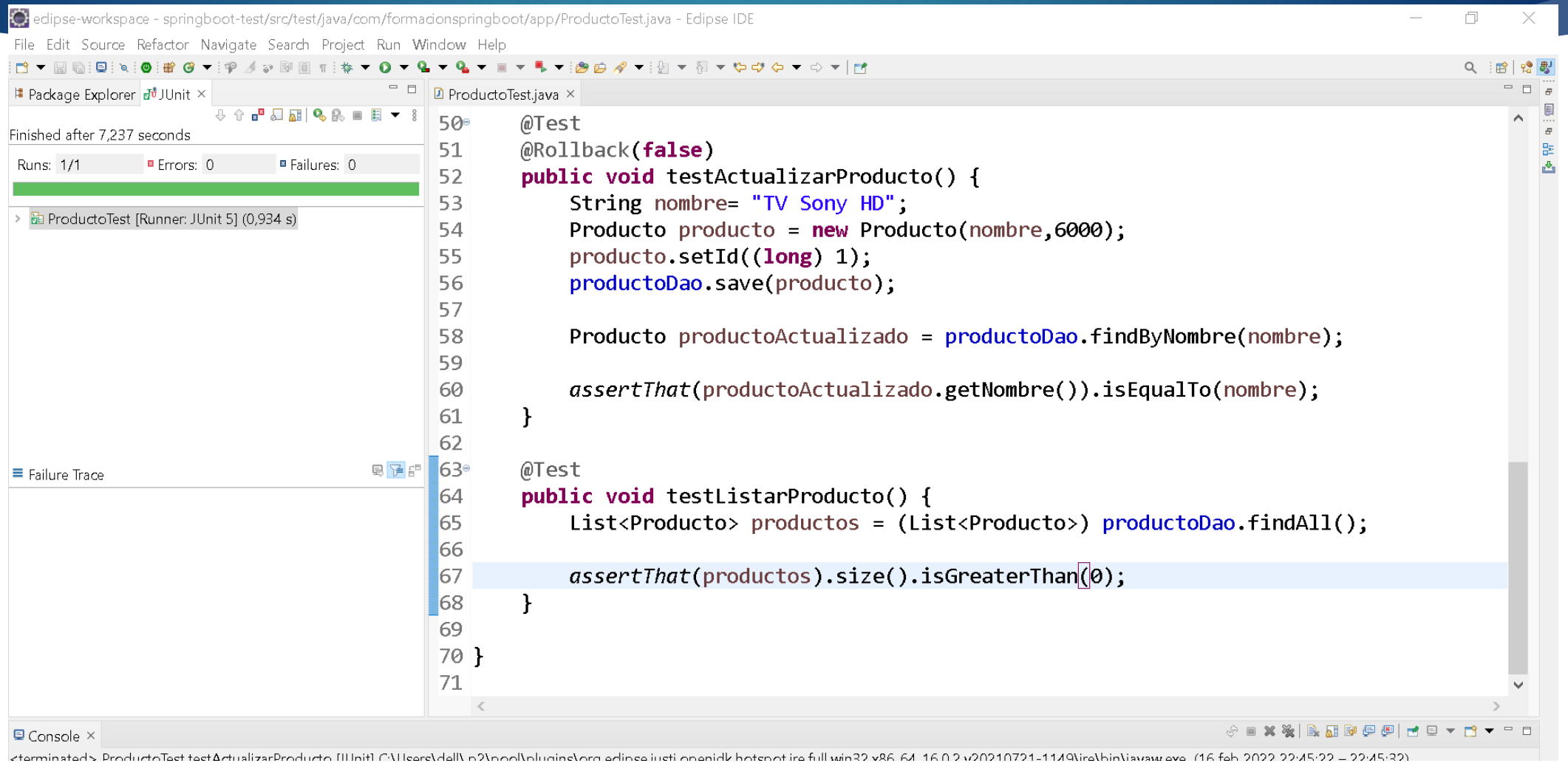
roductoTest.java ×

```
    public void testBuscarProductoPorNombreNoExistente() {
        String nombre= "Apple Iphone 11";
        Producto producto = productoDao.findByNombre(nombre);

        assertNull(producto);
    }

    @Test
    @Rollback(false)
    public void testActualizarProducto() {
        String nombre= "TV Sony HD";
        Producto producto = new Producto(nombre,6000);
        producto.setId((long) 1);
        productoDao.save(producto);

        Producto productoActualizado = productoDao.findByNombre(nombre);

        assertThat(productoActualizado.getNombre()).isEqualTo(nombre);
    }

}
```

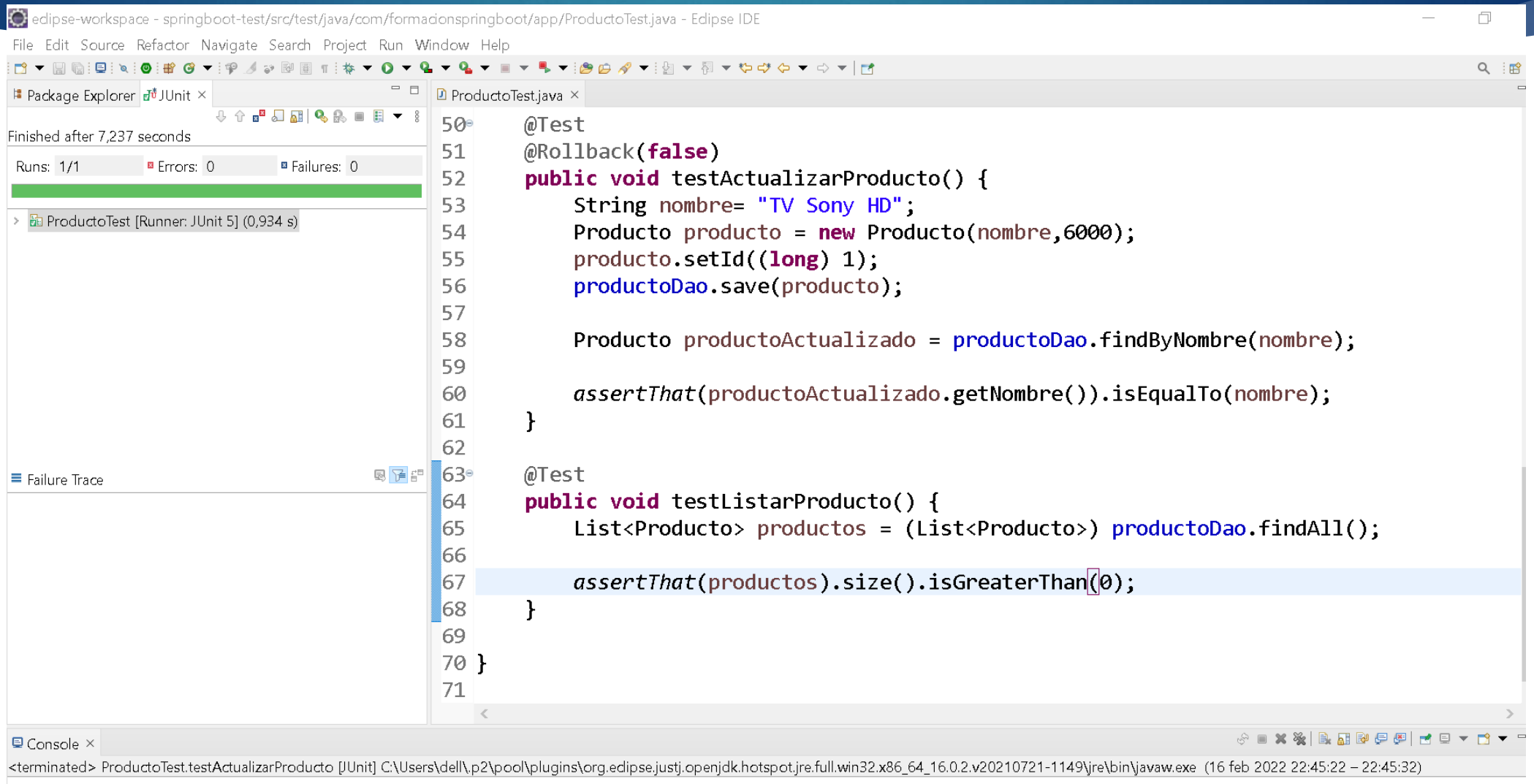
# Ejecutamos


edipse-workspace - springboot-test/src/test/java/com/formacionspringboot/app/ProductoTest.java - Edipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 7,237 seconds
Runs: 1/1
Errors: 0
Failures: 0
ProductoTest [Runner: JUnit 5] (0,934 s)
Failure Trace
ProductoTest.java
40 public void testBuscarProductoPorNombreNoExistente() {
41 String nombre= "Apple Iphone 11";
42 Producto producto = productoDao.findByNombre(nombre);
43
44 assertNull(producto);
45 }
46
47 @Test
48 @Rollback(false)
49 public void testActualizarProducto() {
50 String nombre= "TV Sony HD";
51 Producto producto = new Producto(nombre,6000);
52 producto.setId((long) 1);
53 productoDao.save(producto);
54
55 Producto productoActualizado = productoDao.findByNombre(nombre);
56
57 assertThat(productoActualizado.getNombre()).isEqualTo(nombre);
58 }
59
60 }
61
Console

# Método Listar



# Ejecutamos

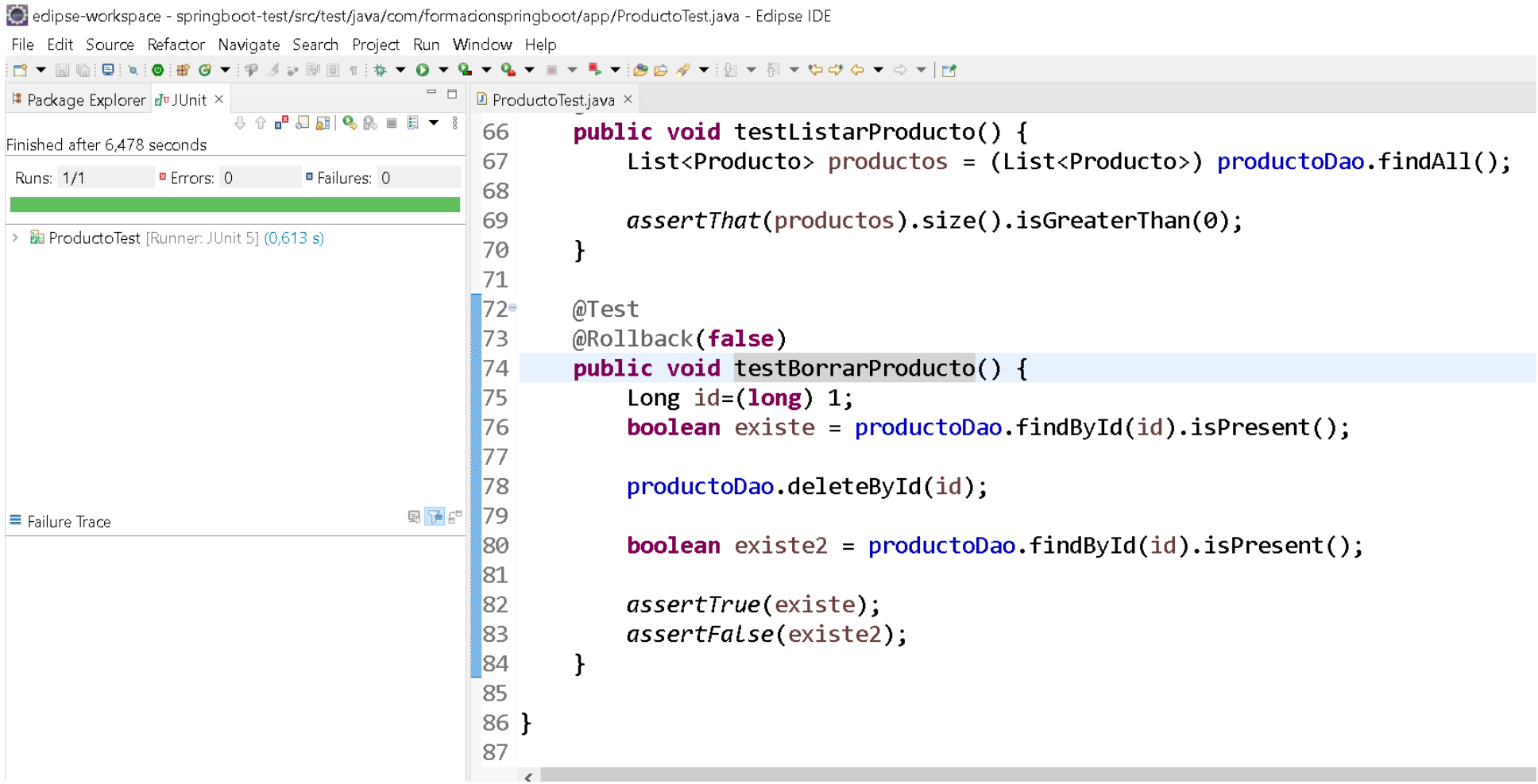

edipse-workspace - springboot-test/src/test/java/com/formacionspringboot/app/ProductoTest.java - Edipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 7,237 seconds
Runs: 1/1
Errors: 0
Failures: 0
ProductoTest [Runner: JUnit 5] (0,934 s)
Failure Trace
ProductoTest.java
50 @Test
51 @Rollback(false)
52 public void testActualizarProducto() {
53 String nombre= "TV Sony HD";
54 Producto producto = new Producto(nombre,6000);
55 producto.setId((long) 1);
56 productoDao.save(producto);
57
58 Producto productoActualizado = productoDao.findByNombre(nombre);
59
60 assertThat(productoActualizado.getNombre()).isEqualTo(nombre);
61 }
62
63 @Test
64 public void testListarProducto() {
65 List<Producto> productos = (List<Producto>) productoDao.findAll();
66
67 assertThat(productos).size().isGreaterThan(0);
68 }
69
70 }
71
Console
<terminated> ProductoTest.testActualizarProducto [JUnit] C:\Users\dell\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe (16 feb 2022 22:45:22 – 22:45:32)

# Método eliminar

ProductoTest.java

```
66      public void testListarProducto() {
67          List<Producto> productos = (List<Producto>) productoDao.findAll();
68
69          assertThat(productos).size().isGreaterThan(0);
70      }
71
72      @Test
73      @Rollback(false)
74      public void testBorrarProducto() {
75          Long id=(long) 1;
76          boolean existe = productoDao.findById(id).isPresent();
77
78          productoDao.deleteById(id);
79
80          boolean existe2 = productoDao.findById(id).isPresent();
81
82          assertTrue(existe);
83          assertFalse(existe2);
84      }
85
86  }
87
```

# Ejecutamos


eclipse-workspace - springboot-test/src/test/java/com/formacionspringboot/app/ProductoTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 6,478 seconds
Runs: 1/1
Errors: 0
Failures: 0
ProductoTest [Runner: JUnit 5] (0,613 s)
Failure Trace
ProductoTest.java
66 public void testListarProducto() {
67 List<Producto> productos = (List<Producto>) productoDao.findAll();
68
69 assertThat(productos).size().isGreaterThan(0);
70 }
71
72 @Test
73 @Rollback(false)
74 public void testBorrarProducto() {
75 Long id=(long) 1;
76 boolean existe = productoDao.findById(id).isPresent();
77
78 productoDao.deleteById(id);
79
80 boolean existe2 = productoDao.findById(id).isPresent();
81
82 assertTrue(existe);
83 assertFalse(existe2);
84 }
85
86 }
87