



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Experiment-4

**Student Name:** Rakshit Chauhan

**Branch:** CSE

**Semester:** 6th

**Subject Name:** System Design

**UID:** 23BCS12628

**Section/Group:** KRG-3B

**Date of Performance:** 04/02/26

**Subject Code:** 23CSH-314

**1. AIM :** To design an scalable OTT platform (Similar to Netflix or Amazon Prime)

**2. Objective:**

- To design and specify an OTT platform that supports user onboarding, subscriptions, content discovery, and video streaming.
- To define functional and non-functional requirements ensuring scalability, availability, and low latency.
- To identify core system entities and design RESTful APIs for seamless client interaction.
- To ensure high availability for video streaming while maintaining consistency in payments and subscriptions.

**3. Tools Required:**

- **Frontend:** HTML, CSS, JavaScript, React.js
- **Backend:** Node.js, Express.js
- **Database:** MongoDB
- **API Testing:** Postman
- **Authentication:** JWT
- **Media Storage:** Cloudinary / AWS S3
- **Version Control:** Git, GitHub
- **Deployment:** AWS / Render / Vercel

**4. SYSTEM DESIGN / SYSTEM SPECIFICATION:**

**4.1. Functional Requirements:**

- Client should be able to create account on the OTT platform.
- After the successful login, client should be able to opt for the subscription plans.
- Client should be able to search for the shows/movies based on the video title or names.
- Client should be able to watch the videos / tv shows in multiple different resolutions (480p, 720p, 1080p, 4k etc.)
- Recommendation for TV shows and movies.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## 4.2. Non-functional Requirements:

- Scalability: 200-300M, for which let's say total videos we are having are 20K videos (~1 hour each)
- CAP Theorem: Availability >>> Consistency  
**Availability on watching TV shows and movie**  
**Consistency in making payments and in subscription plans**
- Latency: 50 - 80 ms  
**Client should be able to see the video with zero or negligible buffering.**

## 4.3. Core-Entites of the System:

1. Client
2. Clients Metadata
3. Video / TV shows
4. Video metadata: (images(Thumbnails + description))

## 4.4. API Endpoints Creation:

### A. Client-Onboarding

1. POST Call: <https://www.netlfix.com/user/register>
2. POST Call: <https://www.netlfix.com/user/login>
3. PUT Call: <https://www.netlfix.com/user/update>
4. User Data Update: PUT API CALL: PUT / api / users / {user\_id} / profile

### B. Subscription

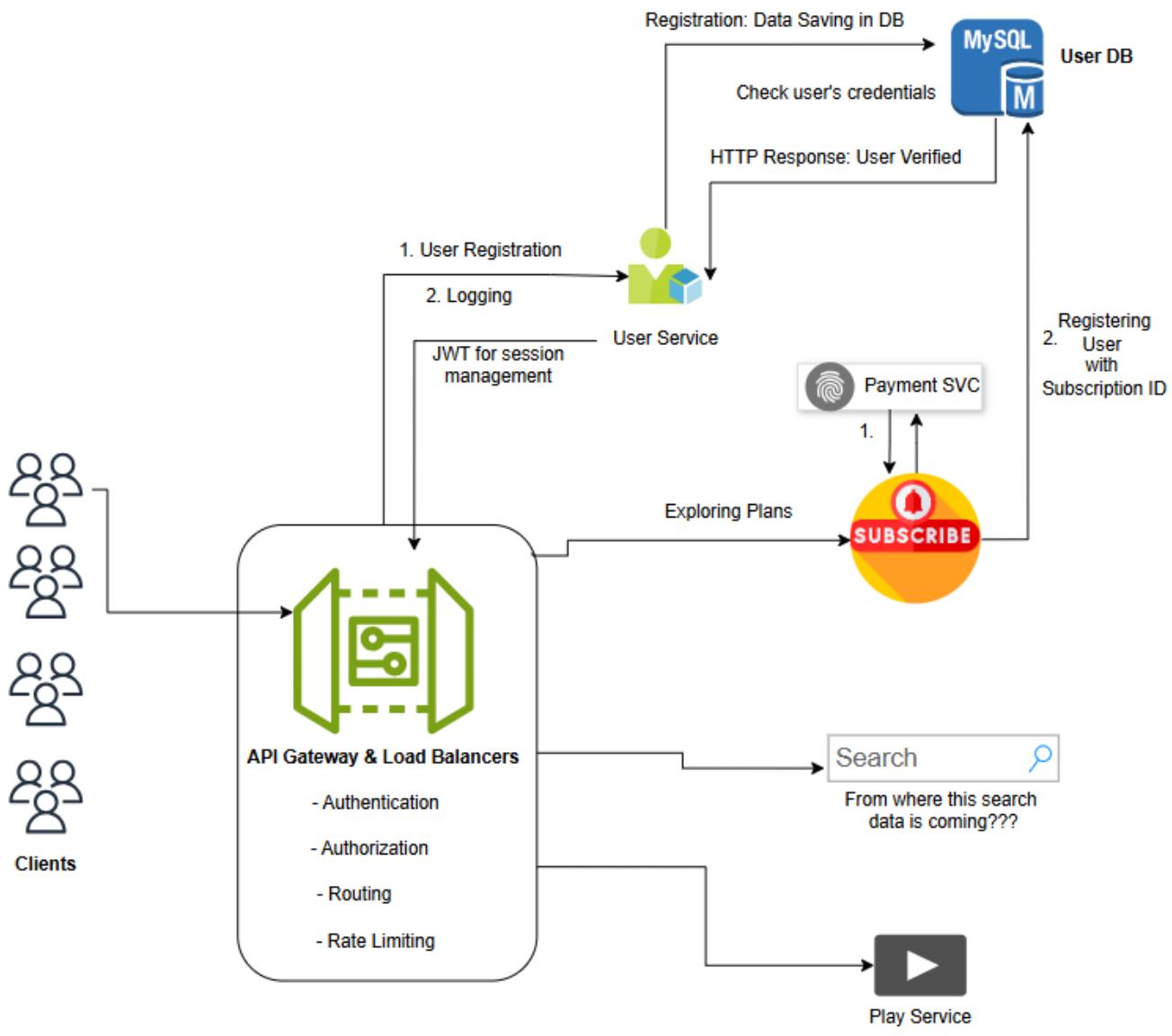
1. GET Call: <https://www.netlfix.com/Get-subscription-plans>
2. POST Call: <https://www.netlfix.com/subscription>

```
{  
    userMetadata,  
    subscriptionID  
}
```

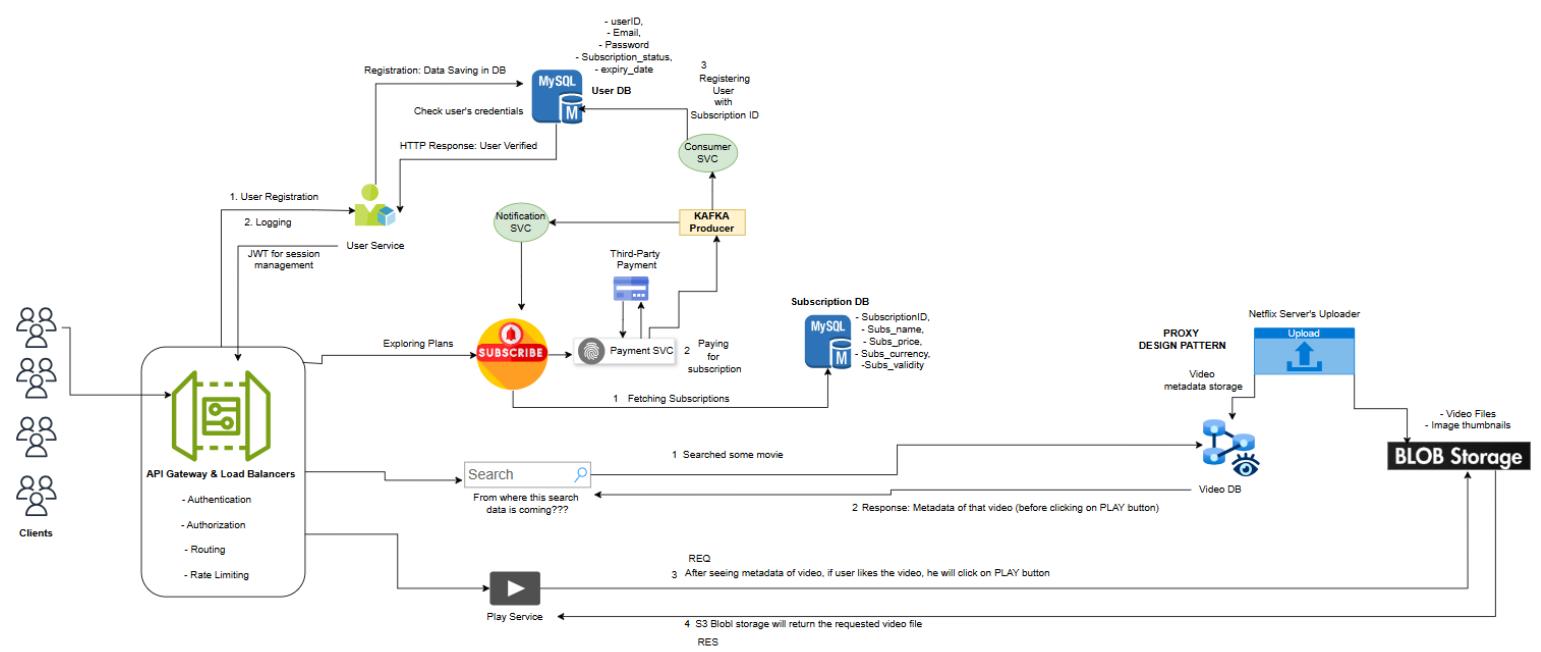
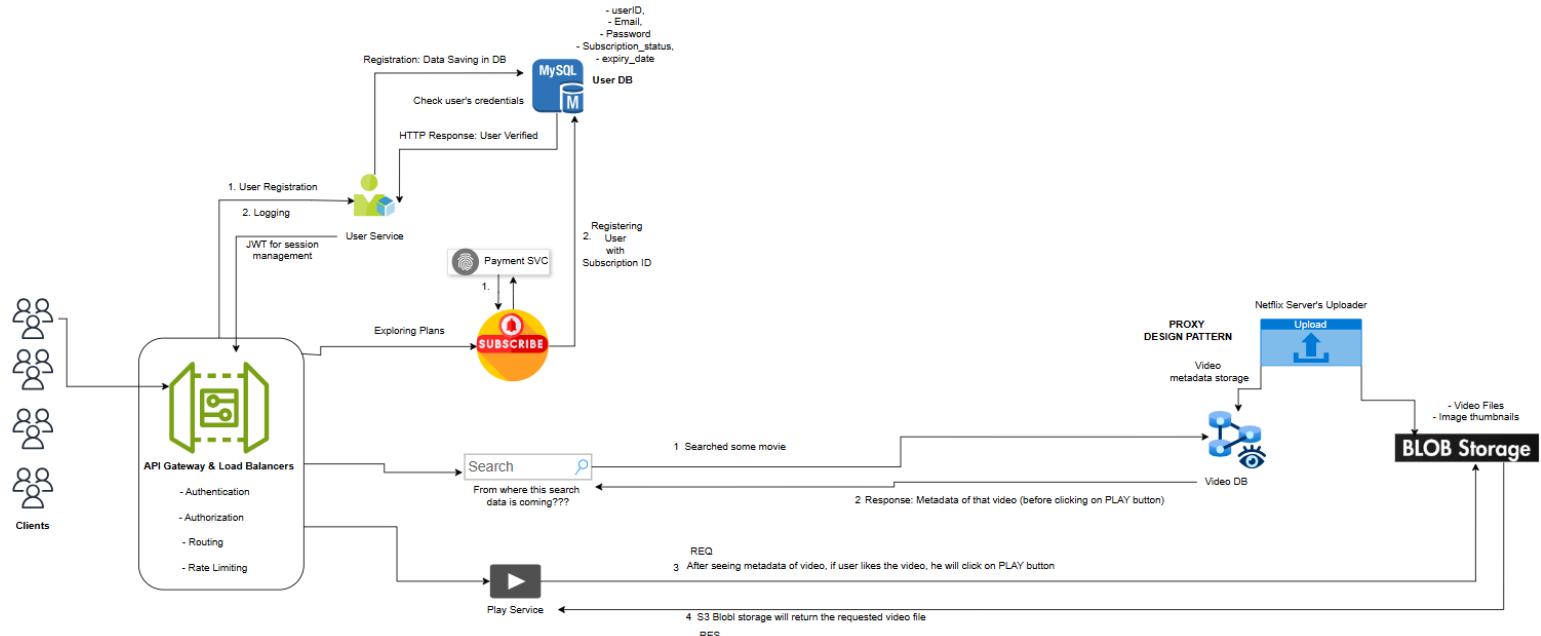
### C. Searching & Video Playing

1. GET Call: [https://www.netlfix.com/search?q={movie\\_name}](https://www.netlfix.com/search?q={movie_name})  
Response: List<Video\_ID> + some meta data of video
2. GET Call: [https://www.netlfix.com/{video\\_ID}](https://www.netlfix.com/{video_ID})  
Response: Metadata of the video (JSON)
3. GET Call: [https://www.netlfix.com/play/{video\\_ID}](https://www.netlfix.com/play/{video_ID})

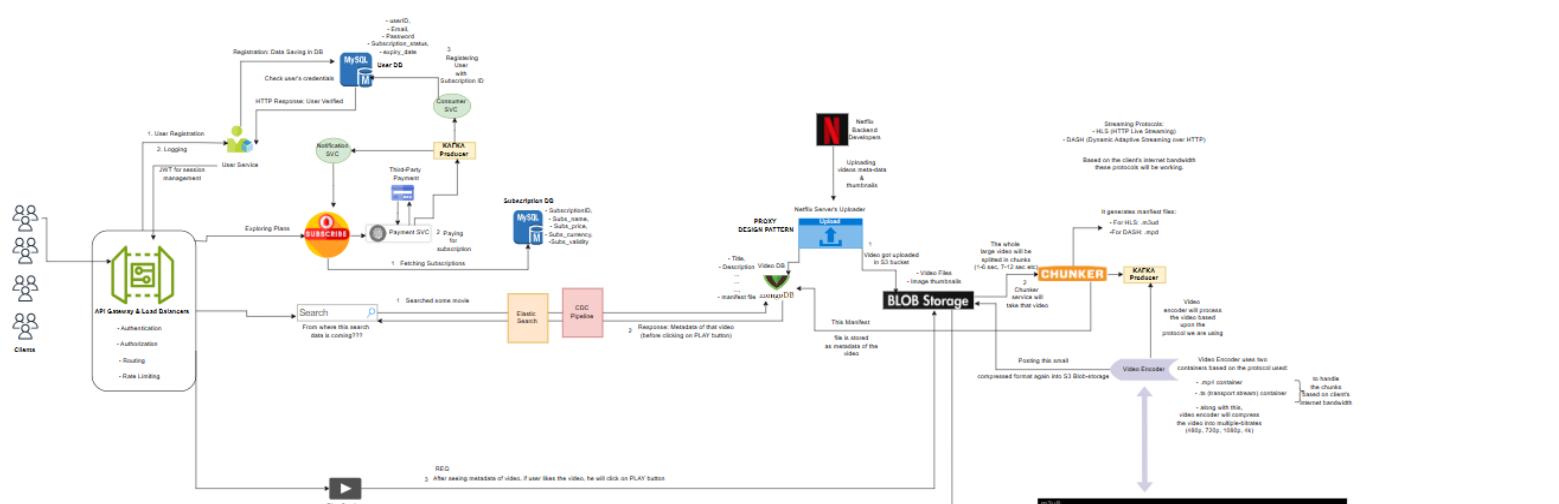
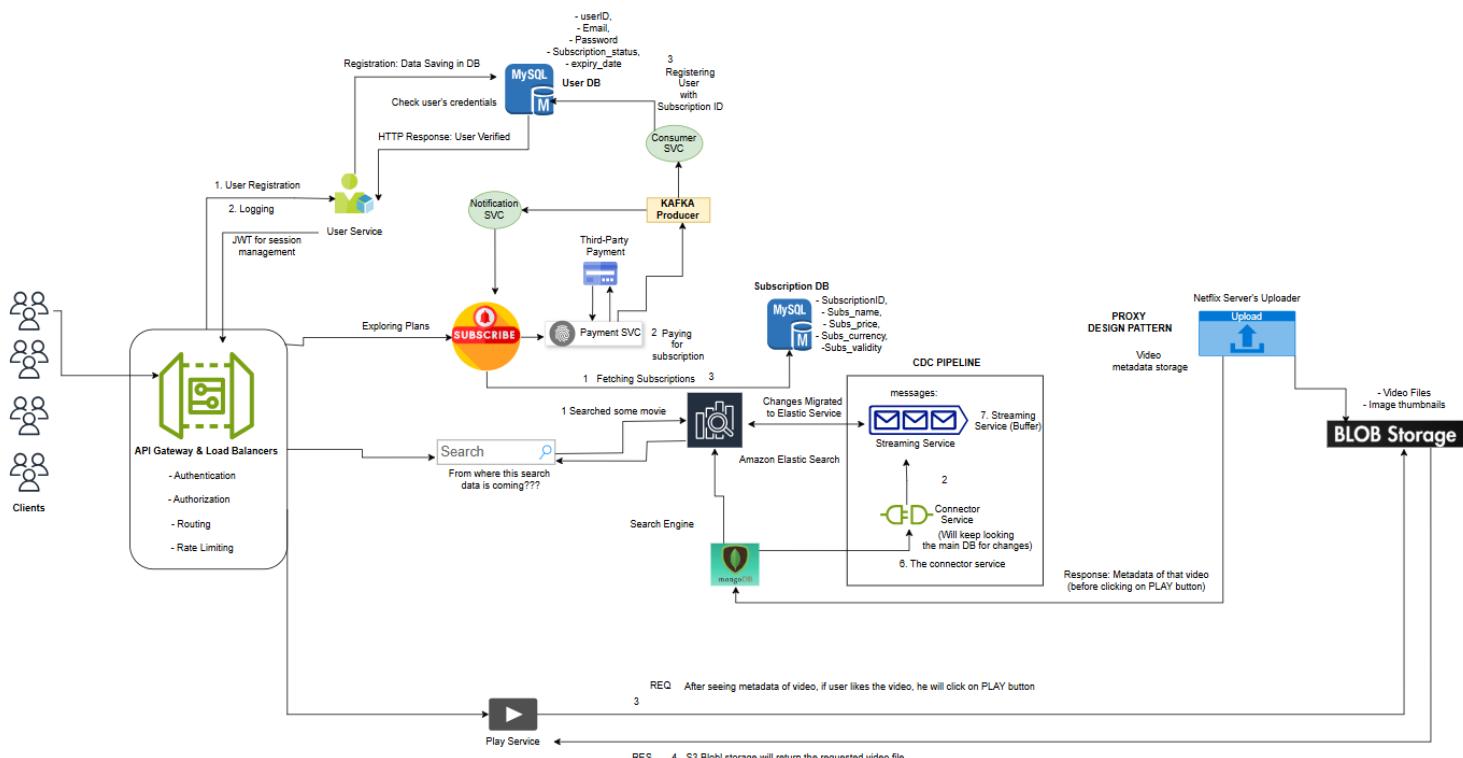
## 5. HLD(High Level Design):



## 6. LLD(Low Level Design)



## Search Service



```

Total Video Duration: 30 Seconds
3 chunks of 10 second each are made
m3u8

#EXTM3U
#EXT-X-TARGETDURATION:10
#EXT-X-MEDIA-SEQUENCE:1
#EXTINF:10.0,
http://example.com/segment1.ts
#EXTINF:10.0,
http://example.com/segment2.ts
#EXTINF:10.0,
http://example.com/segment3.ts
#EXT-X-ENDLIST

```

**Problem:**  
The whole video is in chunks.  
How will we assure the sequence of chunks displayed on the client's machine.

**Solution:**  
we use a MANIFEST file  
where we keep records of the sequence of these chunks.  
Manifest file  
for creating this file.

Combination of these two files is a  
Manifest file  
Mapping of playlist  
to the sequence file

```

m3u8

#EXTM3U
#EXT-X-VERSION:3
#EXT-X-STREAM-INF:BANDWIDTH=500000,RESOLUTION=640x360
http://example.com/low.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=1000000,RESOLUTION=1280x720
http://example.com/medium.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=1000000,RESOLUTION=1920x1080
http://example.com/high.m3u8

Low Bandwidth Playlist Chunk
Medium Bandwidth Playlist Chunk
High Bandwidth Playlist Chunk

```

## **7. Learning Outcomes**

- Understand and document functional and non-functional requirements of a large-scale OTT system.
- Apply CAP theorem concepts in real-world system design scenarios.
- Identify core entities and their roles in system architecture.
- Design RESTful API endpoints for user management, subscriptions, search, and video playback.
- Analyze scalability, availability, and latency requirements for high-traffic applications.