**Requirements here**

Eventual consistency :

1. A delay in a video shown in 1 country

Fault tolearnt

Servcie shouldn't be down

1. It shouldn't be, like, if India has a issue in one of its servers, then we don't get to see any videos. It should be like an anti data center has to go down before we start getting worried. But even then, ideally, I mean, it should not, it should not stop our business. Right, right.

Highly available

Database sharding when fetching

Part 4: performance

1. Low latency between upload and user visibility

which is performance in the sense that when a person uploads a video, they don't want to wait for a very long time before it actually gets published. That is number one. And number two would be like, okay, fine. As soon as you can send it to users, we should send it.

2. Video should not be slow to stream here

**Capacity estimation here**

DAU: 10 mil users serving here

100k creators here

Design the skeleton APIs

1. Write to read ratio here

1: 100 more reads here

2. Monolithic approach first and then a service next:

There are 3 data here that need to be stored

1. Store user data, video meta data and video storage database

A screenshot of a computer

Description automatically generated

So where to read the video here?

1. Use amazon s3 for storing video/ images and all that

2. Use a key value store mongot for video metadata

3. My sql for user data, and then can use recommendation system here

How to make the system fast Using s3 and cdn here for the best

1. Use a queue to take all requests, just like in video, AWS3 is very reliable here, we can tie up S3 to CDN, and have it in diff regions, so the latency video will be below here

Let's say you are watching a video from Mumbai versus someone else watching the same video in us. For you, the S three file that you would be able to access via C D N will be in a center or a region closer to you. Let's say it'll be somewhere near Mumbai in India itself. While for someone else, if we are going to go through CDN approach, it is going to be closer to them physically closer to them, because network latency does play a role

s

**Why choose key-value pair for Video metadata?**

1. Attached to user itself, why not Video duration, upload date, Why choose a

o video metadata first of all is not going to be very organized or relational as user data is going to be. Okay? Okay. It'll just be any video metadata will be just be attached to the video and the user itself. And also metadata properties can change over time. You might want to get rid of some properties. You might want to add some properties. Let's say one video is deleted. So the, when was the video deleted? When was the video uploaded? What was the duration of that video? What is the thumbnail of the video?

2. Key value will be faster for access and failure. This is really important as mentioned above. If you only want the url based on the user, this is really good for scaling here

There r multi services that will access this key value store as said

Schema felxible

Check if file scalable or not

1. 200 k videos per day

200 k \* 1 million = 200 GB \* 2 (types of files = 600 GB

How would the ingestion service work?

1. In terms of conversion?

Like, we cannot wait to the file to be converted into one format. Then after that is finished, then we will convert it into second format. No. Instead of that, we can take the current file, divide it into different chunks, and create a pipeline. So first to first, the first step of the pipeline would be to divide the file into various chunks. Then the second step of the pipeline would be to convert those chunks into four different formats. So there will be four parallel processes from there. So four parallel processes will be converting the file into different formats.

2. So 4 parallel processes for converting files and then here, then into diff resolutions here and then at the end of the process, then 4 files combined and then upload to S3

If 30 sec -> split into 3 vid of 10 sec otherwise split into 10 sec video here.

Multiple format and resolutions here

**How would transport video to user**

1. Once video uploaded and then have replications of copies of video

2. 1 video have

Also we have to upload according to the region, right? So let's say the one file would be available in all the regions. It'll be replicated across all the regions. Let's say we have four regions depending on the demographics of the users. So first of all, one file will go into all four regions, and we have to save 16 or 32 files like this, depending on our formats and resolutions. And also, we might have to replicate these files. Now, why I'm talking about S three and c D n and replication here,

because our system, as we mentioned earlier, that it has to be fault tolerant. So if the one file is not accessible in one region, that doesn't mean that it is not accessible in another region.

**What if 1 node goes down ?**

Secondly, if one of the region machine goes down since we have replicas, the request will be redirected to another node so that we have we still have the file stored in another if one node goes down. So that will, again, help us in achieving high availability as well as fault tolerance. Now,

**Why have our own CDN?**

CDN has cache as well

1. If user has seen this video before, will cache and then request here

What's alkami >

Like, for example, the famous c D N that I know is Akamai, and for Netflix, I think there is something known as Open Connect which is used like,

FTP vs https?

1. We should use ftp if we are using live, if using https then using tcp

2.

**On a more granular level:**

**Let's say a user here visits my profile and says that gimme the profile to that user? End to end request flow all the recent TikTok videos of goup. So how will the, the request flow end to end?**

on the first request return the list of video titles and thumbnails

The request is gonna go to the user database as well as video metadata database, and the mapping where we will have, okay, for this user's like this key value student is going to have the user id. We'll query on that. Yeah, all the video titles and thumbnails will be visible to the user on the first go. So the list of video ids will be returned

THen when user clicks on a video

- video id will hit CDN and start streaming starts from there

The video should

Part 3 here:

Highly available here

1.