Url Shortener

An application that helps in converting the long unreadable url to be in shorter format for sending notifications or for other purpose.

Features

Functional	Non Functional
Shortner	High Availability
Redirection	Low Latency
Url Expiry	Url not Predictable / No collision

Estimating the system Capacity

Storage

- **How much read and write calls will be here**
  Estimating => 100 : 1 [Read : Write]
  Every Month New Url Creation = 1,000,000 (1 Million)
  Every Month URL Redirection = 100,000,000 (100 Million)

- **Query Per Second (qps)**
  qps = 100,000,000 / (30 * 24 * 3600) **~ 50 qps**

- **Every Month New user =** 1,000,000
  Expiry = 10 years
  Total Url in 10 years = 1,000,000 * 12 * 10

- **Size of a URL**
  1 URL Size = 500 Bytes
  Size of Total Urls in 10 years = 1,000,000 * 12 * 10 * 500 { in Bytes } => 60 Gb [**Storage**]

- **Caching**
  We will keep it in RAM for a **TTL** of **1 Day.**
  qps = 50
  qpd = 50 * 24 * 3600 ~ 5,000,000 qpd [5 Million qpd]
  Lets Assume we will only cache 25% of he data = 1.25 Million
  1,250,000 * 500 ~ **1GB [RAM Needed]**

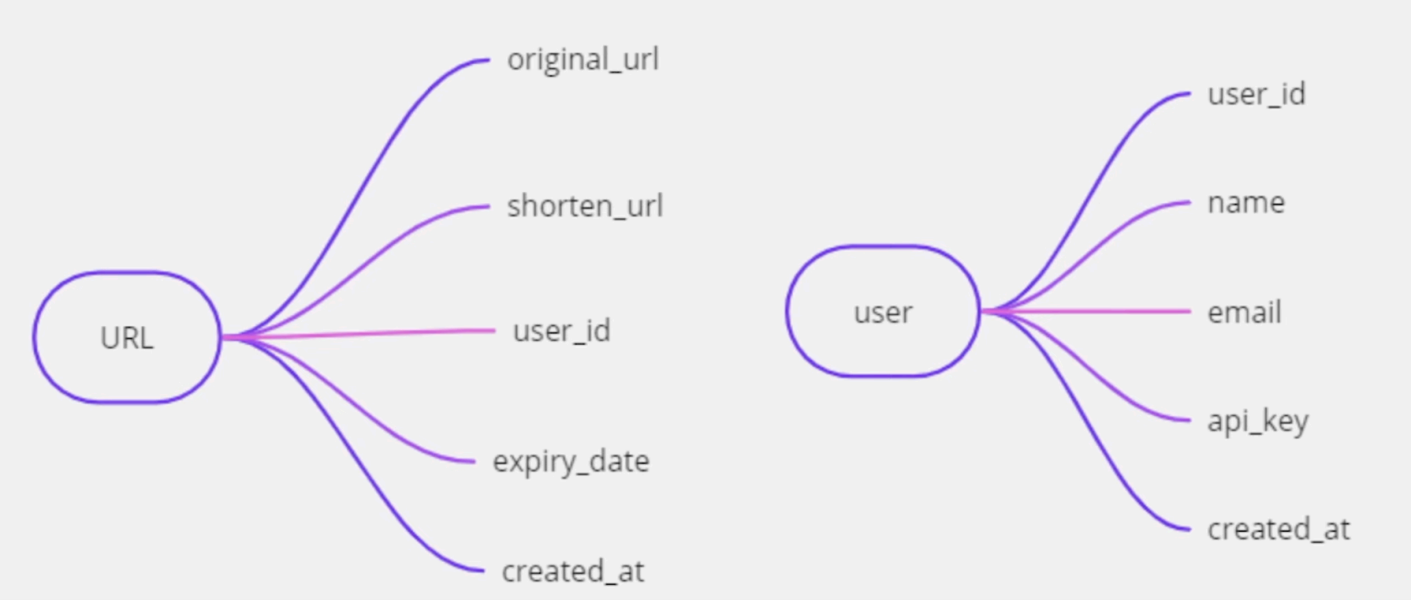
Design Goal

- Read Intensive
- High Availability
- Low Latency
- Security(not every body can shorten Url)

High Level Diagram [HLD]

- **API :**
  - To Shorten a URL
  - Get API to Delete URL
  - Sign Up
  - Login
  - Logout
  - Redirection

- **Database :** MongoDb, we need easily scalable, high available db.



Collection

- **Algorithm**

  Requirement : Send in Long URLs and converting them to short url + avoid collision

  Assumption :

  1. **MD5 hash** : It takes and gives a shortened string of 128 bits.
     We don't need a 128 bits, we need a shorten string that is unique.

  2. **Length of Url**
     Total New user in 10 Years = 1,000,000 * 12 * 10
     Character we can use to generate the short URL with less/no collisions :
     A-Z, a-z, 0-9 => 62 Characters

```
1 - 62
2 - 62 x 62
3 - 62 x 62 x 62
.
.
.
n - 62^n
```

So, here we have a probability of 62^n , where n = number of characters & 62^n = unique combination

So, 62^n **>** Total New user in 10 Years = 1,000,000 * 12 * 10

After taking log both sides,
Length => n = log 64(1,000,000 * 12 * 10) ~ **n = 4.5 ~ 5**

`We can take 5 as the length of the string, but we generally take 7, and also the combination will also increase and collision will decrease.`

Taking **Length = 7**
62 ^ 7 = 3.5 Trillions

So we can take Base 62, instead of MD5 hashing.

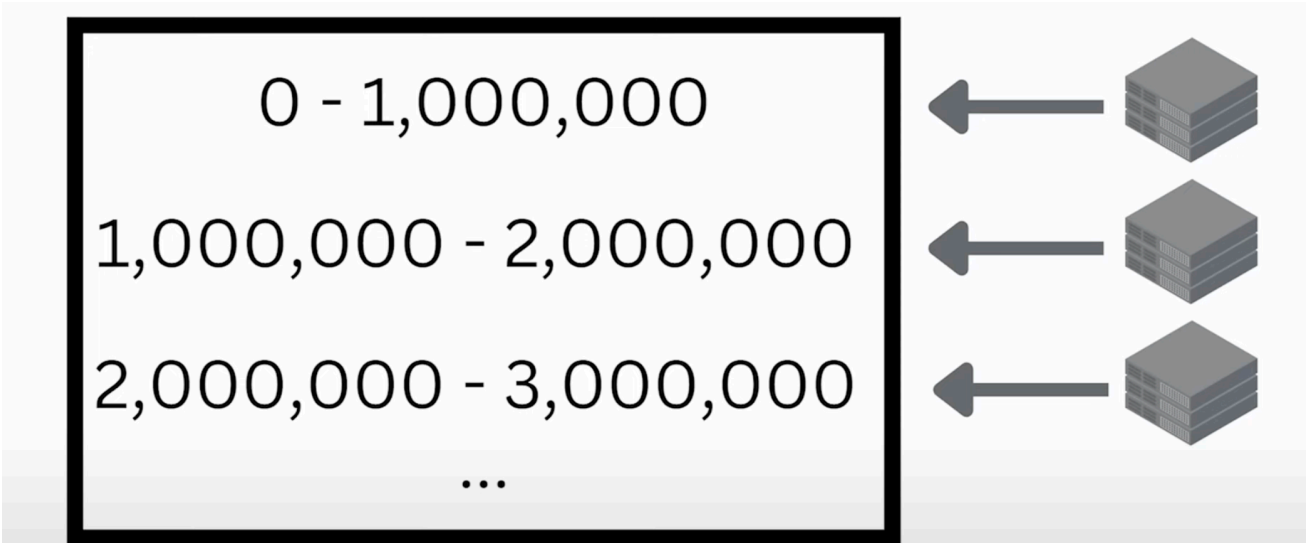
Formula of Base 62 : It takes in a number and gives output as a string

Number → Base 62 → AaZz12

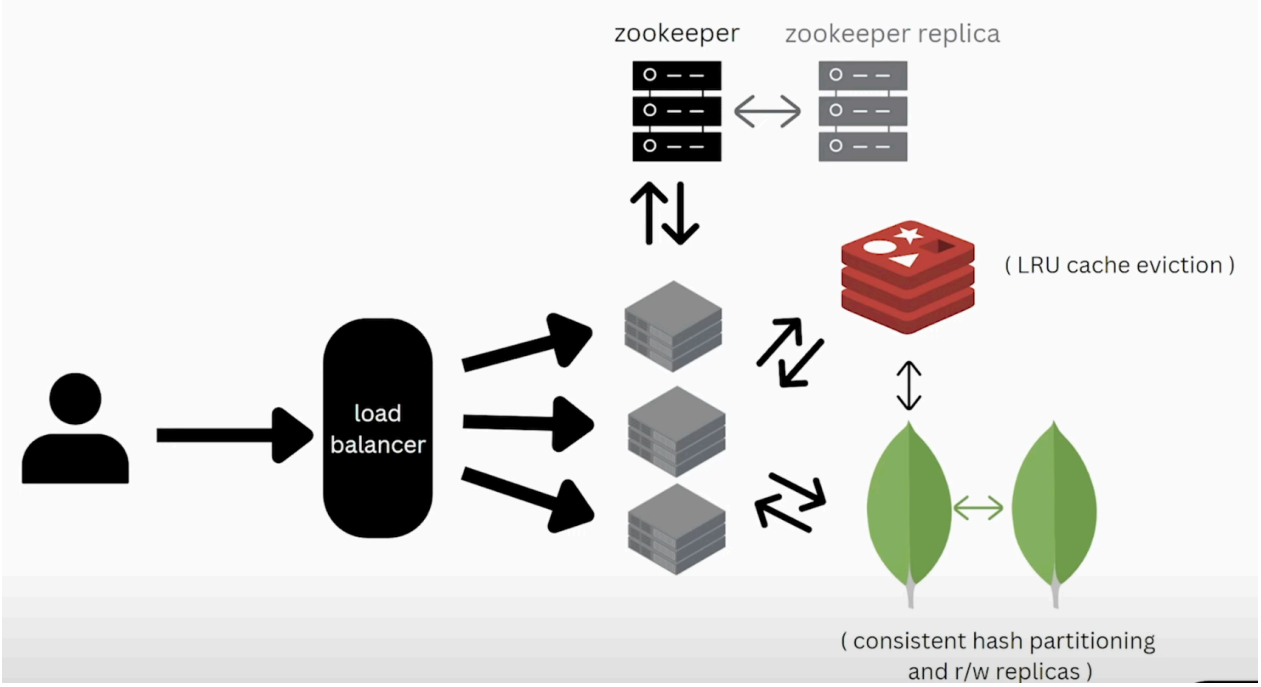
Note : We have a number as an input in the Base 62 and we only have a incoming String.

So to get the number to be inputted and get the short Url, we will be keeping the configuration in some database.

- **Mysql** : As we don't have mysql, we can't take it.
- **Redis :** We can consider this, we can take a counter of the incoming url and save it to redis, but here are some issues with redis:
    - Single Point of Failure
    - If we distribute redistribution, then also we will have sync issues withh all of the redis.
    - Scalability issue, if we need to increase any distribution, its an issue.
- **Apache Zookeeper :** We can use this, as its main focus is to store the configuration in distributed systems with telling zookeeper to range each service, and further to scale we can add a new range with its configs.



Caption



High Level Diagram For Url Shortener