HTTP Caching

like and share

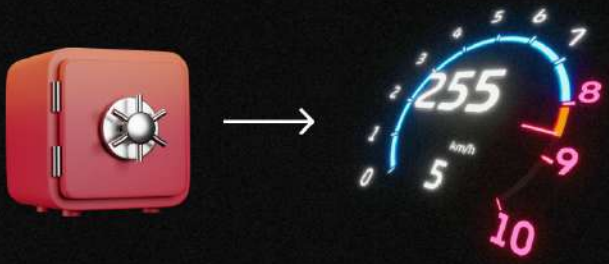
codewithsloba.com

Swipe →

Introduction

1. Caching is a technique used to store and reuse previously fetched resources, improving web performance and reducing server load.

2. Caching in HTTP involves storing resources locally to minimize network traffic and decrease response times.



How Caching Works in HTTP:

1. Caching occurs at different levels, including

   client-side | server-side

   shared caching

   and involves using caching headers to instruct the client on how to cache resources.

2. Caching headers like

   Cache-Control | Expires | ETag

   are used to provide directives on caching behavior, such as setting expiration dates or unique identifiers for resources.

Types of Caching:

1. Client-side caching stores resources locally on the user's device, reducing server load and improving performance by reusing resources.

2. Server-side caching involves caching resources on the server to serve multiple requests efficiently and reduce response times.

3. Shared caching is used in distributed systems where multiple servers can access the same cache.

Swipe →

Caching Headers

1. Cache-Control header provides directives to the client on caching behavior, including max-age, no-cache, no-store, and must-revalidate.

2. Expires header sets an expiration date for a resource, after which the client must revalidate it with the server.

3. ETag header provides a unique identifier for a resource, allowing the client to validate the cached copy with the server.

Caching Strategies:

1. Freshness-based caching involves caching resources for a specific period of time, improving performance by serving cached resources until they expire.

2. Validation-based caching revalidates resources with the server to check if the cached copy is still valid, reducing unnecessary data transfers and ensuring that the client receives the latest version of the resource.

3. Fallback-based caching uses a fallback or default resource when the requested resource is not available in the cache.

Tips for Optimizing Caching Performance:

1. Consider the size of resources and cache storage limits to ensure efficient caching and avoid unnecessary cache evictions.

2. Leverage content delivery networks (CDNs) to cache resources closer to the end-users, reducing latency and improving performance.


codewithsloba.com
Get a weekly digest of my tips and tutorials by subscribing now.
Code with Sloba
Educator • Senior JavaScript Developer • YouTuber