

Uitleg in het Nederlands

Belangrijkste Overwegingen voor het Bouwen van een Hotelbeds Cache API

1) Architectuur & Datamodel:

- Normaliseer hotels, kamers, board types, tariefplannen, promoties en policies.
- Scheid statische content (beschrijving, foto's) van dynamische data (beschikbaarheid, prijzen).
- Sla zowel netto- als brutoprijzen op, inclusief duidelijke vlaggen voor belastingen en toeslagen.

2) Cache Keys & Indexering:

- Gebruik deterministische sleutels met occupancy hashes en rateKeys.
- Houd aparte indexen bij voor minimumprijzen, tariefdetails en dagelijkse beschikbaarheid.

3) TTLs & Verversing:

- Verschillende TTL's: content (24–72u), beschikbaarheid (2–10 min), valutakoersen (1–4u).
- Gebruik Stale-While-Revalidate en pre-warm veelgebruikte zoekopdrachten.

4) Prijs Correctheid:

- Altijd valideren met CheckRates voor boeken.
- Stapelbare kortingen expliciet verwerken.
- Correct afronden en rekening houden met LOS (length of stay), CTA (closed to arrival), CTD (closed to departure).

5) Prestaties & Capaciteit:

- Redis + optionele in-memory LRU cache.
- Payloads comprimeren en reads/writes pipelinen.
- Doel: P95 latency < 150ms bij cache hits.

6) Invalidatie & Consistentie:

- Ongeldige rateKeys selectief invalideren bij mismatch.
- Kortere polling voor data op korte termijn.
- Timestamps en versies bijhouden.

7) Betrouwbaarheid & Rate Limiting:

- Leaky bucket per endpoint, retries met jitter, circuit breaker.
- Fallback naar cache-only als upstream down is.

8) Randgevallen:

- Bezetting met kinderen vereist leeftijden.
- Meerdere kamers en cross-policy validatie.
- Multi-currency opslag met FX timestamp.

9) Observability:

- Metrics: hit/miss, latency, CheckRates verschil.
- Tracing met correlatie-ID's.
- Gestructureerd loggen, geen PII.

10) Beveiliging:

- Vault voor API keys, IP-whitelisting.
- Inputvalidatie om cache poisoning te voorkomen.

11) Testing:

- Golden files, property-based tests, load tests.
- Canary releases met shadow traffic.

12) Praktisch Voorbeeld:

- Keys: min:{dest}:{date}:{occ}, rates:{hotel}:{date}:{occ}, rate:{rateKey}.
- TTLs: min = 5 min, rates = 3–10 min, rate = 15 min, content = 48u.

Explanation in English

Key Considerations for Building a Hotelbeds Cache API

1) Architecture & Data Model:

- Normalize hotels, rooms, board types, rate plans, promotions, and policies.
- Separate static content (descriptions, photos) from dynamic data (availability, pricing).
- Store both net and gross prices with clear tax and fee flags.

2) Cache Keys & Indexing:

- Use deterministic keys with occupancy hashes and rateKeys.
- Maintain separate indexes for minimum prices, rate details, and daily availability.

3) TTLs & Refreshing:

- Different TTLs: content (24–72h), availability (2–10 min), FX rates (1–4h).
- Use Stale-While-Revalidate and pre-warm common searches.

4) Price Correctness:

- Always validate with CheckRates before booking.
- Handle stackable discounts explicitly.
- Ensure correct rounding and handle LOS (length of stay), CTA (closed to arrival), CTD (closed to departure).

5) Performance & Capacity:

- Redis + optional in-memory LRU cache.
- Compress payloads and pipeline reads/writes.
- Target: P95 latency < 150ms for cache hits.

6) Invalidations & Consistency:

- Invalidate rateKeys selectively on mismatch.
- Shorter polling for near-term dates.
- Track fetch timestamps and versions.

7) Reliability & Rate Limiting:

- Leaky bucket per endpoint, retries with jitter, circuit breaker.
- Fallback to cache-only when upstream is down.

8) Edge Cases:

- Occupancy with children requires ages.
- Multi-room and cross-policy validation.
- Multi-currency storage with FX timestamp.

9) Observability:

- Metrics: hit/miss, latency, CheckRates delta.
- Tracing with correlation IDs.
- Structured logging, no PII.

10) Security:

- Vault for API keys, IP whitelisting.
- Validate input to avoid cache poisoning.

11) Testing:

- Golden files, property-based tests, load tests.
- Canary releases with shadow traffic.

12) Practical Example:

- Keys: min:{dest}:{date}:{occ}, rates:{hotel}:{date}:{occ}, rate:{rateKey}.
- TTLs: min = 5 min, rates = 3–10 min, rate = 15 min, content = 48h.