# Capítulo 3: Seguridad

En este capítulo abordaremos todo lo relacionado con la seguridad del sistema web. La seguridad web o seguridad de sitios web es la práctica de salvaguardar las redes, las comunicaciones en línea, el hardware y el software para que no sean manipulados o utilizados con fines maliciosos. En esta era creciente de ciberamenazas y vulnerabilidades, los principales objetivos son los sitios web[1]

## 3.1- Autenticación y Autorización

En las aplicaciones web, es imprescindible garantizar que solamente los usuarios autorizados y autentificados puedan acceder a las funciones y datos críticos del sistema, evitando así que visitantes no autorizados pueda acceder y hacer un mal uso de dicha información.

La autenticación es el proceso de identificar a los usuarios y garantizar que los mismos sean quienes dicen ser. Esto evita que cualquiera pueda entrar en un determinado sistema o iniciar sesión en alguna plataforma de forma indebida, sin que realmente sea el usuario legítimo que tiene el poder para hacerlo.[2]

La autorización es lo que define a qué recursos de sistema el usuario autenticado podrá acceder. Que haya logrado pasar la instancia de la autenticación, no significa que podrá utilizar el sistema por completo como super administrador. De acuerdo a una serie de reglas, normas y regulaciones propias de cada red interna, se determina que el usuario A tendrá acceso a los recursos X e Y. Sin embargo, el usuario B sólo podrá acceder al recurso Z.[2]

### 3.1.1- Spring Security

La seguridad de la aplicación se gestiona con Spring Security, un marco integral que proporciona un conjunto de herramientas para implementar autenticación, autorización, y protección de acceso a aplicaciones basadas en Java. Spring Security permite configurar políticas de seguridad a nivel de aplicación de manera flexible, adaptándose a diferentes necesidades y escenarios.

1. **Configuración General**: Spring Security intercepta todas las solicitudes HTTP entrantes y aplica reglas de autenticación y autorización. Permite definir qué rutas de la aplicación requieren autenticación, qué tipos de autenticación se utilizan, y qué acciones están permitidas para cada tipo de usuario.
2. **Filtros de Seguridad**: Spring Security proporciona una serie de filtros de seguridad que se ejecutan en cada solicitud para autenticar y autorizar usuarios. Estos filtros pueden ser personalizados para adaptarse a necesidades específicas, como la validación de tokens JWT.

### 3.1.2- Autenticación Sin Estado (Stateless)

Una de las principales características de una API REST es que el servicio no tiene estado (*stateless*), lo que implica que cada vez que se recurra a él será necesario recordarle nuestros datos, ya sean credenciales de usuario o cualquier otro tipo de información. Lo que por una parte podría parecer una desventaja, al implicar la tediosa tarea de repetir datos, es en realidad una de sus fortalezas: al no memorizarlos, permite una mayor escalabilidad. No harán falta servidores tan potentes, capaces de almacenar todos los estados de sus clientes.[3]

Este factor es especialmente relevante para cualquier API bancaria, como por ejemplo las de BBVA. Si el *fintech* aspira a atraer a todos los clientes de instituciones bancarias y lograr que los desarrolladores introduzcan en sus aplicaciones herramientas financieras, la escalabilidad de sus API es primordial.[3]

### 3.1.3- JSON Web Token

El token web JSON (JWT) es un estándar abierto (RFC 7519) que define un método compacto y autocontenido para la transmisión segura de información entre partes codificadas como un objeto JSON. Esta información puede verificarse y ser fiable porque está firmada digitalmente. Los JWT pueden firmarse utilizando un secreto (con el algoritmo HMAC) o un par de claves públicas/privadas utilizando RSA.[4]

Estos son algunas situaciones en las que los tokens web JSON son útiles:

* **Autenticación**: Esta es la situación típica para el uso de JWT, una vez que el usuario ha iniciado sesión, cada solicitud posterior incluirá el JWT, permitiendo al usuario acceder a las rutas, servicios y recursos que están permitidos con ese token. El inicio de sesión único es una característica que utiliza ampliamente JWT hoy en día, debido a su pequeña sobrecarga y su capacidad para ser utilizado fácilmente entre los sistemas de diferentes dominios.[4]
* **Intercambio de información**: Los JWT son una buena forma de transmitir información de forma segura entre las partes, porque como se pueden firmar, por ejemplo utilizando un par de claves públicas/privadas, se puede estar seguro de que el remitente es quien dice ser. Además, como la firma se calcula utilizando el encabezado y la carga útil, también puedes verificar que el contenido no ha cambiado.[4]

Spring Security proporciona un marco de trabajo robusto para aplicaciones web, que, junto con el uso de JSON Web Tokens (JWT) y la autenticación sin estado (Stateless Authentication), permite implementar de manera segura y eficiente los procesos de autenticación y autorización en la aplicación. Estas herramientas combinadas garantizan que solo los usuarios autorizados accedan a los recursos protegidos, cumpliendo con los más altos estándares de seguridad.

## 3.2- Configuración del uso compartido de recursos entre orígenes (CORS)