

Requirements.

Functional:

- Short URL should redirect to the longer version when accessed. Mark the HTTP response as 30x (i.e. a permanent redirect).
- No de-duplication.
- URLs with TTL (i.e. time to live).
- Long-term:
 - Private URLs.
 - Analytics or performance metrics: for e.g., number of clicks.

Non-functional:

- High available but prefer reads over writes if need to make a tradeoff.

Assumptions:

- 100M users.
- 10 URLs created per user per month.
- 100:1 create to read ratio.
- Average URL size: 100 bytes.
- 80:20 cache storage.

Storage requirements:

- $100 * 10 * 100M / 10^9 = 100$ GB per month.
 - 6 TB for 5 years.
 - Cold storage.
- $0.8 * 100 = 80$ GB.

Request rate:

- Writes: $10 * 100M / 2.5 M = 400$ rps.
- Reads: 40k rps.

- CreateShortUrl
 - Request:
 - User id.
 - Long URL.
 - Idempotency token.
 - [Optional] Expiration time.
 - Response:
 - Short URL.
- GetShortUrlRedirectionDetail
 - Request:
 - [Optional] User id.
 - Short URL.
 - Response:
 - Long URL.

Frontend service:

- Fronted by a load balancer.
- Authentication and authorization.
- Interacts with the DB and cache.

Cache:

- When to write to cache?
 - Write on read if missing.
 - 1-hit wonders.
 - Most reads happen soon after writes.
- Key eviction: LRU.

Key generation:

- On-demand:
 - Hash(long URL, user id, idempotency token).
 - MD5 or SHA - 128 bits.
 - 6 char & [a-zA-Z0-9_.] (i.e 64 characters) -> $64^6 = 68B$.
 - 6 bits per character. So, 36 bits.
- Generate keys offline:
 - Start:
 - Key Id.
 - User-id: null.
 - Creation timestamp:
 - By frontend service:
 - Key Id.
 - User-id: 123.

Sharding the data:

- Shard the database on key level.

