

**PROFESSIONAL TRAINING REPORT**  
**at**  
**Sathyabama Institute of Science and Technology**  
**(Deemed to be University)**

Submitted in partial fulfillment of the requirements for the award of  
Bachelor of Technology in Information Technology

By  
**GOBIGA B**  
**REG. NO. 39120032**



**DEPARTMENT OF INFORMATION TECHNOLOGY**  
**SCHOOL OF COMPUTING**  
**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**  
**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**  
**CHENNAI – 600119, TAMILNADU**

**APRIL 2022**



# **SATHYABAMA**

## **INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC**  
(Established under Section 3 of UGC Act, 1956)  
JEPPIAAR NAGAR, RAJIV GANDHI SALAI  
CHENNAI– 600119  
[www.sathyabama.ac.in](http://www.sathyabama.ac.in)

---

### **DEPARTMENT OF INFORMATION TECHNOLOGY**

#### **BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **GOBIGA B(Reg. No: 39120032)** who carried out the project entitled “**NETFLIX CLONE**” under my supervision from February 2022 to April 2022.

**Internal Guide**

**Dr. R.M.GOMATHI, M.Tech., Ph.D.**

**Head of the Department**

**Dr. R. Subhashini, M.E., Ph.D.**

---

**Submitted for Viva voce Examination held on \_\_\_\_\_**

**Internal Examiner**

**External Examiner**

## DECLARATION

I, **GOBIGA B** hereby declare that the project report entitled “**NETFLIX CLONE**” done by me under the guidance of **Dr.R.M.GOMATHI, M.Tech.,Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Information Technology.

**DATE:10-04-2021**

**PLACE: CHENNAI**

A handwritten signature in black ink, appearing to read 'B. Gobiga', with a stylized flourish at the end. Below the signature, there is a small, faint rectangular stamp.

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D, Dean**, School of Computing, **Dr. R. Subhashini, M.E., Ph.D. Head of the Department** of **Information Technology** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. R.M.GOMATHI, M.Tech., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

# **TRAINING CERTIFICATE**

## **ABSTRACT**

Video Streaming, in various forms of video on demand (VOD), live, and 360 degree streaming, has grown dramatically during the past few years. In comparison to traditional cable broadcasters whose contents can only be watched on TVs, video streaming is ubiquitous and viewers can flexibly watch the video contents on various devices, ranging from smartphones to laptops, and large TV screens. Such ubiquity and flexibility are enabled by interweaving multiple technologies, such

as video compression, cloud computing, content delivery networks, and several other technologies. As video streaming gains more popularity and dominates the Internet traffic, it is essential to understand the way it operates and the interplay of different technologies involved in it.

Accordingly, the first goal of this paper is to unveil sophisticated processes to deliver a raw captured video to viewers' devices. In particular, we elaborate on the video encoding, transcoding, packaging, encryption, and delivery processes. We survey recent efforts in academia and industry to enhance these processes. As video streaming industry is increasingly becoming reliant on cloud computing, the second goal of this survey is to explore and survey the ways cloud services are utilized to enable video streaming services. The third goal of the study is to position the undertaken research works in cloud-based video streaming and identify challenges that need to be obviated in future to advance cloud-based video streaming industry to a more flexible and user-centric service.

## LIST OF ABBREVIATIONS

TERM	ABBREVIATION
JS	JAVASCRIPT
HTML	HYPertext MARKUP LANGUAGE
CSS	CASSCADING STYLESHEETS

## LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO
4.1	SYSTEM ARCHITECTURE/ IDEATION MAP	12

	TABLE OF CONTENTS	
CHAPTER	TITLE	PAGE No
	ABSTRACT	6
	LIST OF ABBREVIATIONS AND LIST OF FIGURES	7
1	INTRODUCTION	9
2	AIM AND SCOPE	10
	2.1 AIM	
	2.2. SCOPE	
3	SYSTEM SPECIFICATION	11
	3.1 HARDWARE SPECIFICATION	
	3.2 SOFTWARE SPECIFICATION	
4	METHODOLOGY	12
5	CONCLUSION AND FUTURE SCOPE	17
	REFERENCES	18
	APPENDIX SNAPSHOTS	19
	SOURCE CODE	25



# **CHAPTER 1**

## **1.1.INTRODUCTION**

Video streaming has received much attention for some time now, and its popularity has grown manifold in the last decade, particularly with the deployment of widely known video streaming services, such as YouTube, Netflix, Hulu, Daily Motion, and others. Video streaming now accounts for a large share of the total traffic the Internet, and as such the transmission mechanisms used by video servers have an impact not only on the quality of the video presentation to the clients, but also on the total traffic impact on the network. As a result, many new techniques for video streaming have developed. Moreover, with the continual expansion of computing, now encompassing everyday objects, the concepts of ubiquitous and pervasive computing describe a world in which the boundaries between the virtual and real world have been blurred. Wireless networks and mobile computing now feature in most distributed applications, leading to a substantially new challenge for video streaming: the computing/memory/energy capabilities of small mobile devices, and the low-quality channels interconnecting them and linking them to the Internet, impose limitations on the quality of videos that can be streamed.

It also suggests the need for multiple encoding levels for each title, so that devices and networks of different characteristics can be supported. Streaming is the continuous transmission of audio or video files from a server to a client. In simpler terms, streaming is what happens when consumers watch TV or listen to podcasts on Internet-connected devices. With streaming, the media file being played on the client device is stored remotely, and is transmitted a few seconds at a time over the Internet.

# **CHAPTER 2**

## **AIM AND SCOPE**

### **2.1. AIM:**

The aim to create a video streaming application using ReactJS. This application is a clone of Netflix website built using React.JS & Typescript as a Front-end & Firebase as Back-end. It's not a replica, and it doesn't have all the features of Netflix website. It's a similar version of Netflix with my own design touch, showing my abilities in React.JS to build something advanced like Netflix. It contains the home page, sign-in page, sign-up page, browse page, and movie player.

### **2.2. PROJECT SCOPE:**

Web applications have become the primary necessity for modern consumers. They help out customers to get their necessary products or services in a short time. The incorporation of technology in our day to day services like paying bills to booking cabs involves complex procedures and innumerable iterations to serve customers efficiently. As on-demand apps offer the comfort to get products delivered at the doorsteps, people need not have to commute to buy food, groceries, clothes, and other commercial necessities. Technology has made it possible for people to get their necessities with a few simple taps and swipes on their smartphones.

## **CHAPTER 3**

### **3.1.SYSTEM SPECIFICATIONS**

#### **3.1.HARDWARE REQUIREMENT**

- ☐ RAM : 4GB
- ☐ Processor Intel Core i3 and above

#### **3.2. SOFTWARE REQUIREMENT**

- Front-end: ReactJS and TypeScript
- Back-end: Firebase

## CHAPTER 4

### 4.1.METHODOLOGY

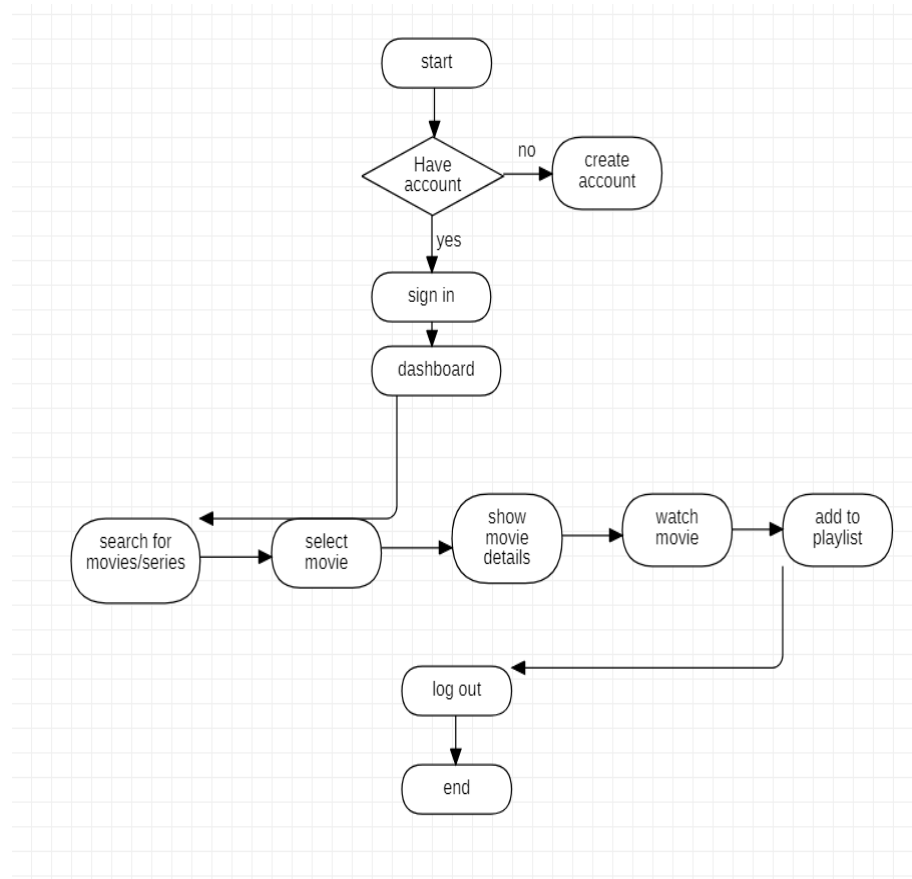


Fig.4.1. System Architecture/ Ideation Map

### 4.2.Technology Used

I have built this project using the following tools & techniques:

- React.JS
- Typescript.
- Firebase.
- Styled-Components.
- React Router.
- React Hooks.
- Compound Components.

## 4.3. FEATURES:

This video streaming application has following features in it:

### HOME PAGE:

The Home Page consists of 5 main sections:

1) Header, which includes:

- Logo: it redirects you to the home page when you click on it.
- Sign-in button: it redirects you to the sign-in page.
- Feature title & subtitle: It shows the main sentences of the website.

2) Optform: It's a text input field and a button, It redirects you to the sign-up page once you click on the button.

3) Jumbotron: This section contains some images and paragraphs beside it, showing the advantages of Netflix. The data of this Jumbotron came from jumbo.json file.

4) Frequently Asked Questions: This section contains the FAQs in a form of accordion, when you click anywhere in the grey area of the question the answer appears below it, and then you can close the answer by clicking again on the same grey area of the question. The data of these FAQs came from faqs.json file.

5) Footer: It contains useful links users may need it.

### SIGN-IN PAGE:

The Sign-in Page consists of 3 main sections:

1) Header, which includes:

- Logo: it redirects you to the home page when you click on it.

2) Sign-in Form, which includes:

- Email address input field.
- Password input field.
- Sign-in Button: It has a validation option, if any field in the form is empty it will be disabled. If the form fields have any data it will be active and will send the data to the firebase database in the backend for authentication. It has also an error handling function.
- Link to Sign-up Page: it redirects you to the sign-up page.

3) Footer: It contains useful links users may need it.

## **SIGN-UP PAGE:**

The Sign-up Page consists of 3 main sections:

*1) Header, which includes:*

- Logo: it redirects you to the home page when you click on it.

*2) Sign-up Form, which includes:*

- First Name input field.
- Email address input field.
- Password input field.

- Sign-up Button: It has a validation option, if any field in the form is empty it will be disabled. If the form fields have any data it will be active and will send the data to the firebase database in the backend for registration. It has also an error handling function.
- Link to Sign-in Page: it redirects you to the sign-in page.

3) *Footer*: It contains useful links users may need it.

## **BROWSE PAGE:**

The Browse Page consists of 5 main sections:

1) Header, which includes:

- Logo: it redirects you to the home page whenever you click it.
- Categories Links: It shows the movies of a specific category when you click on it, for example, if you click on the films link it will be active and the browse page will show only the films. And if you click on the series link it will be active and the browse page will show only the series.
- Featured Movie Title & Description: It shows the title and description of the featured movie.
- Play Button: it shows the video player to play the movie.

2) Movies Slides: It's a slides shows the movies based on their genre. The genres and all movie information had been retrieved from the Firebase database.

3) Movie Card: It's an image represent the movie, when you hover over it became bigger and it will show its card feature if you click on it.

4) Card Feature: it's another section that appears under the movie slide if you click on any movie card, it contains more information about the movie like title, description, a special background represent the movie, and play button. when you click on the play button it shows the video player to play the movie. and you can close the card feature by clicking on the close icon in the top right corner of the card feature.

5) Video Player: it's a video player that has full controls, appears in the middle of the screen when you click on any play button, and you have to click on the play icon in the video player after it show up because it doesn't have an autoplay option currently. And when the video player show up the whole screen became an overlay, and only the video appears in the middle, and when you scroll up and down the video player moves with you. The video player should show the video of this movie which you clicked on it, but for this project purpose, it shows only one video as a sample for all movies. You can close the video player anytime by clicking anywhere else on the screen.

#### **4.3.BENEFITS OF USING THIS WEBSITE**

- 1)Creates space for community building.
- 2)Creates a platform for budding creators.
- 3) Various contents bring diversity in people by making them aware of different culture.



## **CHAPTER 5**

### **CONCLUSION**

#### **5.1 CONCLUSION**

- Video Streaming is a technology that has completely changed the entertainment industry as well as consumption models among audience members. A lot has changed since that very first Real Player transmission in 1995. Since then, technology has been constantly improving, making content delivery and access easier no matter the platform trying to access it. Various contents bring diversity in people by making them aware of different culture.

#### **5.2. FUTURE SCOPE**

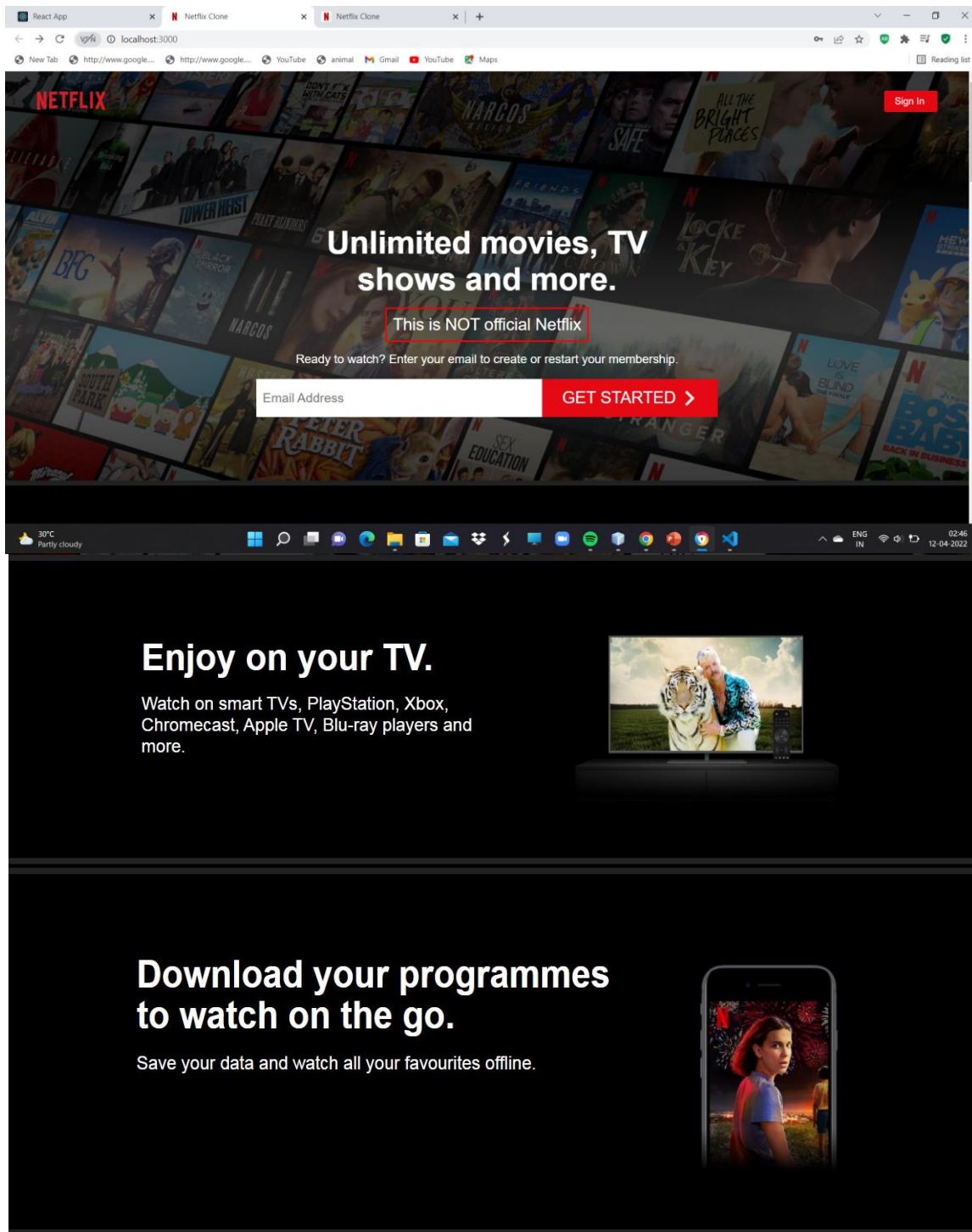
Netflix, Inc. is one of the best examples regarding commercial applications for Video Streaming. With millions subscribing to the service all over the world, the company has found a way to capitalize its services using its title stock and outsourced infrastructure

## REFERENCES

1. [www.w3schools.com](http://www.w3schools.com)
2. [www.tutorialpoint.com](http://www.tutorialpoint.com)
3. Clever Programmer-Youtube

## APPENDIX SNAPSHOTS

Home page:



# Watch everywhere.

Stream unlimited films and TV programmes on your phone, tablet, laptop and TV without paying more.



## Frequently Asked Questions

What is Netflix?	+
How much does Netflix cost?	+
Where can I watch?	+
How do I cancel?	+
What can I watch on Netflix?	+

Ready to watch? Enter your email to create or restart your membership.

GET STARTED >

Questions? Contact us.

[FAQ](#)

[Help Center](#)

[Account](#)

[Media Center](#)

[Investor Relations](#)

[Jobs](#)

[Ways to Watch](#)

[Terms of Use](#)

[Privacy](#)

[Cookie Preferences](#)

[Corporate Information](#)

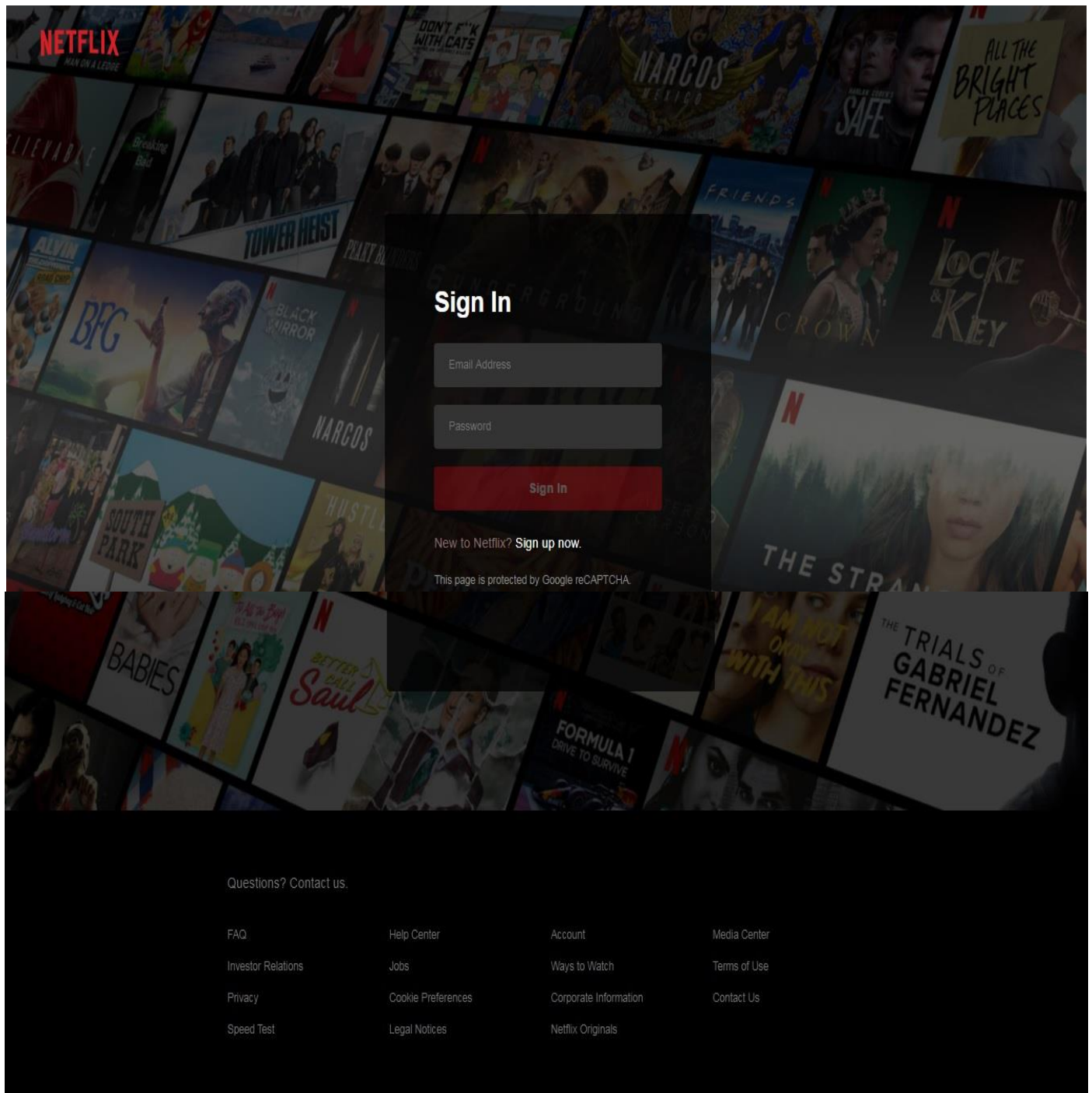
[Contact Us](#)

[Speed Test](#)

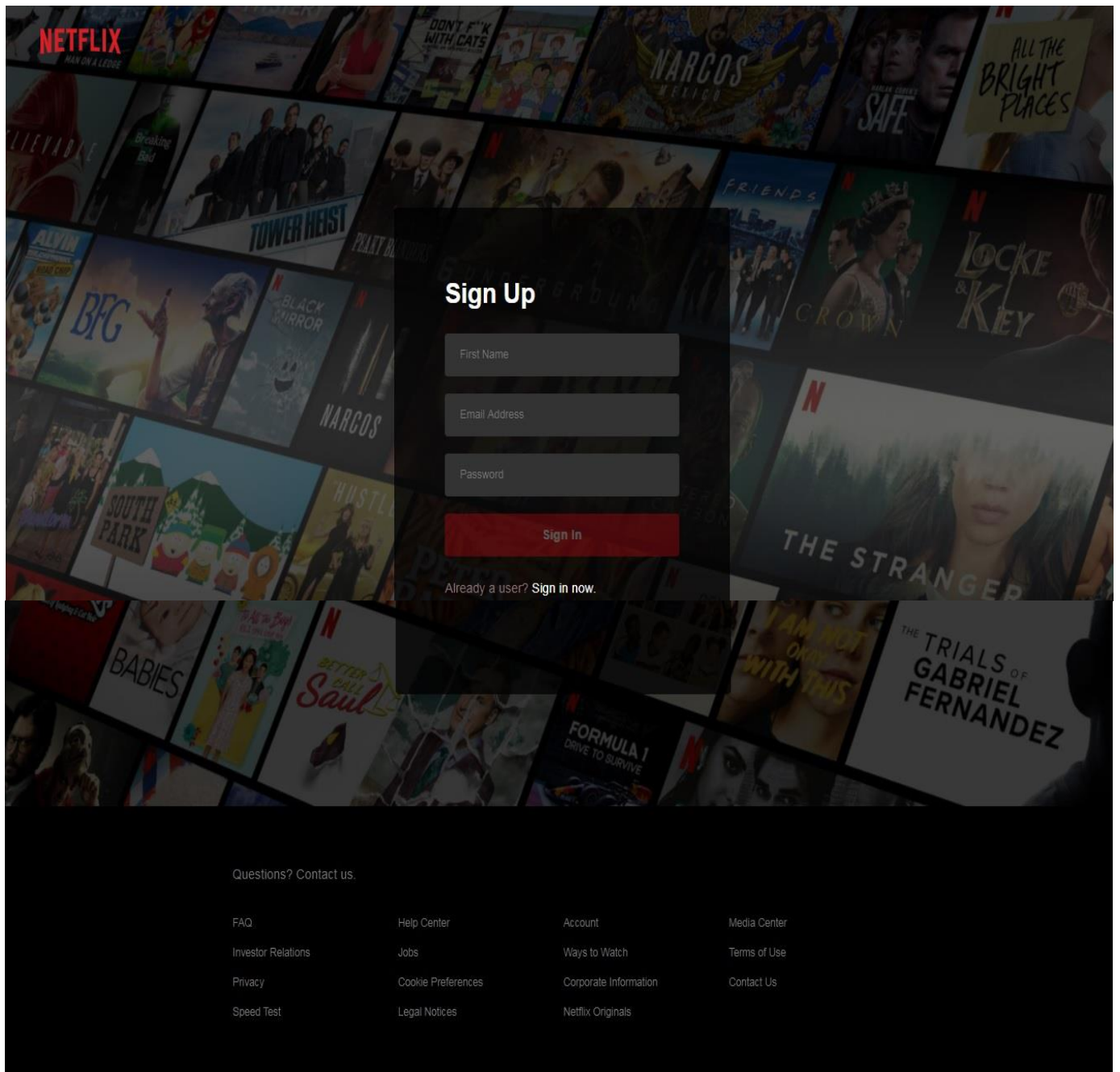
[Legal Notices](#)

[Netflix Originals](#)

**SignIn page:**



SignUp page:

The image shows the Netflix sign-up page. The background is a collage of various Netflix titles including 'Narcos', 'Friends', 'The Stranger', 'South Park', 'Breaking Bad', 'The Crown', 'Locke & Key', 'The Trials of Gabriel Fernandez', 'Formula 1', 'I Am Not Okay With This', 'Babies', 'Saul', 'Hustlers', 'The BFG', 'Black Mirror', 'Tower Heist', 'Pleasantville', 'Don't F\*ck With Cats', 'Safe', 'All the Bright Places', 'Liev Schaefer', 'Alvin', 'The Christmas', 'Katie Couric', 'The Christmas', 'Katie Couric', 'The Christmas', 'Katie Couric'. In the center, there is a white 'Sign Up' form with three input fields: 'First Name', 'Email Address', and 'Password'. Below these fields is a red 'Sign In' button. At the bottom of the form, it says 'Already a user? Sign in now.' At the bottom of the page, there is a dark footer with the text 'Questions? Contact us.' and a grid of links: FAQ, Help Center, Account, Media Center, Investor Relations, Jobs, Ways to Watch, Terms of Use, Privacy, Cookie Preferences, Corporate Information, Contact Us, Speed Test, Legal Notices, Netflix Originals.



NETFLIX

Films

Series

## Watch Patman Now

Forever alone in a crowd, failed comedian Arthur Fleck seeks connection as he walks the streets of Gotham City. Arthur wears two masks, the one he paints for his day job as a clown, and the guise he projects in a futile attempt to feel like he is part of the world around him.

Play

### Drama



### Fight Club

Discontented with his capitalistic lifestyle, a white-collared insomniac forms an underground fight club with Tyler, a careless soap salesman. The project soon spirals down into something sinister.

Play



### Thriller



### Children



### Suspense



## Romance



Questions? Contact us.

[FAQ](#)

[Help Center](#)

[Account](#)

[Media Center](#)

[Investor Relations](#)

[Jobs](#)

[Ways to Watch](#)

[Terms of Use](#)

[Privacy](#)

[Cookie Preferences](#)

[Corporate Information](#)

[Contact Us](#)

[Speed Test](#)

[Legal Notices](#)

[Netflix Originals](#)

Play

Drama



Thriller





## SOURCE CODE

### **APP.JS:**

```
import React from "react";
import { BrowserRouter as Router, Switch, Route } from "react-router-dom";
import HomePage from "../pages/HomePage";
import SigninPage from "../pages/SigninPage";
import SignupPage from "../pages/SignupPage";
import BrowsePage from "../pages/BrowsePage";
```

```
function App() {
  return (
    <Router>
      <Switch>
        <Route exact path="/">
          <HomePage />
        </Route>
        <Route path="/signin">
          <SigninPage />
        </Route>
        <Route path="/signup">
          <SignupPage />
        </Route>
        <Route path="/browse">
          <BrowsePage />
        </Route>
      </Switch>
    </Router>
  );
}
```

```
export default App;
```

### **HOME PAGE.JS:**

```
import React from "react";
import HeaderComponent from "../compounds/HeaderCompound";
import OptFormCompound from "../compounds/OptFormCompound";
import JumboCompound from "../compounds/JumboCompound";
import Seperator from "../components/Seperator/Seperator";
import AccordionCompound from "../compounds/AccordionCompound";
import FooterCompound from "../compounds/FooterCompound";
```

```
function HomePage() {
  return (
    <>
      <HeaderCompound>
        <OptFormCompound />
      </HeaderCompound>
      <Seperator />
      <JumboCompound />
    </>
  );
}
```

```

    <AccordionCompound />
    <OptFormCompound />
    <Seperator />
    <FooterCompound />
  </>
);
}

export default HomePage;

```

### **SIGNINPAGE.IS:**

```

import React, { useState, useContext } from "react";
import { useHistory } from "react-router-dom";
import { FirebaseContext } from "../context/FirebaseContext";
import HeaderWrapper from "../components/Header/HeaderWrapper";
import NavBar from "../components/Header/NavBar";
import Logo from "../components/Header/Logo";
import FooterCompound from "../components/FooterCompound";
import SignFormWrapper from "../components/SignForm/SignFormWrapper";
import SignFormBase from "../components/SignForm/SignFormBase";
import SignFormTitle from "../components/SignForm/SignFormTitle";
import SignFormInput from "../components/SignForm/SignFormInput";
import SignFormButton from "../components/SignForm/SignFormButton";
import SignFormText from "../components/SignForm/SignFormText";
import SignFormLink from "../components/SignForm/SignFormLink";
import SignFormCaptcha from "../components/SignForm/SignFormCaptcha";
import SignFormError from "../components/SignForm/SignFormError";
import Warning from "../components/Header/Warning";

function SigninPage() {
  const history = useHistory();
  const { firebase } = useContext(FirebaseContext);

  const [emailAddress, setEmailAddress] = useState("");
  const [password, setPassword] = useState("");
  const [error, setError] = useState("");

  const IsInvalid = password === "" || emailAddress === "";

  function handleSubmit(event) {
    event.preventDefault();

    firebase
      .auth()
      .signInWithEmailAndPassword(emailAddress, password)
      .then(() => {
        setEmailAddress("");
        setPassword("");
        history.push("/browse");
      });
  }
}

```

```

    })
    .catch((error) => setError(error.message));
  }

  return (
    <>
    <HeaderWrapper className="header-wrapper-home">
      <NavBar className="navbar-signin">
        <Logo />
      </NavBar>
      <SignFormWrapper>
        <SignFormBase onSubmit={handleSubmit} method="POST">
          <Warning>NOT official Netflix</Warning>
          <SignFormTitle>Sign In</SignFormTitle>
          {error ? <SignFormError>{error}</SignFormError> : null}
          <SignFormInput
            type="text"
            placeholder="Email Address"
            value={emailAddress}
            onChange={({ target }) => setEmailAddress(target.value)}
          />
          <SignFormInput
            type="password"
            placeholder="Password"
            autoComplete="off"
            value={password}
            onChange={({ target }) => setPassword(target.value)}
          />
          <SignFormButton disabled={isInvalid}>Sign In</SignFormButton>
          <SignFormText>
            New to Netflix?
            <SignFormLink href="/signup">Sign up now.</SignFormLink>
          </SignFormText>
          <SignFormCaptcha>
            This page is protected by Google reCAPTCHA to ensure you are not a
            bot.
          </SignFormCaptcha>
        </SignFormBase>
      </SignFormWrapper>
    </HeaderWrapper>
    <FooterCompound />
  </>
  );
}

export default SigninPage;

```

### **SIGNUPPAGE.JS:**

```

import React, { useState, useContext } from "react";
import { useHistory } from "react-router-dom";
import { FirebaseContext } from "../context/FirebaseContext";

```

```

import HeaderWrapper from "../components/Header/HeaderWrapper";
import NavBar from "../components/Header/NavBar";
import Logo from "../components/Header/Logo";
import FooterCompound from "../compounds/FooterCompound";
import SignFormWrapper from "../components/SignForm/SignFormWrapper";
import SignFormBase from "../components/SignForm/SignFormBase";
import SignFormTitle from "../components/SignForm/SignFormTitle";
import SignFormInput from "../components/SignForm/SignFormInput";
import SignFormButton from "../components/SignForm/SignFormButton";
import SignFormText from "../components/SignForm/SignFormText";
import SignFormLink from "../components/SignForm/SignFormLink";
import SignFormCaptcha from "../components/SignForm/SignFormCaptcha";
import SignFormError from "../components/SignForm/SignFormError";
import Warning from "../components/Header/Warning";

```

```

function SignupPage() {
  const history = useHistory();
  const { firebase } = useContext(FirebaseContext);

  const [firstName, setFirstName] = useState("");
  const [emailAddress, setEmailAddress] = useState("");
  const [password, setPassword] = useState("");
  const [error, setError] = useState("");

```

```

  const isValid = password !== "" || emailAddress !== "" || firstName !== "";

```

```

  function handleSubmit(event) {
    event.preventDefault();

```

```

    firebase
      .auth()
      .createUserWithEmailAndPassword(emailAddress, password)
      .then((result) =>
        result.user
          .updateProfile({
            displayName: firstName,
          })
          .then(() => {
            setFirstName("");
            setEmailAddress("");
            setPassword("");
            history.push("/browse");
          })
      )
      .catch((error) => setError(error.message));
  }

```

```

  return (
    <>
      <HeaderWrapper className="header-wrapper-home">
        <NavBar className="navbar-signin">
          <Logo />

```

```

</NavBar>
<SignFormWrapper>
  <SignFormBase onSubmit={handleSubmit} method="POST">
    <Warning>NOT official Netflix</Warning>
    <SignFormTitle>Sign Up</SignFormTitle>
    {error ? <SignFormError>{error}</SignFormError> : null}
    <SignFormInput
      type="text"
      placeholder="First Name"
      value={firstName}
      onChange={({ target }) => setFirstName(target.value)}
    />
    <SignFormInput
      type="text"
      placeholder="Email Address"
      value={emailAddress}
      onChange={({ target }) => setEmailAddress(target.value)}
    />
    <SignFormInput
      type="password"
      placeholder="Password"
      autoComplete="off"
      value={password}
      onChange={({ target }) => setPassword(target.value)}
    />
    <SignFormButton disabled={isInvalid}>Sign Up</SignFormButton>
    <SignFormText>
      Already a user?
      <SignFormLink href="/signin">Sign in now.</SignFormLink>
    </SignFormText>
    <SignFormCaptcha>
      This page is protected by Google reCAPTCHA to ensure you are not a
      bot.
    </SignFormCaptcha>
  </SignFormBase>
</SignFormWrapper>
</HeaderWrapper>
<FooterCompound />
</>
);
}

```

export default SignupPage;

BROWSEPAGE.JS:

```

import React, { useState } from "react";
import useContent from "../custom-hooks/useContent";
import HeaderWrapper from "../components/Header/HeaderWrapper";
import NavBar from "../components/Header/NavBar";
import Logo from "../components/Header/Logo";
import FeatureWrapper from "../components/Header/FeatureWrapper";

```

```

import FeatureTitle from "../components/Header/FeatureTitle";
import FeatureSubTitle from "../components/Header/FeatureSubTitle";
import PlayButton from "../components/Header/PlayButton";
import HeaderLink from "../components/Header/HeaderLink";
import AllSlidesWrapper from "../components/Movies/AllSlidesWrapper";
import SlideWrapper from "../components/Movies/SlideWrapper";
import SlideTitle from "../components/Movies/SlideTitle";
import AllCardsWrapper from "../components/Movies/AllCardsWrapper";
import CardWrapper from "../components/Movies/CardWrapper";
import CardImage from "../components/Movies/CardImage";
import CardTitle from "../components/Movies/CardTitle";
import CardDescription from "../components/Movies/CardDescription";
import CardFeatureWrapper from "../components/Movies/CardFeatureWrapper";
import CardFeatureClose from "../components/Movies/CardFeatureClose";
import PlayerVideo from "../components/Movies/PlayerVideo";
import PlayerOverlay from "../components/Movies/PlayerOverlay";
import FooterCompound from "../compounds/FooterCompound";

```

```

function BrowsePage() {
  let { series } = useContent("series");
  series = [
    {
      title: "Documentaries",
      data: series.filter((item) => item.genre === "documentaries"),
    },
    {
      title: "Comedies",
      data: series.filter((item) => item.genre === "comedies"),
    },
    {
      title: "Children",
      data: series.filter((item) => item.genre === "children"),
    },
    { title: "Crime", data: series.filter((item) => item.genre === "crime") },
    {
      title: "Feel-Good",
      data: series.filter((item) => item.genre === "feel-good"),
    },
  ];

  let { films } = useContent("films");
  films = [
    { title: "Drama", data: films.filter((item) => item.genre === "drama") },
    {
      title: "Thriller",
      data: films.filter((item) => item.genre === "thriller"),
    },
    {
      title: "Children",
      data: films.filter((item) => item.genre === "children"),
    },
  ];
}

```

```

    title: "Suspense",
    data: films.filter((item) => item.genre === "suspense"),
  },
  {
    title: "Romance",
    data: films.filter((item) => item.genre === "romance"),
  },
];

```

```

const [category, setCategory] = useState("films");
const currentCategory = category === "films" ? films : series;
const [showCardFeature, setShowCardFeature] = useState(false);
const [activeItem, setActiveItem] = useState(false);
const [showPlayer, setShowPlayer] = useState(false);

```

```

return (
  <>
    <HeaderWrapper className="header-wrapper-browse">
      <NavBar className="navbar-browse">
        <Logo />
        <HeaderLink
          className={
            category === "films" ? "header-link-bold" : "header-link"
          }
          onClick={() => setCategory("films")}
        >
          Films
        </HeaderLink>
        <HeaderLink
          className={
            category === "series" ? "header-link-bold" : "header-link"
          }
          onClick={() => setCategory("series")}
        >
          Series
        </HeaderLink>
      </NavBar>
      <FeatureWrapper>
        <FeatureTitle className="feature-title-browse">
          Watch Joker Now
        </FeatureTitle>
        <FeatureSubTitle className="feature-subtitle-browse">
          Forever alone in a crowd, failed comedian Arthur Fleck seeks
          connection as he walks the streets of Gotham City. Arthur wears two
          masks, the one he paints for his day job as a clown, and the guise
          he projects in a futile attempt to feel like he is part of the world
          around him.
        </FeatureSubTitle>
        <PlayButton onClick={() => setShowPlayer(true)}>Play</PlayButton>
        {showPlayer ? (
          <PlayerOverlay onClick={() => setShowPlayer(false)}>
            <PlayerVideo src="/videos/video.mp4" type="video/mp4" />

```

```

        </PlayerOverlay>
      ) : null}
    </FeatureWrapper>
  </HeaderWrapper>

  <AllSlidesWrapper>
    {currentCategory.map((slideItem) => (
      <SlideWrapper key={` ${category}-${slideItem.title.toLowerCase()}`}>
        <SlideTitle>{slideItem.title}</SlideTitle>
        <AllCardsWrapper>
          {slideItem.data.map((cardItem) => (
            <CardWrapper key={cardItem.docId}>
              <CardImage
                onClick={() => {
                  setShowCardFeature(true);
                  setActiveItem(cardItem);
                }}
                src={` ../images/${category}/${cardItem.genre}/${cardItem.slug}/small.jpg`}
              />
            </CardWrapper>
          ))}
        </AllCardsWrapper>
        {showCardFeature &&
          slideItem.title.toLowerCase() === activeItem.genre ? (
            <CardFeatureWrapper
              style={{
                backgroundImage:
                  `url(../images/${category}/${activeItem.genre}/${activeItem.slug}/large.jpg)`,
              }}
            >
              <CardTitle>{activeItem.title}</CardTitle>
              <CardDescription>{activeItem.description}</CardDescription>
              <CardFeatureClose onClick={() => setShowCardFeature(false)} />
              <PlayButton onClick={() => setShowPlayer(true)}>
                Play
              </PlayButton>
              {showPlayer ? (
                <PlayerOverlay onClick={() => setShowPlayer(false)}>
                  <PlayerVideo src="../videos/video.mp4" type="video/mp4" />
                </PlayerOverlay>
              ) : null}
            </CardFeatureWrapper>
          ) : null}
        </SlideWrapper>
      ))}
    </AllSlidesWrapper>
    <FooterCompound />
  </>
);
}

```

```
export default BrowsePage;
```