Conclusión de infraestructura

- Cachear lo máximo posible (siempre y cuando tenga sentido) haciendo buen uso de las keys → tiempos de respuestá más rapidos
- Soportar múltiples data centers
- Hostear assets en un CDN (CDN ← → BALDE (S3 1usd + Glacier (antiguedad)0.1 tb)
- Redundancia (escalado horizontal) tanto en instancias BE como en Bases de datos
- Monitorear nuestro sistema y utilizar herramientas de automatización (despliegue con docker / kubernetes)
- Realizar test de carga de nuestros servicios para determinar minimo y máximo de instancias (pods)

Consideraciones

- La caché es rápida el disco es lento (evitar buscar en disco resulta en mejor performance / tiempo de respuesta)
- Comprimir los datos antes de enviarlos a un servidor
- Data centers deben estar en diferentes regiones (toma tiempo enviar la información entre cliente y server)

Cálculos (back of the envelope) → estimar la capacidad de un sistema

Alta disponibilidad: Es la capacidad de un sistema de estar continuamente en operación por un tiempo determinado. La alta disponibilidad se mide en % donde 100% significa (no tiene tiempo de downtime)

SLA (Server level agreement)

Availability	Downtime per day	Downtime per year
99%	14.4 minutos	3.6 días
99.9%	1.44 minutos	9hs (hagan las cuentas)
99.99%	8.6 segundos	<1hs

Ejemplo de estimación básico de **X**

Hipotesis

- ¿Cuantos usuarios activos hay mensualmente? 300 millones
- 60% de usuarios estan activos diariamente
- Un usuario publica 2 tuits por día (promedio)
- 10% de los usuarios publica contenido multimedia (promedio)
- Los datos se almacenan por 4 años

DAU (Daily active users) = 300M * 0.6 = 180M

QPS (Queries por segundo) = 180M [dia] * 2 [tweets] = 360.000.000 [tuits / dia] / (24*60*60) = 4200 tuits / segundos → minimo de 5 pods y un máximo de 15

Peak QPS (coeficiente de seguridad máximo) = 2 * QPS = 8400 tuits / segundo (minimo de 10 pods y un máximo de 30)

Media storage

tweet_id (64 bytes)	text (140 bytes)	media 1mb

¿Cuanto storage necesito para media para 4 años de data?

180M * 2 tuits / dia * 10% * 1mb = 36MILLONES mb = 36000 GB = 36 TB / dia

En 4 años → 36TB * 365 * 4 = 54 PETA BYTES

Preguntas comunes

- QPS, PeakQPS (coeficiente seguridad), storage, numero de servidores

---

Consigna: Diseñar un acortador de URL's

Pregunta: ¿Como debería funcionar este url shortener?
Respuesta: **Imagina una URL http://lvlup.tech/curso/bla al pasar por el servicio de acortar de URL's debería devolver algo como http://tinyurl.com/ha1na2**

Pregunta: ¿Que pasa cuando clickeas sobre la url acortada?
Respuesta: cuando clickeamos sobre esa url nos debería redirigir a la página original

Pregunta: ¿Cuánto tráfico esperamos al día?
Respuesta: 100 millones de URL's al día

Pregunta: ¿Cuánto tiempo debería estar almacenada la URL?
Respuesta: 2 años

Pregunta: ¿Que tan corta tiene que ser la URL?
Respuesta: Lo más corta posible (porque el espacio es platita)

Pregunta: ¿Que caracteres acepta la URL?
Respuesta: alfanumericos [0-9][a-z][A-Z]

Pregunta: ¿Qué tasa de lectura-escritura tiene la aplicacion?
Respuesta: 10-1

Casos de uso:

- Dada una URL retornar una URL acortada
- Cuando se hace click sobre una url acortada redirigir a la original
- Se pide alta disponibilidad, escalabilidad y tolerancia a fallas

Cálculos:

QPS (escritura)
100 [millones/ dia] → 100.000.000 / (60*60 * 24) = 1157 req/s

QPS (leectura) = 11.157 req/s

Sabemos que la URL quedará almacenada por 2 años:

#CantidadRegistros = 100M * 365 * 2 = 73000M de registros

Pregunta: ¿Cual es la longitud promedio de las URL's que se van a guardar sin acortar?
Respuesta: 100 caracteres → cada caracter es 1 byte

Almacenamiento: 73000M * 100 bytes = 7300 GB = 7.3 TB

Adoptando un coeficiente de seguridad de 2 el almacenamiento es de 15TB

Propuesta a alto nivel del acortador

- Para crear una URL acortada el cliente enviará un POST request a un endpoint POST /api/v1/shorten. Se envía dentro de un body { url: 'http://….my-long-url' }
- Existe un servicio que se encarga de acortar la URL a la menor cantidad de caracters posibles y relacionar la url corta con la url original guardando esto en una base de datos
- Redirección: Para redirigir de una url corta a una larga realiza una petición [GET] /api/v1/:shorturl y este endpoint devuelve la URL original con un status code …

Problema 2: Diseñar un sistema de notificaciones

Consideraciones:

- Un sistema de notificaciones es más que una push en una app. Existen 3 tipos de notificaciones:
    - SMS
    - Push notification
- La notificación tiene que llegar lo antes posible al usuario pero puede no ser en tiempo real
- Las notificaciones son disparadas mediante una aplicación en el cliente y también pueden ser programadas desde el servidor
- Los usuarios pueden silenciarlas y así no recibirlas más
- El sistema debe aceptar 5 millones de notificaciones push diarias, 1 millon de SMS