TinyURL

(1) Functional Requirements

- System should be able to create a custom URL which is short enough to copy and paste
- User should be able to select the custom URL
- User should be redirected to actual URL when accessed
- User should be able to specify the expiry time.

Non-functional Requirements

- Shortened URLs should not be predictable
- URL redirection should happen in real-time with minimum latency
- Security
- Highly available.
- scalability.

Back of Envelope Calculation:-

(1) Traffic: Read : write = 100 : 1

Per month new URL req(write) = 500M

Per month read req = $500 \times 10 \times 10^6 = 5 \times 10^9 = 5$GB.

Per sec = $\frac{500M \times 12}{3600 \times 24 \times 30}$ = 192 write req/sec

= $\frac{5 \times 10^9}{3600 \times 24 \times 30}$ = 19K read req/sec.

(2) Storage: 500 M req $\times$ 12 $\times$ 5 (years) $\times$ 500 B

= 15TB

(3) Bandwidth: 500 Bytes $\times$ 200 req per sec = 100 KB/s write req

20K $\times$ 500B = 10 MB/sec.

(4) Memory (cache) :- 20K $\times$ 3600 $\times$ 24 $\times$ 0.2 $\times$ 500B = 170 GB.

APIs

1. createURL(user_id, url, expiry_date, custom_url)
2. deleteURL(url)

Database:-

User
user_id
email
last_login

URL
custom_url
original_url
expiry_date
user_id

Generation of unique URL:-

1st Approach: Assuming the unique url to be 6 character length,

Create a hash of 128 bits ⇒ convert it to Base62 encoding

⇒ Choose randomly 5 digits

⇒ Append userId

This approach might give same url if two users are generating url at the same time.

Efficient Approach:- Generate 62^6 Keys offline and store them

Actual Design: