MEDIA STREAMING WITH IBM CLOUD VIDEO STREAMING

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

SUBMITTED BY

1.K.KARTHIKA (422421205017)

2.M.KARPOORAVALLI (422421205016)

3. B.NERSHA (422421205023)

4. B.JENISHA (422421205012)

ABSTRACT

The landscape of media streaming is undergoing a profound transformation, and IBM Cloud Video Streaming emerges as a leading player in this dynamic environment. This project delves deeply into the real of media streaming, with a specific focus on IBM's cloud service. IBM Cloud Video Streaming provides an extensive suite of features and capabilities, offering organizations the means to seamlessly deliver, manage, and protect high-quality video content to audiences around the globe.

Within this exploration, we thoroughly examine the key components and functionalities of IBM Cloud Video Streaming, encompassing comprehensive content management tools, live streaming capabilities, and the dynamic realm of video-on-demand services. We shed light on the platform's user-friendly interfaces and advanced analytics tools, empowering content providers to gain invaluable insights into viewer engagement patterns and preferences, thus allowing for the refinement of content delivery strategies. Furthermore, we highlight the paramount importance of the robust security measures and content protection mechanisms implemented by IBM Cloud Video Streaming. These measures ensure that sensitive content remains safeguarded in an increasingly interconnected digital landscape, offering peace of mind to content creators and distributors. Through a comprehensive analysis of real-world case studies and practical implementations, this project underscores how IBM Cloud Video Streaming can be a transformative force for organizations seeking to deliver captivating media content efficiently and securely, ultimately enhancing their digital presence and engagement with audiences worldwide.

Phase 3: Development Part 1

Project Objectives:

In this part we will begin building our project. We start building the virtual cinema platform using IBM Cloud Video Streaming. We will define the platform's features and design an intuitive user interface. We will set up user registration and authentication mechanisms to ensure secure access to the platform.

Project Tasks:

1:Authentication Introduction:

TASK:

A. Why Authentication ?:

Access Control: Authentication ensures only authorized users can use the app, preventing unauthorized access to content.

User Data Protection: It safeguards sensitive user information, like payment details and viewing history, from potential breaches.

Personalization: Authentication allows for the creation of user profiles, enabling tailored content recommendations and user experiences.

Revenue Protection: It enforces subscription limits, ensuring fair usage and protecting the app's financial interests

B. What is the use of application:

Functionality: Applications are designed to perform specific tasks or provide functionality, such as communication, productivity, entertainment, or information access.

User Convenience: Applications enhance user convenience by offering intuitive interfaces, streamlined access to services, and efficient ways to perform tasks on various devices.

Solving Problems: Applications address specific needs and problems, whether it's managing finances, tracking fitness, editing photos, or connecting with others, making life more efficient and enjoyable.

Economic and Social Impact: Applications contribute to economic growth and influence society by creating job opportunities, driving innovation, and connecting people globally through social and communication apps.

C. Why authentication ?:

Security: Authentication ensures that only authorized users can access the app, protecting sensitive data and preventing unauthorized access or breaches.

Personalization: Authentication allows the app to create and maintain user profiles, enabling personalized experiences, recommendations, and tailored content.

Access Control: It enforces subscription limits and user privileges, preventing misuse and protecting the app's revenue model.

Trust and Accountability: Authentication establishes trust with users, as they know their data is secure, and it helps maintain accountability by tracking and logging user activities within the app..

D. Application of authentication :

Cybersecurity: Used to protect networks and data from unauthorized access.

Finance and Banking: Ensures secure online transactions and identity verification.

Healthcare: Safeguards electronic health records and patient data.

E-commerce: Secures user accounts and online transactions to prevent fraud

2: Why authentication for this application:

TASK:

A. Security:

Access Control: Authentication serves as a gatekeeper, allowing only authorized individuals to access systems, data, and resources, thus preventing unauthorized entry.

Data Protection: Authentication safeguards sensitive information and confidential data from exposure to unauthorized users, reducing the risk of data breaches.

Dentity Verification: It ensures that users are who they claim to be, enhancing overall system security by reducing the risk of impersonation or unauthorized access.

Audit Trails: Authentication systems help create detailed logs of user activities, facilitating forensic analysis and accountability in the event of security incidents or breaches.

B. Check user's count:

Usage Monitoring: Authentication can track the number of users accessing a system, helping administrators monitor usage patterns and detect any irregularities.

Access Limitations: By counting and managing user access, authentication can enforce subscription limits, ensuring fair usage and revenue protection for services.

Resource Management: Monitoring user counts assists in resource allocation and optimization, ensuring that system performance remains consistent even during peak usage.

Security: Verifying user counts can help detect and prevent unauthorized access or suspicious activities, enhancing the overall security of the system.

C. Reduce account exploitation:

Identity Verification: Authentication methods verify the identity of users, reducing the risk of unauthorized individuals gaining access to accounts.

Multi-Factor Authentication (MFA): MFA adds an extra layer of security, making it significantly more challenging for attackers to exploit accounts, even if they have stolen login credentials.

Account Recovery: Authentication helps in establishing secure processes for account recovery, preventing unauthorized access even when users forget their credentials.

User Behavioral Analysis: Some authentication systems use behavioral analysis to detect anomalies in user behavior, further reducing the risk of account exploitation by identifying suspicious activities.

3: Program (Coding Part):

TASK:

A. Coding for authentication:

Code part -1: unauth-page

```
"use client"; import { signIn } from "next-
auth/react"; import { motion } from "framer-motion";
import { PlusIcon } from
"@heroicons/react/24/outline"; import { useRouter }
from "next/navigation"; import { useState } from
"react";


const questions = [
  {
```

ques: "What is Netflix?",

ans: `Netflix is a streaming service that offers a wide variety of award-winning TV shows, movies, anime, documentaries and more – on thousands of internet-connected devices. You can watch as much as you want, whenever you want, without a single ad – all for one low monthly price. There's always something new to discover, and new TV shows and movies are added every week!`,

},

{

ques: "How much does Netflix cost", ans: `Watch Netflix on your smartphone, tablet, Smart TV, laptop, or streaming device, all for one fixed monthly fee. Plans range from ₹ 149 to ₹ 649 a month.

No extra costs, no contracts.`,

},

{

ques: "What can I watch on Netflix?", ans: `Watch anywhere, anytime. Sign in with your Netflix account to watch instantly on the web at netflix.com from your personal computer or on any internet-connected device that offers the Netflix app, including smart TVs, smartphones, tablets, streaming media players and game consoles.

You can also download your favourite shows with the iOS, Android, or Windows 10 app. Use downloads to watch while you're on the go and without an internet connection. Take Netflix with you anywhere.`, },

{

ques: "How do I cancel?", ans: `Netflix is flexible. There are no annoying contracts and no commitments. You can easily cancel your account online in two clicks. There are no cancellation fees – start or stop your account anytime.`, },

{

```
    ques: "What can I watch on Netflix?", ans: `Netflix has an extensive
    library of feature films, documentaries, TV
    shows, anime, award-winning Netflix originals, and more. Watch as much as
    you want, anytime you want.`,
  },
  {
    ques: "Is Netflix good for kids?", ans: `The Netflix Kids experience is
    included in your membership to give parents control while kids enjoy family-
    friendly TV shows and films in their own space.

Kids profiles come with PIN-protected parental controls that let you restrict the
    maturity rating of content kids can watch and block specific titles you don't
    want kids to see.`,
  },
];

function UnauthBanner({ router }) {
  return (
    <div className="h-[65vh] sm:h-[90vh] xl:h-[95vh] bg-cover bg-no-repeat bg-
    [url('https://assets.nflxext.com/ffe/siteui/vlv3/84526d58-475e-4e6f-9c81-
    d2d78ddce803/e3b08071-f218-4dab-99a2-80315f0922cd/LK-en-20221228-
    popsignuptwoweeks-perspective_alpha_website_small.jpg')] border-b-8
    border-gray-800 ">
      <div  className="bg-black  bg-opacity-70  h-
      [100vh]


    "
      >
        <div className="flex items-center justify-between">
          <img
```

```
      src="https://rb.gy/ulxxee" alt="netflix" width={120}
      height={120} className="w-28 sm:w-36 lg:w-52 ml-4
      sm:ml-8 pt-4" onClick={() => router.push("/")}
    />
    <div className="flex mr-4 sm:mr-10">
      <button onClick={() => signIn("github")} className="h-8 px-1
        sm:px-4 m-2 text-white bg-[#e50914] rounded"
      >
        Sign In
      </button>
    </div>
  </div>
  <div className="h-[55vh] sm:h-[80vh] w-[90%] md:w-[80%] mx-[5%]
md:mx-[10%] flex flex-col items-center justify-center text-white text-center">
    <h1 className="text-2xl sm:text-4xl lg:text-5xl xl:text-6xl sm:px-[15%]
md:px-[15%] lg:mx-14 lg:px-[7%] xl:px-[15%] font-medium"> Unlimited
movies, TV shows, and more..
    </h1>
    <h2 className="text-lg sm:text-1xl lg:text-2xl font-medium m-2 sm:m- 4">

      Watch anywhere. Cancel anytime.
    </h2>
    <div className="flex justify-center">
      <button onClick={() => signIn("github")} className="bg-
        red-600 hover:bg-[#e50914] p-4 rounded"
      >
        Sign In to Get Started
      </button>
    </div>
```

```
      </div>
    </div>
  </div>
 );
}


export default function UnauthPage() {
 const router = useRouter();
 const [showCurrentAns, setShowCurrentAns] = useState(null);


 return (
  < motion.div initial={{
   opacity: 0 }}
    whileInView={{ opacity: 1 }} viewport={{
    once: true }}
  >
   <main>
    <div className="bg-[#000000]">
     <UnauthBanner router={router} />
     <div className="border-b-8 border-gray-800 pb-8">
      <div className="flex flex-col h-[85vh] lg:h-[95vh] text-white px-8 sm:px-14 md:px-28 lg:px-48 xl:px-80 mt-3 sm:mt-14">
       <h1 className="mb-5 text-xl sm:text-3xl md:text-4xl lg:text-5xl text-bold text-center px-14 md:px-0">
        Frequently asked questions
       </h1>
       {questions.map((item, index) => (
        <div className="flex flex-col gap-3">
```

```
            <div onClick={() => setShowCurrentAns(showCurrentAns ===
      index ? null : index)}
              className="flex justify-between p-3 lg:p-5 mt-2 bg-[#303030]
      cursor-pointer"
            >
              <h2>{item.ques}</h2>
              <PlusIcon className="h-7 w-7" color="white" />
            </div>
            {showCurrentAns === index && (
              <div className="p-3 lg:p-5 mt-2 bg-[#303030] cursor-pointer">
                {item.ans}
              </div>
            )}
          </div>
        ))}
      </div>
    </div>
  </main>
 </motion.div>
);


  }
```

Coding Part -2 : account-form :

```
"use client";
import { motion } from "framer-motion";

export default function AccountForm({
  showAccountForm,
  formData,
  setFormData,
  handleSave
```

```
}) { return
  (
    showAccountForm && (
      <motion.div initial={{ opacity:
        0 }} whileInView={{ opacity:
        1 }} viewport={{ once: true
        }}
      >
        <div className="px-8 py-8 h-[300px] fixed top-[10px] gap-3 flex flex-col
items-start right-[10px] bg-black opacity-[0.85] z-[999]">
          <div className="flex flex-col gap-5">
            <input name="name"
              type="text"
              value={formData["name"]
              } onChange={(e) =>
              setFormData({
                 ...formData,
                 [e.target.name]: e.target.value,
               }) }
              placeholder="Enter your name"
              className="px-5 py-3 rounded-lg placeholder:text-red-700 text-lg text-
[#e5b109] outline-none focus:outline-none"
            />
             <input
              name="pin"
              type="password"
              value={formData["pin"]}
              onChange={(e) =>
              setFormData({
                 ...formData,
                 [e.target.name]: e.target.value,
               })
              }
              maxLength={4} placeholder="Enter
              your PIN"
              className="px-5 py-3 rounded-lg placeholder:text-red-700 text-lg text-
[#e5b109] outline-none focus:outline-none"
            />
            <button
            onClick={handleSave}
            className="border p-4 bg-[#e5b109] outline-none rounded-lg text-black
text-lg font-bold"
```

```
          >Save</button>
        </div>
      </div>
    </motion.div>
  )
 );
}
```

Coding part -3 : auth-provider :

```
"use client";

import { SessionProvider } from "next-auth/react";

export default function NextAuthProvider({ children }) {
  return <SessionProvider>{children}</SessionProvider>;
```

4:Tools and system requirements used:

Code Editor: VS code.

Frame work : React.js

1.6 GHz or faster processor

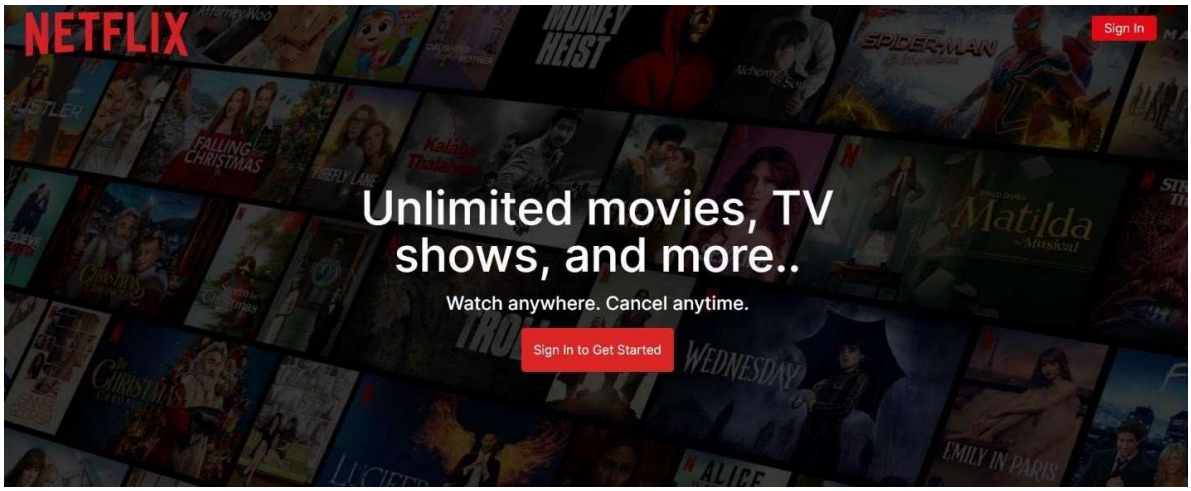
1 GB of RAM

OS X Yosemite

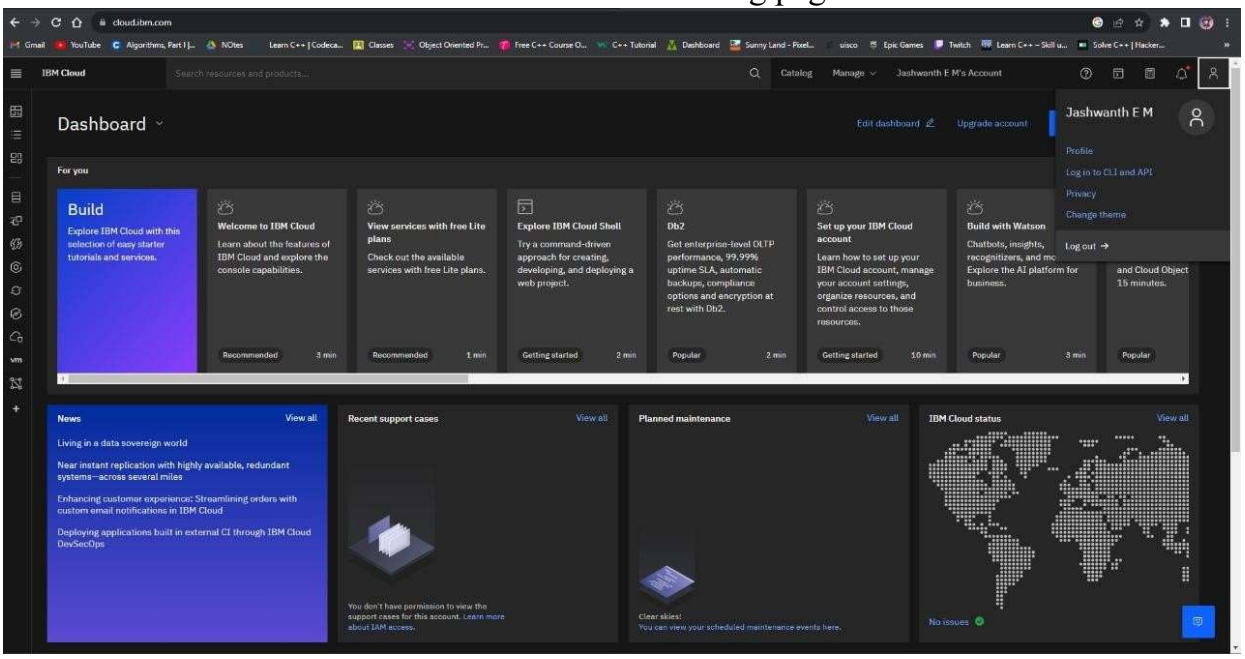
Windows 7 (with .NET Framework 4.5)

Linux with GLIBCXX version 3.4.15

5:Final output (sample screenshots):

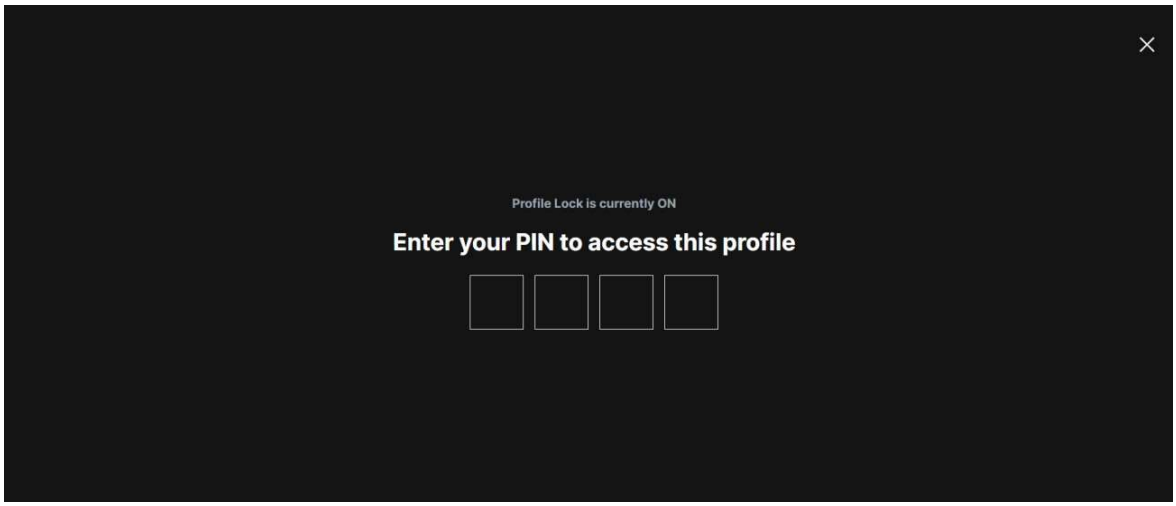


Screen shot - 1 : Landing page

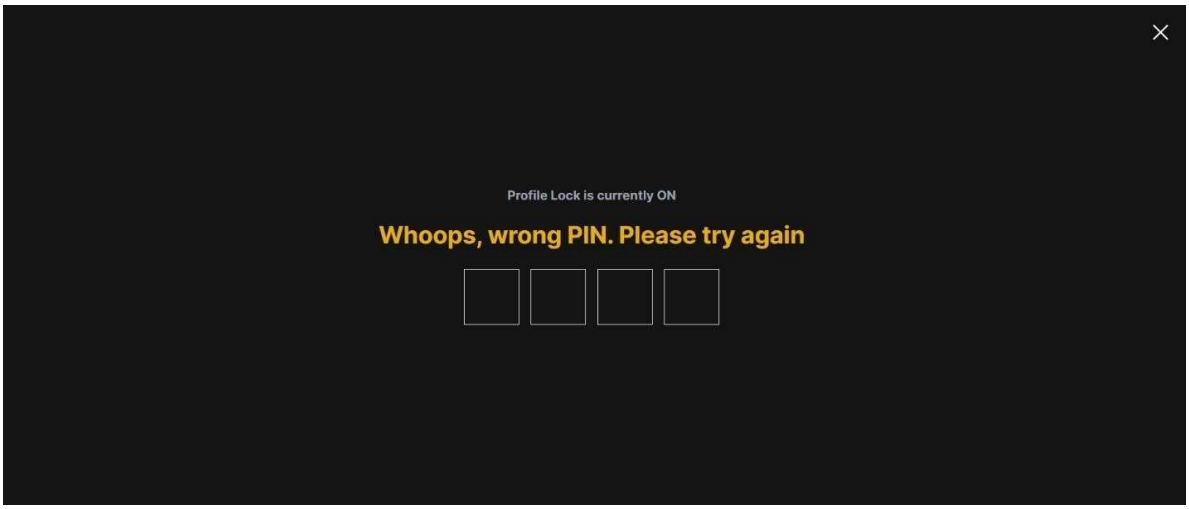


Screen shot - 2 : Using IBM cloud account for backend

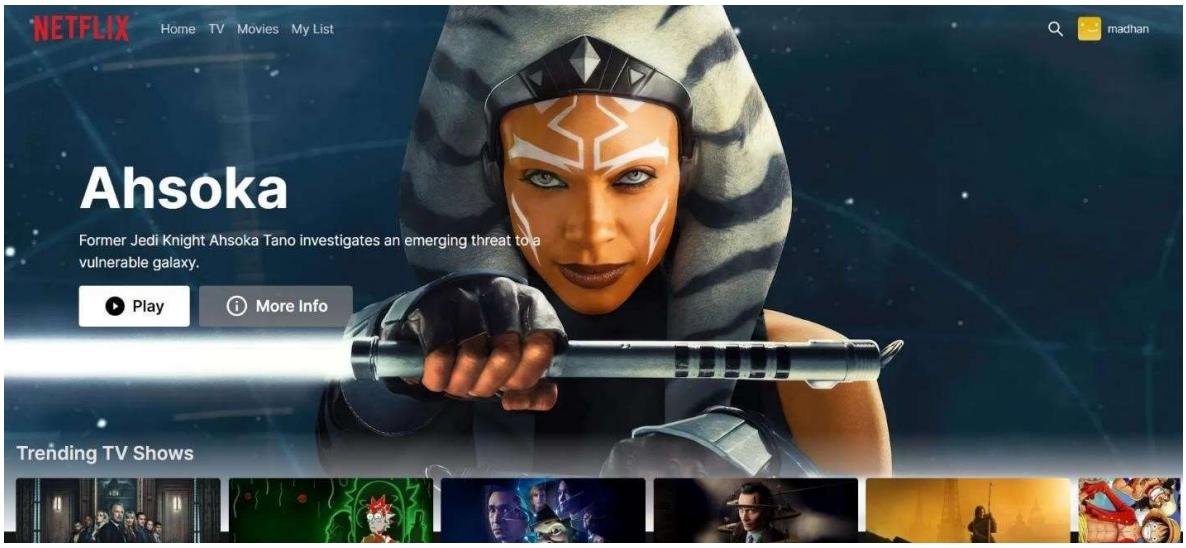
Screen shot - 3 : Authentify each user



Screen shot - 4 : Authentify using PIN



Screen shot - 5 : When users enter wrong pin



Screen shot – 6 : Final Page after login

Conclusion:

In Phase 3 of Media streaming app project, our objective is to achieve these milestones through the successful completion of tasks. These innovative features will provide users with a more engaging and personalized show watching experience, leading to increased user satisfaction and retention.