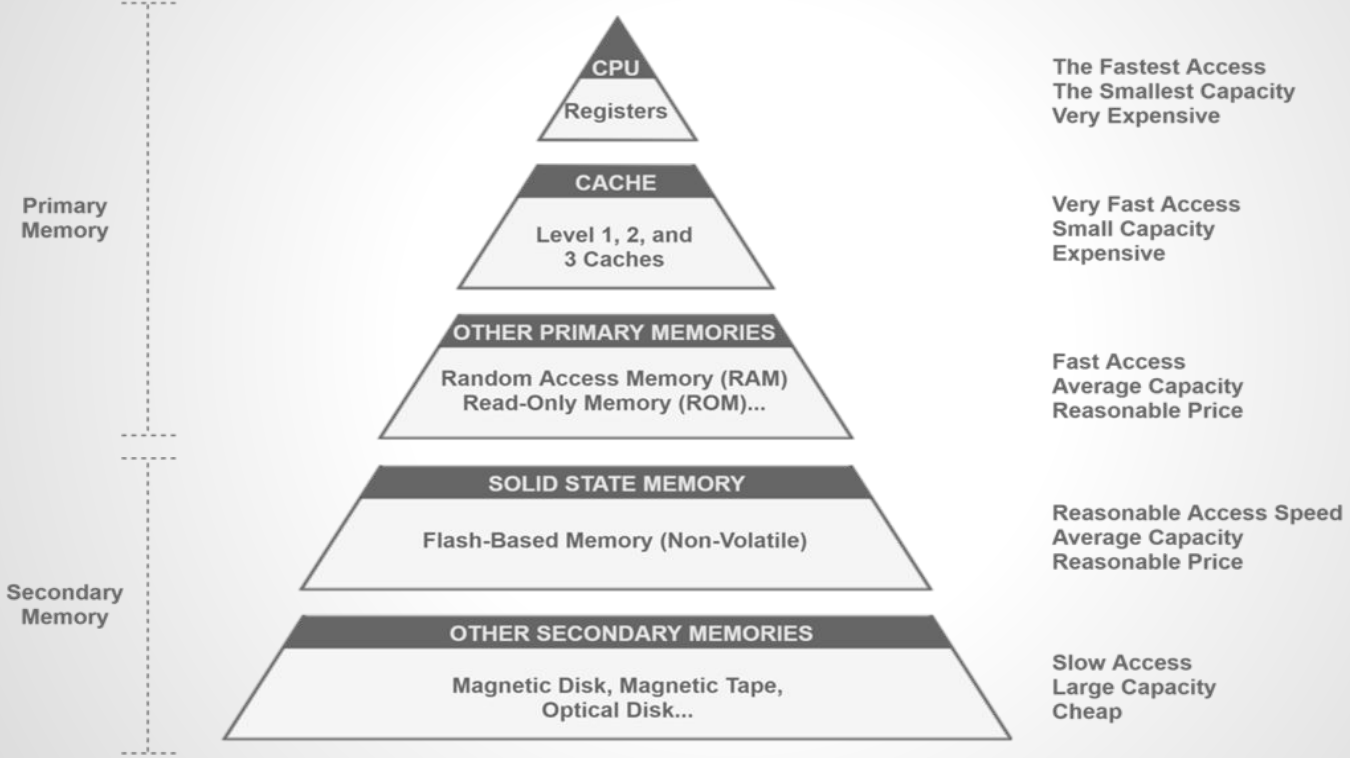
CACHING

By: Robel Shewangzaw

CACHING

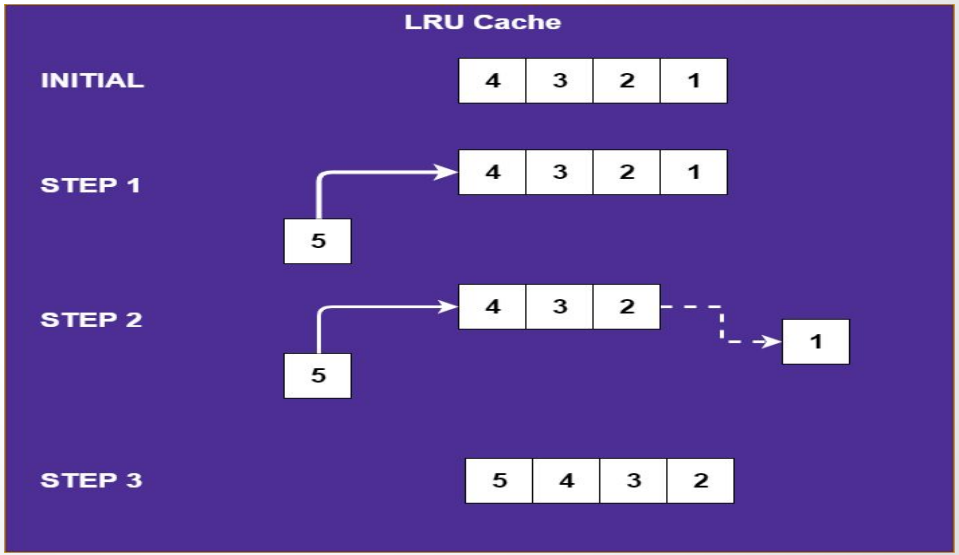
caching is hardware or software component that stores data temporarily to serve future requests more quickly.

Memory Hierarchy Levels



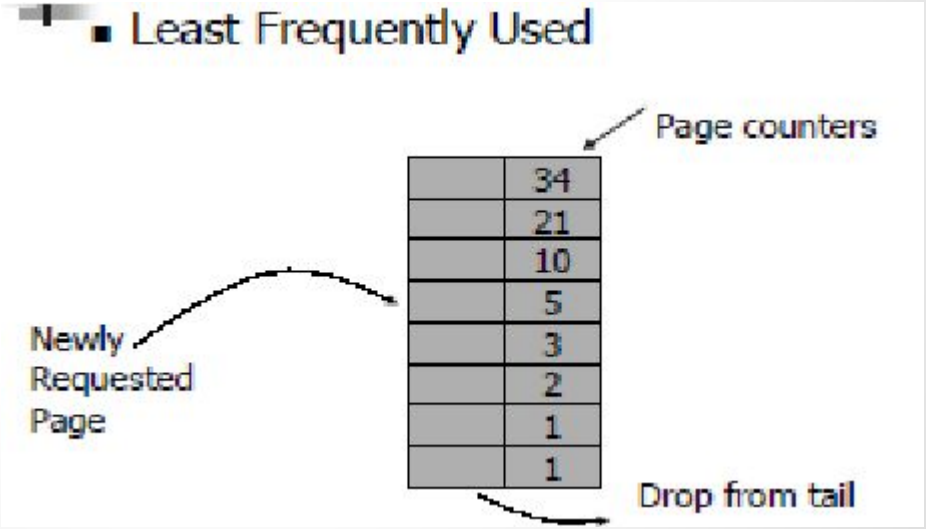
Cache Replacement Algorithms

Least Recently Used (LRU) : a memory storage system that removes the least recently used items when full, keeping the most recently accessed items for quicker retrieval.



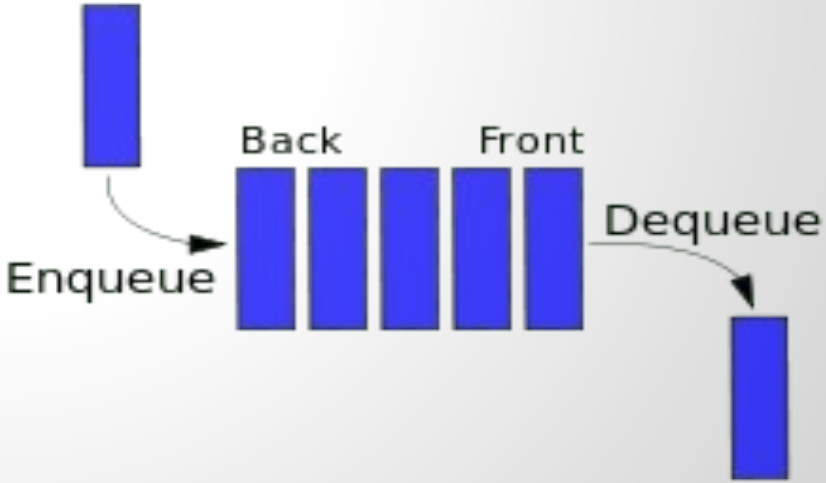
Cache Replacement Algorithms

Least Frequently Used(LFU) : a memory storage system that prioritizes keeping items that are accessed the least frequently, removing them when the cache is full to make room for more frequently accessed items.



Cache Replacement Algorithms

First In First Out (FIFO) : A memory storage system that removes the oldest items first when the cache is full. Maintaining the order in which items were initially.



Why We use Cache

- ➔ Better performance
  - ◆ System loads faster
- ➔ Better scalability
  - ◆ Limit bottlenecks in a system
- ➔ Better robustness
  - ◆ Can support more load

Caching Terminology

[Cache Hit]
when requested data is contained in the cache

[Cache Miss]
when requested not in the cached, has to be recomputed or fetched from original storage

[Cache Key]
unique identifier for a data item in the cache

[Expiration]
item expires at a specific date (absolute), specifies how long after an item was last accessed that is expires

[Cache Scavenging]
deleting items from the cache when memory is scarce

[Local Cache]
caching data on clients rather than on servers

[Distributed Cache]
extension of the traditional concept of cache that may span multiple servers

Types of Caching

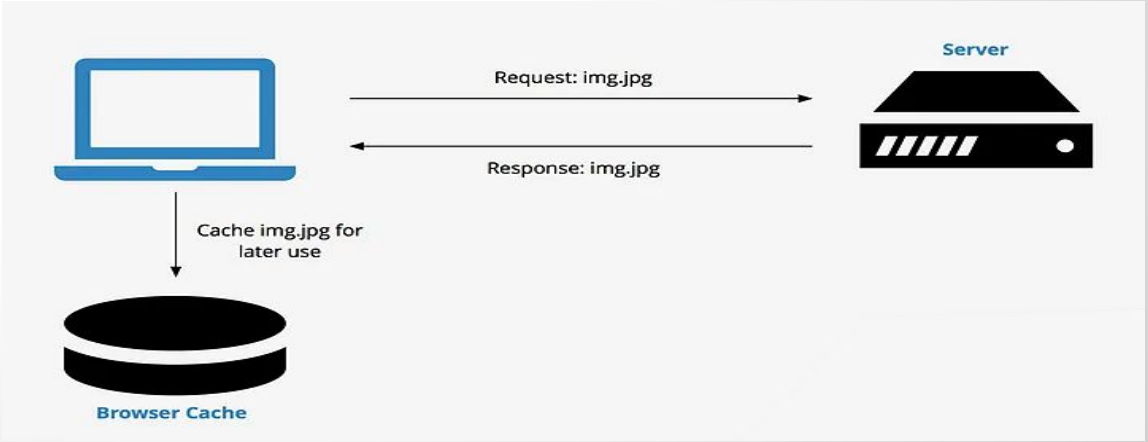
- **Client Caching**
  - browser caches URLs,HTML,CSS ,images for future uses
    - **Mozilla Firefox, Google Chrome**

- **Server-side Caching**
  - server side cache reduces load on server
    - **File System Cache**
    - **In-Memory Cache (MemCached, Redis)**

Client side Caching

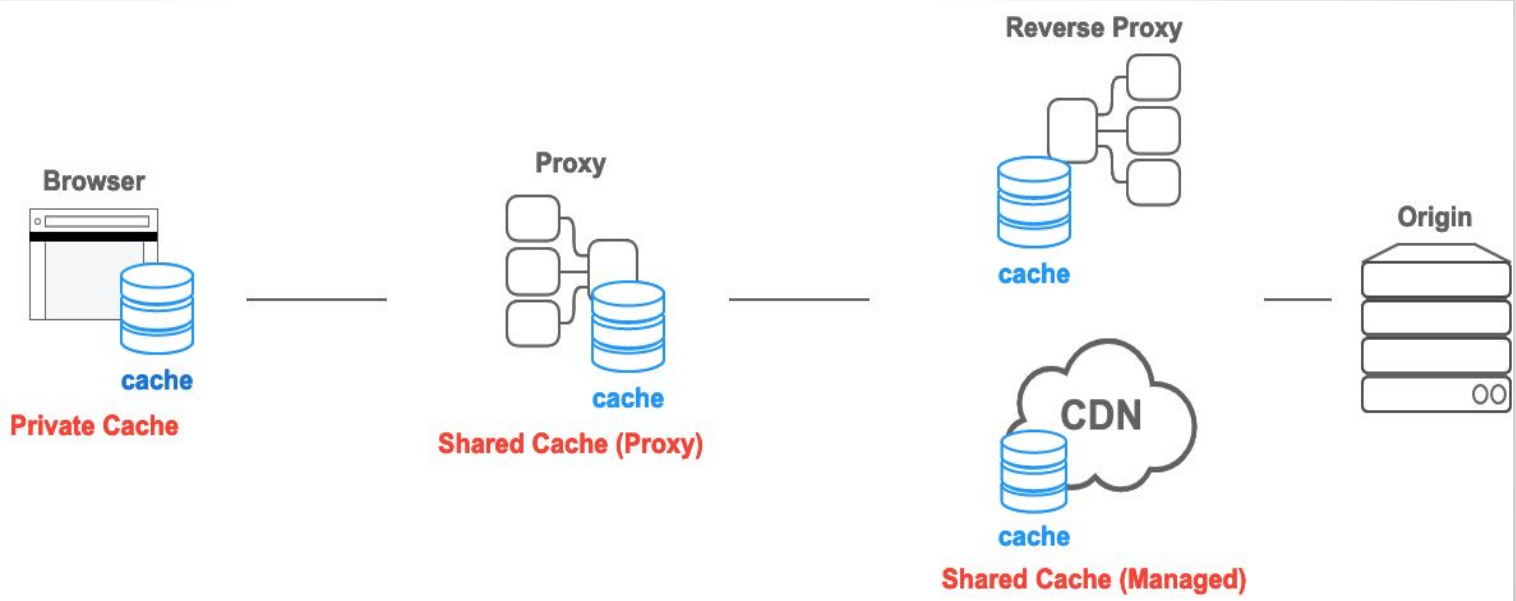
Client side caching, (browse caching): is a web-caching process that temporarily stores the copy of a web page in the browser memory instead of the cache memory in their server.

- Web browsers to store HTML, Pages, images, CSS files and other multimedia files of a website locally. Improving performance by service cached content instead of fetching it from the server for subsequent visits.



Types of Client Side Caching

- HTTP Caching: storing web resources on the client side or proxy servers to improve performance by serving cached content instead of fetching it from the origin server for subsequent request



HTTP Caching Example

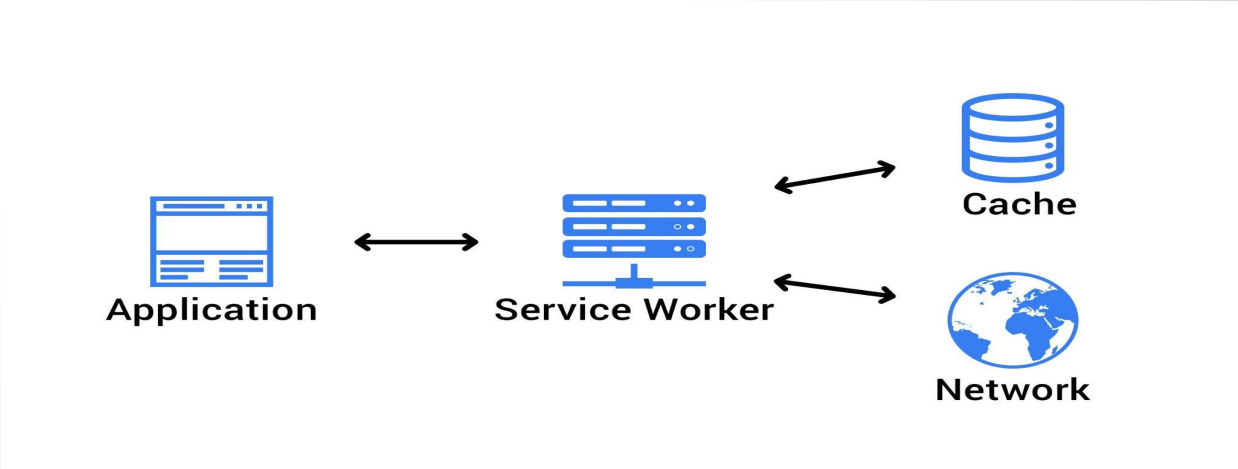
```
async function fetchData() {
    try {
      // Making an HTTP GET request to fetch data

      const response = await fetch('https://api.example.com/data', {
        // Specify caching behavior using headers
        headers: {          },
      });

      if (!response.ok) {
        throw new Error('Failed to fetch data');
      }

      // Extracting JSON data from the response
      const jsonData = await response.json();
      setData(jsonData);
    } catch (error) {
      console.error('Error fetching data:', error);
    }
}
```

A few example of headers and their role in caching websites

- Cach-control
- Expires
- Pragma (no-cache)
- Etag
- Private / public
- No-store
- max-age

Types of Client side Caching

- **Service workers:** is a script that your browser runs in the background , separate from a web page, enabling feature like push notification, background sync, or most relevantly , the ability to cache resources for offline use or faster retrieval.



Pros of Client side caching

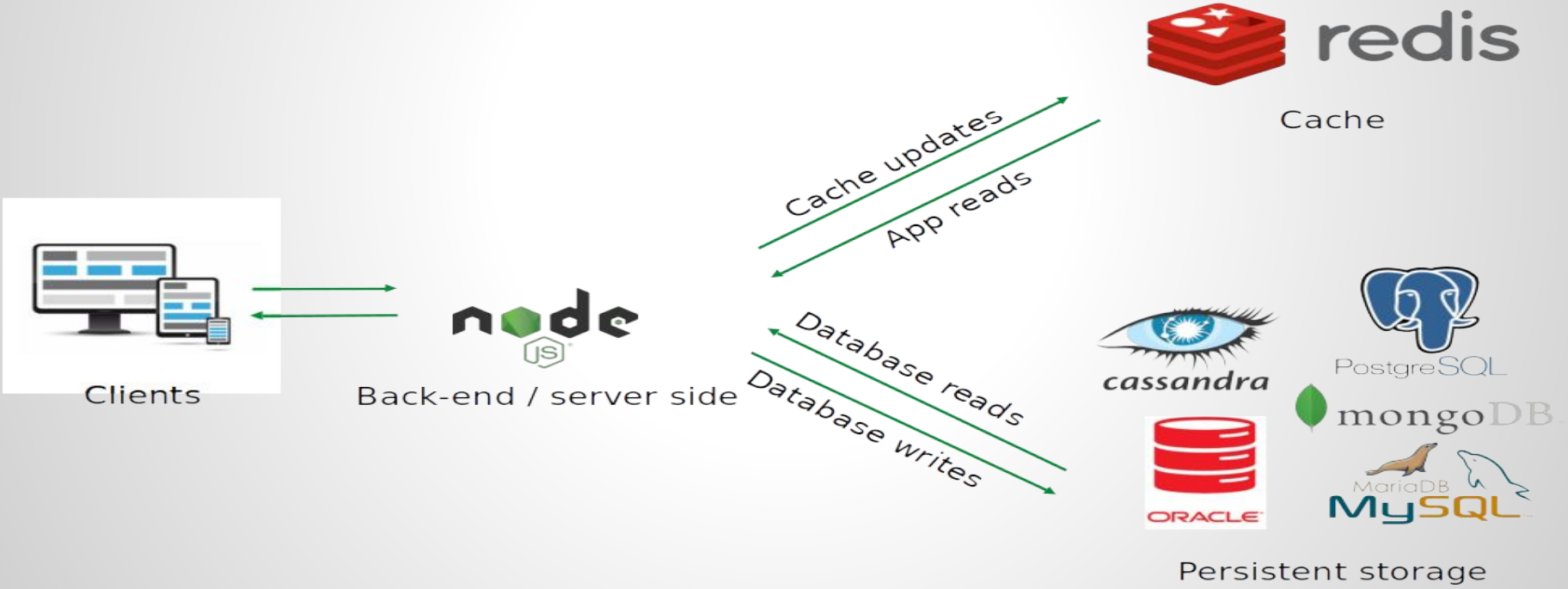
- Reduced server load
- Faster load times
- Offline access
- Improved performance

Cons of Client side Caching

- Storage Limitation
- Stale data
- Security Risks

Server Side Caching

Server side caching: stores precomputed or frequently accessed data on the server, reducing processing time and improving response speed for web applications.



How to use

[File System Caching]

- Works without any additional server most of the CMS by default use File cache without any configuration

[In-Memory cache]

- Need to install and configure server
- Need to client library to access cache
- Most of the popular framework has very good built-in for third party library
- Example Redis, memcached,....

In-Memory means We are Bound By RAM

Introduction of Redis

- **Redis** is open source, In-Memory data structure can be used as database, cache, message broker.
- NoSQL Key/Value store
- It is support atomic operation, ensuring that certain command or transaction are executed as a single indivisible unit , without interference from other clients
- Most operation have a time complexity of O(1),

Caching with Redis

```
Const redisClient = Redis.createClient({

});


(async () => {
 redisClient.on("error", (err) => {
   console.log("Redis Client Error", err);
 });


 redisClient.on("ready", (value) => console.log("Redis is ready", value));
 await redisClient.connect();
 const pong = await redisClient.ping();
})();
```

Optimization of Server side Caching

- **Use Redis Data structure wisely**：Optimize data storage using redis structure
- **Set TTL(Time-to-Live) for Keys**： implement time based key expiration for data freshness
- **Implement cache invalidation**: use a efficient mechanism to update cached data
- **Optimize memory usage**: monitor and manage memory for efficient caching

QUESTIONS / COMMENTS ..