# Flujo de OAuth2 con JWT y Cookies Seguras

Este documento describe cómo implementar un flujo de OAuth2 utilizando JWT y cookies seguras en una aplicación con un frontend en React y un backend en Spring Boot.

## Resumen del Flujo de OAuth2

El flujo de OAuth2 en esta aplicación incluye la autenticación de usuarios, la generación y validación de tokens, y la gestión segura de estos mediante cookies seguras.

### Pasos del Flujo

1. 1. El cliente solicita el login (frontend): El usuario abre la interfaz de inicio de sesión en React y proporciona sus credenciales (usuario y contraseña).
2. 2. El cliente envía las credenciales al backend: Las credenciales se envían a un endpoint de autenticación en tu backend (por ejemplo, /api/auth/login) mediante una solicitud POST.
3. 3. El backend autentica al usuario: El backend verifica las credenciales contra su base de datos. Si son correctas, genera un access token (de corta duración, por ejemplo, 15 minutos) y un refresh token (de mayor duración, por ejemplo, 7 días), enviándolos en cookies seguras.
4. 4. El cliente recibe la cookie del token: El navegador almacena la cookie automáticamente, configurada como httpOnly y no accesible desde JavaScript.
5. 5. El cliente accede a recursos protegidos: Cada vez que el cliente realiza una solicitud a una API protegida, el navegador envía automáticamente la cookie del access token al backend, que valida el token y responde con el recurso solicitado o un error (401) si no es válido.
6. 6. Renovación del token con el refresh token: Cuando el access token expira, el frontend envía automáticamente el refresh token al backend para obtener un nuevo access token.
7. 7. Cierre de sesión (logout): Cuando el usuario cierra sesión, el backend invalida los tokens eliminándolos de la base de datos (si los almacenas) y envía una respuesta para eliminar las cookies en el navegador.

### Diagrama Simplificado del Flujo

Frontend (React) <=> Backend (Spring Boot)  
 ----------------------------------------------------------  
 1. Usuario: Inicia sesión en React  
 2. React: Envía credenciales al backend (/api/auth/login)  
 3. Backend: Verifica credenciales y genera JWTs (Access + Refresh Tokens)  
 4. Backend: Envía los tokens en cookies seguras (httpOnly)  
 5. React: Hace solicitudes a APIs protegidas, cookies envían tokens automáticamente  
 6. Backend: Valida el Access Token, responde con recursos o errores  
 7. Si el Access Token expira, React usa el Refresh Token para renovarlo  
 8. Logout: Backend invalida tokens y elimina cookies

## Ventajas del Enfoque

* Seguridad: Los tokens están protegidos en cookies httpOnly, lo que evita accesos no autorizados desde JavaScript (previene XSS).  
  Renovación automática: El refresh token permite que el usuario siga autenticado sin necesidad de iniciar sesión repetidamente.  
  Separación de responsabilidades: El frontend nunca maneja directamente las credenciales o tokens.