Video Streaming System Design

1. Requirements
    a. Functional
    b. Non_functional
2. Component Architecture
3. Data APIs and Protocol
4. Data Entities
5. Data Store
6. Optimization and Perfromance Improvement
7. Accessibility

Functional Requirements

1. Create a video streaming a app similar to Youtube/prime.
2. User can able to get their recommendation.
3. User can able to search the videos
4. User can able to add Comments / Like / Dislike.
5. Video should autoplay, when user hover on them.

Non-Functional Requirement

1. It should support variet of devices.
2. It should consistent, across the browser.
3. Localization / Internationalization
4. Web Vitals
5. Adaptive Loading.
6. Good to be Responsive

While Designing this, we should also discuss about

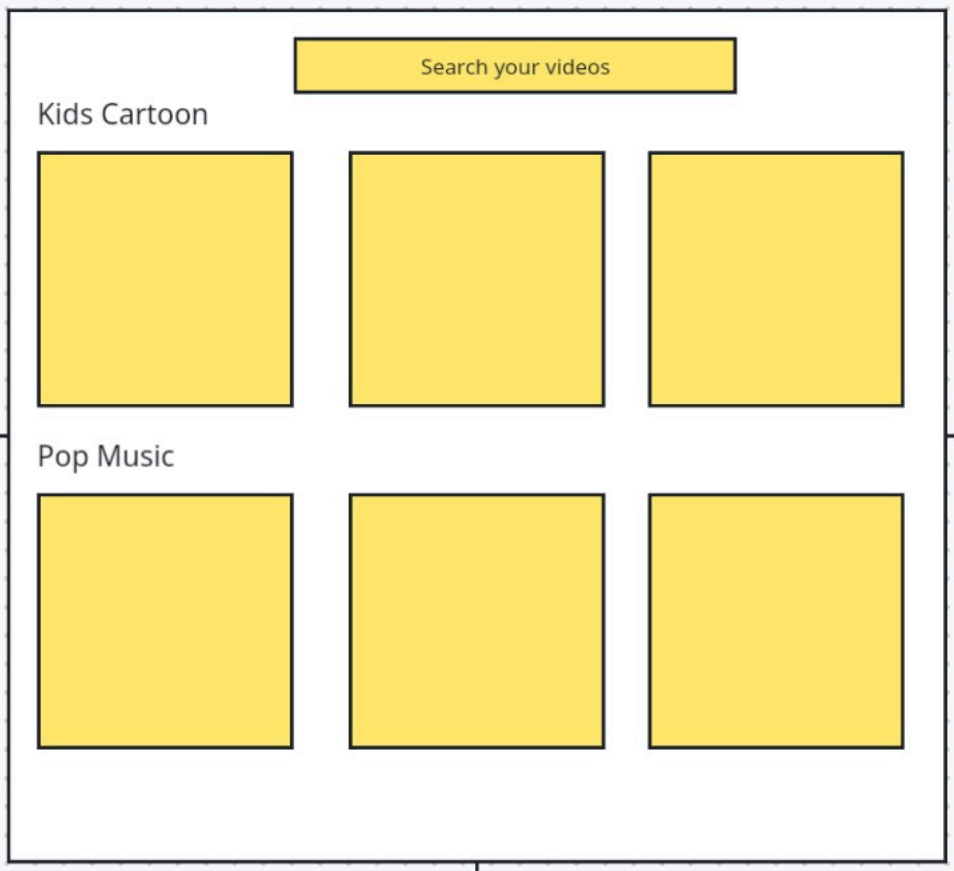
1. Video streaming services
2. Live streaming service.

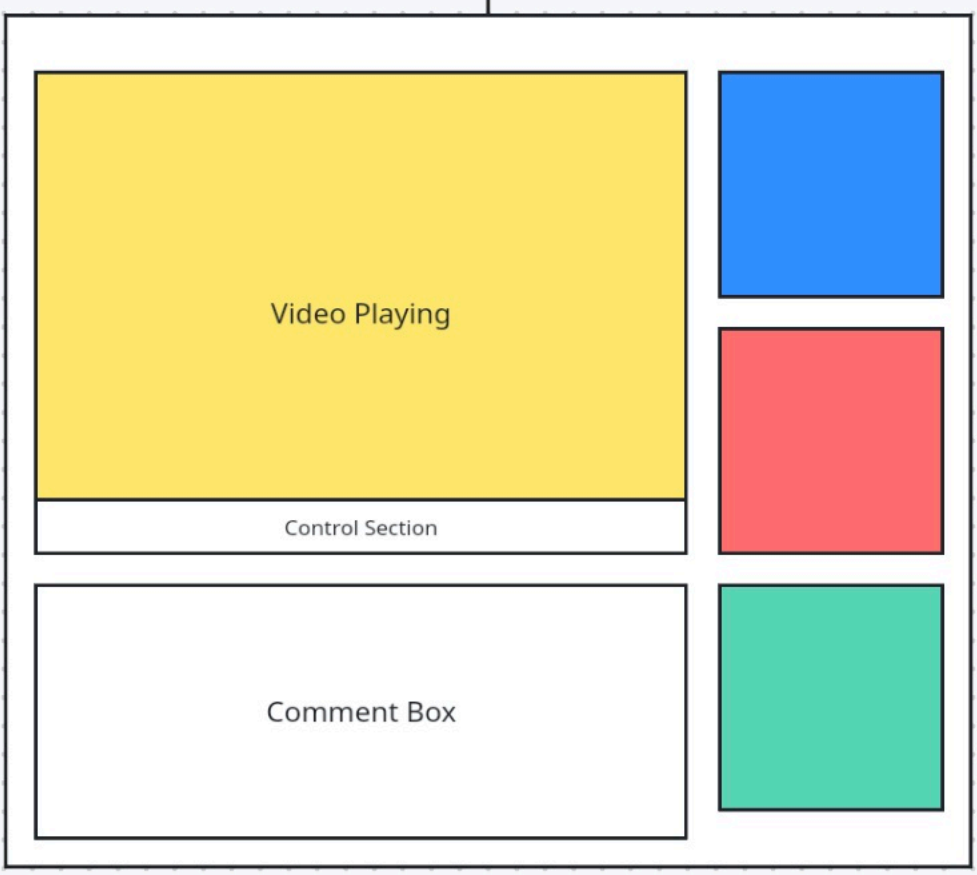
Video Streaming services, offeres pre-recorded content on demand
eg: Youtube videos, prime videos etc.
But

Video Streaming services provide video in real time
eg: Youtube live, news broadcast etc.

Designing the component architecture

We can divide in 2 different parts
1. Recommended List
2. When the user click on any video, that video should start playing and other video should come related to that





Once the Component is Ready now we should talk about CSR or SSR and their pros and cons

How are we going to render our web application

Client Side render(CSR)

1. Initially Broswer, download all the files, and then render our page.
2. SEO is less in CSR
3. Server load is less.
4. Easy to implement.
5. Growing the size of JS files, can degrade the performance.
6. It provide smooth transition, between pages, because we don't have to make call every time.

Library / Framework -> React, Angular, Vue etc.

Server Side render(SSR)

1. From the server we will get the fully render page in the browser, once reach the browser after that hydration process is started.
2. In SSR, SEO is high as compare to CSR.
3. Server load is higher than CSR.
4. Not very much easy.
5. Not very much smooth transition b/w pages.

Library / Framework -> Next.js , Nuxt.js etc.

FollowUP : Why SSR has good SEO, interviewer might ask the question ?

SSR has good SEO, because all the serach engine send web crowler, to check the content of web app, So in case of CSR we have similar to this
So, there may be chance web crower, won't wait for our JS files to get render
that's why SSR has better SEO compare to CSR

What is Hydration ?

It is basically, when the browser get the fully render page, now we have to add interactivity to the page, like hover, client event, move related event.
all those things comes under hydration

Since we have discussed about the componet and which flow we have to use, Now it is better time to discuss about protocol

How do we receive the data from the Backend ?

REST Approach

1. It use HTTP/HTTPS protocol.
2. It has different ends points
    a. /user/id
    b. /user/details/id
    c. user/profile
3. Problem of Overfetching / Underfetching.
4. Need to to end point versioning.
5. For multiple call, we need multiple request
6. Method - GET, POST, PUT, DELETE

GraphQL Approach

1. It use HTTP/HTTPS protocol and also handle websocket for subscription / real time update.
2. It has only one endpoints
    a. /graphql
3. You can get exactly what you need.
4. No need to do because we have only end point /graphql
5. No one call we can me multiple request.
6. Mostly - POST

Data Models

```
Type Video = {
   id: String
   title: String
   description: String
   metaData: VideoMetadata
   tags: [Tag]
   comments: [Comment]
}
```

```
Type Comment = {
   id: String
   originId: String
   userId: String
   comment: String
}
```

```
Type Recommendation = {
   title: String
   videos: [Video]
}
```

```
Type VideoMetadata = {
   userID: String
   craeatedAt: String
   updatedAt: String
  videoLength: String
   thumbnail: String(URL)
   like: String
   dislike: String
   actors: [ActorDetails]
   catption: [Caption]
}
```

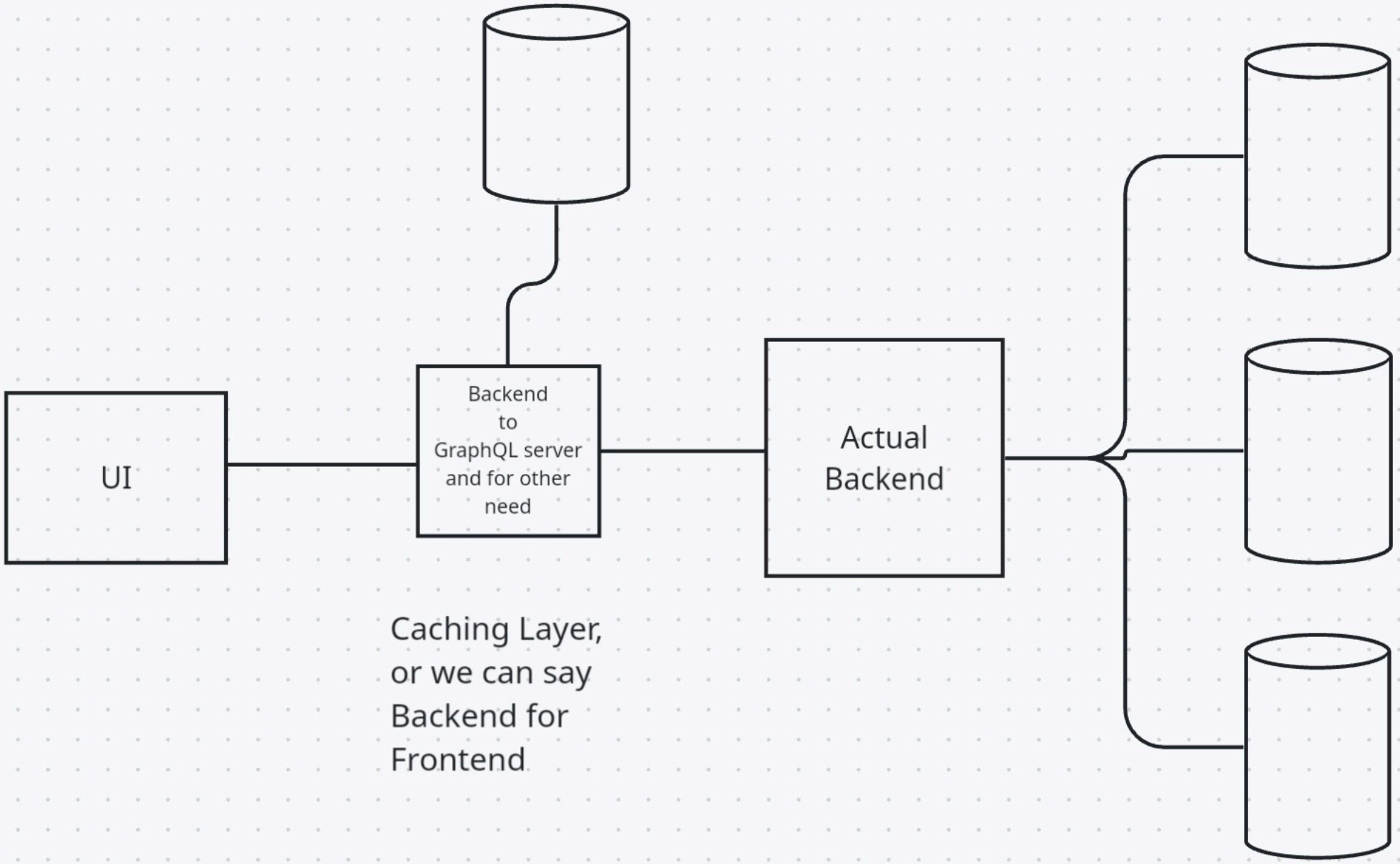
```
enum Tag = {
   MUSIC,
   HIP_HOP
   CARTOON,
   KIDS,
   EDUCATION,
   ANIME
   COMEDY
   SPORT
}
```

```
Type ActorDetails = {
   name: String,
   id: String
   user_profile: String(URL)
}
```

```
Type Caption = {
   id: String
   timeStamp: String
   captionText: String
}
```

We can also talk one more thing, since If we use GraphQL approach, then graphql has it's own server logic implmentation

1. One Way is we will create a Backend for Frontend
2. In the Backend itself have the GraphQL server implementaion.



Caching Layer,
or we can say
Backend for
Frontend

We need a transcoder which help us to break down the media in smaller chunks and different-different resoultion , bitrate and format

Since, everything is ready now we can talk about, how we are going to deliver the video to client from server.

Most of the Modern video platform, they are using "Adaptive Bitrate Streaming".

Which means they provide a few different versions of a video, also known as renditions, for the player to pick from

Different players make different decisions around how and when to switch to the different versions, so the player can make a big difference in the viewer's experience!

You might remember watching videos on Netflix or Youtube and noticing that sometimes in the middle of the video the quality will get worse for a few minutes, and then suddenly it will get better. That is what you saw when the quality changes you are experiencing **adaptive bitrate streaming**.

night remember watching
s on Netflix or Youtube
oticing that sometimes in
iddle of the video the
y will get worse for a few
es, and then suddenly it
et better. That is what you
hen the quality changes
re experiencing **adaptive**
e streaming.

Few Video Streaming Method which, in industry peoples are using

1. HLS (HTTP live streaming)
2. DASH (Dynamic Adaptive Streaming over HTTP)

HLS and DASH , both have similar kind of strategy.

HLS

You take one big video file and break it up into small segments that can be anywhere from 2-12 seconds. So if you have a two-hour-long video, broken up into 10-second segments, you would have 720 segments.

[Video Source]
|
└──> Encoding into different bitrates (e.g. 360p, 720p, 1080p)
|
└──> Split into segments (e.g. 6 seconds each)
|
├──> 360p segments: segment1.ts, segment2.ts, ...
├──> 720p segments: segment1.ts, segment2.ts,
└──> 1080p segments: segment1.ts, segment2.ts, ...

In summary, these are the steps the player goes through to play a video:
1. Load the master manifest which has information about each rendition
2. Find out which renditions are available and pick the best one (based on available bandwidth)
3. Load the rendition manifest to find out where the segments are
4. Load the segments and start playback
5. After playback starts, that is when we get into adaptive bitrate streaming.

A **manifest file** in the context of **media streaming (like HLS or DASH)** is a **playlist or index file** that tells the video player **what media files exist, their order, quality levels, durations, etc**

There are two different kinds of manifest files. For a single video there is **one master manifest** and multiple **rendition manifests**. The master manifest file is the first point of contact for the player.

Picking the Best player, which support all these things

Since, we know how to break down the video in smaller chunks, which strategy we should use,
Now, the important point is which Player we have to use

1. Browser & Device Compatibility
2. Streaming Format Support (HLS or DASH)
3. **Adaptive Bitrate (ABR) Support**
    a. Can the player automatically adjust video quality based on network speed
4. Customizability & UI
5. **Performance & Buffering**
    Does the player buffer smoothly and start quickly?

there are so many points but these are important to take care.

Similar process is also happening with the audio

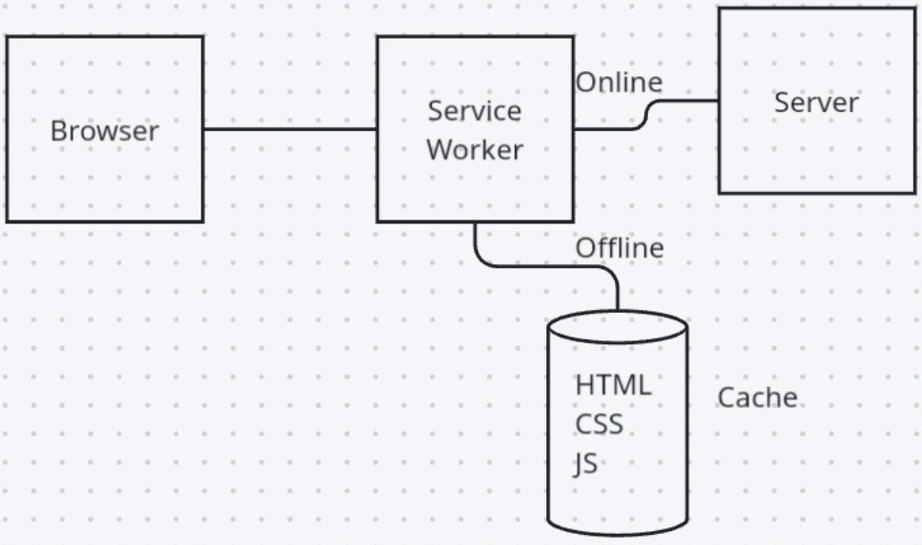
1. Spillinting into smaller chunks
2. and having different-different resolutions (1080, 480, 360 etc)
3. Here also they have Master file which will have so many information related to audio.

Now, we also want our vide should be protected, it should not get stollen
So, we can use **Encrypted Media Extensions**

Service Worker

We can also talke about the service worker, here since user can also able to dowload the video, SO, we should talk about service worker

Without Service Worker



We can also show, in the developer toolBar, where we have this service worker present

This Service Worker, has 3 steps
1. Register Service
2. Install Event
3. Activate Event