**Design a service like TinyUrl**

**1. Functional requirements:**

1)Get Short URL

2) Redirect to long URL

**2. Non-Functional requirements:**

1)Very low latency

2) Very high availability

**3. What should be length of short URL:**

**Traffic: Number of requests:**

**Ex: For example there would x requests/sec**

**X\*60\*60\*24\*365\*10 (No. of request in 10 years)**

**What are chars to be included in URL:**

**Ex: A-Z(26) + a-z(26) + 0-9(10) = 62 characters**

**If url length is = 7 then total url: 62^7 (2.5 trillion)**

There are three layers: API, API layer, Persistence layer

Memory Cache

Data storage/SQL database

Load Balancer

Client/Customer



Web server\_1/short URL service

Web server\_2/short URL serve=ice

**APIs required:**

1. createTinyURL(long url)-> returns tiny url
2. getLongURL(tiny url) -> returns long url

-------------------------------------------------------------------------------------------------------------------------------

Basically client is communicating to server/service using REST API or other methods, which will be received by load balancer. Load balancer is the front end for server/service and load balancer will redirect the request to web servers. Web servers will call write/read API to communicate with Data storage at persistence layer.

**How to generate tiny URL**:

1. Total possible characters: A-Z(26) + a-z(26) + 0-9(10) = total 62
2. If we have to use only 7 characters then it will 62^7 characters in tiny url
3. How many requests will be sent per second?? For ex: 1000 requests/sec

**Load Balancer**:

1. Load balancers improve application performance by increasing response time
2. Reducing network latency.

They perform several critical tasks such as the following: Distribute the load evenly between servers to improve application performance. Redirect client requests to a geographically closer server to reduce latency.