

**Tipo** : Guía de laboratorio  
**Capítulo** : Spring MVC  
**Duración** : 50 minutos

---

## I. OBJETIVO

Elaborar el proyecto de Spring, configurarlo y crear la primera pantalla.

## II. REQUISITOS

Los siguientes elementos de software son necesarios para la realización del laboratorio.

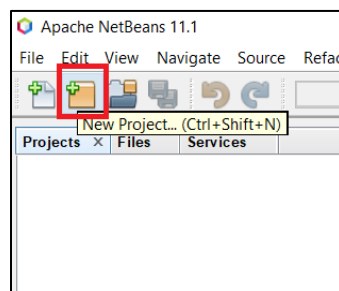
- NetBeans 11
- JDK 11

## III. EJECUCIÓN DEL LABORATORIO

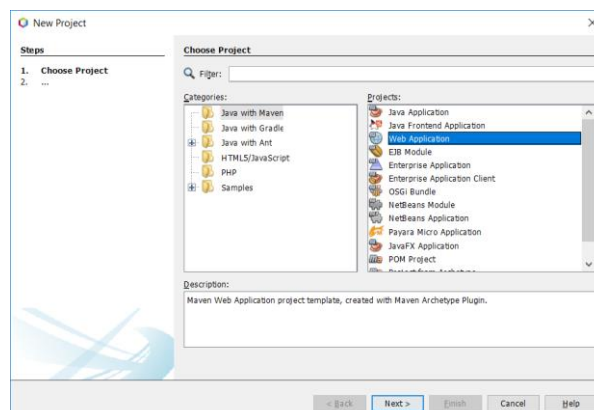
### Ejercicio:

Crea el proyecto con Spring, configura el entorno y empieza con el desarrollo web.

1. Abre el NetBeans 11 y presiona sobre el botón **“New Project”** (en la parte superior izquierda).

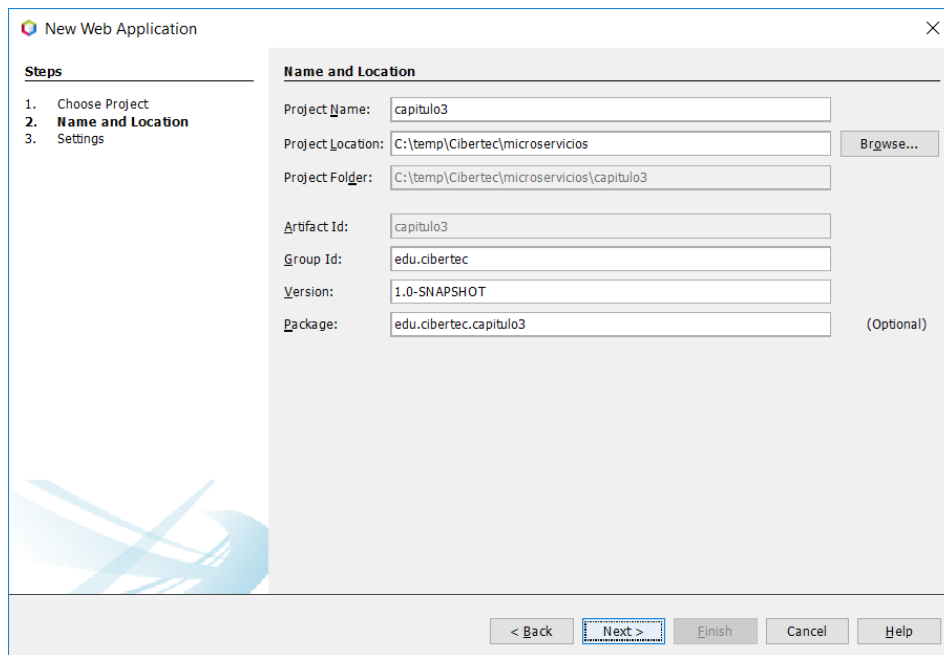


2. Selecciona en **“categorías”** la opción **“Java with maven”** y en **“Projects”** la opción **“Web Application”**.



3. En los datos que solicita el Netbeans, coloca lo siguiente.

- **Nombre del proyecto:** capitulo3
- **Localización del proyecto:** <carpeta de trabajo del alumno>

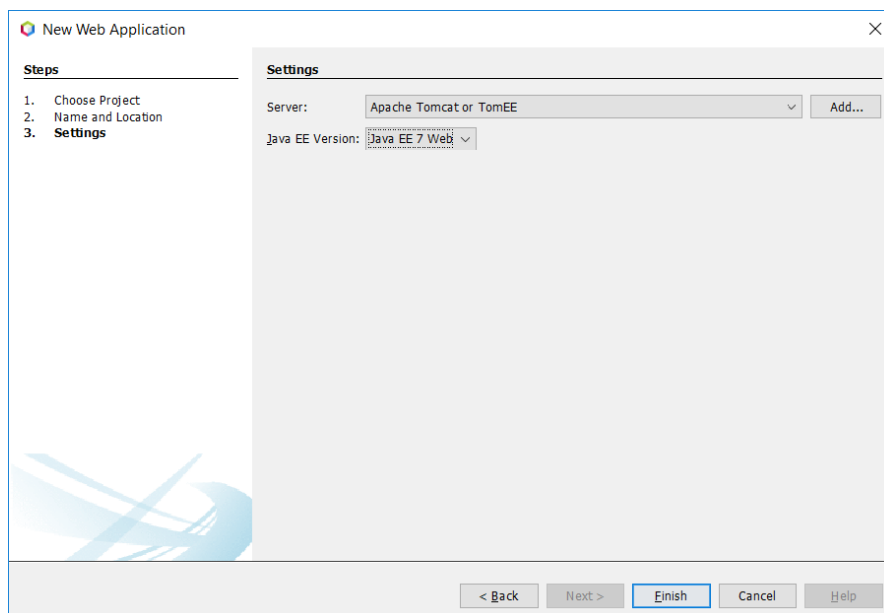


The screenshot shows the 'New Web Application' dialog box in NetBeans. The 'Steps' panel on the left indicates the current step is '2. Name and Location'. The 'Name and Location' tab is active, displaying the following fields:

- Project Name: capitulo3
- Project Location: C:\temp\Cibertec\microservicios (with a 'Browse...' button)
- Project Folder: C:\temp\Cibertec\microservicios\capitulo3
- Artifact Id: capitulo3
- Group Id: edu.cibertec
- Version: 1.0-SNAPSHOT
- Package: edu.cibertec.capitulo3 (marked as '(Optional)')

At the bottom, there are buttons for '< Back', 'Next >' (highlighted with a dashed border), 'Finish', 'Cancel', and 'Help'.

4. Al presionar sobre el botón “**Next**”, el IDE solicita el servidor de aplicaciones a utilizar y la versión del Java Web. Selecciona el **Tomcat** y la **versión 7 de Java**.

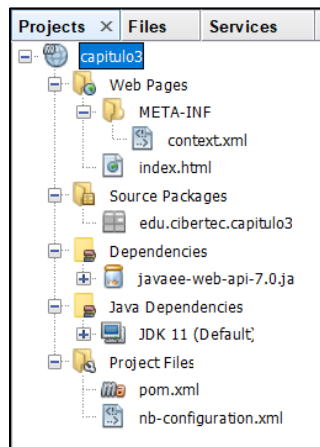


The screenshot shows the 'New Web Application' dialog box in NetBeans, now on the 'Settings' tab. The 'Steps' panel on the left indicates the current step is '3. Settings'. The 'Settings' tab displays the following configuration:

- Server: Apache Tomcat or TomEE (with an 'Add...' button)
- Java EE Version: Java EE 7 Web (with a dropdown arrow)

At the bottom, the buttons are '< Back', 'Next >', 'Finish' (highlighted with a blue border), 'Cancel', and 'Help'.

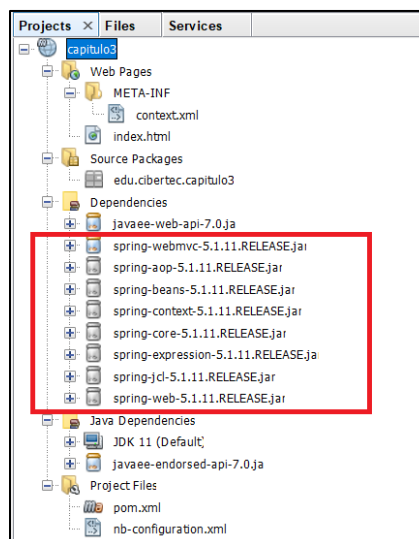
5. Después de presionar sobre el botón **“finish”** se puede observar el proyecto creado dentro del IDE.



6. El proyecto creado no tiene instalado Spring, por lo cual procede a colocar la dependencia en Maven para que administre la librería. Para esto abre el archivo **“pom.xml”** y añade las siguientes líneas.

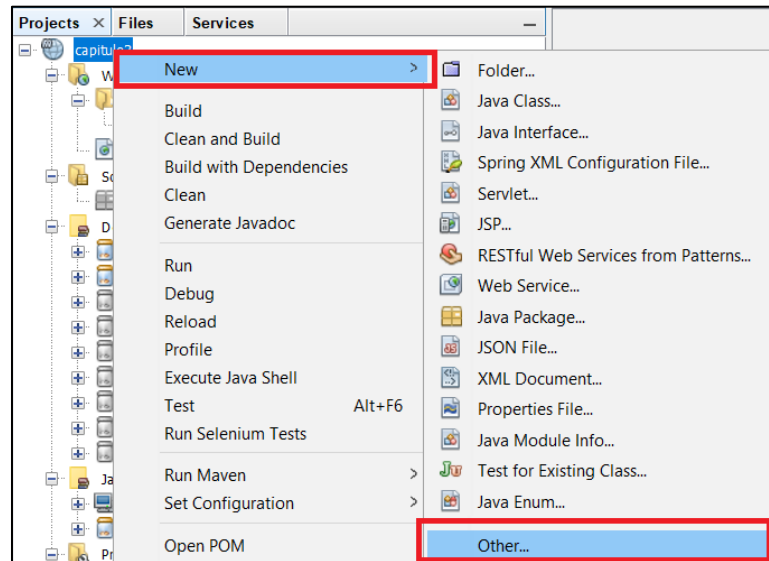
```
<dependencies>
  <dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-web-api</artifactId>
    <version>7.0</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-webmvc</artifactId>
    <version>5.1.11.RELEASE</version>
    <scope>compile</scope>
  </dependency>
</dependencies>
```

Realiza un **“Clean and build”** del proyecto y se podrán apreciar los archivos \*.jar que se han descargado para Spring.

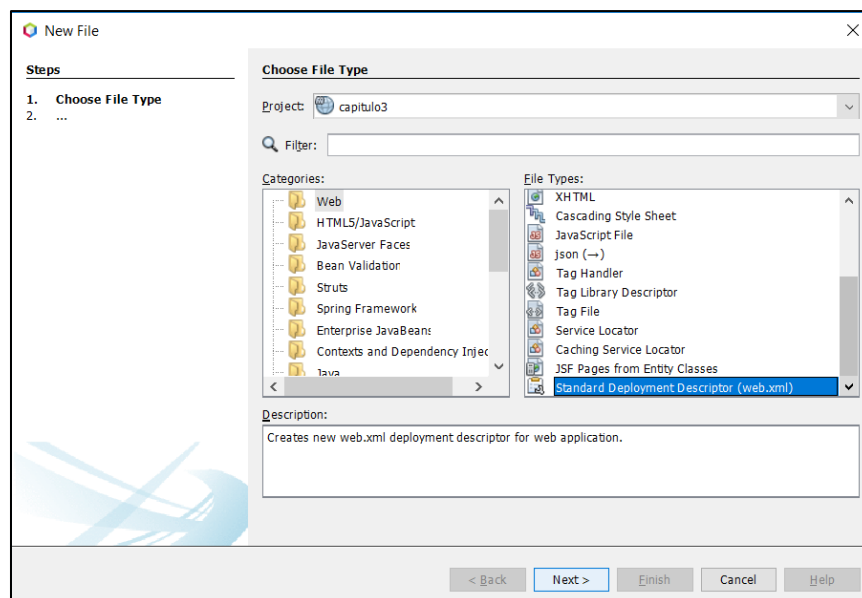


7. Como siguiente paso, configura el **DispatcherServlet** para que sea el único que reciba los requerimientos de los clientes.

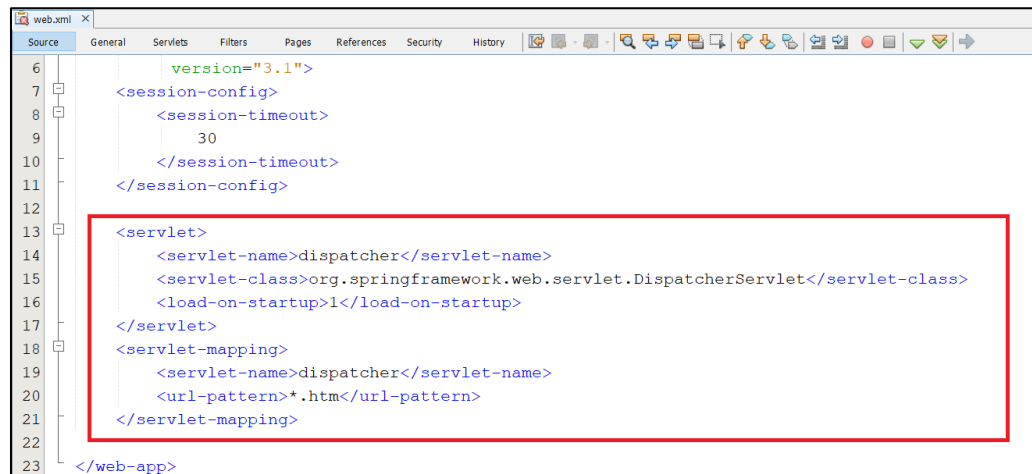
Procede a crear el archivo configurador de todo proyecto web llamado “**web.xml**”. Para realizar esto, haz clic derecho sobre el proyecto y selecciona la opción “**New**” -> “**Other**”.



Cuando aparece el diálogo de “Choose File Type” selecciona la Categoría “**Web**” y el tipo de archivo “**Standard Deployment Descriptor (web.xml)**”.



Presiona sobre el botón **“Next”** y luego sobre el botón **“Finish”** para que se cree el archivo. Abre el archivo y coloca las siguientes líneas para la inscripción del Servlet de Spring.



```
<?xml version="3.1">
<session-config>
  <session-timeout>
    30
  </session-timeout>
</session-config>

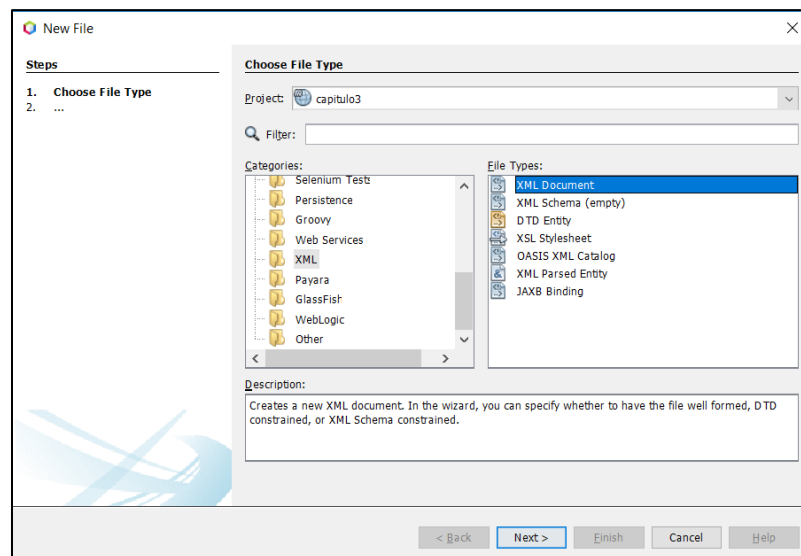
<servlet>
  <servlet-name>dispatcher</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
  <servlet-name>dispatcher</servlet-name>
  <url-pattern>*.htm</url-pattern>
</servlet-mapping>
</web-app>
```

8. Ahora se deberá crear el archivo que configure a este Servlet. Si no se indica un archivo en específico, por defecto, Spring buscará un archivo con el nombre **“WEB-INF/<NOMBRE-SERVLET>-servlet.xml”**, donde **NOMBRE-SERVLET** es lo que se colocó dentro de la etiqueta **“<servlet-name>”** del web.xml. En este caso, el archivo se deberá llamar **“WEB-INF/dispatcher-servlet.xml”**.

Procede a crearlo, para ello haz clic derecho sobre la carpeta **“WEB-INF”** y selecciona la opción **“New”** -> **“Other”**.

Selecciona la categoría **“XML”** y el tipo de archivo **“XML Document”**.



Coloca como nombre de archivo **“dispatcher-servlet”**.

New XML Document

**Steps**

1. Choose File Type
2. **Name and Location**
3. Select Document Type
4. ...

**Name and Location**

File Name:

Project:

Folder:

Created File:

< Back **Next >** Finish Cancel Help

Presiona sobre el botón **“Next”** y en el siguiente cuadro selecciona en el botón **“Finish”**.

New File

**Steps**

1. Choose File Type
2. Name and Location
3. **Select Document Type**
4. ...

**Select Document Type**

Select the type of XML document you want to create based on your document structure, data types, and namespace requirements.

☒ Well-formed Document

☐ DTD-Constrained Document

☐ XML Schema-Constrained Document

< Back Next > **Finish** Cancel Help

9. Una vez creado el archivo, procede a completarlo de la siguiente manera.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:context="http://www.springframework.org/schema/context"
  xmlns:mvc="http://www.springframework.org/schema/mvc"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd
    http://www.springframework.org/schema/context
    http://www.springframework.org/schema/context/spring-context.xsd
    http://www.springframework.org/schema/mvc
    http://www.springframework.org/schema/mvc/spring-mvc.xsd">

  <context:component-scan base-package="edu.cibertec.capitulo3" />

</beans>
```

Como se puede apreciar, la única configuración realizada es que busque los componentes de Spring en la carpeta por defecto.

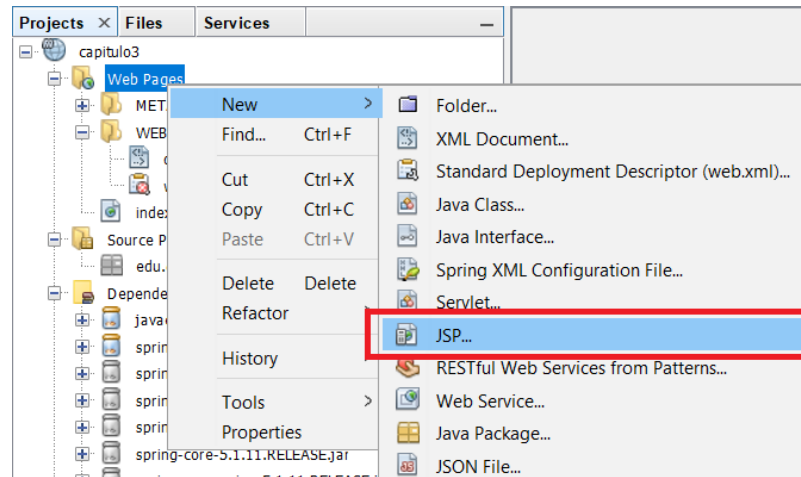
10. Ahora, vas a hacer que la página inicial del proyecto invoque a un primer Controlador y muestre una página de “saludo” ya manejada por Spring.

Para realizar esto, agrega la siguiente línea al archivo index.html que se creó automáticamente por el NetBeans.

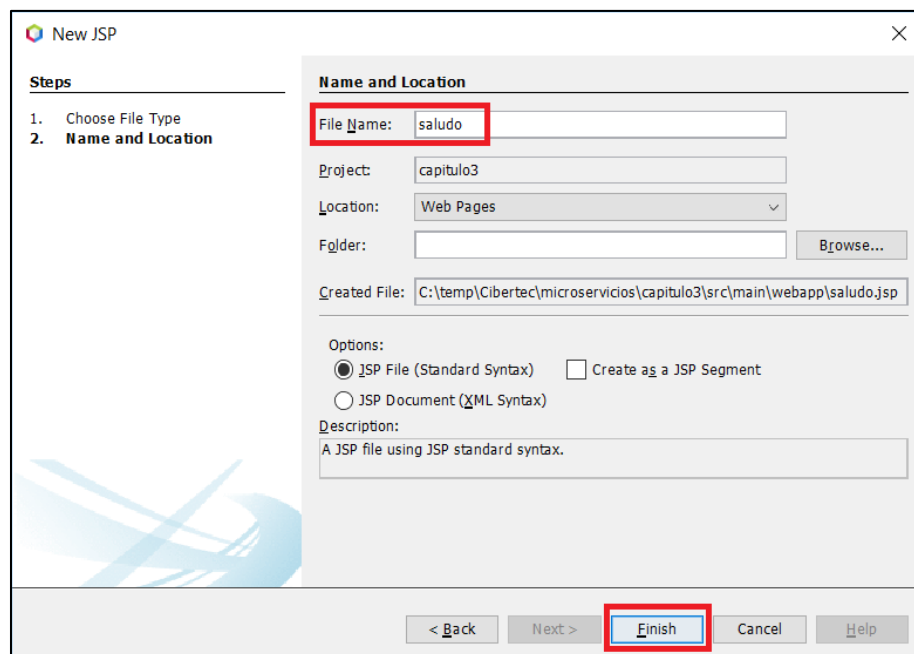
```
<!DOCTYPE html>
<html>
  <head>
    <title>Start Page</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  </head>
  <body>
    <SCRIPT type="text/javascript" >
      window.location.replace("saludo.htm");
    </SCRIPT>
  </body>
</html>
```

11. Crea una página llamada **“saludo.jsp”** en el directorio **“Web Pages”** que simplemente muestre un saludo.

Clic derecho sobre la carpeta **“Web Pages”** -> **“New”** -> **“JSP”**.



Coloca el nombre de **“saludo”** y termina la creación.





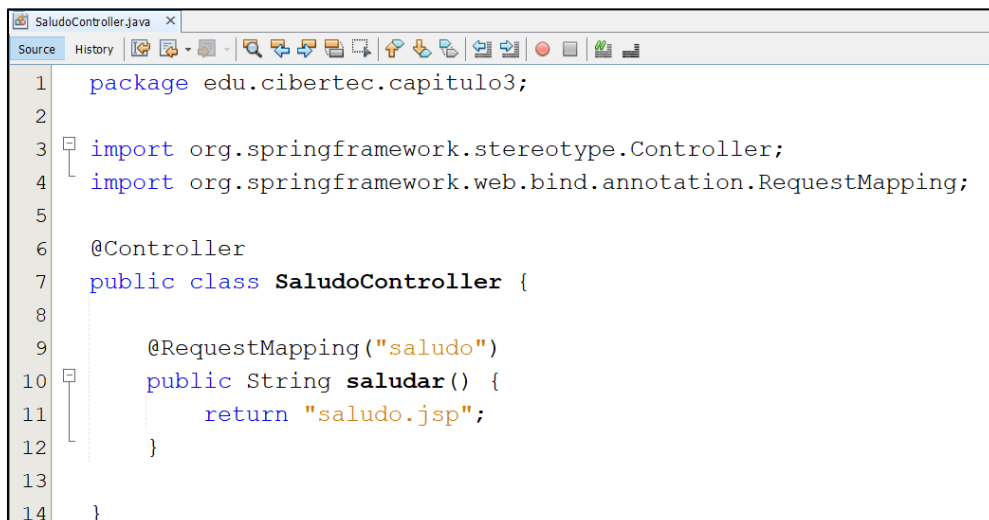
Modifica el contenido del archivo para mostrar un mensaje de saludo.



```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6 <title>Primera Aplicación Spring MVC</title>
7 </head>
8 <body>
9 <h1>Bienvenido a MVC en Spring</h1>
10 </body>
11 </html>
```

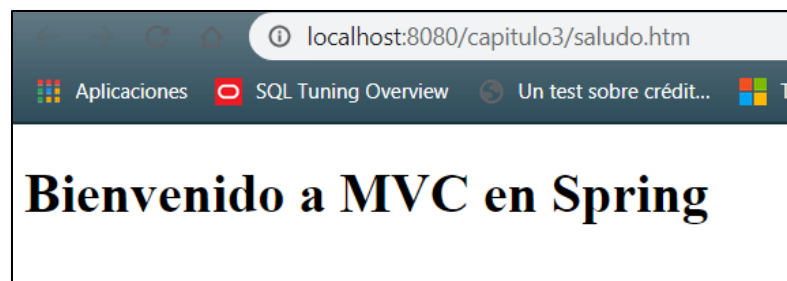
12. Lo que queda por crear es el Controlador que capture el llamado de la página inicial (index.html) y que presente la página de saludo (saludo.jsp).

Crea una clase llamada “**SaludoController**” dentro del paquete por defecto “**edu.cibertec.capitulo3**” con la siguiente implementación.



```
1 package edu.cibertec.capitulo3;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.RequestMapping;
5
6 @Controller
7 public class SaludoController {
8
9     @RequestMapping("saludo")
10    public String saludar() {
11        return "saludo.jsp";
12    }
13
14 }
```

13. Ejecuta el proyecto y obtén la siguiente respuesta.



#### IV. EVALUACIÓN

**Responde la siguiente pregunta.**

1. ¿Cuáles son las anotaciones con las que se puede configurar un proyecto MVC?

Las principales son:

- @Configuration
- @ComponentScan
- @EnableWebMvc

También hay que diseñar una clase que extienda de `AbstractAnnotationConfigDispatcherServletInitializer`.