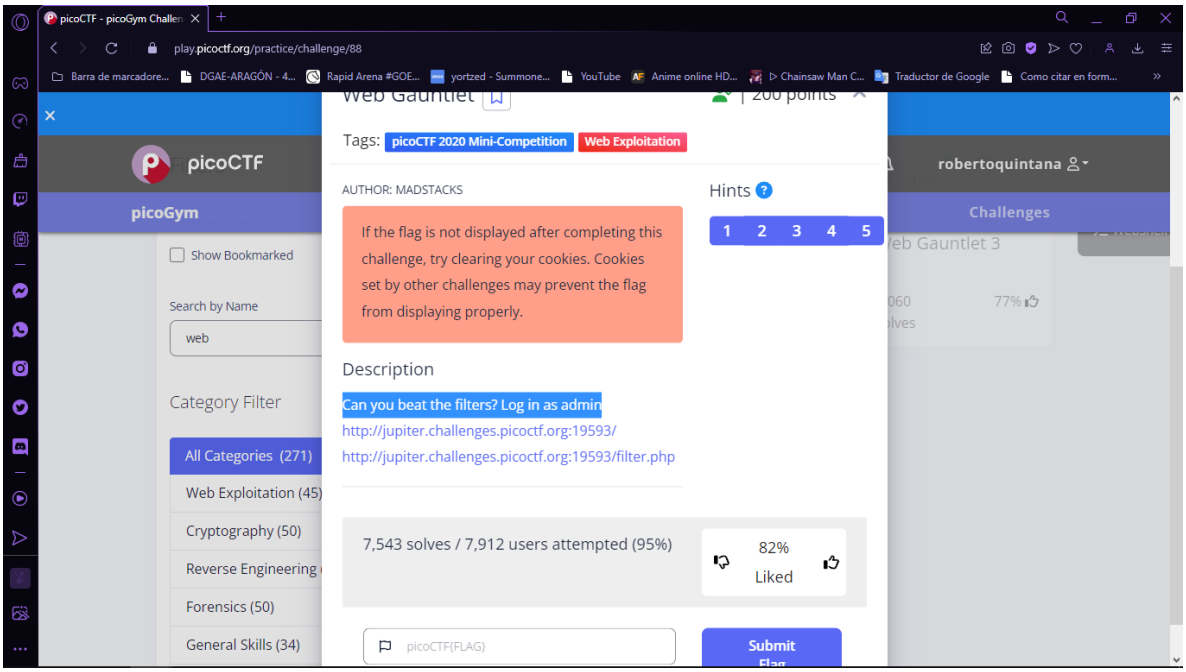
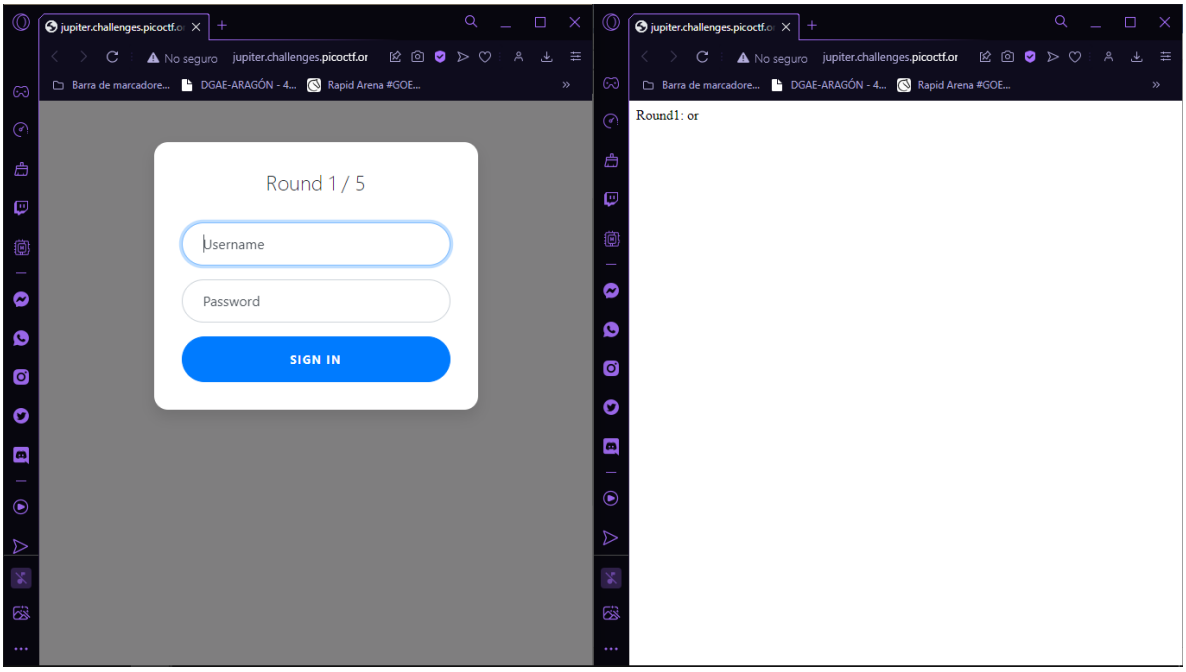
Tarea: SQL Injection

Alumno: Roberto Carlos Quintana Escamilla

Web Gauntlet

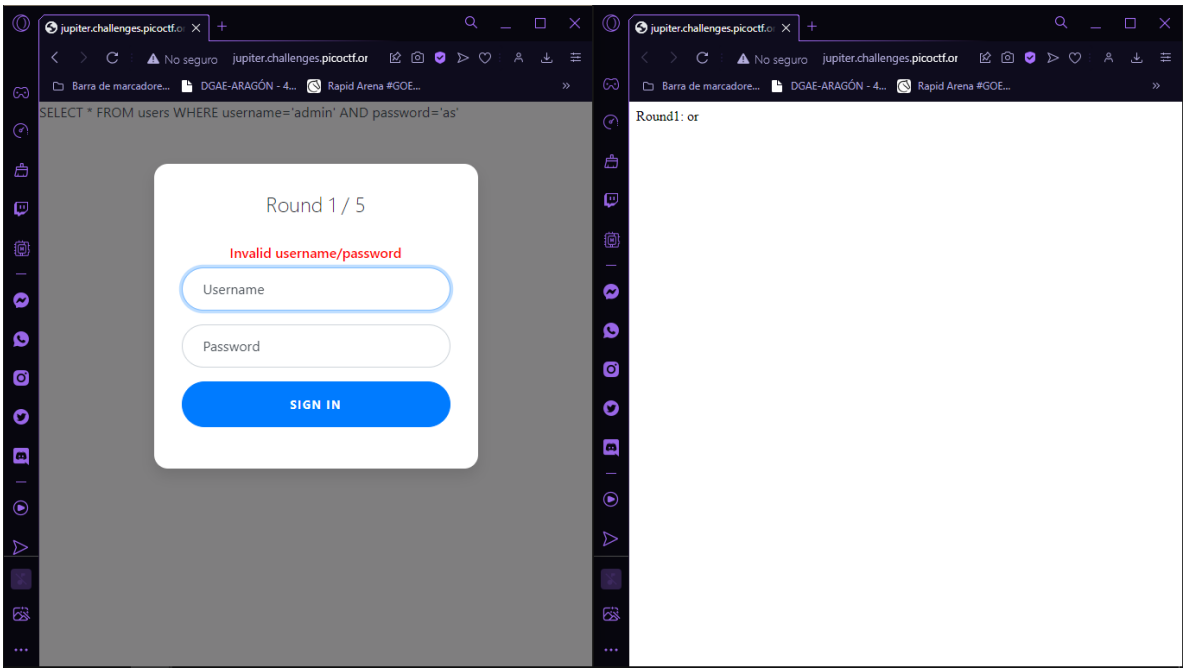
La descripción del ejercicio nos reta a poder logearnos como admin. Para lo cual se nos presentan dos páginas; la primera será sobre la que realizaremos el ataque y la segunda coniene los filtros de php de la primera.





Si intentamos loguearnos con la cuenta admin y una contraseña cualquiera como “as” podemos observar que se nos da una pista de cómo resolver el ejercicio pues aparece la consulta sql que se realizo. Por lo que suponemos que para hacer Loguin la pagina hace el select utilizando las variables obtenidas en los texfiled y pasándolas por el filtro; si la consulta da un resultado valido podremos ingresar.

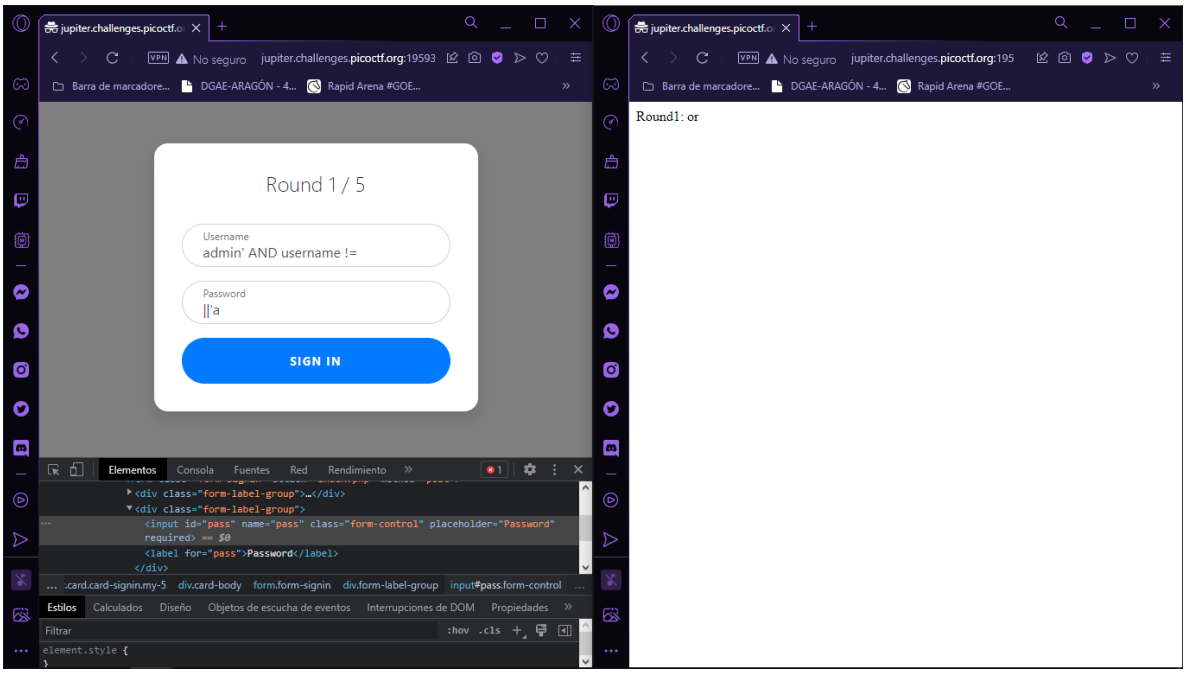
Sin embargo, esto nos permite introducir directamente un string en la consulta para modificarla.



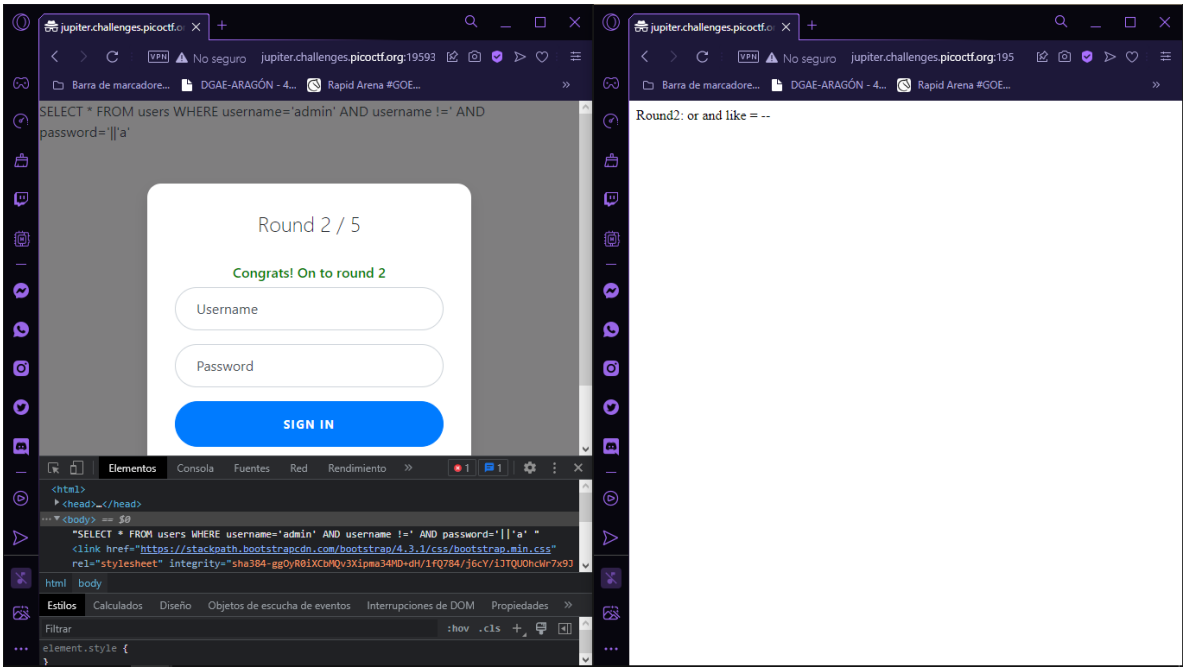
Probemos convirtiendo el resto de la consulta en variable tipo cadena de texto. Para ello primero en el tex file destinado para el nombre de usuario primero indicaremos que nos logaremos como admin poniendo una comilla mediamente para que la consulta interprete que se cerro la cadena. Luego de esto nos inventaremos una sentencia para almacenar la variable tipo cadena que queremos obtener del resto de la consulta sql, en este caso nos vasta con volver a comprobar el nombre de usuario y pediremos que sea diferente a la cadena siguiente, aprovecharemos la comilla que se agrega al final de lo que ingresemos en la casilla para que toda la consulta siguiente se interprete como cadena hasta la comilla con la que normalmente se leiria el paswort. Por ultimo para que no cause errores al ingresar el passwort ingresaremos un signo de concatenación en el texfile de paswort seguido de una comilla y un carácter cualquiera.

Nota: se cambio a modo incognito debido a problemas con el cache.

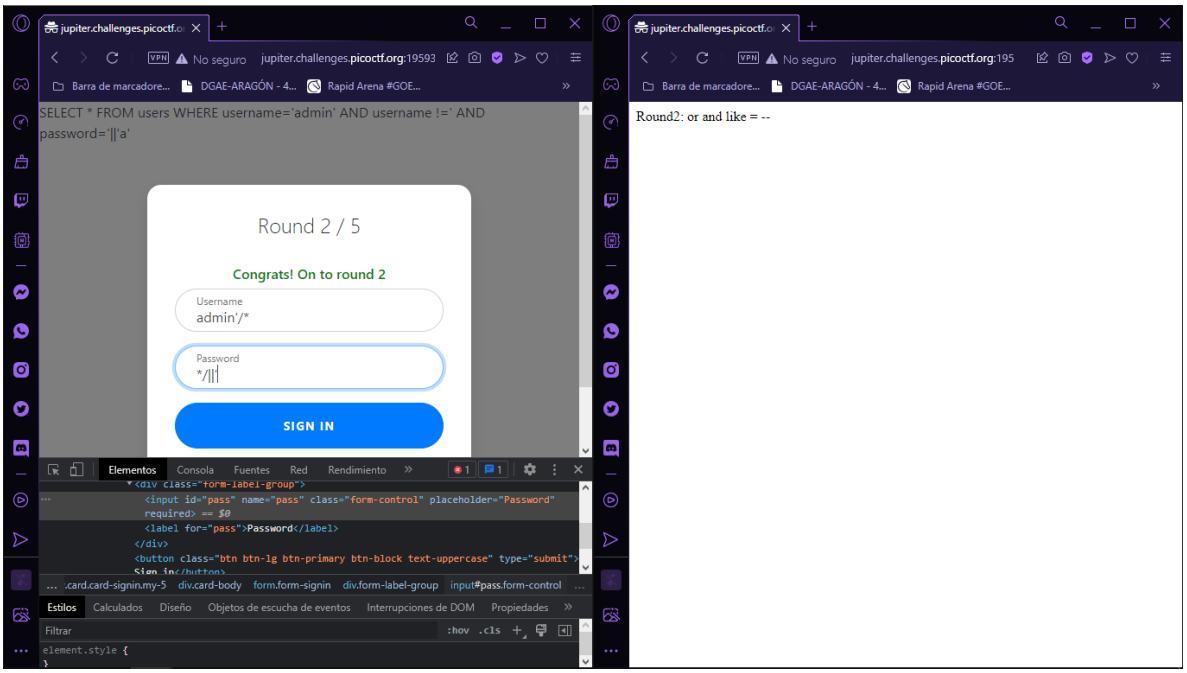
Nota: la consola únicamente se uso para modificar en el html el texfile del passwort para que fuera legible .



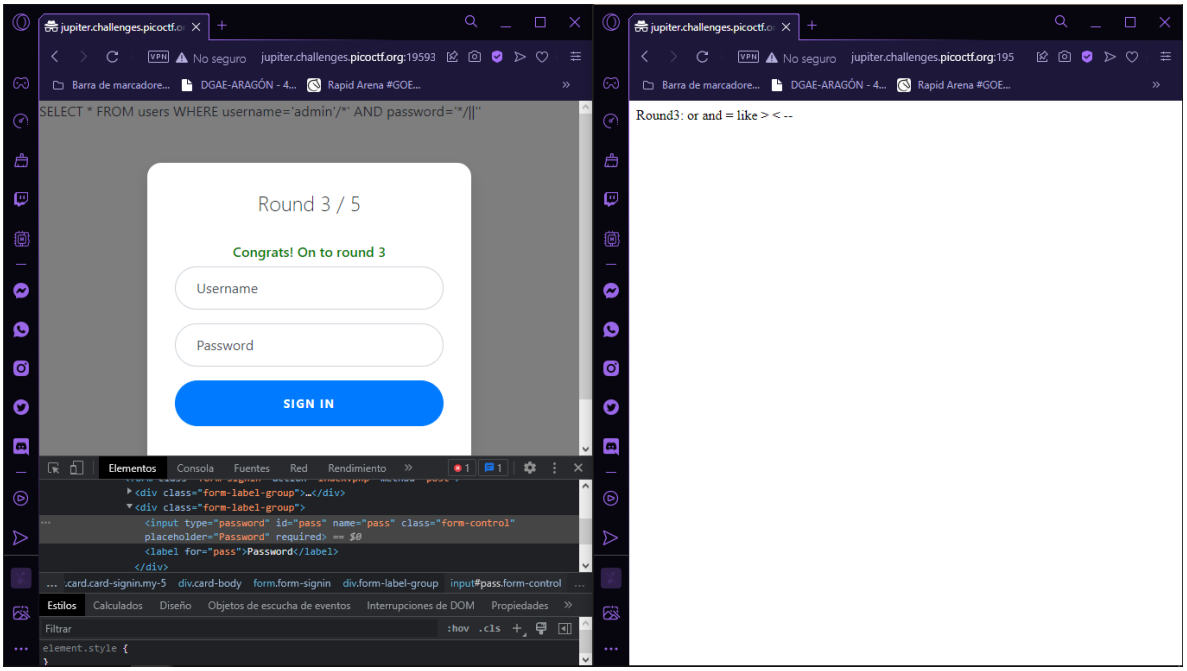
Con esto pasamos de round y vemos que los filtros se volvieron mas restrictivos.



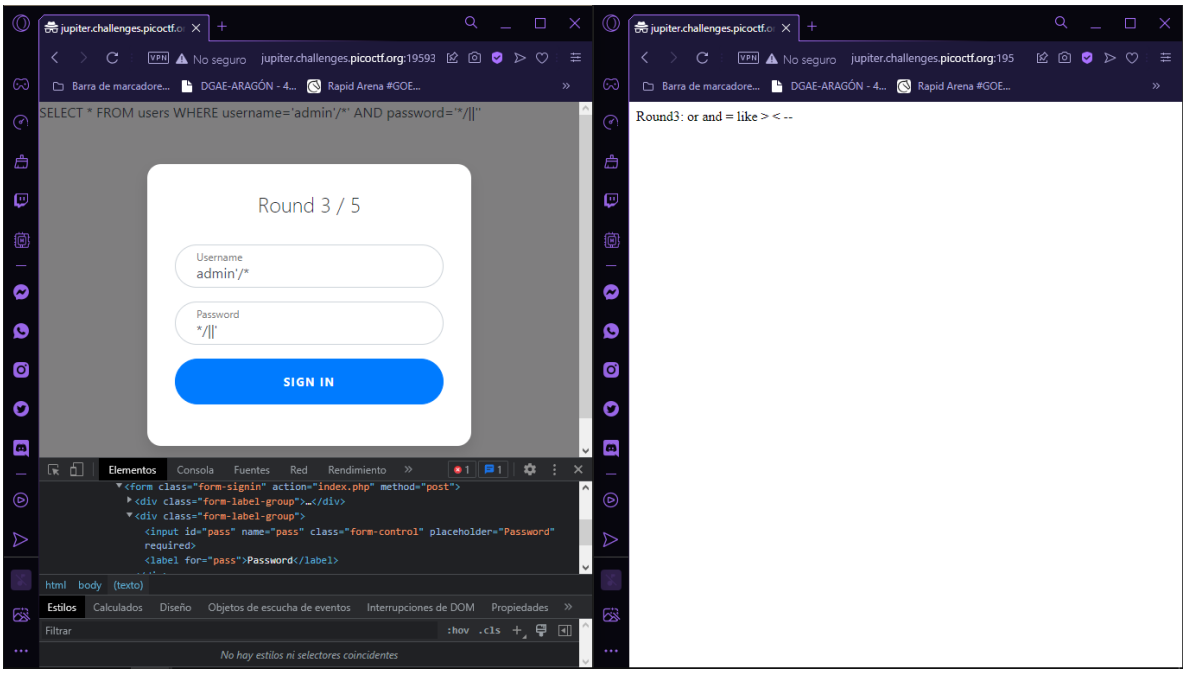
Para saltearnos estos nuevos filtros simplemente en lugar de convertir el resto de la consulta a una cade, procederemos a comentarlo. Para ello simplemente ingresaremos en usernam admin seguido de comilla para cerrar la variable y abriremos un comentario con /* .Para cerrarlo y evitar errores usaremos la casilla del passwort y para deshacernos de la comilla que se agrega al final utilizaremos una concatenación hacia la ultima catena que tenemos, admin en este caso, y como no queremos modificarlo le concatenaremos nada.



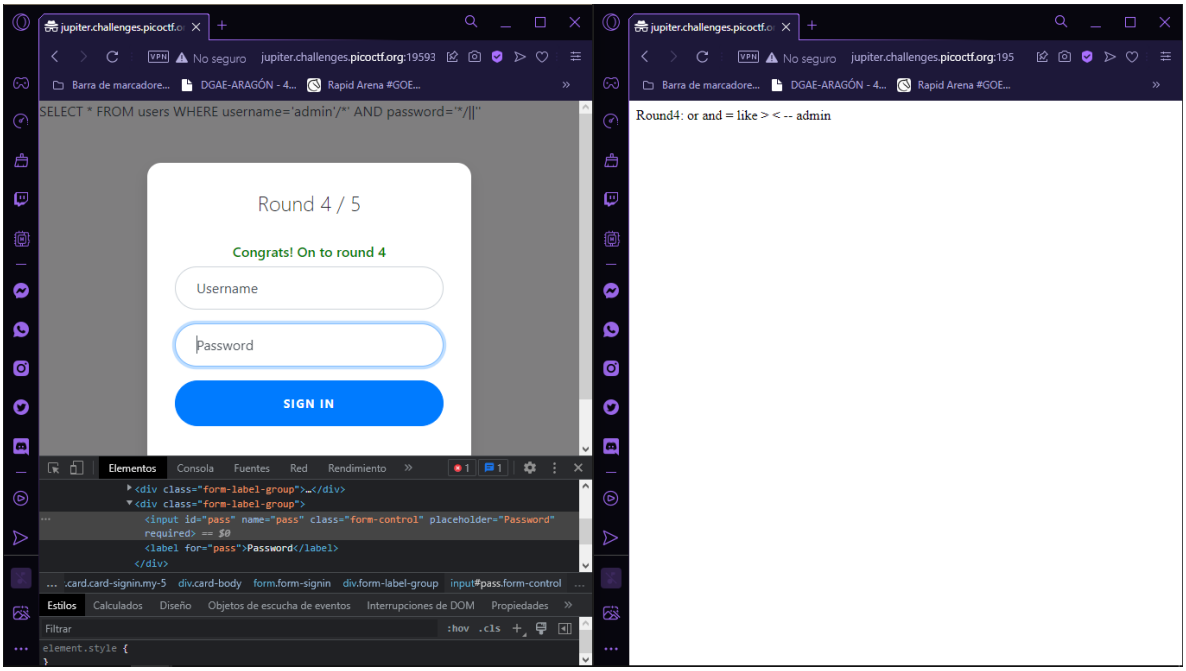
Observamos que los filtros cambiaron nuevamente.



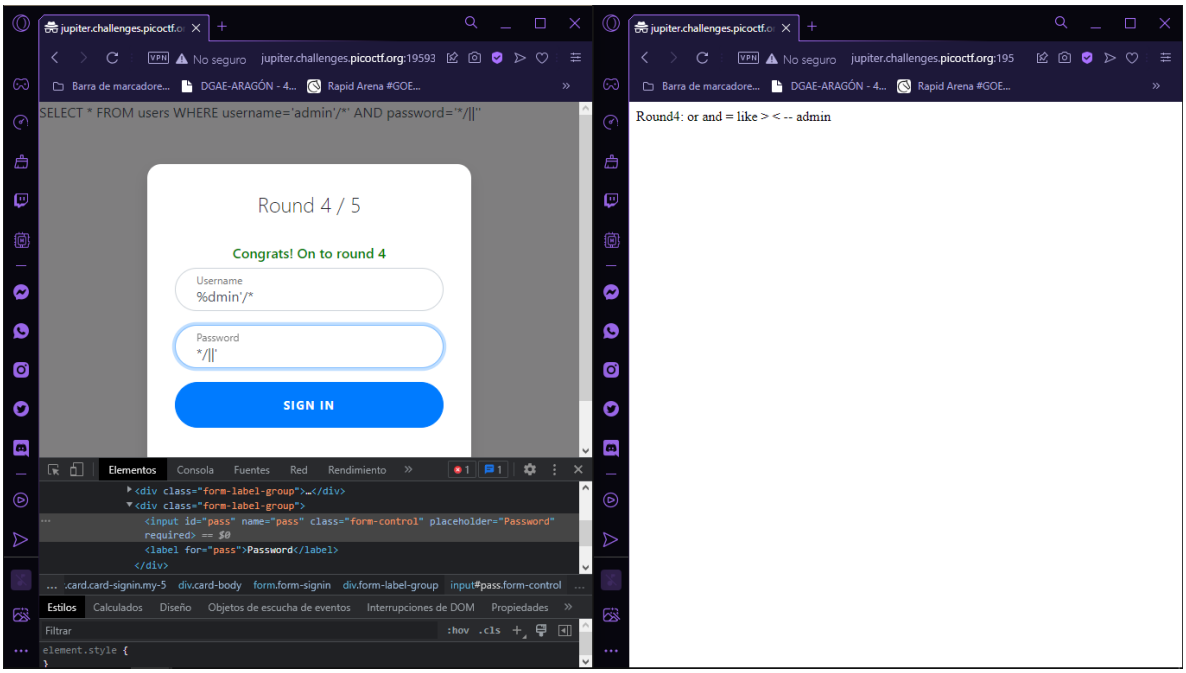
En esta ocasión los filtros no nos impiden utilizar nada que hayamos usado antes por lo cual podemos reutilizar la solución anterior.



Observamos que pasamos y los filtros cambiaron.

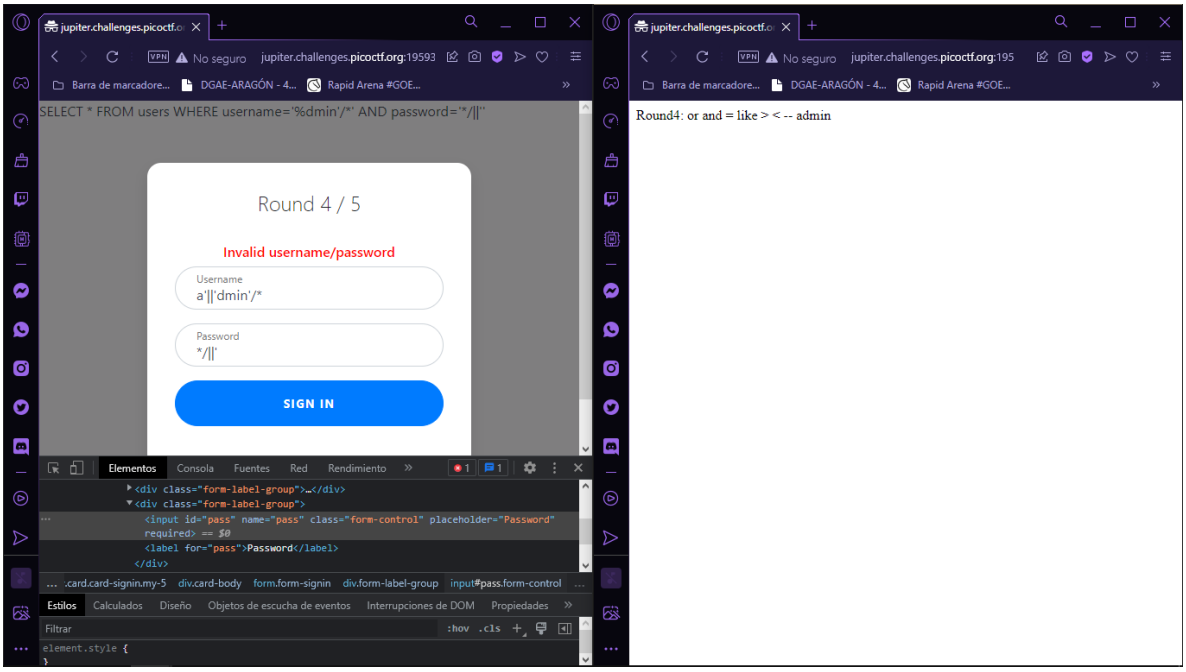


Ya que nos se nos permite usar admin de nuevo.

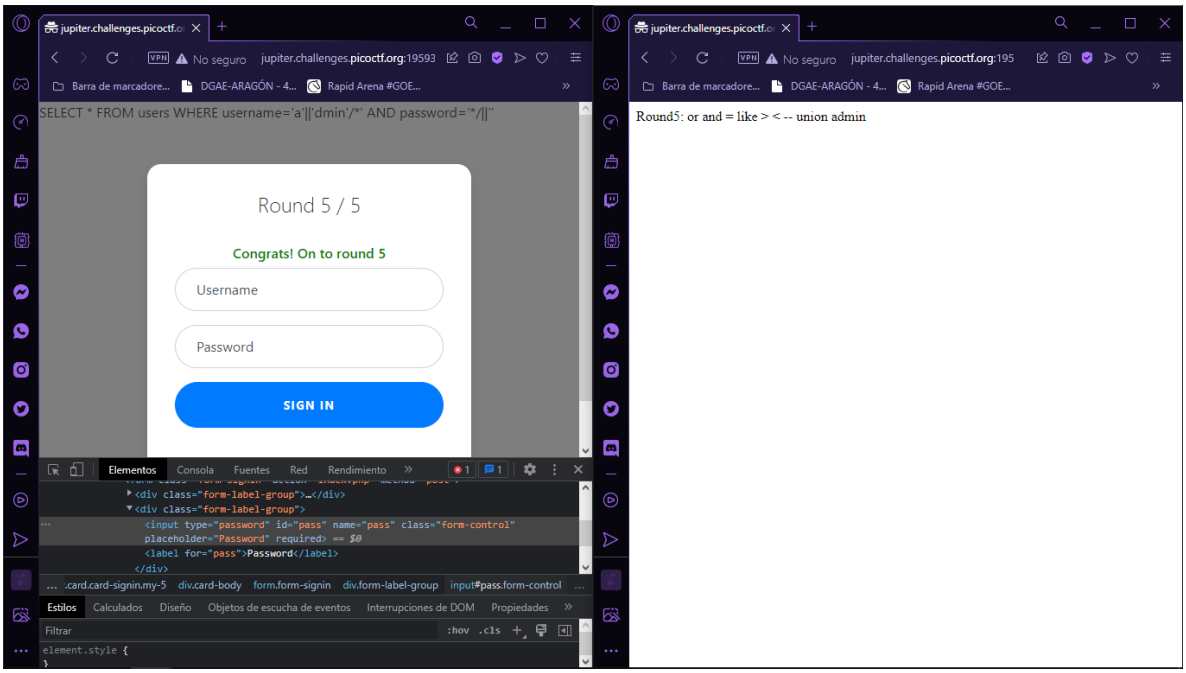


Para podemos seguir logueandonos como admin, lo que haremos será dividir la cadena en dos usando una concatenación intermedia.

Nota: se intentó usar el carácter comodín % al inicio y al final de la cadena, pero no funciono. Probablemente debido a la existencia de usuarios similares.

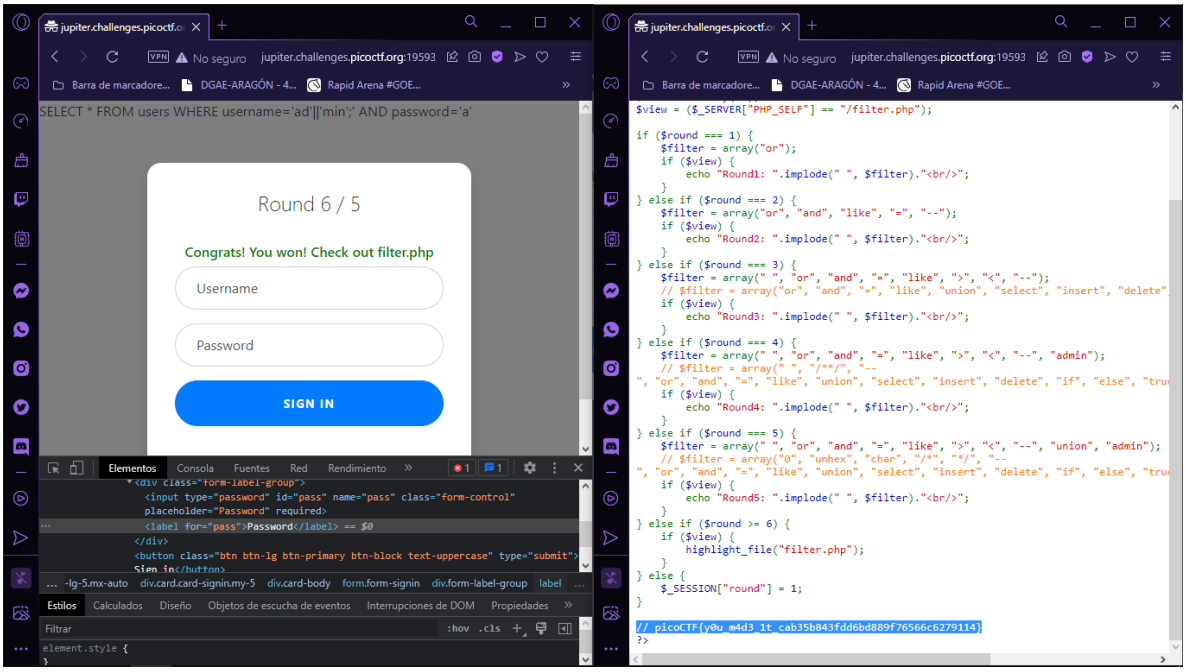


Observamos que logramos pasar y los filtros cambiaron.



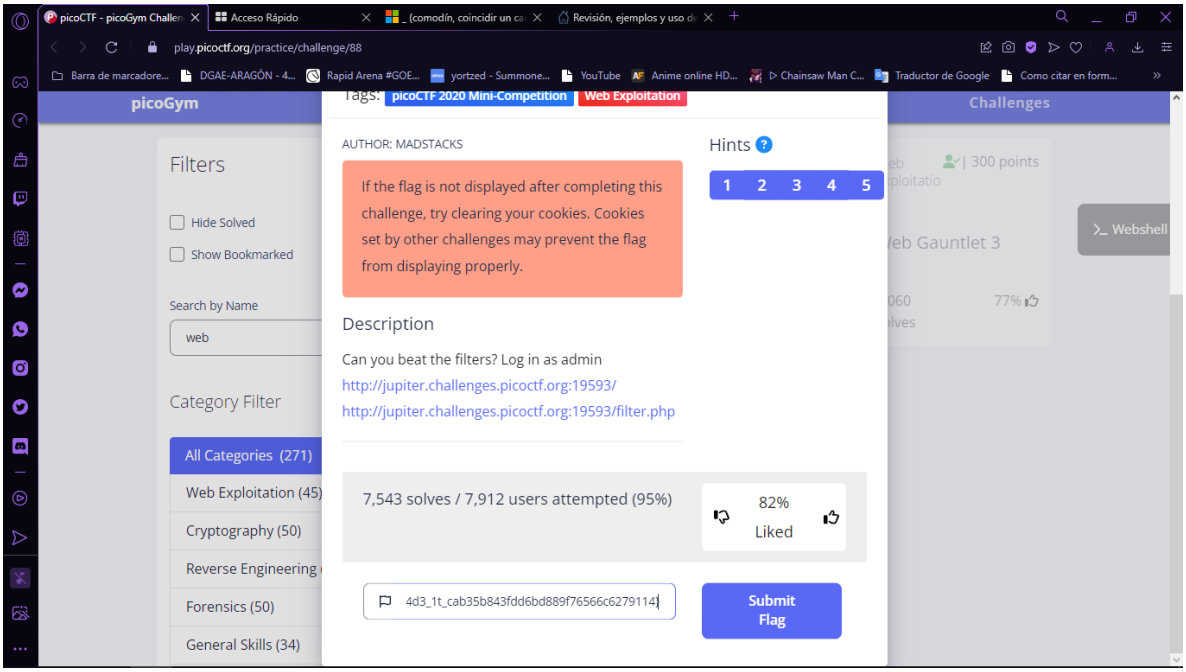
Nuevamente no se nos prohíbe usar algo que nos impida repetir la respuesta. Pero para variar en lugar de comentar el resto de la consulta simplemente terminaremos la sentencia con un ; luego del usuario.

Nota: se probaron diferentes comodines como _ o variantes de [] pero no dieron resultado

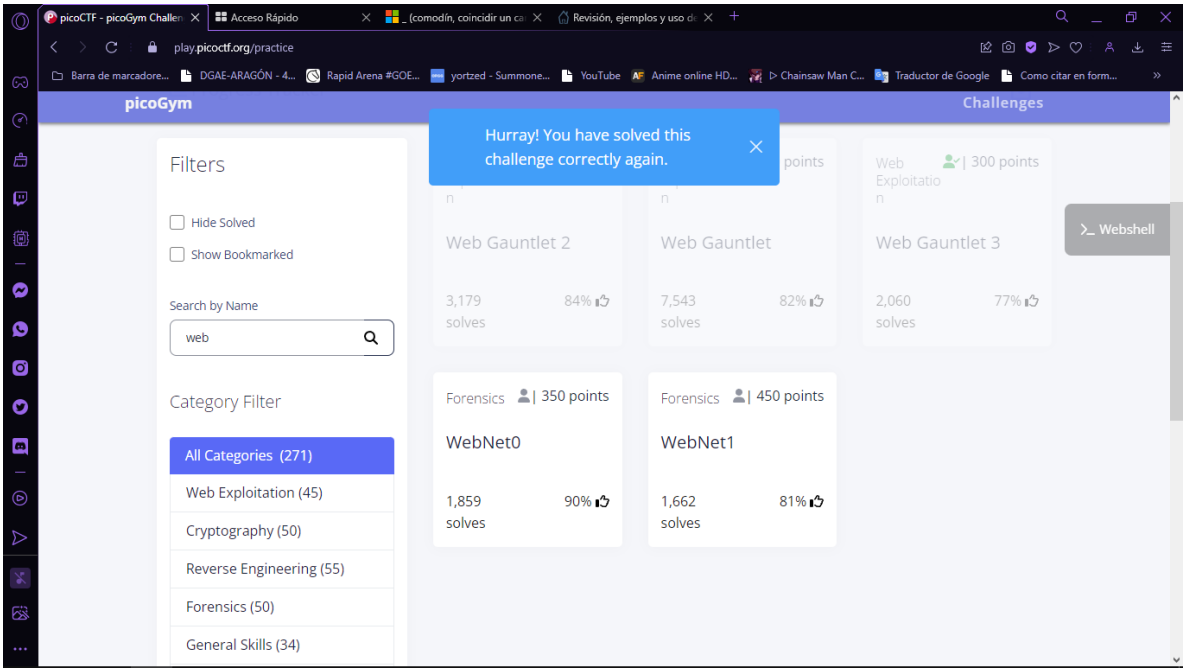


Observamos que obtuvimos resultado, se nos felicita y se nos invita a checar la pagina de fitros donde ahora se nos muestra el texto de los filtros. Con una rápida inspección vemos la flag.

picoCTF{y0u_m4d3_1t_cab35b843fdd6bd889f76566c6279114}

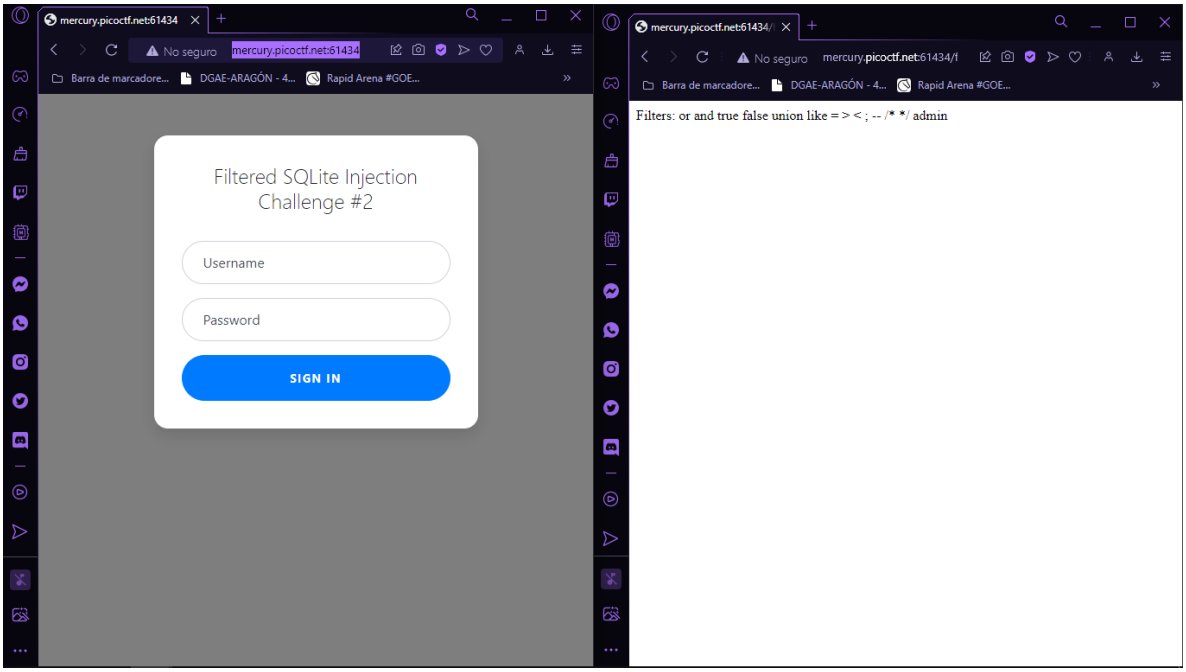


Verificamos

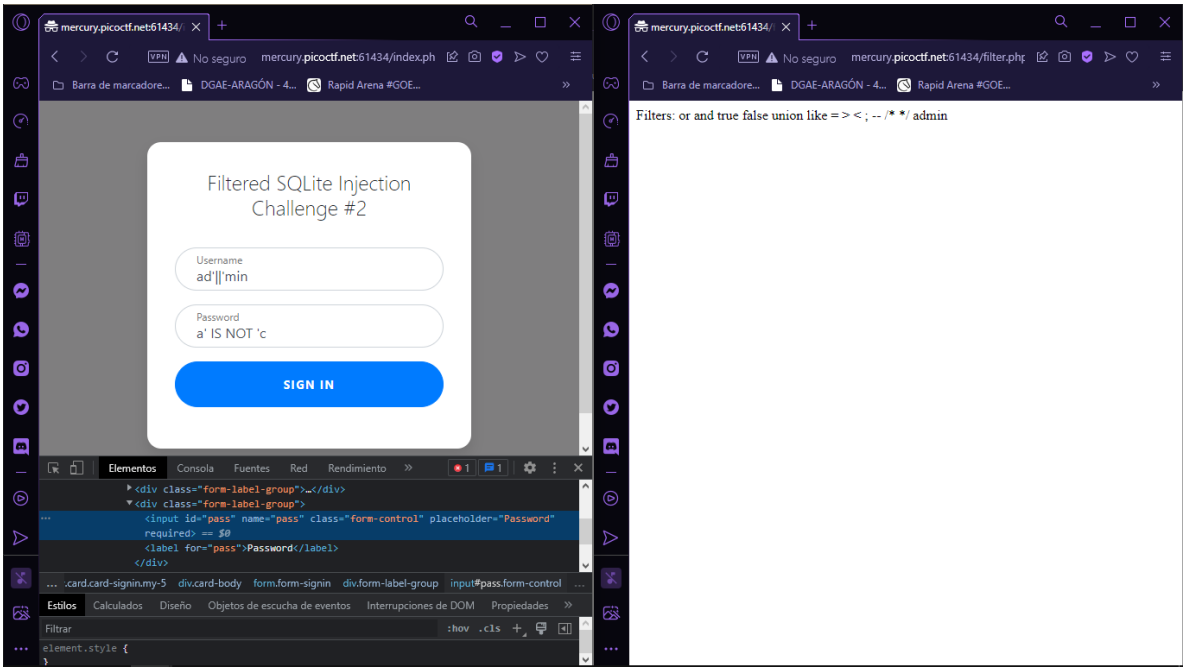


Web Gauntlet 2

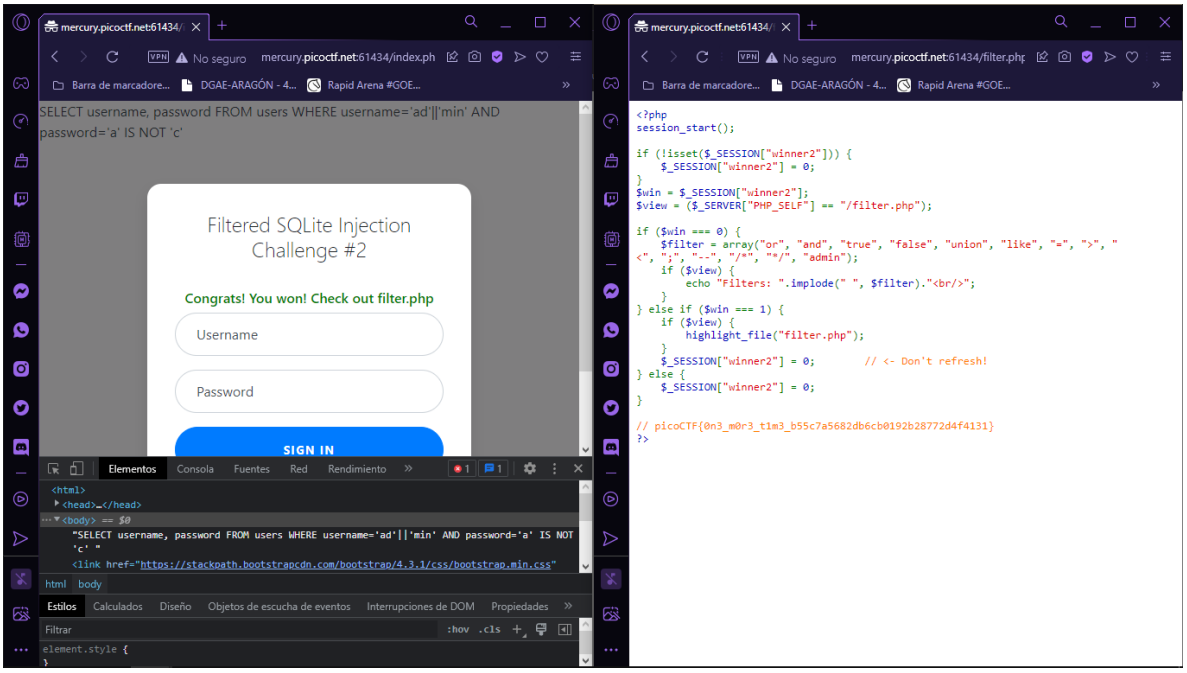
Observamos que es un problema similar a los del ejercicio anterior.



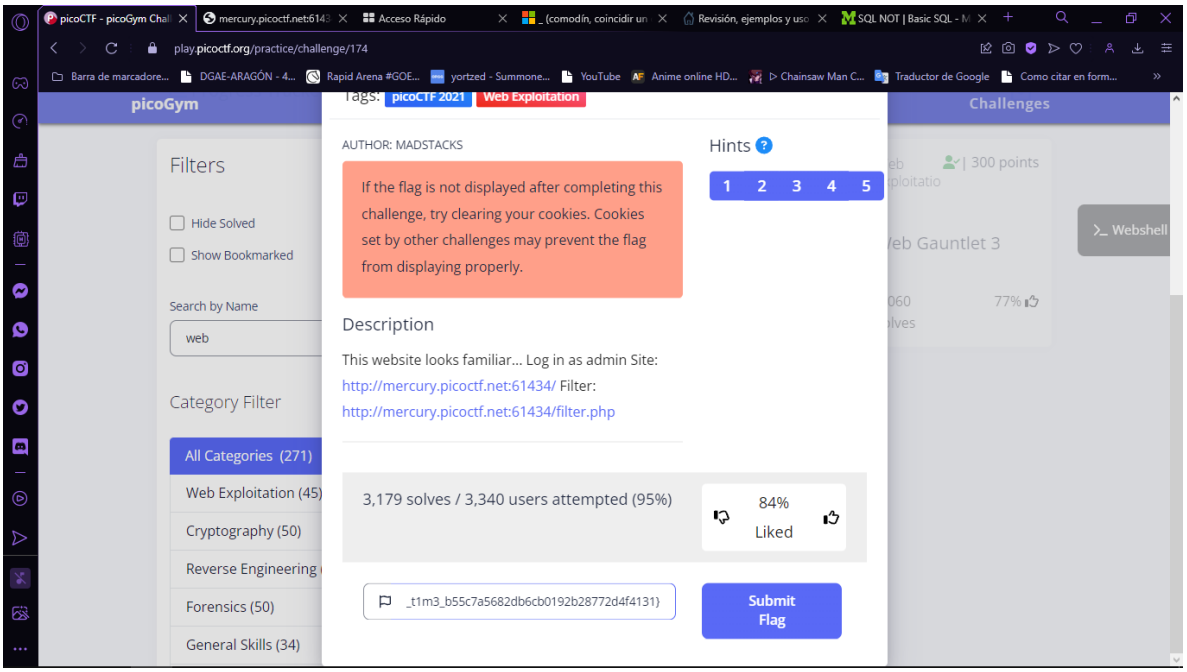
En este caso se nos prohíbe utilizar ; así como /* . Para resolverlo esta vez lo que haremos es utilizar el argumento de passwor t= ‘algo’ como entrada de un argumento de menor jerarquía para que así una vez que se resuelva el primero se ingrese a nuestro nuevo argumento y si este nos da una consulta valida para el usuario admin entonces podremos loguearnos. En este caso bastara con usarlo como entrada de Is NOT.



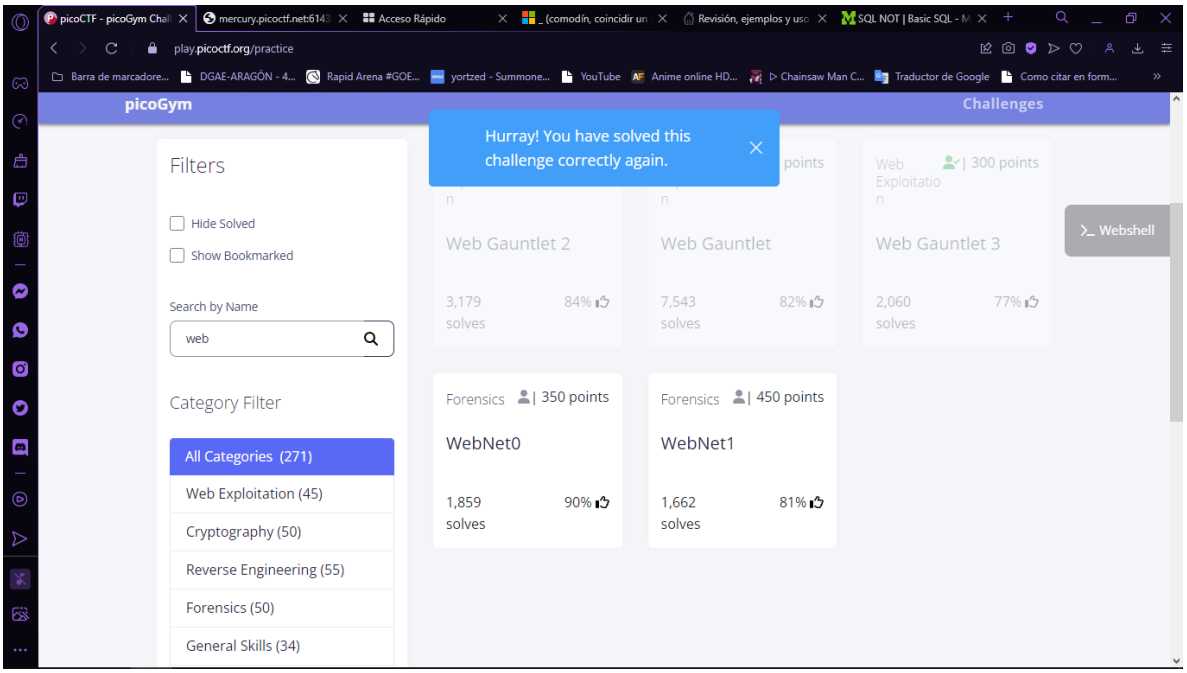
Vemos que hemos logrado pasar el ejercisio y nos dan el archivo de los filtros. Donde vemos la flag



picoCTF{0n3_m0r3_t1m3_b55c7a5682db6cb0192b28772d4f4131}

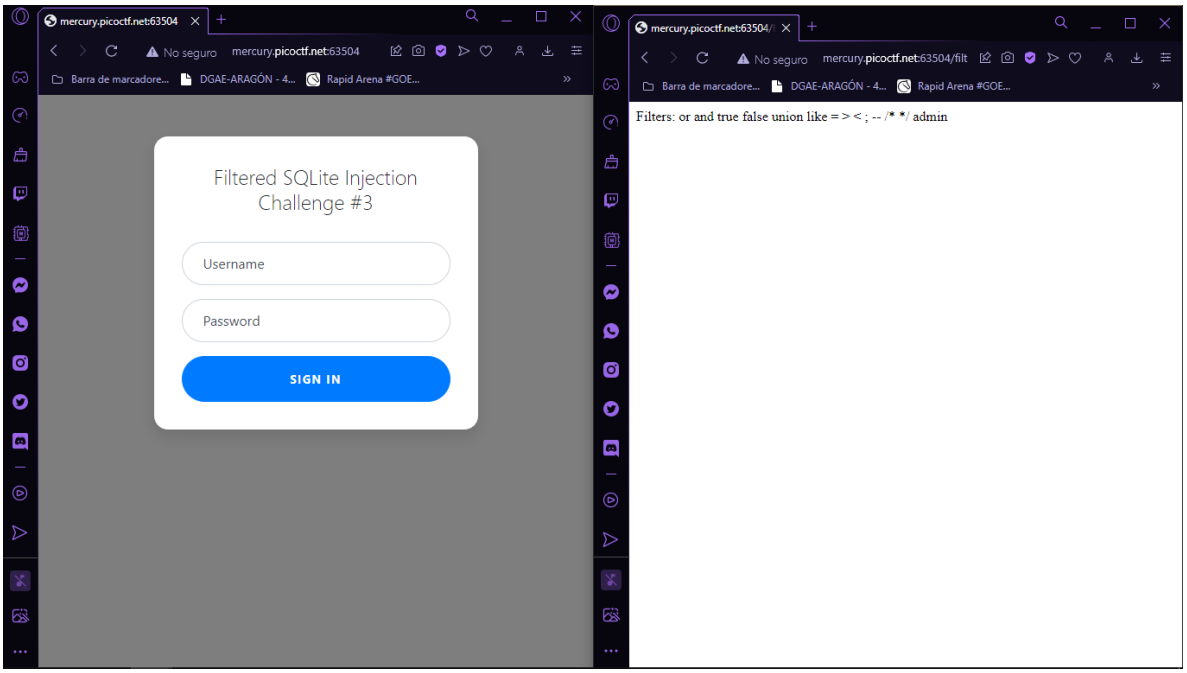


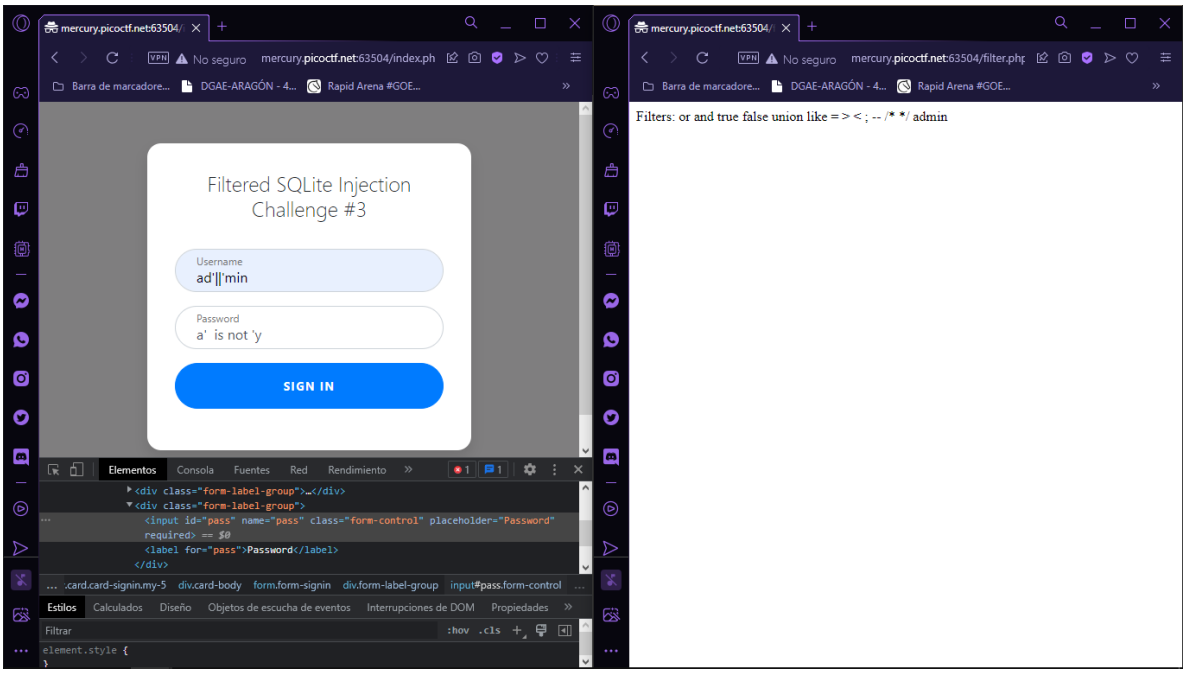
Verificamos.

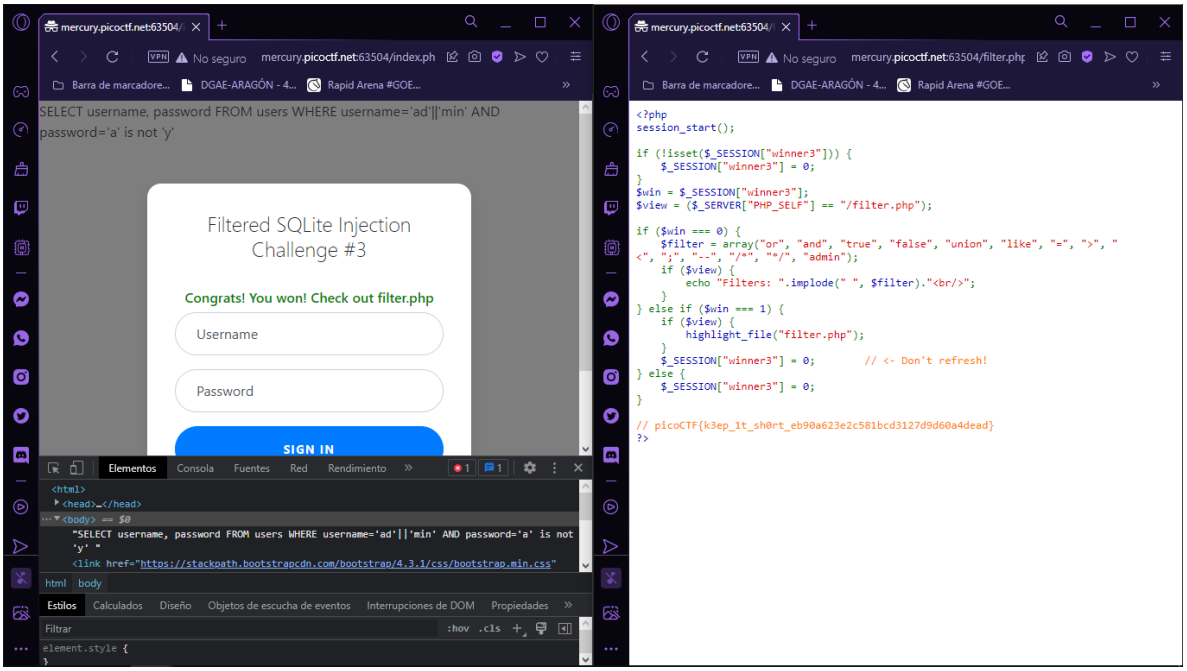


Web Gauntlet 3

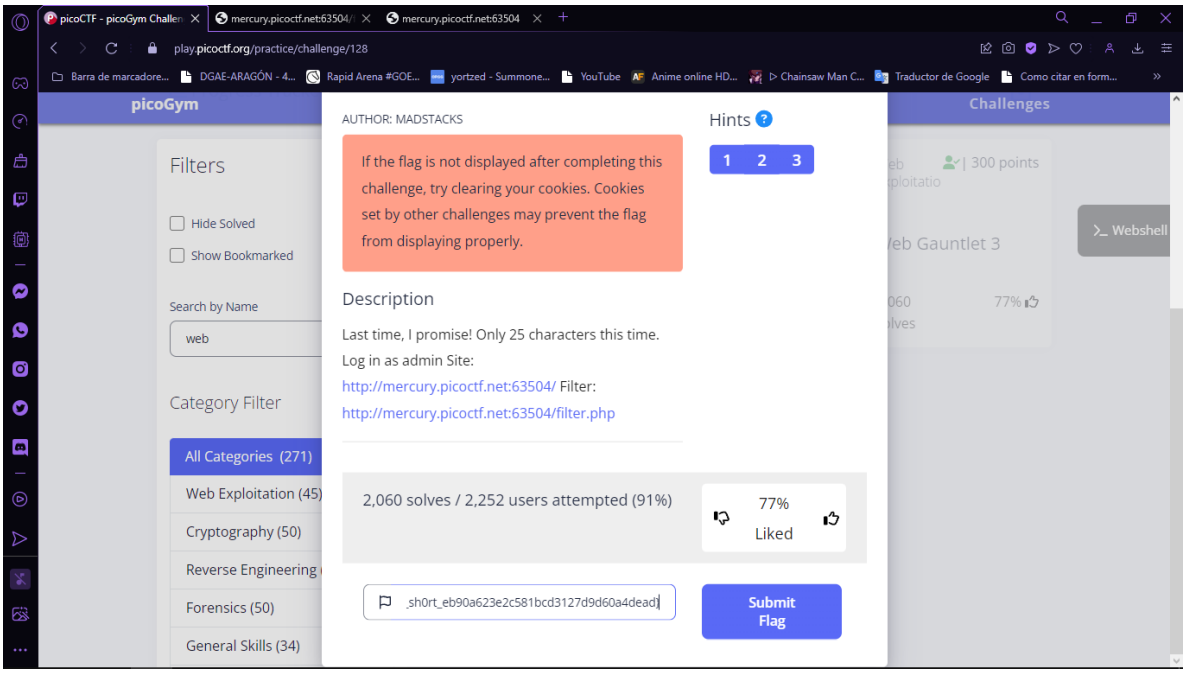
Observamos que no hay ningún cambio en los filtros, por lo que procedemos a repetir los pasos del ejercicio anterior

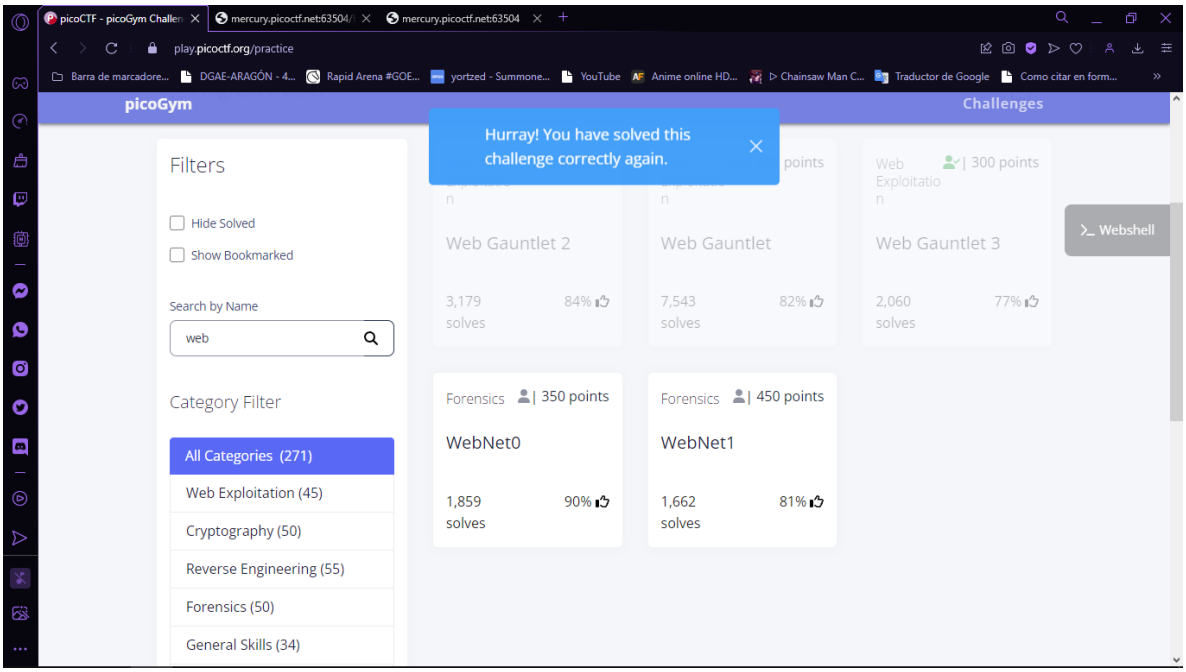






picoCTF{k3ep_1t_sh0rt_eb90a623e2c581bcd3127d9d60a4dead}





Conclusion

Observamos que el Sistema tiene importantes vulnerabilidades al no filtrar correctamente la información de la consulta. De esta manera como vemos se puede insertar código sql para hacer la consulta. Una forma burda de solucionarlo seria prhibir desde el fronter que se ingrese cualquier carácter que pueda cerrar la cadena para iniciar la inyección de código, sin embargo imagino que esto no seria suficiente pues se podría hacer inyección de código simplemente modificando el frontón por lo cual es necesario depurar cualquier petición que se le haga a la bd.