Hoy en día, el rendimiento es una de las métricas más importantes que debemos evaluar al desarrollar un servicio / aplicación web. Mantener a los clientes comprometidos es fundamental para cualquier producto y, por esta razón, es extremadamente importante mejorar el rendimiento y reducir los tiempos de carga de la página. Al ejecutar un servidor web que interactúa con una base de datos, sus operaciones pueden convertirse en un cuello de botella. MongoDB no es una excepción aquí y, a medida que nuestra base de datos MongoDB se amplía, las cosas realmente pueden disminuir. Este problema puede empeorar incluso si el servidor de la base de datos se separa del servidor web. En tales sistemas, la comunicación con la base de datos puede causar una gran sobrecarga. Afortunadamente, podemos usar un método llamado almacenamiento en caché para acelerar las cosas. En este ejemplo, presentaremos este método y veremos cómo podemos usarlo para mejorar el rendimiento de nuestra aplicación utilizando Spring Cache, Spring Data y Redis.

Para tener una mejor idea de cómo lograremos este objetivo, volvamos al ejemplo de las personas sentadas en una biblioteca tradicional. Digamos que quieren encontrar el libro con un título determinado. En primer lugar, mirarán alrededor de la mesa para ver si ya la han llevado allí. Si lo han hecho, eso es genial! Acaban de tener un hit de caché que está encontrando un elemento en el caché. Si no lo encontraron, tuvieron una falla de caché, lo que significa que no encontraron el elemento en el caché. En el caso de que falte un artículo, tendrán que buscar el libro en la biblioteca. Cuando lo encuentren, lo mantendrán en su mesa o lo insertarán en el caché. En nuestro ejemplo, seguiremos exactamente el mismo algoritmo para el método findBookByTitle (). Cuando se le solicite un libro con un título determinado, lo buscaremos en el caché. Si no lo encuentra, lo buscaremos en el almacenamiento principal, que es nuestra base de datos MongoDB.

De forma predeterminada, JHipster usa una sesión HTTP solo para almacenar la información de autenticación y autorizaciones de Spring Security. Por supuesto, puede optar por poner más datos en sus sesiones HTTP. El uso de sesiones HTTP causará problemas si está ejecutando en un clúster, especialmente si no usa un equilibrador de carga con "sesiones pegajosas". Si desea replicar sus sesiones dentro de su grupo, elija esta opción para tener configurado Hazelcast. Elige no si no estás seguro. ¿Quieres usar WebSockets? Websockets se puede habilitar usando Spring Websocket. También proporcionamos una muestra completa para mostrarle cómo usar el marco de manera eficiente.