Documentation for approach to implement caching using cloud front.

You are already caching the JSON in localStorage

That means CloudFront does NOT need to cache the JSON, because your frontend is storing it and serving it locally.

Should we cache the video through CloudFront?

yes, that's the real benefit.
CloudFront will cache the **video content**, not the JSON.

Let me break it down cleanly:

What you have now

✔️ JSON data is cached in `localStorage` → No repeated API calls → Good.

✔️ JSON contains **S3 video URLs** → Every video play → hits backend → goes to S3.

❌ Videos are NOT cached → Users repeatedly load entire video from S3 → costs more + slower.

What you should cache now: Video content

Key point:

You are not caching the S3 URL itself.
You are caching the VIDEO FILE behind the URL.

CloudFront caches **the file**, not the URL text.

Example:

Your JSON:

```
{
   "videoUrl": "https://mybucket.s3.amazonaws.com/videos/abc.mp4"
}
```

When the player opens this URL, CloudFront should sit in front of S3 and cache:

- The `.mp4` file
- The partial chunks.

❗ Wait — important requirement

CloudFront will only cache the video if:

✔️ The URL used is CloudFront domain

Example:

`https://d1234abcd.cloudfront.net/videos/abc.mp4`

❌ If URL is S3 direct:

`https://mybucket.s3.amazonaws.com/videos/abc.mp4`

CloudFront **cannot** cache it.

So your case:

**You are okay with JSON being in localStorage.

Perfect.**

Now your goal is:

✔️ Make CloudFront cache the video file

➡️ By replacing S3 URL with CloudFront URL in your JSON.

How to implement (simple)

Step 1: Create a CloudFront distribution for S3

- Origin: S3 bucket
- Allowed path: `/videos/*`
- Cache TTL: 1 day (or 7 days)

Step 2: Change video URL in JSON

Backend should return:

```
{
  "videoUrl": "https://your-cloudfront-domain/videos/abc.mp4"
}
```

Not the S3 URL.

➢ Now what happens?

User plays video →
CloudFront checks cache →

Case 1: First user

- CF fetches from S3
- Stores in cache
- Sends to user

Case 2: Next 1000 users

- CF serves from edge cache
- No S3 cost
- 3–10x faster
- Backend never touched
- JSON already in localStorage → zero cost

Final Answer (short)

Since you **already cache JSON in localStorage**, you don't need CloudFront for JSON.

You only need CloudFront for **video file caching**, which means:

- Replace S3 URL → CloudFront URL
- CloudFront caches the video content
- Faster load + lower S3 cost

Our response is like this for a video


```
{
    "videoId": 1734,
    "s3url": "https://shopverse-2025.s3.eu-north-1.amazonaws.com/Videos/3752f3a2-9281-41ef-8984-43b6b8c70b4d.mp4",
    "title": "C Programing",
    "thumbnail_url": "https://shopverse-2025.s3.eu-north-1.amazonaws.com/thumbnails/24ecc25b-5600-47b7-b22b-25b9d148b397.jpg",
    "skillTag": "C",
    "tags": "BEGINNER"
  },
```

Field	Origin	Stored Where?	Hits S3?	Cost Impact
videoId	Backend API	localStorage	❌ No	None
title	Backend API	localStorage	❌ No	None
skillTag/tags	Backend API	localStorage	❌ No	None
thumbnail_url	**CloudFront** URL	localStorage	❌ No	Saves cost
video url	**CloudFront** URL	localStorage	❌ No	Saves huge cost

STEP 1 — Create a CloudFront Distribution

Origin = your S3 bucket
Example:

```
Origin: shopverse-2025.s3.amazonaws.com
```

This gives you a CloudFront domain like:

d3abcd1234xyz.cloudfront.net

STEP 2 — In S3, make sure videos + thumbnails follow a folder structure

Example:

```
Videos/<video-id>.mp4
thumbnails/<thumb-id>.jpg
```

CloudFront will mirror this structure.

STEP 3 — Do NOT return S3 URLs in your backend

Currently backend returns:

```
s3url: "https://shopverse-2025.s3.amazonaws.com/Videos/abc.mp4"
```

```
thumbnail_url:
"https://shopverse-2025.s3.amazonaws.com/thumbnails/a.jpg"
```

STEP 4 — Convert S3 URL → CloudFront URL in your backend

Add this logic in backend:

Pseudo-code:

```
const CF_DOMAIN = "https://d3abcd1234xyz.cloudfront.net";


function convertToCloudFront(s3Url) {
    return s3Url.replace(
        "https://shopverse-2025.s3.eu-north-1.amazonaws.com",
        CF_DOMAIN
    );
}
```

Your backend must send:

```
video_url: "https://d3abcd1234xyz.cloudfront.net/Videos/abc.mp4"
thumbnail_url:
"https://d3abcd1234xyz.cloudfront.net/thumbnails/a.jpg"
```

NOT S3 URLs.

✅ **STEP 5 — Frontend saves the response in localStorage**

Frontend receives:

```
{
  "videoId": 1734,
  "title": "C Programing",
```

```
  "video_url":
"https://d3abcd1234xyz.cloudfront.net/Videos/abc.mp4",

  "thumbnail_url":
"https://d3abcd1234xyz.cloudfront.net/thumbnails/a.jpg",

  "skillTag": "C",

  "tags": "BEGINNER"

}
```

Store only the metadata in localStorage:

```
localStorage.setItem("recommendedVideos",
JSON.stringify(apiResponse));
```

This includes CloudFront URLs – that's fine.

✅ STEP 6 – CloudFront now caches the actual video + thumbnail

How it works:

✔ First user

CloudFront → S3 → fetch video → cache it

✔ Next 10,000 users

CloudFront serves cached version
❌ No S3 cost
✔ Massive savings

🔥 STEP 7 — Confirm caching using response headers

In browser → Network tab → select video request:

Correct response will show:

Server: CloudFront

CF-Cache-Status: Hit

If you still see:

.s3.amazonaws.com

or

CF-Cache-Status: Miss

something is wrong.

❌ THIS WILL NOT CACHE WITH CLOUDFRONT
❌ THIS WILL HIT S3
❌ THIS WILL INCREASE COST