Uso de la cache en un web api

El manejo de la cache es manejado por la mayoría de los navegadores mediante estándares, para reducir tiempos de respuesta y la carga en el servidor (cantidad de recursos).

El cliente hace un requerimiento al servidor, y este responde con un response con una cabecera, donde esta son instrucciones para que el cliente sepa cómo acceder, o que hacer con la cache del servidor (instrucciones para el manejo dela cache del servidor).

Heders principales:

Cache-control

Expires

Expires:

Fue el primer archivo de cabecera para el manejo de cache, se encarga de definir el tiempo de expiración de un recurso puede ser utilizado.

Cache-control:

Es la practica actual de uso de cabecera de cache, permite especificar qué condiciones de almacenamiento, que puede almacenar y cuánto tiempo puede ser almacenado un recurso.

Public:

Cualquier navegador de un usuario final o proxis intermedios pueden almacenar mi información.

Private:

La cache solo almacenara recursos del cliente. Utilizado para guardar información que solo les interese a los usuarios.

Elemento max-age:

Establece un intervalo de tiempo para almacenar un recurso en caché.

Elemento No-store:

Si el servidor setea cache-control: No-store indica al cliente que no buscara nada en memoria cache, sino que siempre buscara el recurso directamente en el servidor.

Elemento no-cache:

Para un recurso que está almacenado en cache pero se prohíbe al usuario acceder hasta que primero pregunta al servidor si puede o no usar el recurso

Requerimientos condicionales:

El navegador tiene almacenado algún recurso, pero no sabe si usarlo o no, por lo que pregunta al servidor si puede usarlo o no. Este decide si el recurso que tiene el cliente es un recurso más actual o si debe actualizarlo. Si es un recurso actualizado la respuesta es un 304.

Requerimiento condicional basado en el tiempo:

Cuando el cliente hace un requerimiento al servidor, el servidor además de enviar dicho recurso setea un last-modified que es fecha de última modificación del recurso, si el cliente vuelve a acceder al sitio por el mismo recurso, se verifica la fecha del recurso para saber si este se ha actualizado, de no ser así se envía un 304, de no se vuelve a enviar el recurso actualizado.

Requerimiento condicional basado en el contenido:

Se toma en cuenta una función hash, que convierte un valor a otro valor, o un checksum, que sirve para verificar la integridad de un paquete, cualquiera de ellas genera un código que es utilizado por el navegador (ETag), donde para la próxima vez que el cliente solicita un recurso con el ETag, si este ETag es el mismo que el del servidor, quiere decir que no se ha actualizado el recurso y se envía un 304.

ETag tiene dos validaciones: Fuerte y débil, Fuerte se analiza bit por bit, mientras que debil si el contenido es igual o no.