**Chapter 8**

Design a URL Shortener

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sample Interview**   |  |  | | --- | --- | | **Questions** | **Answers** | | What is the traffic volume? | 100 million URLs per day | | How long is the shortened URL? | As short as possible | | What characters are allowed in the shortened URL? | A combination of numbers (0-9) and characters (a-z, A-Z) | | Can shortened URLs be deleted or updated? | For simplicity, assume shortened URLs cannot be deleted or updated | | How long do we expect the service to stay operational? | 20 years |   **Back-of-the-envelope Estimation**   * Write operation per second = 100 million/24/3600 = 1160 URLs/second * Read operation   + Assuming the ratio of read to write operation is 10:1   + Read operation per second will be 11,600 URLs/second * Amount of lifetime records   + 100 million/day \* 365 day/year \* 20 years = 730 billion records * Storage   + Assuming average URL length is 100 bytes   + 365 billion records \* 100 bytes/record = 36.5 TB |

**High Level Design**

API Endpoints – facilitates the communication between clients and servers

URL shortener requires 2 API

* URL shortening – accepts a long URL and return a short URL
  + POST request
  + Request parameter – original long URL
  + Return – short URL
* URL redirecting – using the shortened URL will redirect the link to the original URL
  + GET request
  + Return – original long URL in status code 301 or 302

*URL Redirecting*

* Most intuitive way to store shortURL and longURL pair in hash tables
* A request using the shortened URL will go to a server to retrieve the original URL

A diagram of a computer

Description automatically generated

301 redirect – request URL is permanently moved to the long URL

* Browser caches the response
* Subsequent requests for the same URL will not be sent to the URL shortening service
* Advantages – reduces server load

302 redirect – temporarily moved to the long URL

* No caching, subsequent request will visit the shortening service every time
* Advantages – easy to track click rate and traffic metrics

*URL Shortening*

* Hash function must satisfy the following requirements
  + Each longURL must be hashed to one hash value
  + Each hash value can be mapped back to the longURL

**Hash Function Deep Dive**

* Store shortURL, longURL mapping in a relational database

*Hash value length*

* Counting the characters gives the amount of possible characters
  + 10 + 26 + 26 = 62 possible characters
* Find the smallest n such that

|  |  |  |  |
| --- | --- | --- | --- |
| n | Equation | Maximum number of URLs | Estimation |
| 1 |  | 62 | - |
| 2 |  | 3,844 | 3,800 |
| 3 |  | 238,328 | 230,000 |
| 4 |  | 14,776,336 | 14 million |
| 5 |  | 916,132,832 | 916 million |
| 6 |  | 56,800,235,584 | 56 billion |
| 7 |  | 3,521,614,606,208 | 3.5 trillion |

* 3.5 trillion is more than enough to cover their estimated 730 billion URLs
* 2 types of hash functions for URL shortener
  + Hash + collision resolution
  + Base 62 conversion

*Hash + collision resolution*

* Concept
  + Apply a well-known hash function like CRC32, MD5 and SHA-1
  + Collect the first sets of characters up to the number you need in length
* Challenges
  + Hash collisions - two pieces of data sharing the same hash value
    - Eliminate collisions by recursively append a new predefined string until no more collisions are discovered
  + Expensive to query from database every recursion
    - Bloom filter can improve that performance

*Base62 Conversion*

* Base conversion converts the same number between its different number representation systems
* Base62 Conversion used for 62 possible characters for hash value



A close-up of a card

Description automatically generated

*Comparison of the two approaches*

|  |  |
| --- | --- |
| Hash + collision resolution | Base62 conversion |
| Fixed short URL length | URL length goes up with the ID |
| It does not need a unique ID generator | Depends on a unique ID generator |
| Collision is possible and must be resolved | Collision is impossible because ID is unique |
| Impossible to figure out the next available short URL because it does not depend on ID | It is easy to figure out the next available short URL if ID increments by 1 for a new entry. This can be a security concern |