**Máster Universitario en Ingeniería del Software para la Web**



**Trabajo Intermedio Asignatura:**

Herramientas de Desarrollo Web

**Profesores:**

Dr. D. Salvador Otón Tortosa – [salvador.oton@uah.es](mailto:salvador.oton@uah.es)

Dr. D. Roberto Barchino Plata – [roberto.barchino@uah.es](mailto:roberto.barchino@uah.es)

**Alumno:**

Piero Rospigliosi Beltrán – [piero.rospigliosi@uah.es](mailto:piero.rospigliosi@uah.es), [pierinho13@gmail.com](mailto:pierinho13@gmail.com)

**SOLUCION ENCONTRADA:**

Para realizar este trabajo se ha trabajado en Spring Framework que está basado en Eclipse y nos permite trabajar con proyectos Maven con dependencias de Spring, como Spring MVC por ejemplo.

Se han utilizado las siguientes tecnologías:

Maven, que es una herramienta para la gestión construcción de proyectos Java.

Spring MVC, Spring Security (para la gestión de los usuarios).

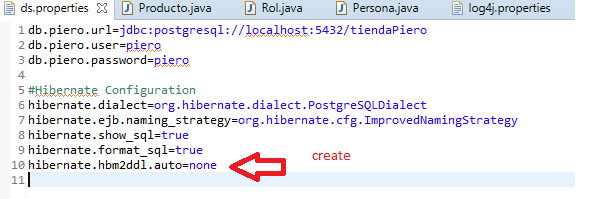
Spring JPA-DATA.

En el front, JSP.

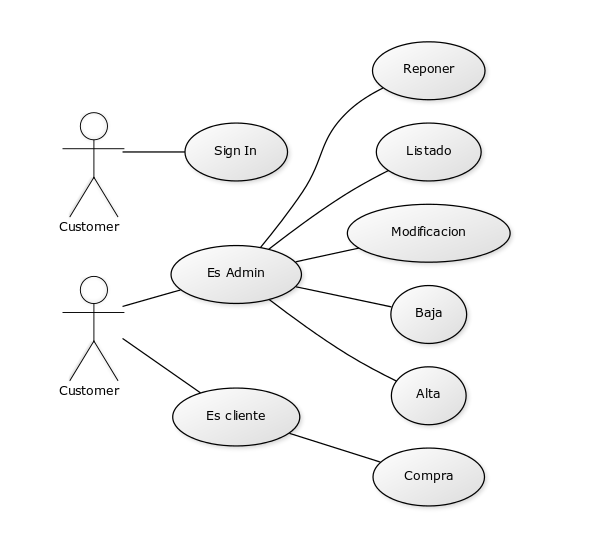
Postgres SQL.

Se ha adjuntado el archivo SCRIPT.SQL, el proyecto Maven a importar (si ese fuera el caso), y el WAR llamado ROOT.  
Para ejecutar la aplicación debemos ejecutarlo en localhost:8080 (dependiendo de tu número de puerto, pero hago énfasis que se abre en la raíz).

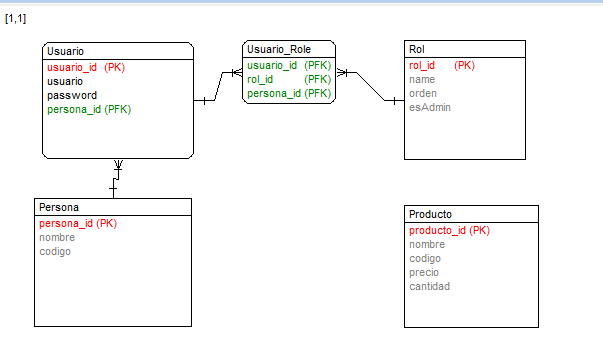
La aplicación también ha configurado para que si quisiéramos, solo ejecutar el script de crear base de datos y usuario. Y que Hibernate sea quien cree tablas y realice los insert del archivo import.sql del proyecto. Para ello tendríamos que cambiar el valor que podemos ver abajo por créate.



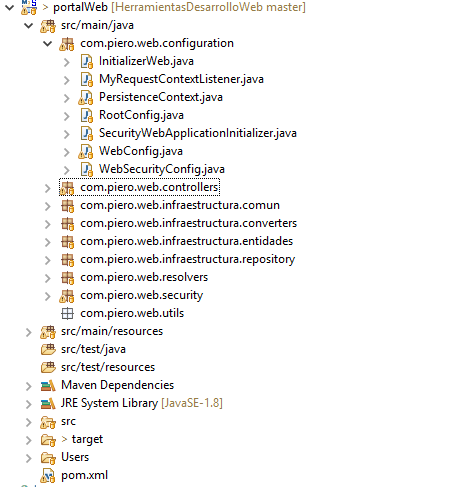
**CASOS DE USO:**



**DIAGRAMA ENTIDAD RELACIÓN:**



Esta es la estructura de nuestro proyecto Maven.



**Los usuarios de la aplicación son:**

Para el ROL ADMIN : admin 1234

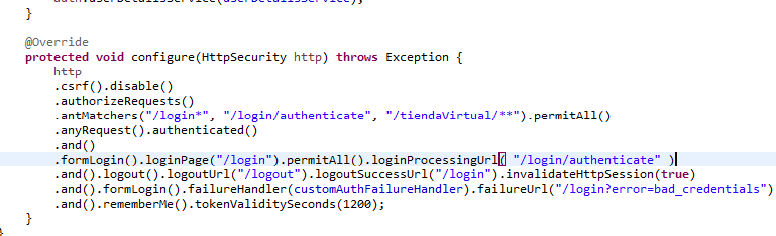
Para el ROL CLIENTE : cliente 1234

Donde admin y cliente son los usuarios y 1234 las contraseñas.

Esta es nuestra pantalla de login:

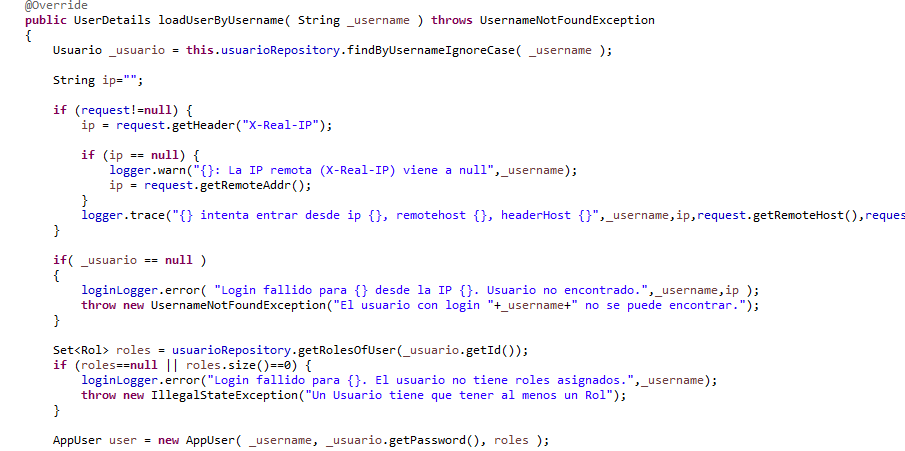


El cual se ha configurado mediante Spring Security:

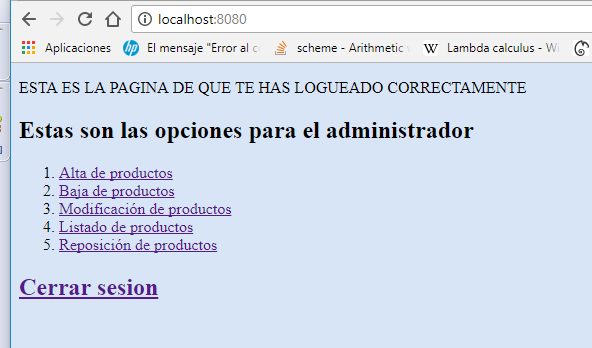


En esta pantalla podemos ver como busca en la tabla usuario de nuestra base de datos.

Con la llegada de Spring Security 4, la comparación la realiza él si hemos creado correctamente nuestras tablas.



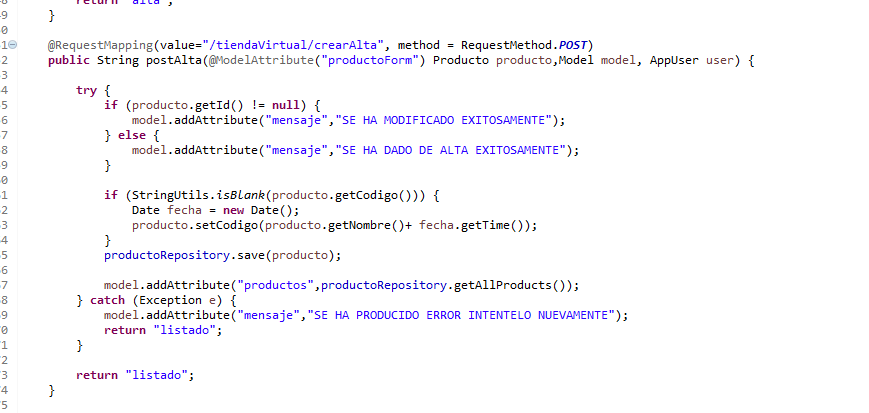
Esto es lo que ve un usuario ADMIN.



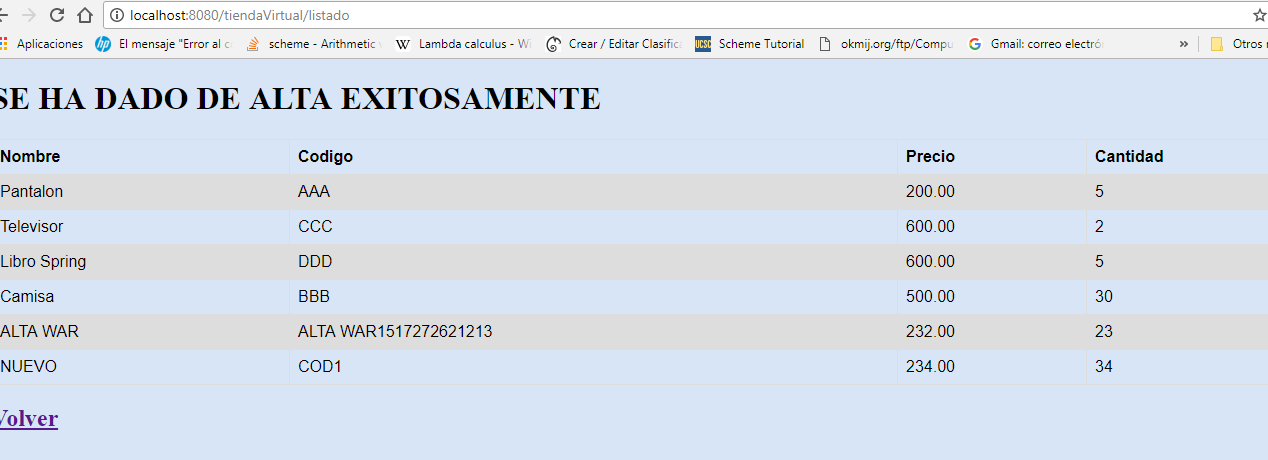
Esta es la pantalla de Alta.



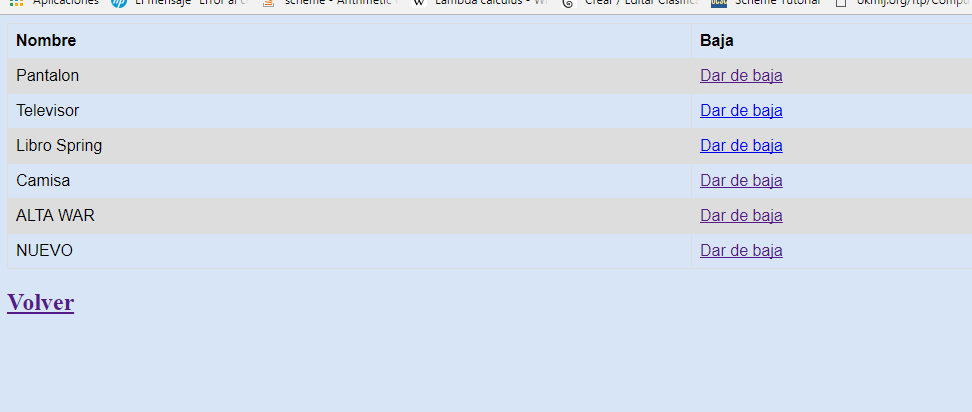
Que se realiza mediante un sencillo POST que es recibido por nuestro requestMapping de crear Alta y recibe de la vista lo enviado a través del uso de @ModelAttribute



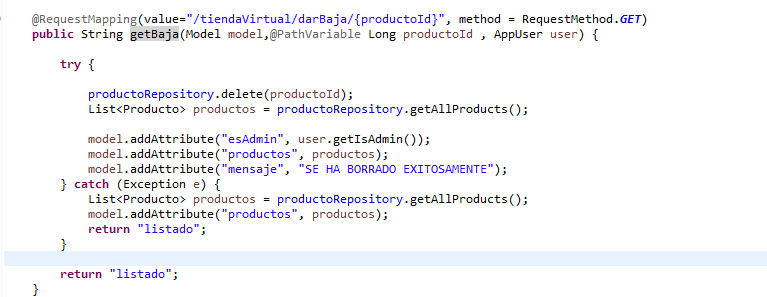
Esta es nuestra pantalla de listados que hemos reutilizado cuando damos de alta, baja y modifiación con distinto mensaje en cada caso.



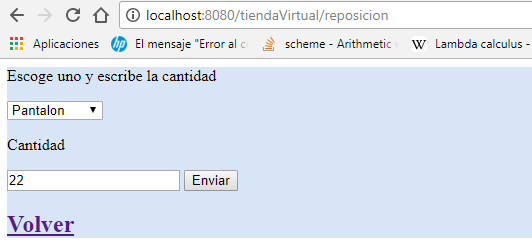
Esta es la pantalla de dar de baja.



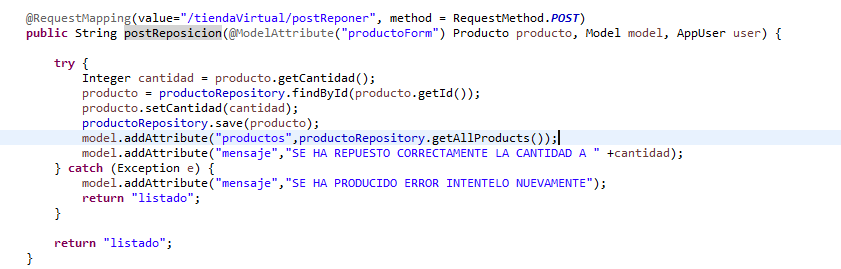
Para ello, mediante la inyección de un @Repository hacemos un delete del id que nos llega de la vista.



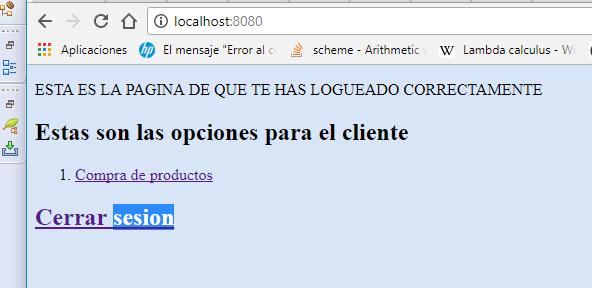
Esta es nuestra pantalla de reponer la cantidad.



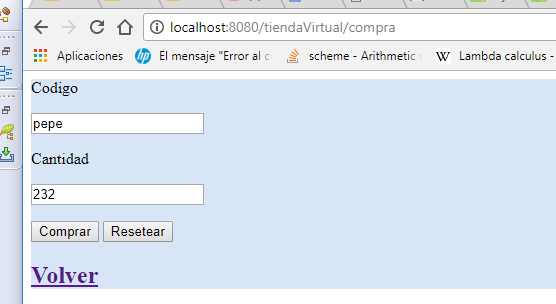
Aquí se muestra el código que es similar al de crear Alta.



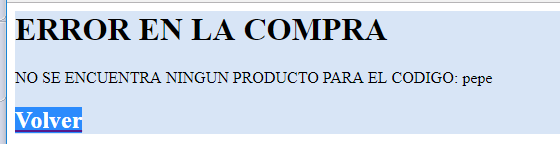
Esta es la pantalla de un usuario CLIENTE



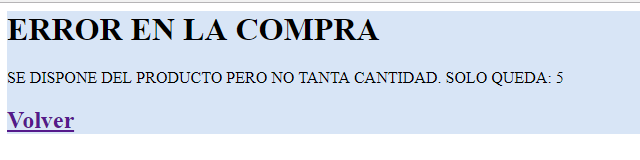
Ellos pueden pasar un código y una cantidad.



Si no existe el código nos lanzará mensaje de error.



Y también si no tenemos cantidad suficiente



Si existe el producto y hay suficiente cantidad entonces podremos comprar y se actualizará el producto en nuestra base de datos.