Caching

Caching is an important topic of system design and is a widely used technique with multiple benefits.

You must have noticed that many websites are super fast and load the elements very fast, while some new websites take a lot of loading time. Slow loading gives a poor user experience. In order to prevent slow loading like excessive buffering during streaming a video, websites use caching, which helps speed up the process.

What is Caching?

In computers, a cache is a high-speed data storage layer that caches a chunk of data that is typically transient so that subsequent requests for that data can be delivered up faster than if the data were accessed directly from its primary storage location. Caching allows you to reuse data that has been previously retrieved or computed quickly. Caching helps in the reduction of the number of read calls, API calls and network I/O calls by creating a local instance of the static information.

Some commonly cached items include:

1. HTML pages (partial or complete)
2. API responses
3. Database queries

How does Caching work?

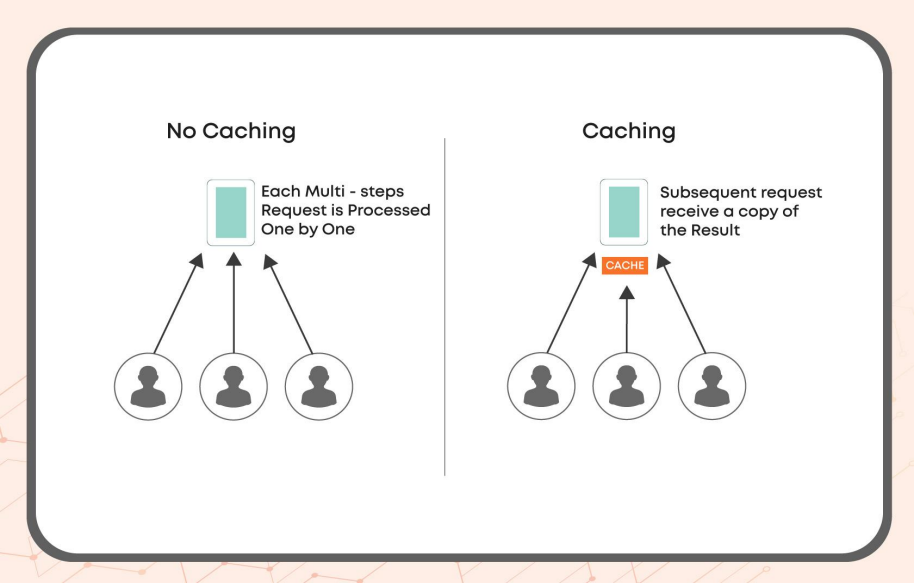

Fig: This is how Caching functions

The data in a cache is most of the times kept in rapid access hardware like RAM (Random-access
memory), but it can also be utilised with a software component. The fundamental goal
is to improve data retrieval performance by eliminating the need to contact the slower
storage layer behind it.
In contrast to persistent storage, a cache often stores a subset of data transiently.

What places can caching be used?

A web application would have a data storage layer, frontend and backend layer if the data in the database is not frequently changed or updated, instead of again and again accessing the required data from the database, which adds on the response time. The Backend layer can have a temporary instance of the database, which is known as a cache. Caching is inbuilt in the Operating System for this above algorithm. Caching can be used for JS or CSS components if they do not change frequently. Caching can also be used in the frontend if some element is not changing frequently; we can cache it. Caching can be used at various places.

Significant applications of caching are:

1. CDN(Content Delivery Network)
2. Application Server Cache

Application Server Cache:

We can locally store this information in memory for a web application consisting of front-end, heavy computation or a database, and the elements are not frequently changed or updated. This local storage is known as a cache.

CDN(Content Delivery Network):

When the frontend has multiple static elements like CSS, JavaScript, etc., which are not changed frequently, instead of serving them from the backend, they can be delivered from a third party layer known as Content Delivery Network. They cache static content and provide the data at a faster rate to the frontend or application.

Types of Caching Solutions

There are two types of cache:

1. In Memory/ Local Cache:
   It is used for a single system when the cache must be stored in the local memory.
   Example: Google Guava Cache, Memcache

2. Distributed Cache/External Cache:
   It is used when the cache needs to be shared between multiple systems. Therefore, we store the cache as a distributed system and can be shared by all the servers.
   Example: Redis

Deciding when to use caching:

1. Static Data:
   Caching would be useful if the data is not changing very frequently, we can store it and use it directly. In case the data is changing quickly, caching would not help much.

2. Type of Application:
   The application can be either read-intensive(number of read operations are more than the number of write operations like Wikipedia) or write-intensive(number of write operations are more than the number of read operations like twitter). Caching would be more useful for the read-intensive application. For a write-intensive application, data would be changing very fast and therefore caching should not be used.