Resilience patterns cheat sheet: 12 patterns, when to use them and when not to

Four families · each card: micro-diagram, use when, avoid when · numbers match the sections of the article · November 2026

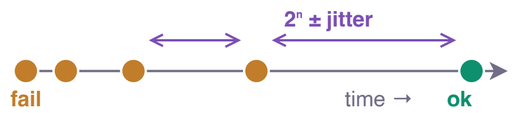
A · Bound the wait

1 Timeouts and deadlines



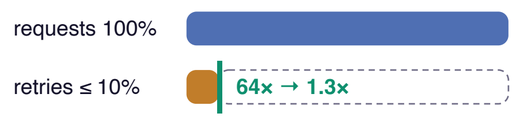
Use when: every network call. Derive from the callee's p99.9; pass the remaining deadline downstream.
Avoid when: you keep library defaults, or an inner timeout is longer than the outer one.

2 Retries with backoff and jitter



Use when: transient faults on idempotent calls. Capped exponential backoff with full or decorrelated jitter.
Avoid when: the call is a non-idempotent write, the error is a 4xx, or every layer retries too.

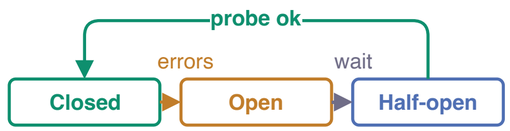
3 Retry budgets



Use when: always, next to per-request limits. Retries at most ~10% of requests; retry at one layer.
Avoid when: you rely on per-request limits alone: 3 retries at 3 layers is 64 calls at the bottom.

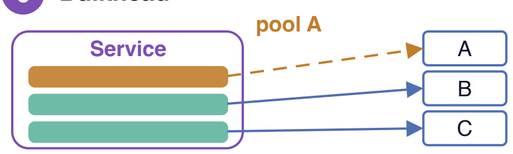
B · Contain the failure

4 Circuit breaker



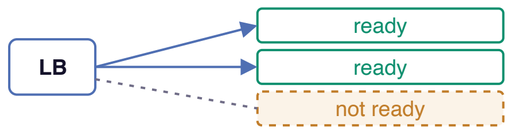
Use when: a dependency is failing or slow. Fail fast, give it room to recover, probe in half-open.
Avoid when: samples are tiny, one breaker spans many hosts, or nothing sensible happens when it opens.

5 Bulkhead



Use when: one slow dependency or tenant could drain shared threads, connections or memory.
Avoid when: pools get so small that normal peaks queue, or nobody watches rejections per pool.

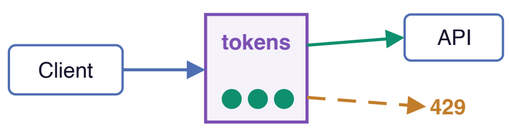
6 Health checks



Use when: readiness gates traffic; liveness only detects a stuck process. Keep both cheap.
Avoid when: liveness checks dependencies: one shared outage then restarts the whole fleet.

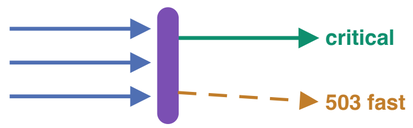
C · Protect capacity

7 Rate limiting



Use when: protecting a shared API from any single client. Per-client quotas; 429 with Retry-After.
Avoid when: it is your only overload defence. Quotas are static; real capacity is not.

8 Load shedding



Use when: the server is past capacity. Reject early and cheaply, lowest-priority traffic first.
Avoid when: rejection happens after the expensive work, or health checks get shed too.

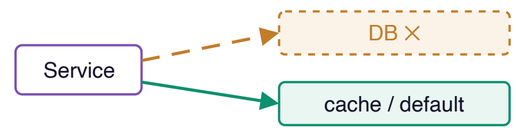
9 Queue-based load levelling



Use when: bursty asynchronous work. Bounded queue, drop expired items, scale workers on queue age.
Avoid when: a synchronous caller waits behind an unbounded queue: latency hides, then explodes.

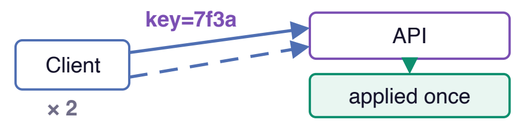
D · Fail soft, retry safely

10 Fallback and degradation



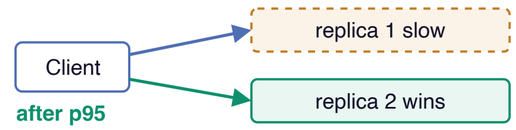
Use when: a cached, default or reduced answer beats an error: stale prices labelled 'as of'.
Avoid when: the fallback calls another fragile service, or has never been exercised in a test.

11 Idempotency keys



Use when: retrying writes. Client sends a unique key; server stores the outcome and replays it.
Avoid when: keys contain personal data, or expire before the client's retries have ended.

12 Hedged requests



Use when: tail latency on idempotent reads. Send a second copy after ~p95, cancel the loser.
Avoid when: the slowness is overload (hedges add load exactly then), or the call is a write.

Legend Request flow · Control / safe path · Failure or overload · Rejected or fallback · 1 Section number in the article

conceptualise.de · CC BY 4.0