Media Streaming with IBM Cloud Video Streaming

TEAM MEMBER

312621243023:Shalini.J

Phase-2 Document submission

The Objective

project involves creating a virtual cinema platform using IBM Cloud Video Streaming. The objective is to build a platform where users can upload and stream movies and videos on-demand. This project encompasses defining the virtual cinema platform, designing the user interface, integrating IBM Cloud Video Streaming services, enabling on-demand video playback, and ensuring a seamless and immersive cinematic experience.

Introduction

1.Sign up for IBM Cloud Video Streaming: If you haven’t already, sign up for an IBM Cloud Video Streaming account.

2.Upload Your Movies and Videos: Prepare your favourite movies and videos and upload them to your IBM Cloud Video Streaming account. Make sure they are in a compatible format and quality for streaming.

3.Set Up Your Virtual Cinema Space: Create a dedicated virtual space for your cinema platform. You can design a website or an app where users can access the movies and videos you’ve uploaded.

4.Customize Your Platform: Customize the platform with features like user authentication, movie categorization, and a user-friendly interface. Ensure that users can easily browse, search, and select the content they want to watch.

5.Implement IBM Cloud Video Streaming API: Integrate the IBM Cloud Video Streaming API into your platform to enable seamless video playback and streaming. This API will handle the streaming infrastructure, ensuring high-quality video delivery.

6.Implement User Management: Create user accounts and implement access control. You can set up different user roles, such as administrators and viewers, and manage permissions accordingly.

7.Enable On-Demand Streaming: Implement on-demand streaming so that users can choose when to watch their favourite movies and videos.

8.Optimize for Different Devices: Ensure that your virtual cinema platform is responsive and works well on various devices, including smartphones, tablets, and computers.

9.Implement Social Features: Add social features like chat rooms or comments to allow viewers to interact and discuss movies in real-time.

10.Monetize: If you want to monetize your platform, consider implementing payment processing for rentals or subscriptions.

11.Test and Debug: Thoroughly test your platform to ensure that video streaming works flawlessly, and there are no usability issues.

12.Launch Your Virtual Cinema: Once everything is set up and tested, launch your virtual cinema platform and start promoting it to friends, family, and a broader audience.

13.Provide Support and Maintenance: Offer customer support and regularly update and maintain your platform to ensure a smooth and enjoyable experience for users.

Innovation

User-Generated Playlists:

- Allow registered users to create and curate their own playlists of movies and videos.

- Provide an easy-to-use interface for users to add, remove, and reorder content in their playlists.
- Enable users to share their playlists with friends and other platform users.
- Implement features for users to browse and discover playlists created by others.

Real-Time Chat:

- Integrate a real-time chat system within the virtual cinema platform.
- Create chat rooms or channels dedicated to specific movies or genres to facilitate discussions.
- Implement moderation tools to ensure a friendly and respectful chat environment.
- Allow users to send text messages, emoji, and perhaps even share reactions during movie playback.

- Consider adding chat features like private messaging and group chat rooms for different viewing parties.

Notifications:

- Implement a notification system to inform users about new chat messages, playlist updates, and upcoming movie screenings.
- Notify users when a movie they added to their playlist is about to start streaming.

User Engagement Features:

- Encourage users to rate and review movies after watching them to help others discover content.
- Implement a recommendation system based on user preferences and viewing history.

Moderation and Reporting:

- Have a reporting system in place so that users can report inappropriate content or behaviour in the chat.
- Appoint moderators if necessary to ensure a safe and enjoyable environment.

Synced Playback:

- Offer a feature for users to start a movie or video simultaneously, allowing friends and family to watch together even if they are physically apart.

Cross-Platform Compatibility:

- Ensure that chat and playlist features work seamlessly across different devices and web browsers.

Feedback and Iteration:

- Collect feedback from users and iterate on the platform based on their suggestions and needs.

Conclusion

By incorporating these features, the virtual cinema platform will not only provide an immersive movie-watching experience but also foster a sense of community and interaction among users.

Code :

```
#Replace with your IBM Cloud video streaming credentials
api_key ="YOUR_API_KEY"
channel_id="YOUR_CHANNEL_ID"
#Replace with the URL of the media you want to stream
media_url="https://example.com/media.mp4"
#Create a live event
response=requests.post(f"https://api.video.ibm.com/channels/{channel_
id}/events",
 Headers={"Authorization": api_key},
 json={"event_type":"live","start",:true},
)
event_id=response.json()["event"]["id"]
#upload media file
response=request.post(f"https://api.video.ibm.com/channels /{channel_
id}/events/
{event_id}/uploads",)
headers={"Authorization": api_key},
json={"url": media_url},
)
upload_id = response.json()["upload"]
["id"]
# Start the live stream
response = requests.post(
f"https://api.video.ibm.com/channels /{channel_id}/events/
{event_id} /broadcasts",
 headers={"Authorization": api_key},
 json={"upload_id": upload_id},
)
broadcast_id = response.json()
["broadcast"]["id"]
# Get the stream URL
```

```
response = requests.get(
 f"https://api.video.ibm.com/channels /{channel_id}/events/
{event_id}/
broadcasts
/{broadcast_id}/stream_url",
headers={"Authorization": api_key},
)
stream_url = response.json()["stream_url"]
# Print the stream URL
 print("Stream URL:", stream_url)
```

Make sure to replace "YOUR_API_KEY", "YOUR_ CHANNEL_ID", and "https:// example.com/media.mp4" with your own credentials and