

MEDIA STREAMING

PROBLEM DEFINITION:-

The 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.

INNOVATION:-

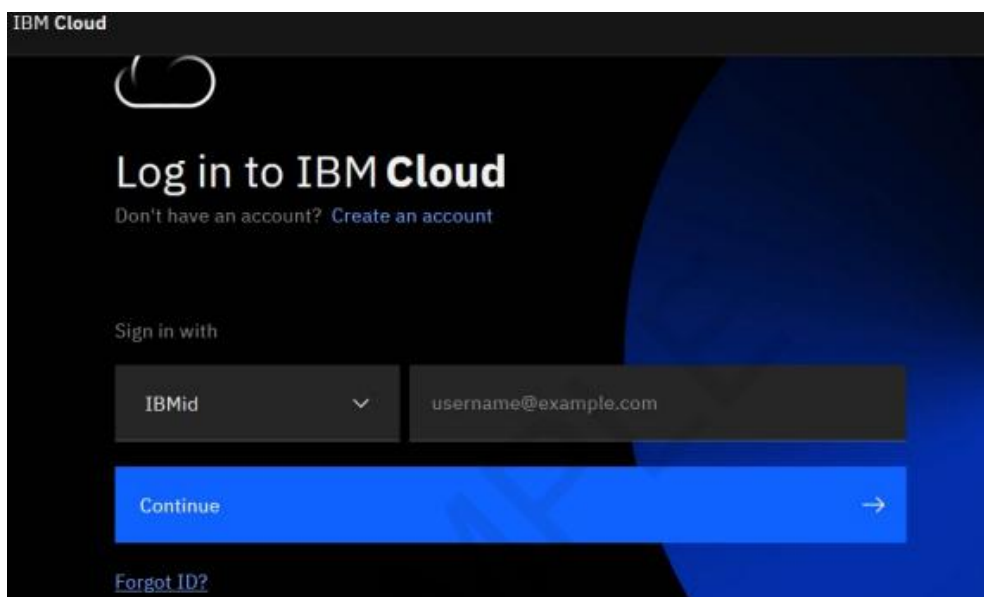
Considering incorporating features like user-generated playlists or real-time chat for a more engaging movie-watching experience.

CREATING DB2 AND COGNOS SERVICES ON IBM CLOUD ACCOUNT:-

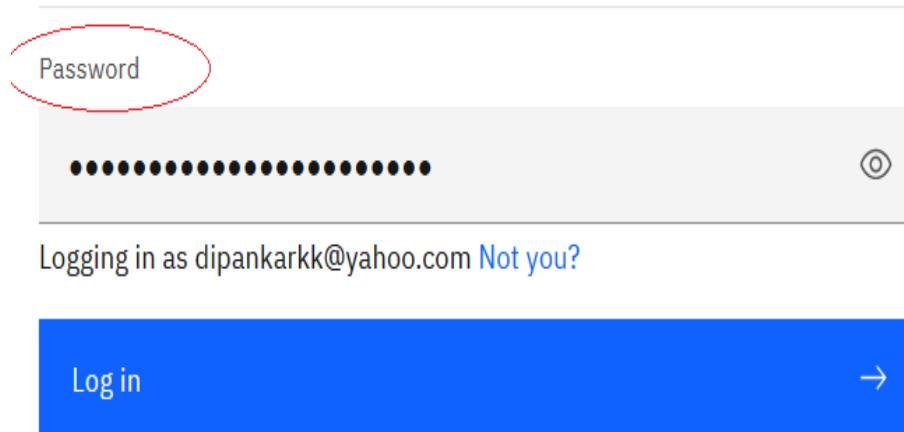
After create a Lite account (Trail Free account) as a student. Login to IBM cloud.

<https://cloud.ibm.com/login>

Entering our registered mail id.

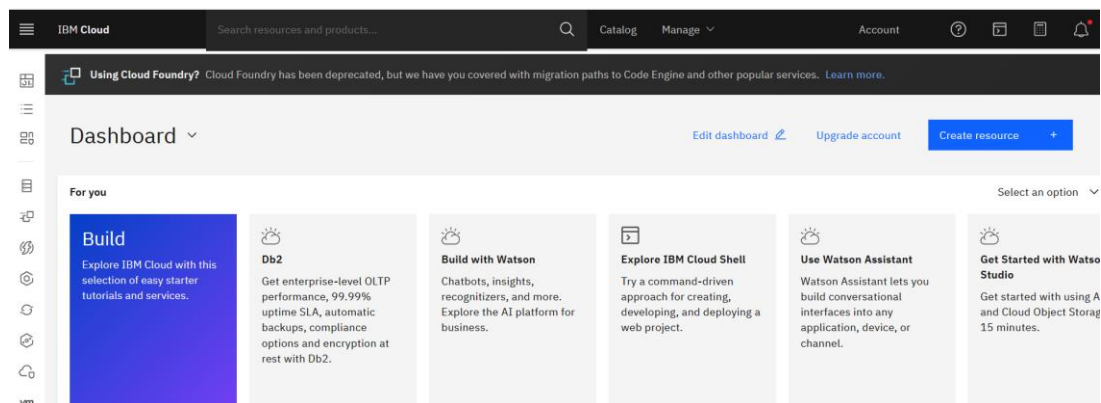


Then Entering Password

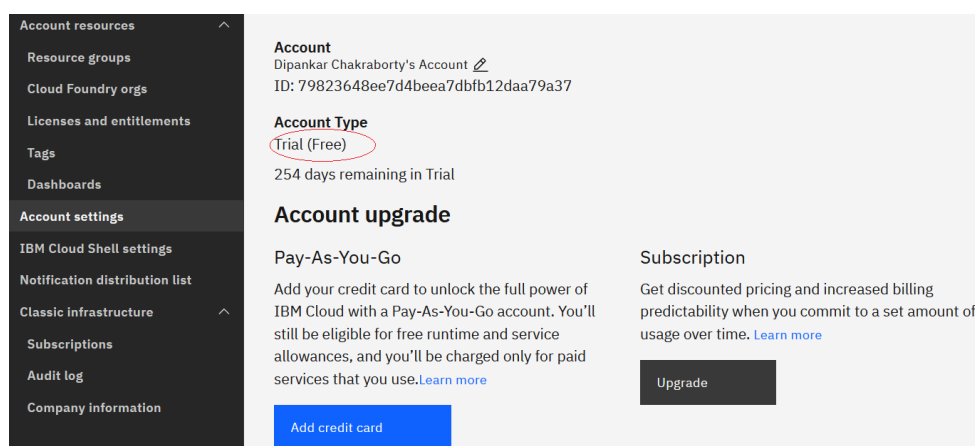


The login form consists of a 'Password' label with a red circle around it, followed by a password input field with a strength indicator (dots) and an eye icon. Below the input field is the text 'Logging in as dipankarkk@yahoo.com' with a link 'Not you?'. At the bottom is a large blue 'Log in' button with a right-pointing arrow.

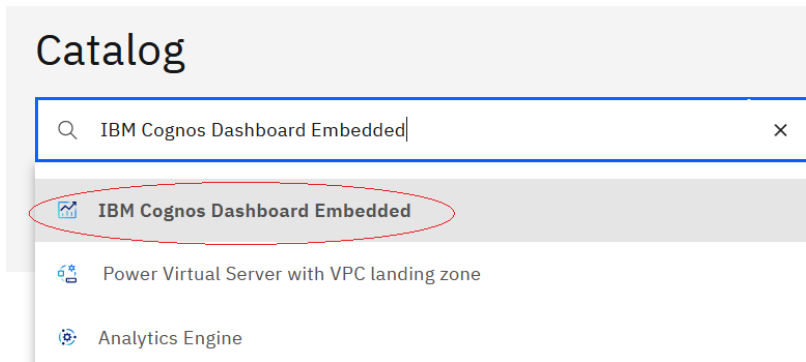
After successfully login, user will be redirected to account Dashboard page.



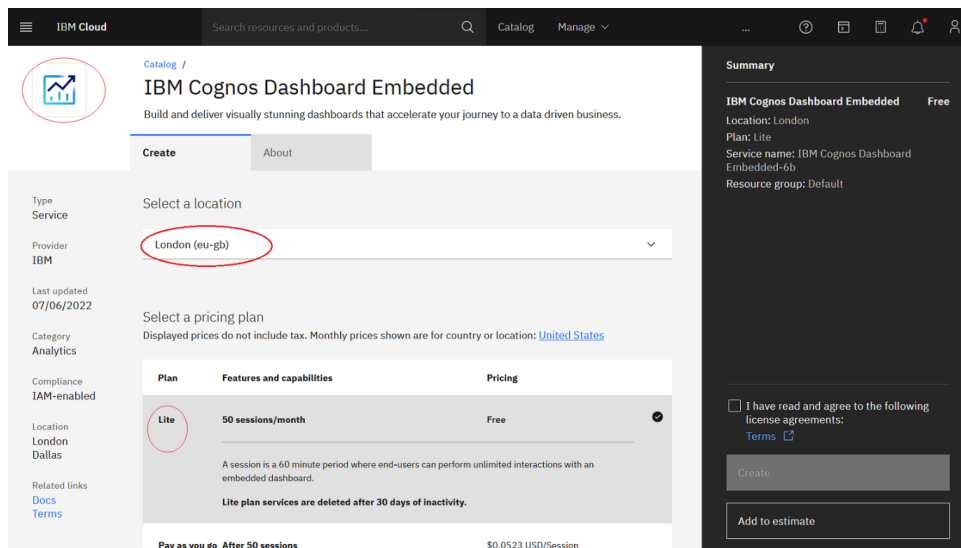
ACCOUNT SETTING:-



Clicking on **Catalog** menu on top page. Then entering Cognos in search box.



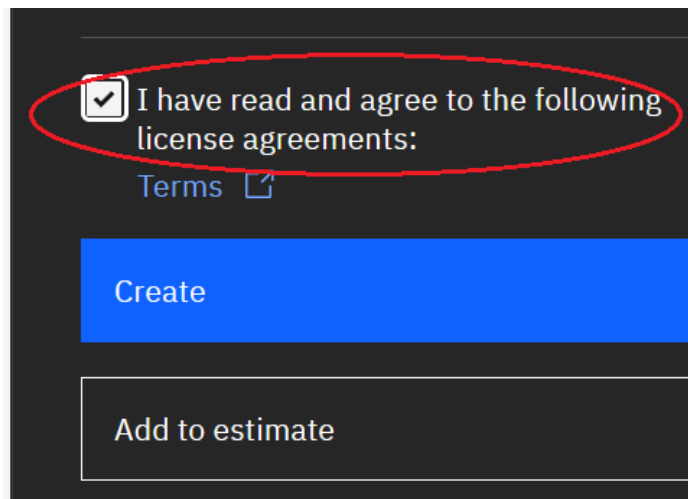
Click “IBM Cognos Dashboard Embedded”, it will redirect to Cognos page. Once you are on Cognos page.



Select a Location: **London**

Select a pricing plan: **Lite**.

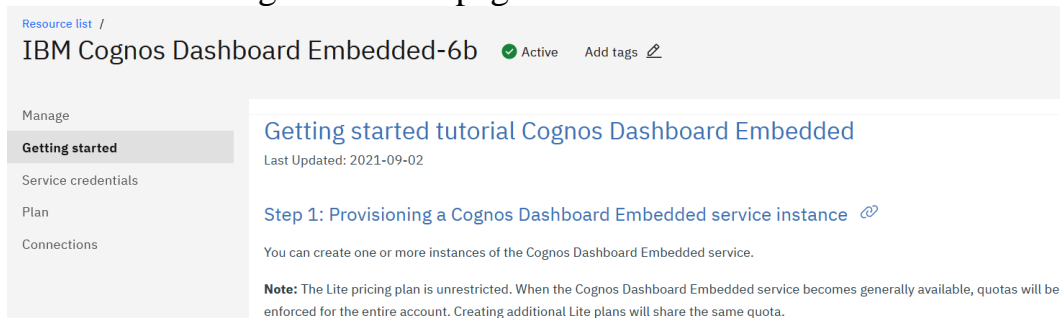
Select terms & condition (license agreements)



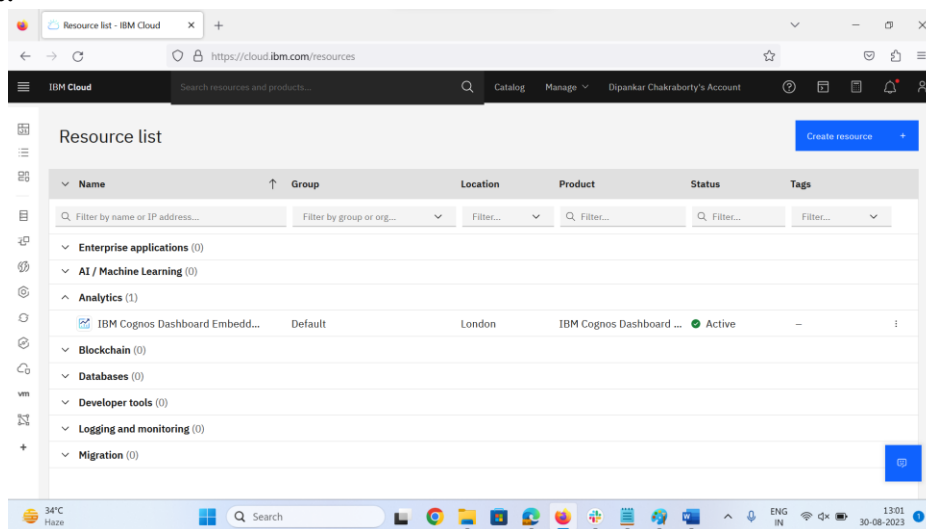
Then click on **Create** button.

After you click on create, it will take some time to create the service.

User will be redirected to Cognos tutorial page.

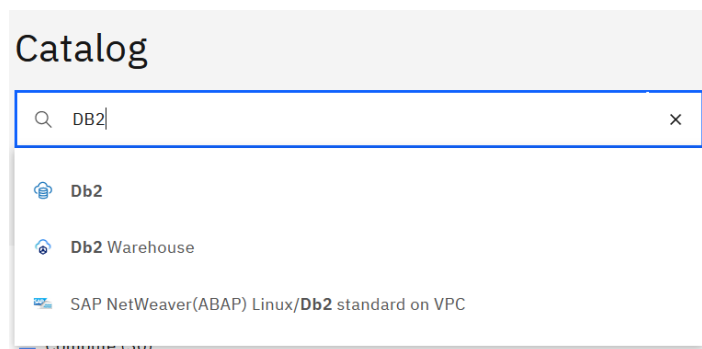


It takes few minutes to service created at cloud account. After few minutes you can check resource list.



ADDING DB2 SERVICES IN THE ACCOUNT:-

Now again search DB2 on catalog



Click on **Db2**, you will be redirected to DB2 service page.

The screenshot shows the IBM Cloud console for creating a Db2 instance. The 'Create' tab is active, and the 'Location' dropdown is set to 'London (eu-gb)'. The 'Plan' is set to 'Lite'. The 'Summary' panel on the right shows the configuration details. A checkbox for 'I have read and agree to the following license agreements:' is checked. The 'Create' button is highlighted in blue.

Select a Location: **London**

Select a pricing plan: **Lite**.

Select terms & condition (license agreements)

Then click on **Create** button.

After you click on create, it will take some time to create the service.

FEATURES

MULTIPLE ADMINS AND PERMISSIONS:-

Higher level plans can create a management structure inside their accounts. This can include granular permissions, such as setting someone up so they can manage published videos but prevent them from deleting or publishing content.

VIDEO REVIEW AND APPROVAL WORK FLOW:-

Setup custom roles that designate users as moderators of content, reviewing and approving with automated email notifications to content generators.

SIMULTANEOUS LIVE STREAMING:-

Creating multiple channels and conduct live streams simultaneously across them.

MULTILANGUAGE AUDIO TRACKS:-

Live stream video in multiple languages, appearing as language tracks during the live stream and for on-demand version.

4KVIDEO STREAMING:-

Creating 4K resolution live video streams or upload 4K video files.

SIMULATED LIVE:-

Simulate a live broadcast using previously recorded video. Choose one or multiple videos and schedule a start time from an experience that is presented as a live stream.

VIDEO EDITING / TRIMMING:-

On demand videos can be edited through the online dashboard. This can be used to remove unwanted parts at the start or end of video content. These edits are nondestructive and can be reverted at any time. Segments can also be saved as separate videos, as teasers or to break up long form content.

DYNAMIC MANUAL VIDEO PLAYLIST:-

Creating video playlists, a collection of on demand video content. These can be created manually or dynamically, evolving automatically with content based on assigning metadata criteria.

PLAYBACK SPEED CONTROL:-

Inside the player is an option to control the playback speed of on-demand videos. This allows viewers to slow videos down or do the opposite and speed up the rate of the video for playback.

CONTINUE WATCHING:-

Video on demand content has the playback position saved locally, and will start from the moment the viewer left off from if they return to the content within three months.

REGISTRATION GATE:-

Leverage a mandatory or optional registration gate, requiring users to enter details like their name, email address or phone number before accessing video content or attending an online event.

E-MAIL CONFIRMATION:-

When registration gate form is submitted, an email confirmation can be automatically sent to the registrant

CHAT MODULE:-

An embeddable text chat module can be shared on websites or via channel pages. Full moderation is available.

BREAKOUT CHAT ROOM:-

Spin off additional text chat rooms that live inside the same player experience, offering the ability for watch parties or separate topic discussion.

LIVE POLLING:-

Broadcasters can push a live poll inside the video player to audiences. Virtually real-time results are given.

CONCENTRATING INTO

1. Bandwidth Limitations:

Limited internet bandwidth can lead to buffering, lower video quality, and interruptions in streaming.

2. Latency:

Latency issues can cause delays in video playback and affect real-time interactions, such as live streaming or video conferencing.

3. Congestion:

Network congestion during peak usage times can result in slower streaming speeds and reduced video quality.

4. Codec Compatibility:

Incompatibility between codecs used by the streaming service and the viewer's device can lead to playback issues.

5. Device and Platform Fragmentation:

The wide variety of devices and platforms used for streaming (e.g., smartphones, smart TVs, gaming consoles) can make it challenging to ensure a consistent streaming experience across all devices.

6. Content Delivery:

Efficient content delivery is crucial. Content delivery networks (CDNs) help distribute content geographically to reduce latency and improve streaming quality.

7. Quality of Service (QoS):

QoS issues can impact streaming quality, especially in cases where network providers do not prioritize streaming traffic.

8. Buffering and Start-ups Delays:

Buffering delays can frustrate users, especially when content takes a long time to start playing.

9. Ad Insertion:

The insertion of ads during streaming can sometimes disrupt the viewing experience if not properly managed.

10. Content Piracy:

Protecting copyrighted content from piracy is an ongoing challenge for streaming providers.

11. Security Concerns:

Protecting user data and ensuring secure streaming experiences is vital, especially for paid streaming services.

12. Content Licensing:

Negotiating and maintaining licensing agreements with content providers can be complex and costly.

13. User Experience:

Providing a seamless and user-friendly interface for content discovery and playback is critical for retaining viewers.

14. Accessibility:

Ensuring that multimedia content is accessible to people with disabilities is a legal and ethical requirement in many regions.

15. Geographic Restrictions:

Some content may be subject to regional licensing restrictions, leading to limited availability in certain areas.

16. Live Streaming Challenges:

Live streaming involves real-time delivery, making it susceptible to issues like network instability and scalability challenges.

START UP PROGRAM FOR PROJECT USING VISUAL STUDIO CODE**room.html**

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
    <meta charset = 'utf = 8'>
```

```
    <meta http - equiv = 'x - UA - Compatible' content = 'IE = edge'>
```

```
    <title> Room </title>
```

```
    <meta name = " viewpoint' content = 'width = device - width, initial - scale = 1'>
```

```
    <link rel = 'stylesheet' type = 'text/css' media = 'screen href = 'styles/main.css'>
```

```
    <link rel = 'stylesheet' type = 'text/css' media = 'screen href = 'styles/room.css'>
```

```
</head>
```

```
<body>
```

```
    <header id = "nav">
```

```
        <h1> Cinimaful </h1>
```

```
    </header>
```

```
    <main id = "room__container">
```

```
        <section id = "participants__container">
```

```
            <div> id = "participants__heaader">
```

```
                <p>Room Members</p>
```

```
                <p>5</p>
```

```
            </div>
```

```
            <div class = "member__wrapper">
```

```
                <span class = "green_dot"></span>
```

```
                <p>Felix</p>
```

```
            </div>
```

```
<div class = "member__wrapper">
  <span class = "green_dot"></span>
  <p>Vishal</p>
</div>
```

```
<div class = "member__wrapper">
  <span class = "green_dot"></span>
  <p>Bhakkiyalakshmi</p>
</div>
```

```
<div class = "member__wrapper">
  <span class = "green_dot"></span>
  <p>Asmitha</p>
</div>
```

```
<div class = "member__wrapper">
  <span class = "green_dot"></span>
  <p>Iswarya</p>
</div>
```

```
</section>
```

```
<section id = "chat__container"></section>
```

```
<div class = "message_wrapper">
  <p>Felix</p>
  <p class = "message">Stream And Host Your Video Reliably
And Securely With Cinimaful.</p>
</div>
```

```
<div class = "message_wrapper">

    <p>Felix</p>

    <p>class = "message">Stream And Host Your Video Reliably
    And Securely With Cinimaful.</p>

</div>

<div class = "message_wrapper">

    <p>Felix</p>

    <p>class = "message">Stream And Host Your Video Reliably
    And Securely With Cinimaful.</p>

</div>

</main>

</body>

</html>
```

main.css

```
@import url
('https://fonts.googleapis.com/css2?family=Poppins;500;600;700;800&display=swap');
{
    font-family: 'Poppins', sans-serif;
}

body
{
    background-color: #1a1a1a;
    font: 14px;
    padding: 0;
```

```
margin: 0;
box-sizing: border-box;
}

#nav
{
    background-color: #1a1a1a;
    height: 50px;
    border-bottom: 1px solid #000;
}

.green__dot
{
    height: 10px;
    width: 10px;
    background-color: #zaca3e;
    border-radius: 50%;
}
```

room.css

```
#room__container
{
    display: grid;
    grid-template-columns: 200px 800px;
}

#participants__container
```

```
{  
    border - right: #797a79;  
}  
  
.message__wrapper  
{  
    background-color: #363739;  
    border-radius: 10px;  
    padding: 10px;  
    margin top: 10px;  
}
```

----- END OF THE DOCUMENT -----