Web Caching

An approach to optimizing data storage in web applications.

WE'LL COVER THE FOLLOWING

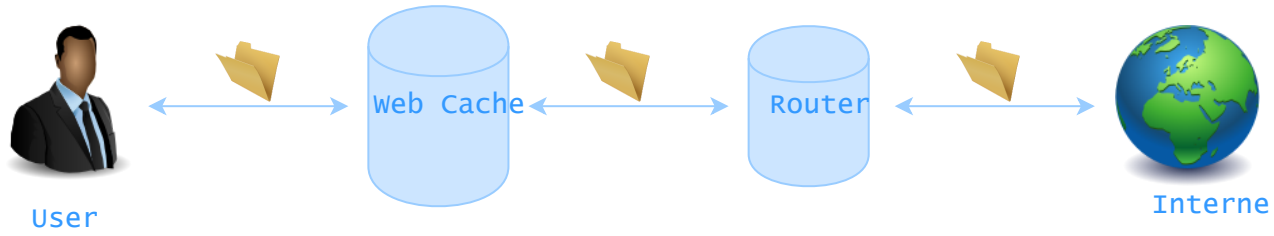
- Introduction to web caching
- Benefits of caching
- Quick quiz on web caching!
- MongoDB
- Redis
- CouchDB
- Memcached

In the previous lesson, we discussed the latency that frequent database calls introduce in web applications. Now, we will look into web caching as a means of eliminating this latency.

Introduction to web caching

Web caching is a design feature of the HTTP protocol that is meant to minimize the amount of traffic an application is handling at any given point in order to improve the responsiveness of the web application, as perceived by users, as a whole. In order to do this, caches are used at each level starting from the server itself all the way to users' browsers, and each of them is meant to store data that users are very likely to request.

Essentially, web caching works by caching HTTP responses for certain requests according to a probabilistic analysis of requests that are frequently observed on the server. Subsequent requests for cached content can then be fulfilled from a cache closer to the user instead of having to send the request all the way back to the web server, which then makes a call to the database for the required data.



How Caching Works

Benefits of caching

There are multiple benefits of web caching, each of which has been enumerated below:

1. Data can be cached at several different points in the path between the client and the server. When the required data is cached closer to the client, requests do not increase network traffic too much since they are resolved much earlier along the path.
2. Extending on the first point, since requests are resolved earlier along the path, responses are sent back quicker as well, thus improving the responsiveness of web applications.
3. Aggressive caching along the network can also allow the application to sustain higher loads of data since a significant chunk of the data can be stored within caches.
4. In case the server has trouble accessing the database for some reason, data that has been stored in caches can still be served to end-users.

Quick quiz on web caching!

Quiz on Web Caching

1 Which of the following is a benefit of caching?

Now that we know how important caching is let's look into exactly how you can cache your web application data!

MongoDB

MongoDB has inbuilt mechanisms to handle caching and keeps the most recently used data in the RAM. If users have created indexes for their queries and the working data set fits in the RAM, MongoDB serves all queries from memory. However, MongoDB does not cache query results in order to return results from the cache for all future identical queries.

Redis

Redis, as we have already learned, stores all data in memory by default, and it is often used as a cache itself. It is, therefore, optimal to use as a database system in modern web applications.

CouchDB

CouchDB, like the other two NoSQL databases we have discussed, aims to cache all the data it possibly can. The smaller the file size, the more of the file can be cached by CouchDB. It is, therefore, a good idea to think about the data you want to store when it comes to NoSQL databases so that your web application can run seamlessly and unburdened by unnecessary amounts of data.

Memcached

Memcached is an open-source, high-performance, distributed memory object

Memcached is an open-source, high-performance, distributed memory object caching system that is intended to speed up dynamic web applications by alleviating database load. Memcached stores key-value pairs of data from database calls in memory in order to speed up the database lookup process. The underlying idea of Memcached is that it allows you to take memory from parts of your system where you have more than you need and make it accessible to areas where you have less than you need. Memcached is, therefore, a highly useful tool.

That concludes the discussion on databases and how their use can be made more efficient through caching. In the next chapter, we will move on to discuss another integral component of any web application; the front-end.