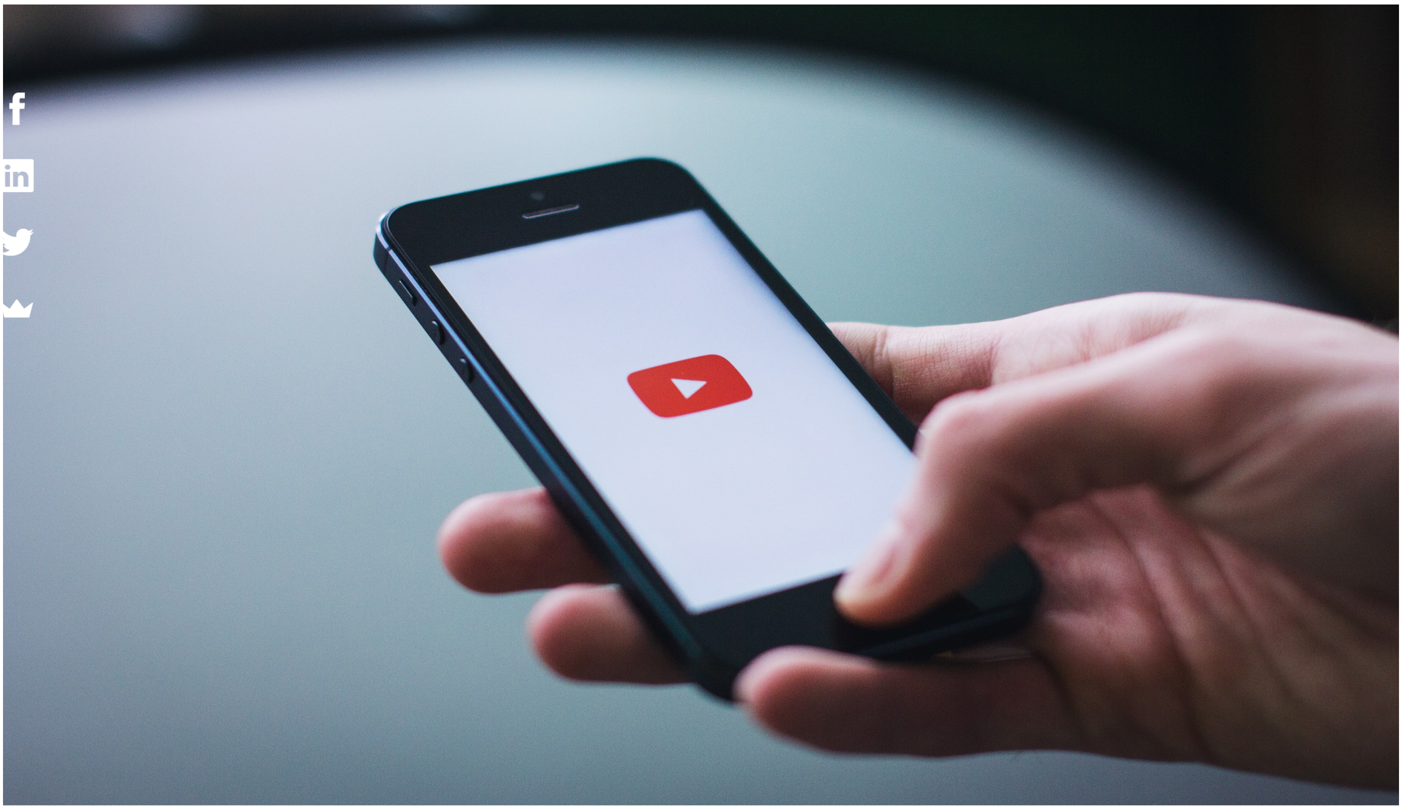

How to Design Youtube (Part I)

One of the most common types of system design interview questions is to design an existing popular system. For example, in the past, we've discussed How to Design Twitter (http://blog.gainlo.co/index.php/2016/02/17/system-design-interview-question-how-to-design-twitter-part-1/), Design Facebook Chat Function (http://blog.gainlo.co/index.php/2016/04/19/design-facebook-chat-function/) and so on so forth.

Part of the reason is that the question is usually general enough so that there are a lot of areas to discuss. In addition, if candidates are generally curious enough, they are more likely to explore how existing products are designed.

So this week, we're going to talk about how to design Youtube (http://www.youtube.com). It's a broad question because building Youtube is like building a skyscraper from scratch and there are just too many things to consider. Therefore, we'll cover most of the "major" components from interviewer's perspective, including database model, video/image storage, scalability, recommendation, security and so on.

Overview

Facing this question, most people's minds go blank as the question is just too broad and they don't know where to start. Just the storage itself is non-trivial as serving videos/images seamlessly to billions of users is extremely complicated.

As suggested in 8 Things You Need to Know Before a System Design Interview (http://blog.gainlo.co/index.php/2015/10/22/8-things-you-need-to-know-before-system-design-interviews/), it's better to start with a high-level overview of the design before digging into all the details. This is true especially for problems like this that has countless things to consider and you'll never be able to clarify everything.

Basically, we can simplify the system into a couple of major components as follows:

- Storage. How do you design the database schema? What database to use? Videos and images can be a subtopic as they are quite special to store.
- Scalability. When you get millions or even billions of users, how do you scale the storage and the whole system? This can be an extremely complicated problem, but we can at least discuss some high-level ideas.

- Web server. The most common structure is that front ends (both mobile and web) talk to the web server, which handles logics like user authentication, sessions, fetching and updating users' data, etc.. And then the server connects to multiple backends like video storage, recommendation server and so forth.
- Cache is another important components. We've discussed in details about cache before (http://blog.gainlo.co/index.php/2016/05/17/design-a-cache-system/), but there are still some differences here, e.g. we need cache in multiple layers like web server, video serving, etc..
- There are a couple of other important components like recommendation system, security system and so on. As you can see, just a single feature can be used as a stand-alone interview question.

Storage and data model

If you are using a relational database (https://en.wikipedia.org/wiki/Relational_database) like MySQL, designing the data schema can be straightforward. And in reality, Youtube does use MySQL as its main database from the beginning and it works pretty well.

First and foremost, we need to define the user model, which can be stored in a single table including email, name, registration data, profile information and so on. Another common approach is to keep user data in two tables – one for authentication related information like email, password, name, registration date, etc. and the other for additional profile information like address, age and so forth.

The second major model is video. A video contains a lot of information including meta data (title, description, size, etc.), video file, comments, view counts, like counts and so on. Apparently, basic video information should be kept in separate tables so that we can first have a video table.

The author-video relation will be another table to map user id to video id. And user-like-video relation can also be a separate table. The idea here is database normalization (https://en.wikipedia.org/wiki/Database_normalization) – organizing the columns and tables to reduce data redundancy and improve data integrity.

Video and image storage

It's recommended to store large static files like videos and images separately as it has better performance and is much easier to organize and scale. It's quite counterintuitive that Youtube has more images than videos to serve. Imagine that each video has thumbnails of different sizes for different screens and the result is having 4X more images than videos. Therefore we should never ignore the image storage.

One of the most common approaches is to use CDN (Content delivery network) (https://en.wikipedia.org/wiki/Content_delivery_network). In short, CDN is a globally distributed network of proxy servers deployed in multiple data centers. The goal of a CDN is to serve content to end-users with high availability and high performance. It's a kind of 3rd party network and many companies are storing static files on CDN today.

The biggest benefit using CDN is that CDN replicates content in multiple places so that there's a better chance of content being closer to the user, with fewer hops, and content will run over a more friendly network. In addition, CND takes care of issues like scalability and you just need to pay for the service.

Popular VS long-tailed videos

If you thought that CDN is the ultimate solution, then you are completely wrong. Given the number of videos Youtube has today (819,417,600 hours of video (https://www.google.com/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=how%20many%20videos%20youtube)), it'll be extremely costly to host all of them on CDN especially majority of the videos are long-tailed, which are videos have only 1-20 views a day.

However, one of the most interesting things about Internet is that usually, it's those long-tailed content that attracts the majority of users. The reason is simple – those popular content can be found everywhere and only long-tailed things make the product special.

Coming back to the storage problem. One straightforward approach is to host popular videos in CDN and less popular videos are stored in our own servers by location. This has a couple of advantages:

- Popular videos are viewed by a huge number of audiences in different locations, which is what CND is good at. It replicates the content in multiple places so that it's more likely to serve the video from a close and friendly network.
- Long-tailed videos are usually consumed by a particular group of people and if you can predict in advance, it's possible to store those content efficiently.

Summary

There are just too many topics we'd like to cover for the question "how to design Youtube". In our next post, we'll talk more about scalability, cache, server, security and so on.

By the way, if you want to have more guidance from experienced interviewers, you can check Gainlo (http://www.gainlo.co/?utm_source=blog&utm_campaign=blog&utm_medium=How%20to%20Design%20Youtube%20(Part%20I)) that allows you to have mock interview with engineers from Google, Facebook ,etc..

The post is written by **Gainlo (http://www.gainlo.co/?utm_source=blog&utm_medium=footer-link http%3A//blog.gainlo.co/index.php/2016/10/22/design-youtube-part/&utm_campaign=blog)** - a platform that allows you to have mock interviews with employees from Google, Amazon etc..

I'd like to learn more (http://www.gainlo.co/?utm_source=blog&utm_medium=footer http%3A//blog.gainlo.co/index.php/2

(http://www.facebook.com/sharer.php?u=http://blog.gainlo.co/index.php/2016/10/22/design-youtube-part/) 193 (http://twitter.com/share?url=http://blog.gainlo.co/index.php/2016/10/22/design-youtube-part/&text=How+to+Design+Youtube+%28Part+I%29+) (http://www.linkedin.com/shareArticle?mini=true&url=http://blog.gainlo.co/index.php/2016/10/22/design-youtube-part/) 3 (http://reddit.com/submit?url=http://blog.gainlo.co/index.php/2016/10/22/design-youtube-part/&title=How to Design Youtube (Part I)) 0

Subscribe

We'll email you when there are new posts here.

Enter your email address here...

Subscribe

Related Posts:

1. **Design a Recommendation System (http://blog.gainlo.co/index.php/2016/05/24/design-a-recommendation-system/)**
2. **Design News Feed System (Part 1) (http://blog.gainlo.co/index.php/2016/03/29/design-news-feed-system-part-1-system-design-interview-questions/)**
3. **Design Facebook Chat Function (http://blog.gainlo.co/index.php/2016/04/19/design-facebook-chat-function/)**
4. **8 Things You Need to Know Before a System Design Interview (http://blog.gainlo.co/index.php/2015/10/22/8-things-you-need-to-know-before-system-design-interviews/)**

LEAVE A REPLY

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

❮ 8 Secrets to Software Engineer Self Introduction (http://blog.gainlo.co/index.php/2016/10/14/8-secretes-software-engineer-self-introduction/)

How to Design Youtube (Part II) ❯ (http://blog.gainlo.co/index.php/2016/11/04/design-youtube-part-ii/)

CATEGORIES

Coding Interview Questions (http://blog.gainlo.co/index.php/category/coding-interview-questions/)

Common Pitfalls (http://blog.gainlo.co/index.php/category/common-pitfalls/)

Common Questions (http://blog.gainlo.co/index.php/category/common-questions/)

Facebook Interview Questions (http://blog.gainlo.co/index.php/category/facebook-interview-questions/)

Mock Interview Thoughts (http://blog.gainlo.co/index.php/category/mock-interview-thoughts/)

Others (http://blog.gainlo.co/index.php/category/others/)

System Design Interview Questions (http://blog.gainlo.co/index.php/category/system-design-interview-questions/)

The Complete Guide to Google Interview Preparation (http://blog.gainlo.co/index.php/category/google-interview-preparation/)

Uber Interview Questions (http://blog.gainlo.co/index.php/category/uber-interview-questions/)

^