What is caching, and what are the common caching strategies?

1 November 2026 · 5 min read · blogs.iictp.org/what-is-caching

Caching stores a copy of data in a faster place so that repeat requests are answered without redoing the slow work. It improves speed and reduces load, at the cost of the copy sometimes being out of date.

TL;DR The short version

- Caching keeps a copy of data somewhere faster so repeat requests skip the slow work, at the cost of sometimes being out of date.
- Common strategies are cache-aside, write-through and write-behind, with expiry and invalidation to keep data fresh.
- Guard against stampedes when popular entries expire, and do not cache data that must always be current, such as a balance at payment.

Read online, listen to it, or change the text size: https://blogs.iictp.org/what-is-caching

Where can caching happen?

In the browser, at a content delivery network near the user, in the application's memory, in a shared cache such as an in-memory store, and in the database itself. Each layer trades reach and freshness differently.

What is cache-aside?

The application checks the cache first. On a miss it reads the database, stores the result in the cache and returns it. When the data changes, the application deletes or updates the cached copy. It is the most common pattern.

What are write-through and write-behind?

In write-through, every write goes to the cache and the database together, so the cache stays fresh. In write-behind, writes go to the cache first and are saved to the database later, which is fast but can lose data if the cache fails.

How do you keep cached data fresh?

Set a time to live so entries expire, and invalidate an entry when its source changes. Use both: the expiry limits how stale an entry can get if an invalidation is missed.

What can go wrong?

A popular entry expiring can send many requests to the database at once, called a stampede, which a single refresh or a stale copy can prevent. Personal data cached where others can read it, or keys that omit something that changes the value such as language, serve the wrong content.

When should you not cache?

When data must always be current, such as an account balance at the moment of payment, or when the cost of a wrong value is high. Make sure the system still works, if slower, when the cache is empty.

Related credential: Associate Technical Practitioner. Explore the exam facts and syllabus.

Sources and further reading

1. IETF. RFC 9111, HTTP Caching. rfc-editor.org/rfc/rfc9111.
2. Kleppmann, M. Designing Data-Intensive Applications. O'Reilly, 2017.

Published by IICTP, an independent technical credential body, on 1 November 2026. The current version of this post is at https://blogs.iictp.org/what-is-caching. More guides: blogs.iictp.org.