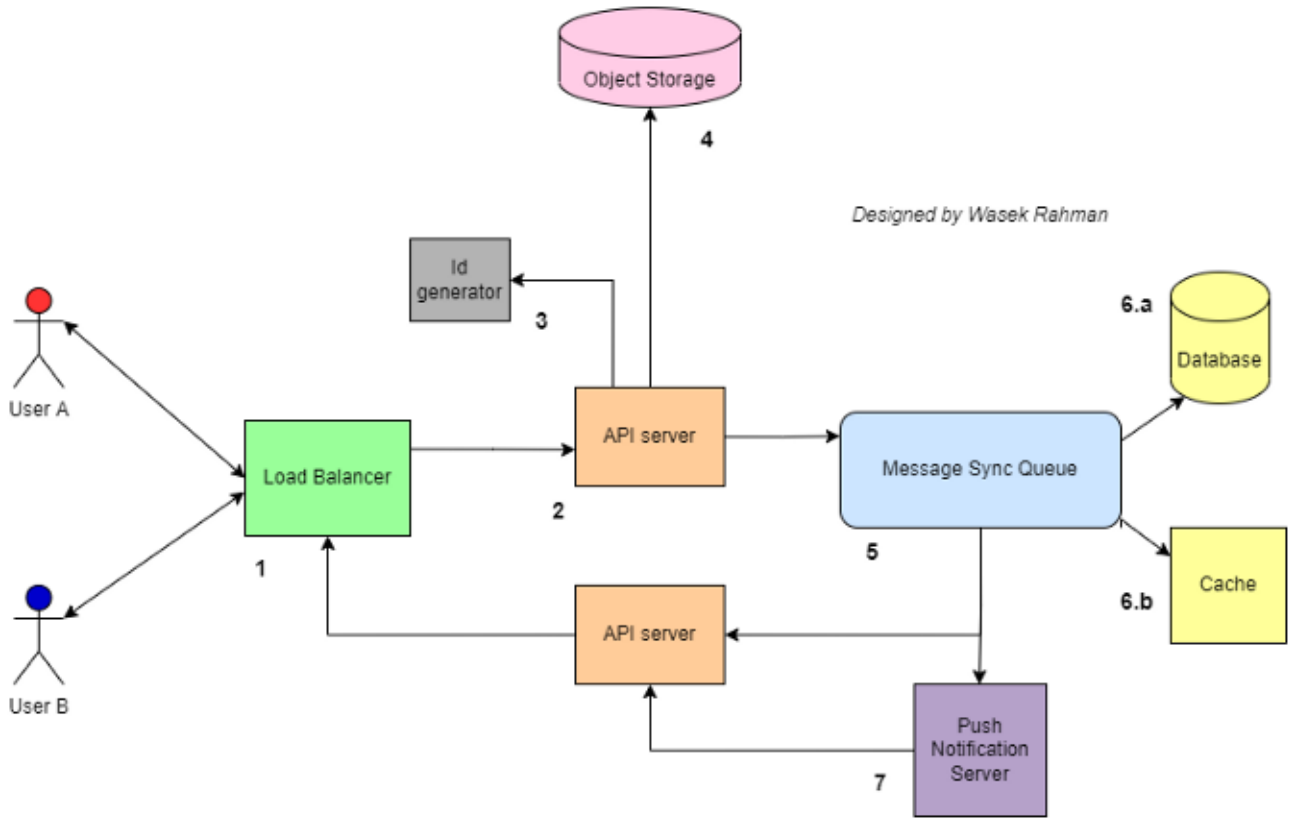
Published in CodeX

Wasek Rahman Follow
Jan 22 · 3 min read ★ · Listen

System Design Interview Prep: Real-Time Chat Application



When you start designing a system you will need to ask a few clarifying questions to your interviewer and make some assumptions.

> *In our case, we will assume —*
> *1. The user is already logged in.*
> *2. We have over a million users.*
> *3. The database is already designed.*
> *4. We are only focusing on how User A sends a message to User B.*

Okay then, let's get started. I have referenced every number with the numbers in the design for better understanding.

1. User A writes a message and hits enter. This message is retrieved by a **load balancer** which basically handles traffic to avoid overloading the server.

2. The message is then distributed to a suitable **API Server** that processes this message request. This is where the request is processed.

3. The **API Server** generates a **unique ID** if any **media files** are sent through the message. The reason for this is we want to store these media files in a cloud storage system with a unique ID and this unique ID will be stored in the database with the message details for reference.

4. The **Object Storage** is the cloud storage system where we would store all of these media files using a unique ID. The most popular cloud storage system would be **AWS S3.**

5. After the media files are stored, the message is sent to a message sync queue. Now, **message queues consist of a publishing service and multiple consumer services that communicate via a queue.** This **communication is typically one way** where the publisher will issue commands to the consumers. The publishing service will typically put a message on a queue or exchange and a single consumer service will consume this

But while the message is being sent, we make sure a couple of other things are achieved for better functionality and efficiency.

6 a. The message is stored along with the sender details such as time and the unique ID of the media file to a **database system**. For a chat application, it is preferred to have a **NoSQL database** due to scalability and efficiency.

6 b. We also make sure the message details are **cached to a temporary memory like Redis**. This is because when we want to load messages, it would be more efficient to get data from the cache rather than making requests over and over again to a database. This reduces the cost and makes loading faster.

7. Remember the pop-ups you get on your phone when someone sends you a text on a chat app? Well, the way we get that flowing is by using a **Push Notification Server.** A push notification server consists of two parts: storing device IDs and sending push notifications. To achieve this we create two endpoints: register and send.

That's it. This would be how a message is sent from **User A** to **User B**.

If you guys learnt something new, please drop a clap and follow me for more tutorials.

Recommendation : The most popular book that will help you ace any System Design interview —

System Design Interview - An Insider's Guide: Volume 2

System Design Interview - An Insider's Guide: Volume 2 [Xu, Alex, Lam, Sahn] on Amazon.com. *FREE* shipping on...

amzn.to

The tool I used for designing — https://app.diagrams.net/

Connect with me on LinkedIn — *https://www.linkedin.com/in/wasek-rahman/*