

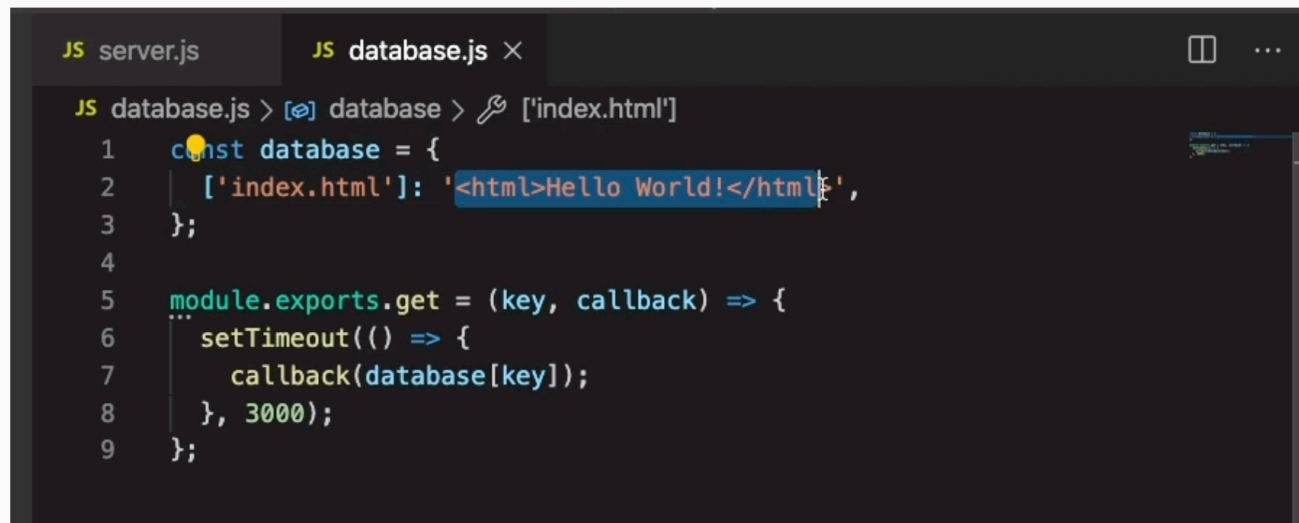
4/11/2021-

Caching

(Client-level Cache) (Server-level Cache) (Database)

In-Btw Client and Server Cache

Multiple servers



```
JS server.js JS database.js ×  
JS database.js > [?] database > 🔑 ['index.html']  
1  const database = {  
2    ['index.html']: '<html>Hello World!</html>',  
3  };  
4  
5  module.exports.get = (key, callback) => {  
6    ...  
7    setTimeout(() => {  
8      callback(database[key]);  
9    }, 3000);  
10 }
```

```
JS server.js × JS database.js
JS server.js > ...
1  const database = require('./database');
2  const express = require('express');
3
4  const app = express();
5  const cache = {};
6
7  app.get('/nocache/index.html', (req, res) => {
8    database.get('index.html', page => {
9      res.send(page);
10    });
11  });
12
13  app.get('/withcache/index.html', (req, res) => {
14    if ('index.html' in cache) {
15      res.send(cache['index.html']);
16      return;
17    }
18
19    database.get('index.html', page => {
20      cache['index.html'] = page;
21      res.send(page);
22    });
23  });
24
25  app.listen(3001, function() {
26    console.log('Listening on port 3001!');
27  });
```

Write-through cache/Write_back cache

Consistency/Staleness of data

3 Prerequisites

Latency

The time it takes for a certain operation to complete in a system. Most often this measure is a time duration, like milliseconds or seconds. You should know these orders of magnitude:

- **Reading 1 MB from RAM:** 250 μ s (0.25 ms)
- **Reading 1 MB from SSD:** 1,000 μ s (1 ms)
- **Transfer 1 MB over Network:** 10,000 μ s (10 ms)
- **Reading 1MB from HDD:** 20,000 μ s (20 ms)
- **Inter-Continental Round Trip:** 150,000 μ s (150 ms)

Throughput

The number of operations that a system can handle properly per time unit. For instance the throughput of a server can often be measured in requests per second (RPS or QPS).

Memory

Short for **Random Access Memory (RAM)**. Data stored in memory will be lost when the process that has written that data dies.

5 Key Terms

Cache

A piece of hardware or software that stores data, typically meant to retrieve that data faster than otherwise.

Caches are often used to store responses to network requests as well as results of computationally-long operations.

Note that data in a cache can become **stale** if the main source of truth for that data (i.e., the main database behind the cache) gets updated and the cache doesn't.

Cache Hit

When requested data is found in a cache.

Cache Miss

When requested data could have been found in a cache but isn't. This is typically used to refer to a negative consequence of a system failure or of a poor design choice. For example:

If a server goes down, our load balancer will have to forward requests to a new server, which will result in cache misses.

Cache Eviction Policy

The policy by which values get evicted or removed from a cache. Popular cache eviction policies include **LRU** (least-recently used), **FIFO** (first in first out), and **LFU** (least-frequently used).

Content Delivery Network

A **CDN** is a third-party service that acts like a cache for your servers. Sometimes, web applications can be slow for users in a particular region if your servers are located only in another region. A CDN has servers all around the world, meaning that the latency to a CDN's servers will almost always be far better than the latency to your servers. A CDN's servers are often referred to as **PoPs** (Points of Presence). Two of the most popular CDNs are **Cloudflare** and **Google Cloud CDN**.