ACADEMIC TASK

Degree: Bachelor of Science

Specialization

Computer Science and Engineering

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

From-
SUBMITTED BY

Name
Ayush Verma (11908044)

Areek Rizvi (11906802)

Irfan Ahmad Khan (11807243)

Nemali Venkata Subba Reddy (11801090)

Submitted to:

Sudhanshu Prakash Tiwari

Acknowledgement

We would like to extend my special thanks to my teacher Sudhanshu Prakash Tiwari, who gave us the golden opportunity to do this great work on this topic, which also helped us with a lot of research and came to know many new things.

Secondly, we would also like to thank each and every team mates and friends who have helped us so much in completing the work on time.

Topic: StarVee▭

Introduction

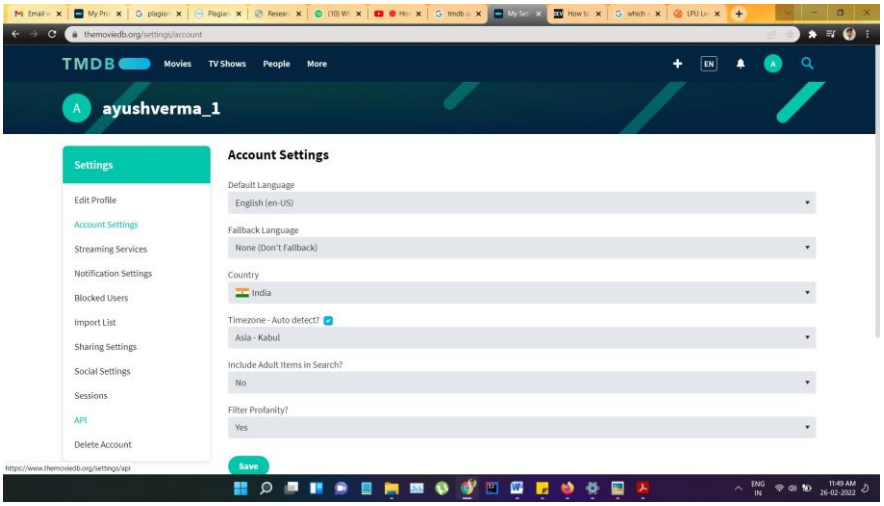
Traditionally, the use of movies and other audio and video content has been relatively stable on social media such as theater and television. As technology advanced, it became more accessible at home and whenever necessary with the introduction of VHS, DVDs, Blu-rays and disk rental services. In addition, cable television delivers content via Co-axial cables and fiber optic cables. Something better, the service emerged as a Direct-to-home (DTH) technology with satellite and vessel communications deliver the highest quality streaming and content required directly to the consumer. Recently advances in technology have made movie or TV watching much easier online Streaming or Video on Demand (VoD) services VoD refers to the streaming of video content over Internet, through applications commonly referred to as Over-The-Top (OTT). Viewers can access the video content via OTT applications on any connected device such as smartphone, smart TV, tablet, desktop computer, laptops, etc. Unlike traditional media, broadcast media tells a variety of non-existent stories restricted censors, box office or demographics. It offers a highly advanced viewing experience audio and visual quality, as long as consumers have a stable internet.

OTT bypasses cable, broadcast, satellite television and other platforms that generally act as a controller or distributor and enables disintermediation. The sole gateway to consumers', in the age of traditional media, was through film distributors, theatre runners, television networks or Multiple System Operators (MSOs). With OTT, the content creators can interact with their audience directly through a web page or Smartphone app. This offers the comfort of viewing movie and other entertainment at one's convenient time and place. Once considered a luxury, an increasingly growing number of Indians are shifting towards

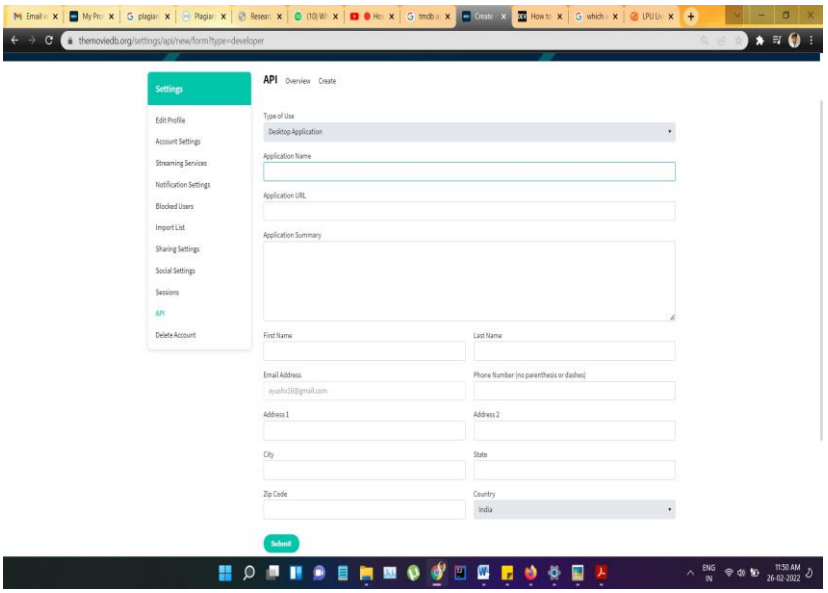
cord-cutting or online streaming. While the figures show that the VoD industry is still at its nascent stage, the entry of almost 40 VoD companies in a span of just three years indicates the massive potential of the industry. Out of five Smartphone owners in India, at least four people watch content in at least one OTT app. The OTT apps have become the most downloaded app category ahead of social networking apps like Facebook, messaging apps like WhatsApp, and e-commerce apps like Amazon and Flipkart. The streaming market will collectively account for 46% of the overall growth in the Indian entertainment and media industry from 2017 to 2022

DISCRIPTION OF PROJECT

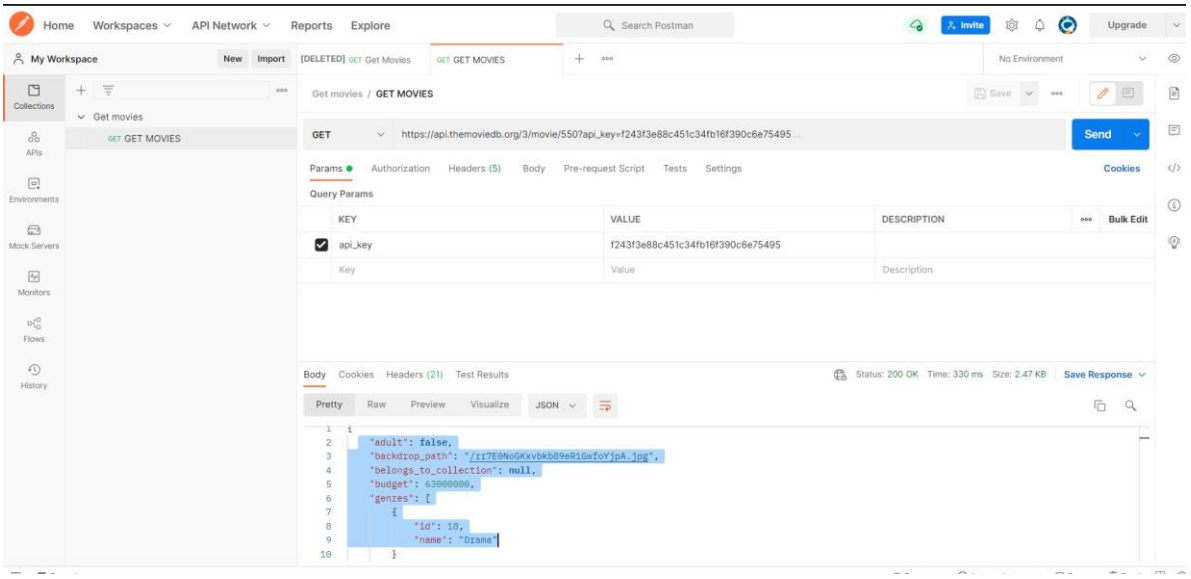
STEP 1:
First of all we are accessing a service which is actually going to access a service or in other words we are going to make a request a service from TMBD API and they are going to give us back a bunch of information about different films and anime
and the actual difference between This service and other is that is that they actually give us a bunch of different GENRES so we have like
Netflix originals trending now top rated action movie comedy etc.
And then we pull that information and show it in front end
To do so first we need to create an account on TMBD platform www.themoviedb.org Then we are fetching the API key by selecting developer option



After creating an account go to settings and select API -> click request API-> Then select developer option -> after that accept all the terms and condition.

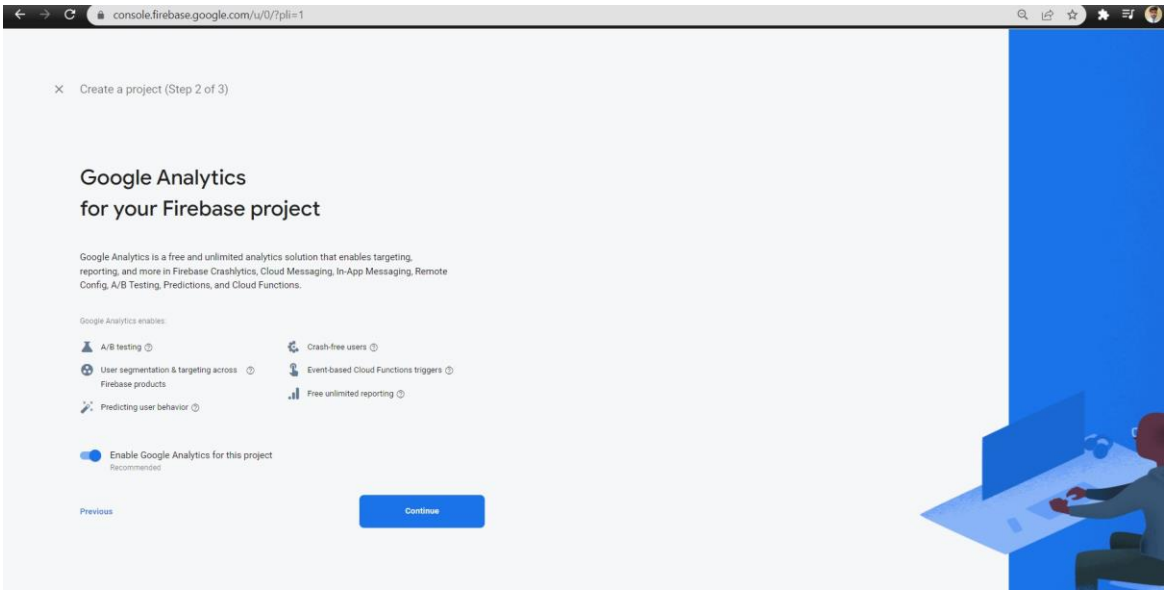


Then after completing the above details we get API key i.e f243f3e88c451c34fb16f390c6e75495 With the help of this api key we can access all the features.
After that
we used postman: post man is allows us to send request to any server or API



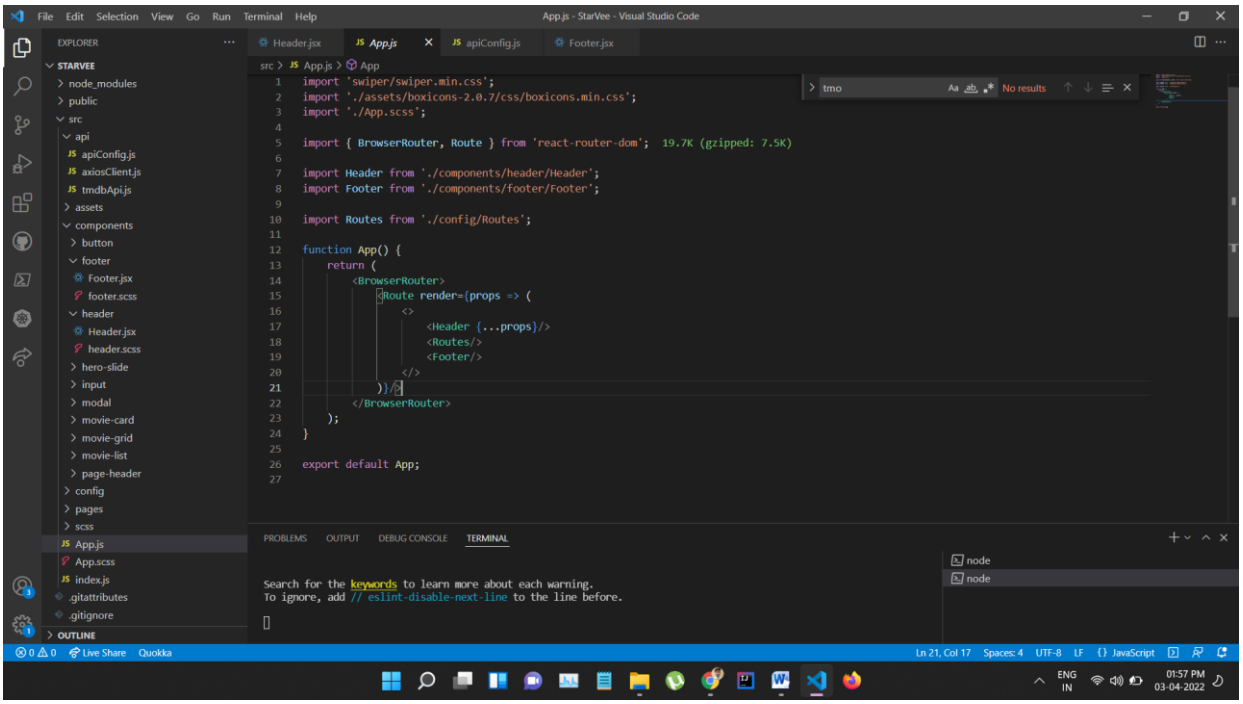
Step2: Creating an react application
First of all the application we are using to create the app is VS Code i.e Visual Studio Code

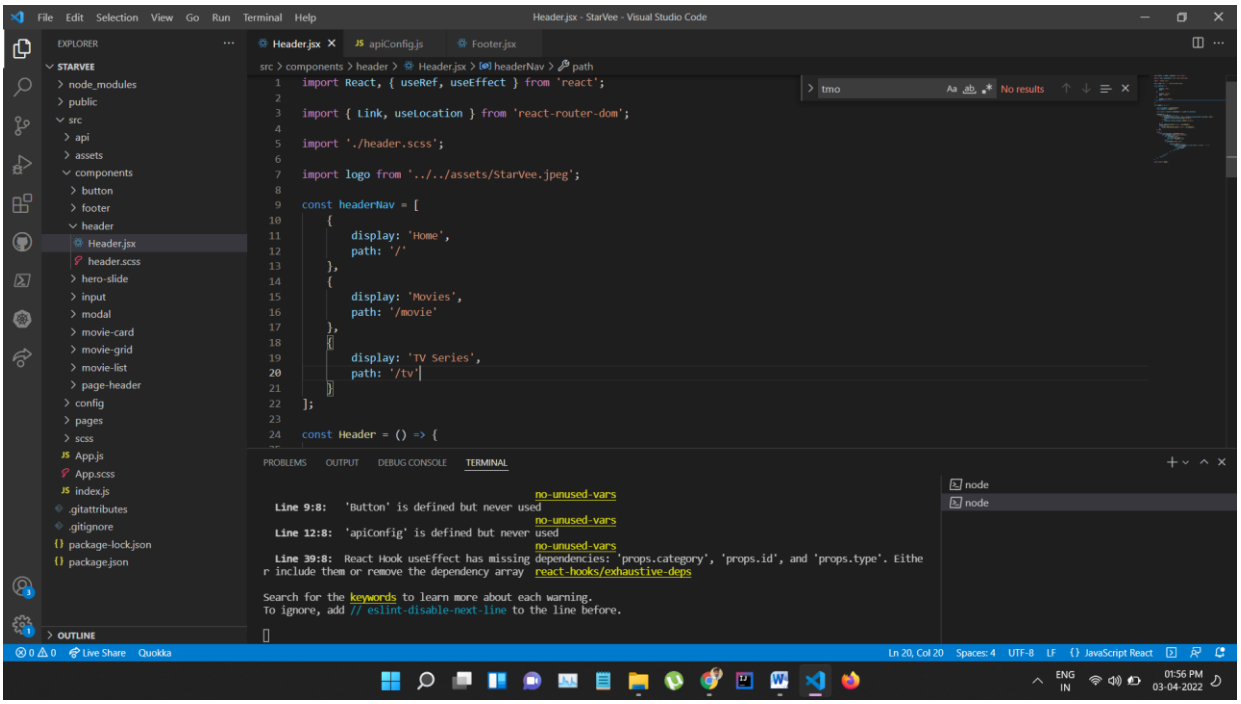
Step 3 Setting Up Firebase hosting

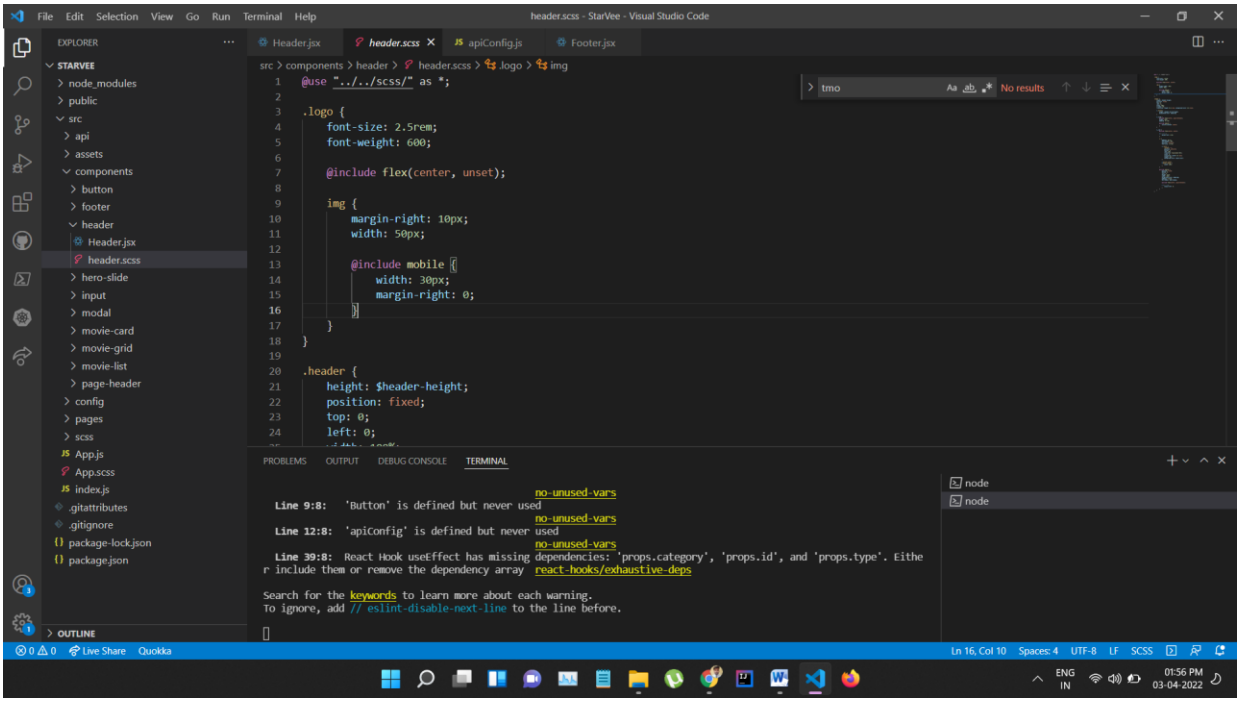


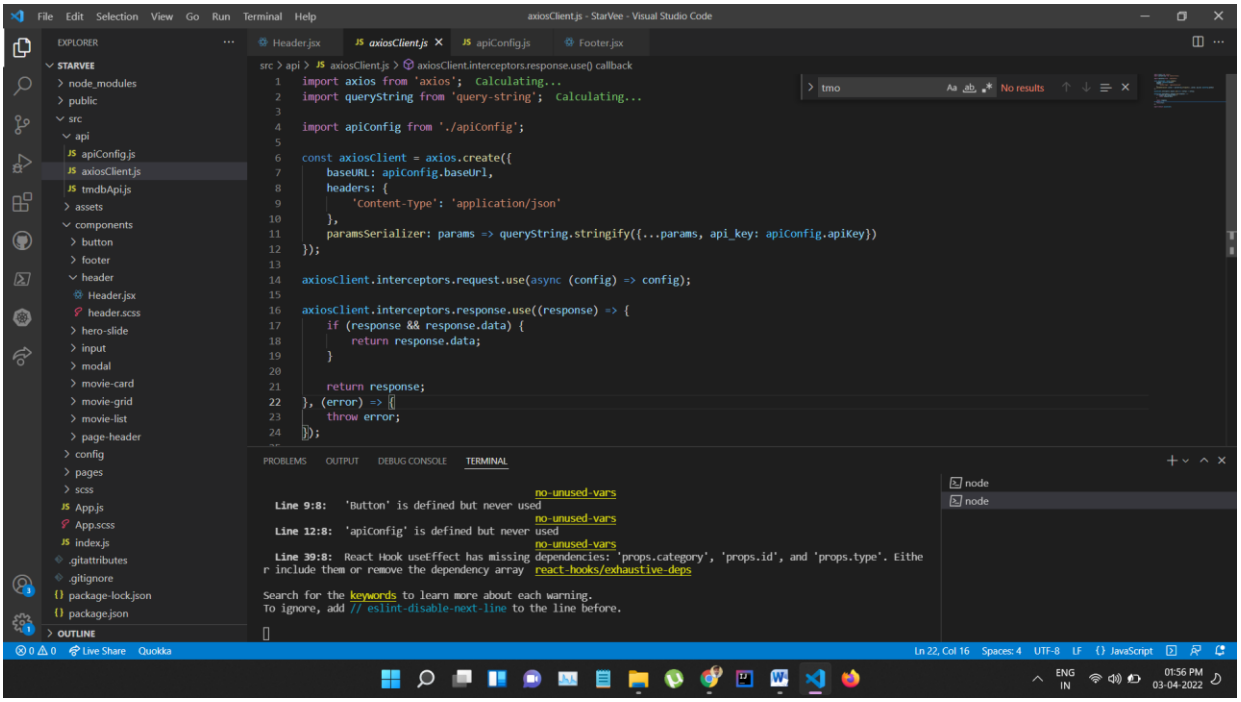
we are setting firebase for the deployment of the application

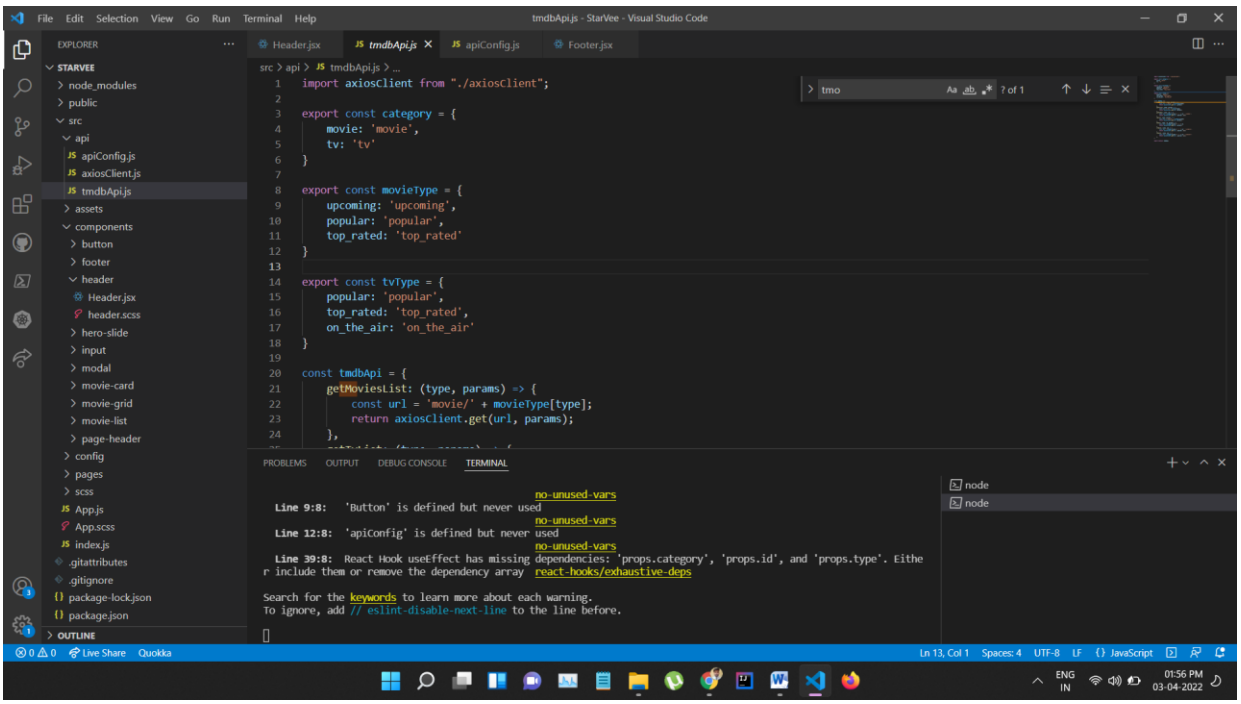
Step 4 Now we are going to get all the movies Build a lo

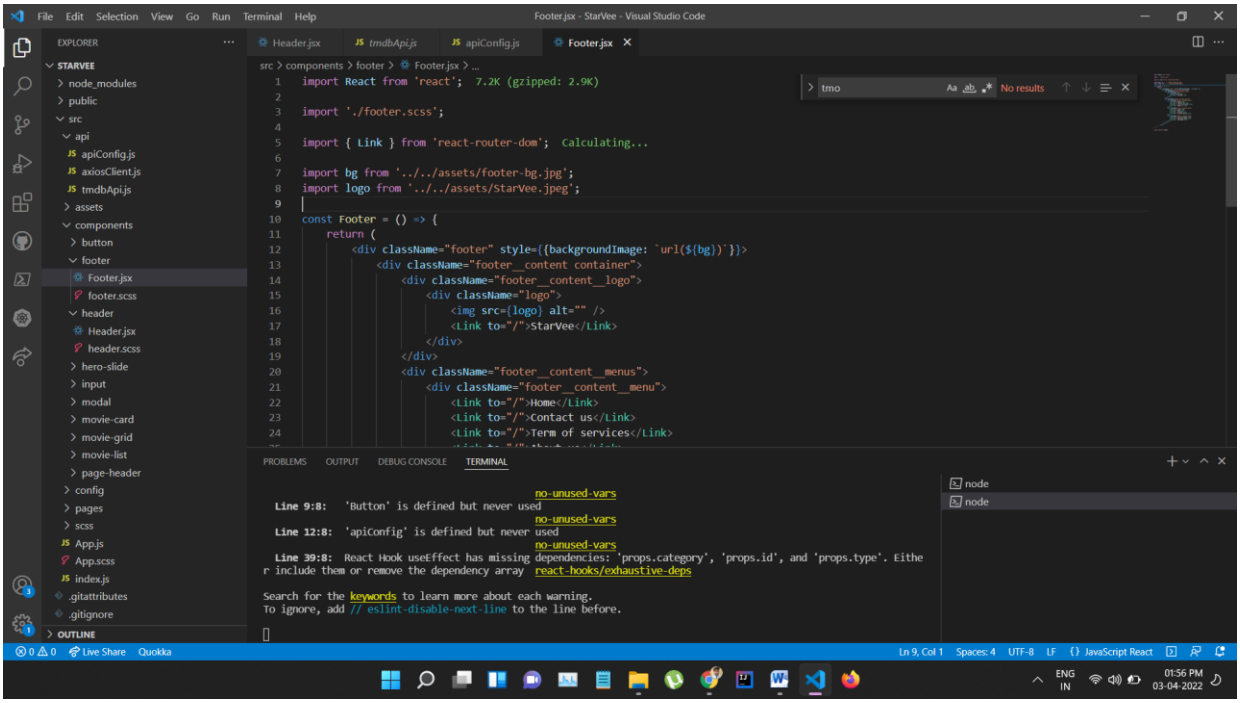



File Edit Selection View Go Run Terminal Help
Header.jsx - StarVee - Visual Studio Code
EXPLORER
STARVEE
node_modules
public
src
api
assets
components
button
footer
header
Header.jsx
header.scss
hero-slide
input
modal
movie-card
movie-grid
movie-list
page-header
config
pages
scss
App.js
App.scss
index.js
.gitattributes
.gitignore
package-lock.json
package.json
OUTLINE
Header.jsx
apiConfig.js
Footer.jsx
src > components > header > Header.jsx > headerNav > path
import React, { useRef, useEffect } from 'react';
import { Link, useLocation } from 'react-router-dom';
import './header.scss';
import logo from '../../assets/StarVee.jpeg';
const headerNav = [
{
display: 'Home',
path: '/'
},
{
display: 'Movies',
path: '/movie'
},
{
display: 'TV Series',
path: '/tv'
}
];
const Header = () => {
tmo
No results
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
no-unused-vars
Line 9:8: 'Button' is defined but never used
no-unused-vars
Line 12:8: 'apiConfig' is defined but never used
no-unused-vars
Line 39:8: React Hook useEffect has missing dependencies: 'props.category', 'props.id', and 'props.type'. Either include them or remove the dependency array react-hooks/exhaustive-deps
Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.
node
node
Ln 20, Col 20
Spaces: 4
UTF-8
LF
{} JavaScript React
Live Share
Quokka
ENG
IN
01:56 PM
03-04-2022


File Edit Selection View Go Run Terminal Help
header.scss - StarVee - Visual Studio Code
EXPLORER
STARVEE
node_modules
public
src
api
assets
components
button
footer
header
Header.jsx
header.scss
hero-slide
input
modal
movie-card
movie-grid
movie-list
page-header
config
pages
scss
App.js
App.scss
index.js
.gitattributes
.gitignore
package-lock.json
package.json
OUTLINE
Header.jsx
header.scss
apiConfig.js
Footer.jsx
src > components > header > header.scss > .logo > img
@use "../../scss/" as *;
.logo {
font-size: 2.5rem;
font-weight: 600;
@include flex(center, unset);
img {
margin-right: 10px;
width: 50px;
@include mobile {
width: 30px;
margin-right: 0;
}
}
}
.header {
height: $header-height;
position: fixed;
top: 0;
left: 0;
tmo
No results
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
no-unused-vars
Line 9:8: 'Button' is defined but never used
no-unused-vars
Line 12:8: 'apiConfig' is defined but never used
no-unused-vars
Line 39:8: React Hook useEffect has missing dependencies: 'props.category', 'props.id', and 'props.type'. Either include them or remove the dependency array react-hooks/exhaustive-deps
Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.
node
node
Ln 16, Col 10
Spaces: 4
UTF-8
LF
SCSS
Live Share
Quokka
ENG
IN
01:56 PM
03-04-2022

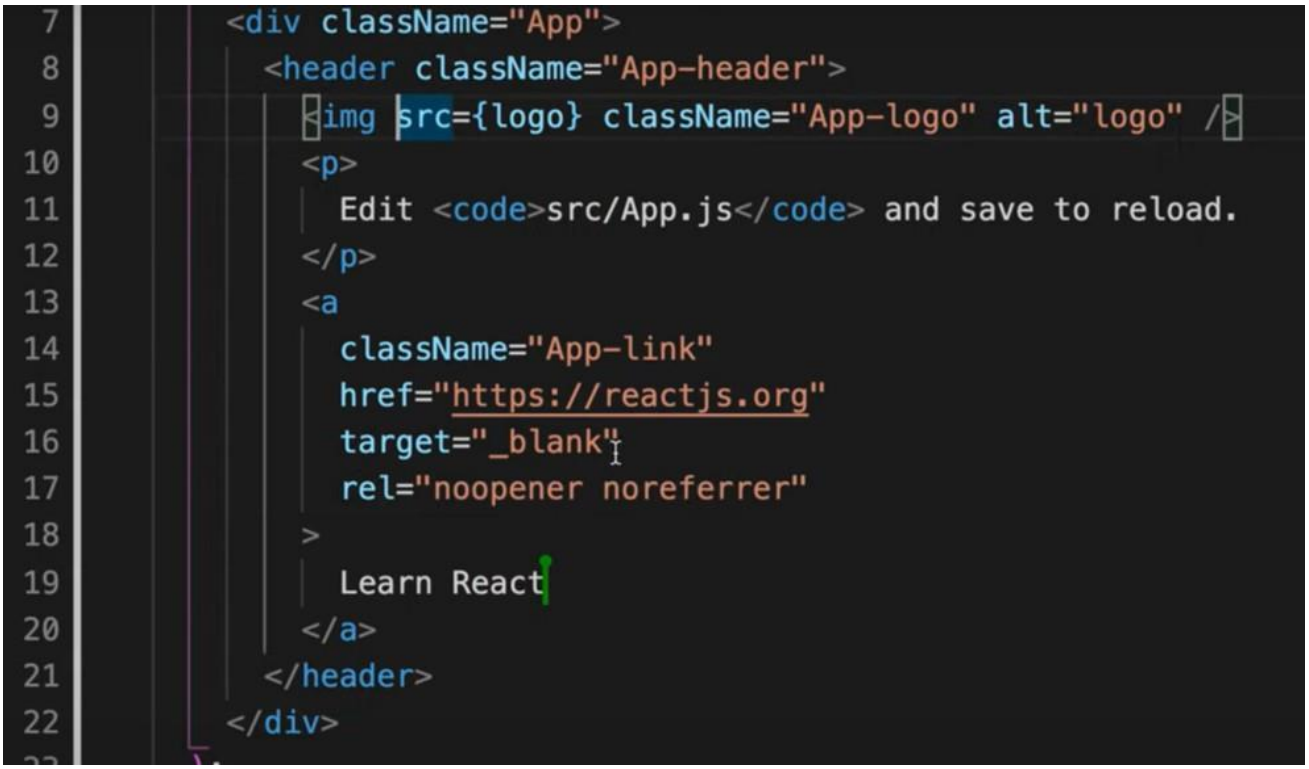

File Edit Selection View Go Run Terminal Help
axiosClient.js - StarVee - Visual Studio Code
EXPLORER
STARVEE
node_modules
public
src
api
apiConfig.js
axiosClient.js
tmdbApi.js
assets
components
button
footer
header
Header.jsx
header.scss
hero-slide
input
modal
movie-card
movie-grid
movie-list
page-header
config
pages
scss
App.js
App.scss
index.js
.gitattributes
.gitignore
package-lock.json
package.json
OUTLINE
Header.jsx
axiosClient.js
apiConfig.js
Footer.jsx
src > api > axiosClient.js > axiosClient.interceptors.response.use() callback
import axios from 'axios'; Calculating...
import queryString from 'query-string'; Calculating...
import apiConfig from './apiConfig';
const axiosClient = axios.create({
baseURL: apiConfig.baseUrl,
headers: {
'Content-Type': 'application/json'
},
paramsSerializer: params => queryString.stringify({...params, api_key: apiConfig.apiKey})
});
axiosClient.interceptors.request.use(async (config) => config);
axiosClient.interceptors.response.use((response) => {
if (response && response.data) {
return response.data;
}
return response;
}, (error) => {
throw error;
});
tmo
No results
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
node
node
no-unused-vars
Line 9:8: 'Button' is defined but never used
no-unused-vars
Line 12:8: 'apiConfig' is defined but never used
no-unused-vars
Line 39:8: React Hook useEffect has missing dependencies: 'props.category', 'props.id', and 'props.type'. Either include them or remove the dependency array react-hooks/exhaustive-deps
Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.
Ln 22, Col 16
Spaces: 4
UTF-8
LF
JavaScript
Live Share
Quokka
ENG
IN
01:56 PM
03-04-2022


File Edit Selection View Go Run Terminal Help
tmdbApi.js - StarVee - Visual Studio Code
EXPLORER
STARVEE
node_modules
public
src
api
apiConfig.js
axiosClient.js
tmdbApi.js
assets
components
button
footer
header
Header.jsx
header.scss
hero-slide
input
modal
movie-card
movie-grid
movie-list
page-header
config
pages
scss
App.js
App.scss
index.js
.gitattributes
.gitignore
package-lock.json
package.json
OUTLINE
Header.jsx
tmdbApi.js
apiConfig.js
Footer.jsx
src > api > tmdbApi.js > ...
import axiosClient from "./axiosClient";
export const category = {
movie: 'movie',
tv: 'tv'
}
export const movieType = {
upcoming: 'upcoming',
popular: 'popular',
top_rated: 'top_rated'
}
export const tvType = {
popular: 'popular',
top_rated: 'top_rated',
on_the_air: 'on_the_air'
}
const tmdbApi = {
getMoviesList: (type, params) => {
const url = 'movie/' + movieType[type];
return axiosClient.get(url, params);
},
tmo
2 of 1
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
node
node
no-unused-vars
Line 9:8: 'Button' is defined but never used
no-unused-vars
Line 12:8: 'apiConfig' is defined but never used
no-unused-vars
Line 39:8: React Hook useEffect has missing dependencies: 'props.category', 'props.id', and 'props.type'. Either include them or remove the dependency array react-hooks/exhaustive-deps
Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.
Ln 13, Col 1
Spaces: 4
UTF-8
LF
JavaScript
Live Share
Quokka
ENG
IN
01:56 PM
03-04-2022


File Edit Selection View Go Run Terminal Help
Footer.jsx - StarVee - Visual Studio Code
EXPLORER
STARVEE
node_modules
public
src
api
apiConfig.js
axiosClient.js
tmdbApi.js
assets
components
button
footer
Footer.jsx
footer.scss
header
Header.jsx
header.scss
hero-slide
input
modal
movie-card
movie-grid
movie-list
page-header
config
pages
scss
App.js
App.scss
index.js
.gitattributes
.gitignore
OUTLINE
Header.jsx
tmdbApi.js
apiConfig.js
Footer.jsx
src > components > footer > Footer.jsx > ...
import React from 'react'; 7.2K (gzipped: 2.9K)
import './footer.scss';
import { Link } from 'react-router-dom'; Calculating...
import bg from '../../assets/footer-bg.jpg';
import logo from '../../assets/StarVee.jpeg';
const Footer = () => {
return (
<div className="footer" style={{backgroundImage: `url(${bg})`}}>

<Link to="/">StarVee</Link>

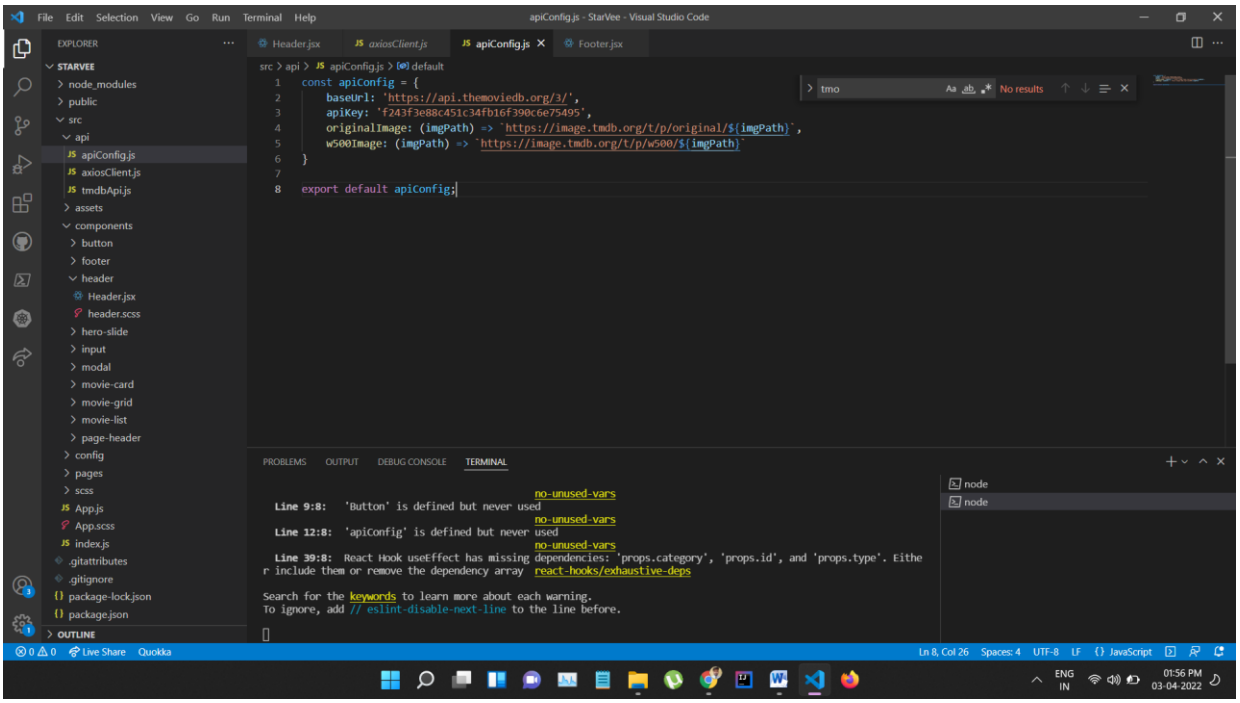
<Link to="/">Home</Link>
<Link to="/">Contact us</Link>
<Link to="/">Term of services</Link>
tmo
No results
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
node
node
no-unused-vars
Line 9:8: 'Button' is defined but never used
no-unused-vars
Line 12:8: 'apiConfig' is defined but never used
no-unused-vars
Line 39:8: React Hook useEffect has missing dependencies: 'props.category', 'props.id', and 'props.type'. Either include them or remove the dependency array react-hooks/exhaustive-deps
Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.
Ln 9, Col 1 Spaces: 4 UTF-8 LF {} JavaScript React
Live Share Quokka
ENG IN
01:56 PM 03-04-2022

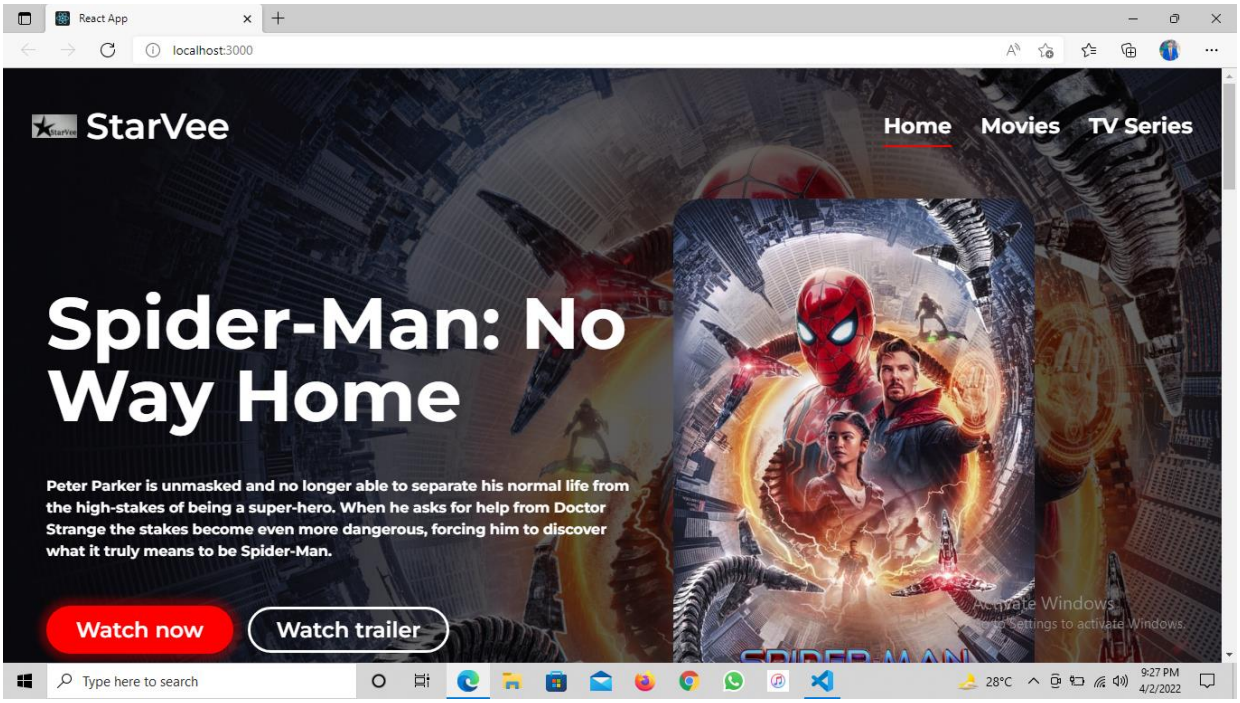


Step 5. Let's build a front end application and it will be a part of the website users can see and interact wth such as the graphical user interface(GUI) and the command line including the design navigation menu and buttons etc.

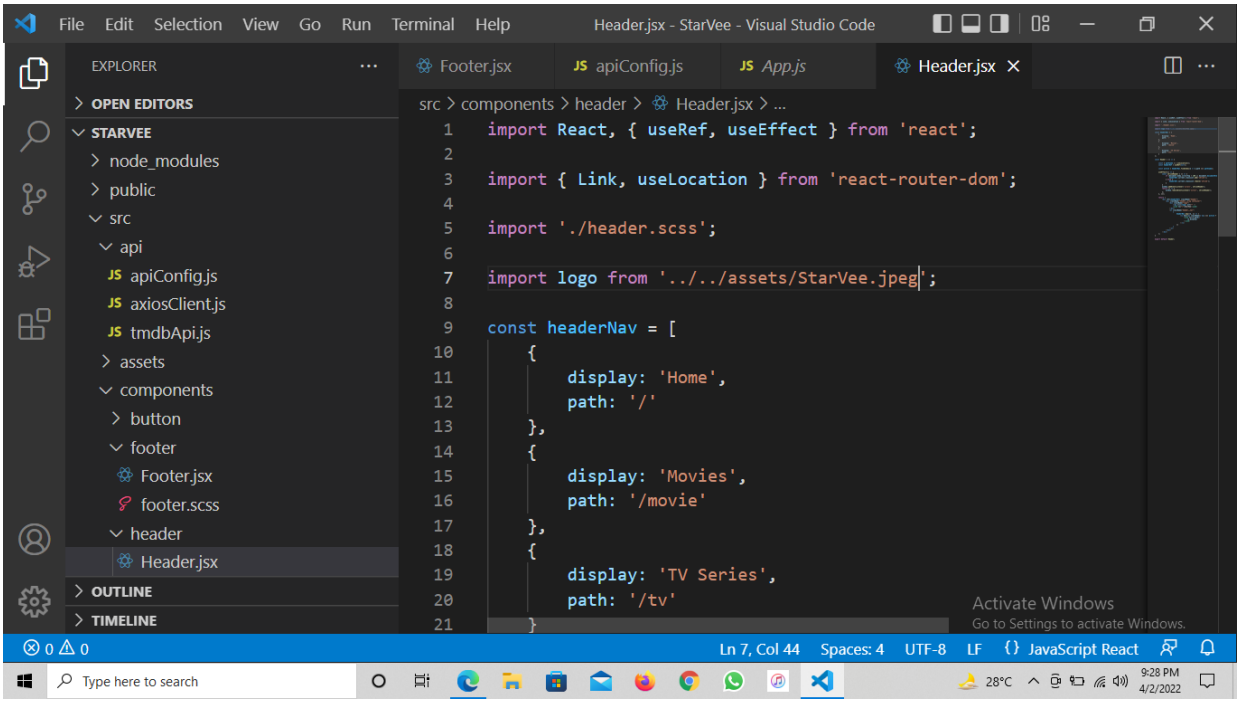
Backend on the contrary is the part of the website users cannot see and interact with.

The logo of the project

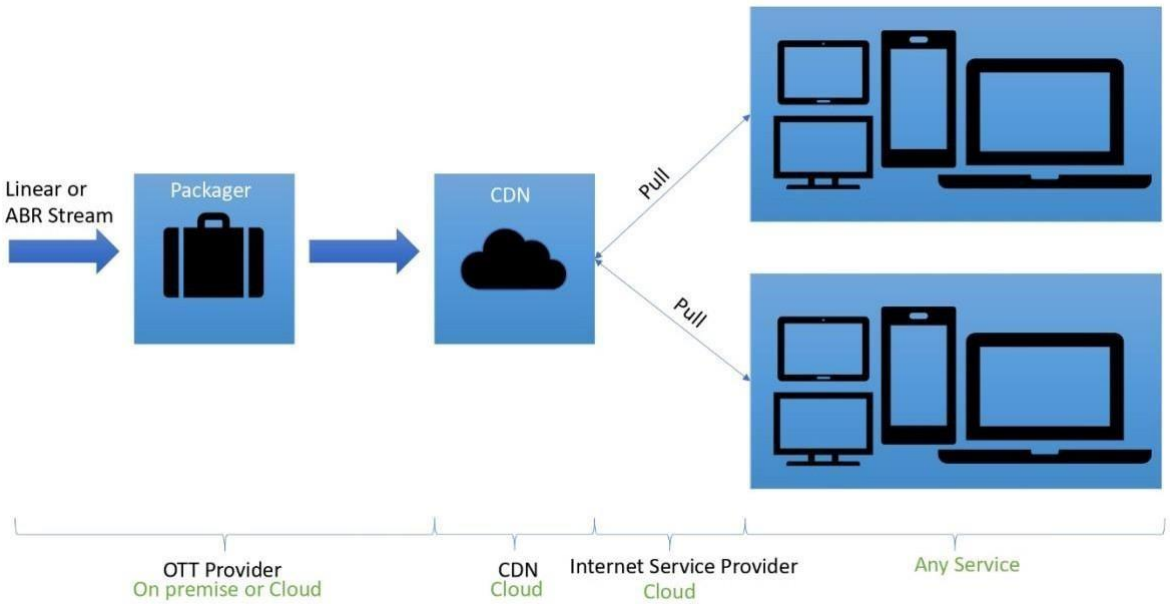




Step 6. Get all the movies and build the rows



Step 6. Now we are going to build two components there is one which is the navigate bar should be on the top and there is another one which were going to call a banner.



I. Literature Review/Survey

- **The global development of OTT**

The global development of OTT The evolution from digital viewing into online viewing had everything to try and do with the breakthrough of computer technologies. OTT bypasses cable, broadcast, satellite TV, and other platforms that generally act as a controller or distributor and enables disintermediation. The evolution from digital viewing to online viewing had everything to try and do with the breakthrough of computer technologies. during this phase, the net was used mainly to drive viewers to observe the TV programs the standard way. Eventually, the web became another major battlefield for full streaming of TV and online programs.

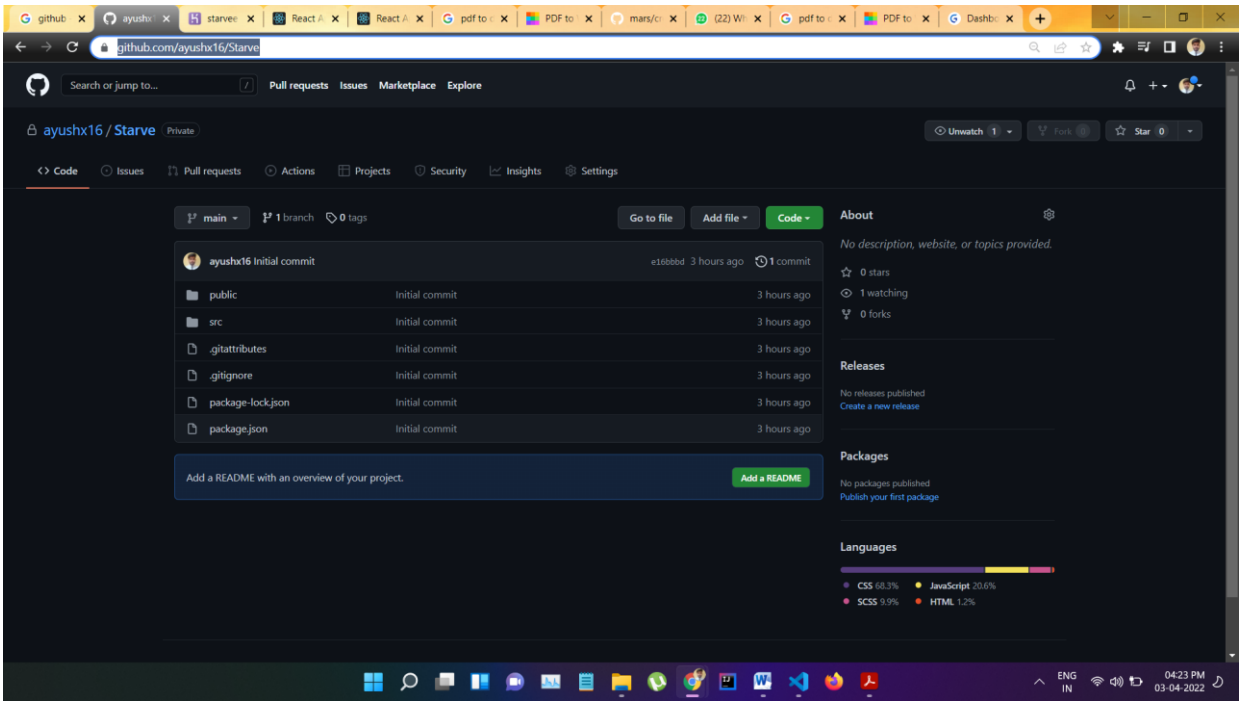
- **Viewing behaviour on mobile devices**

The most recent innovation that has tremendously affected the event of OTT is mobile viewing. The emergence of mobile devices like smartphones, tablets, and Ultrabooks has inexorably challenged the boundary between new media and traditional television, reshaping the ways within which programs are both produced, and viewers watch them. This ground-breaking age has prompted global academics to seek out those factors whereby the audience is inclined to decide on one medium over another, additionally because of the relationships among media types.

GitHub and Deploying the Project on Heroku

To upload a code on GitHub we first created a GitHub repository and with the help of pull request we uploaded the complete project on GitHub as you can see from the given link you can navigate to it

https://github.com/ayushx16/Starve



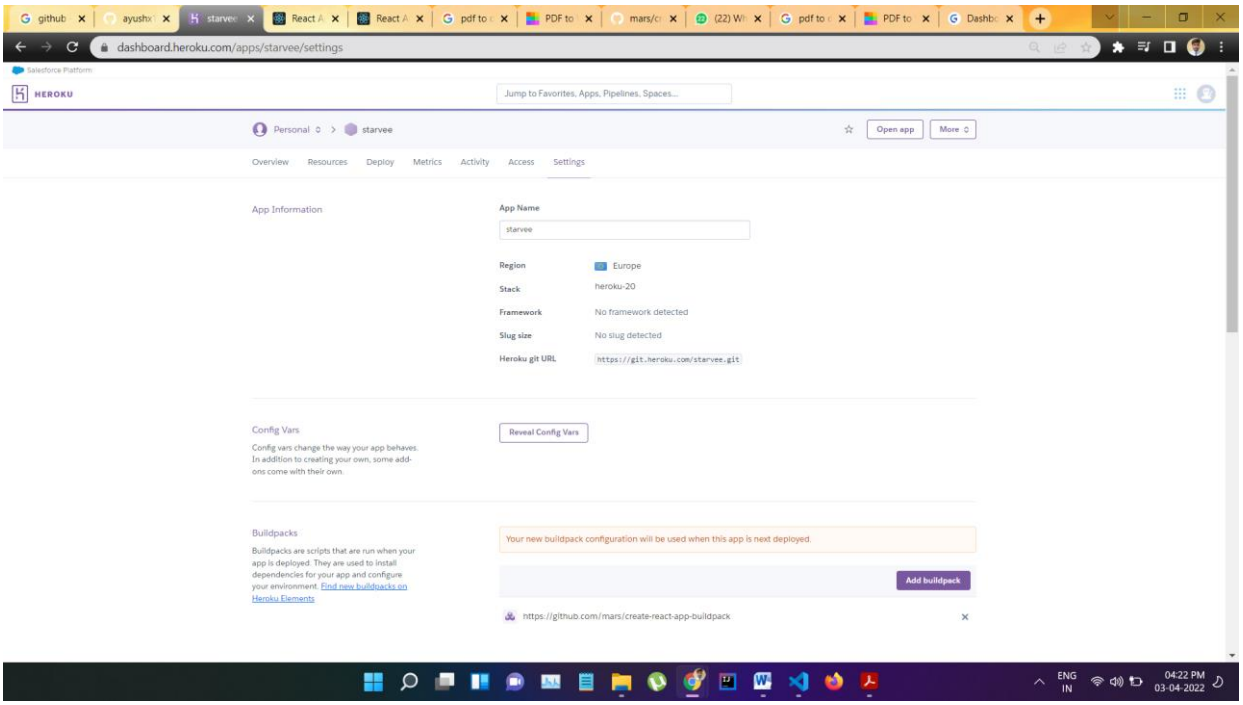
Now for Deployment on Heroku

We first created Heroku account then

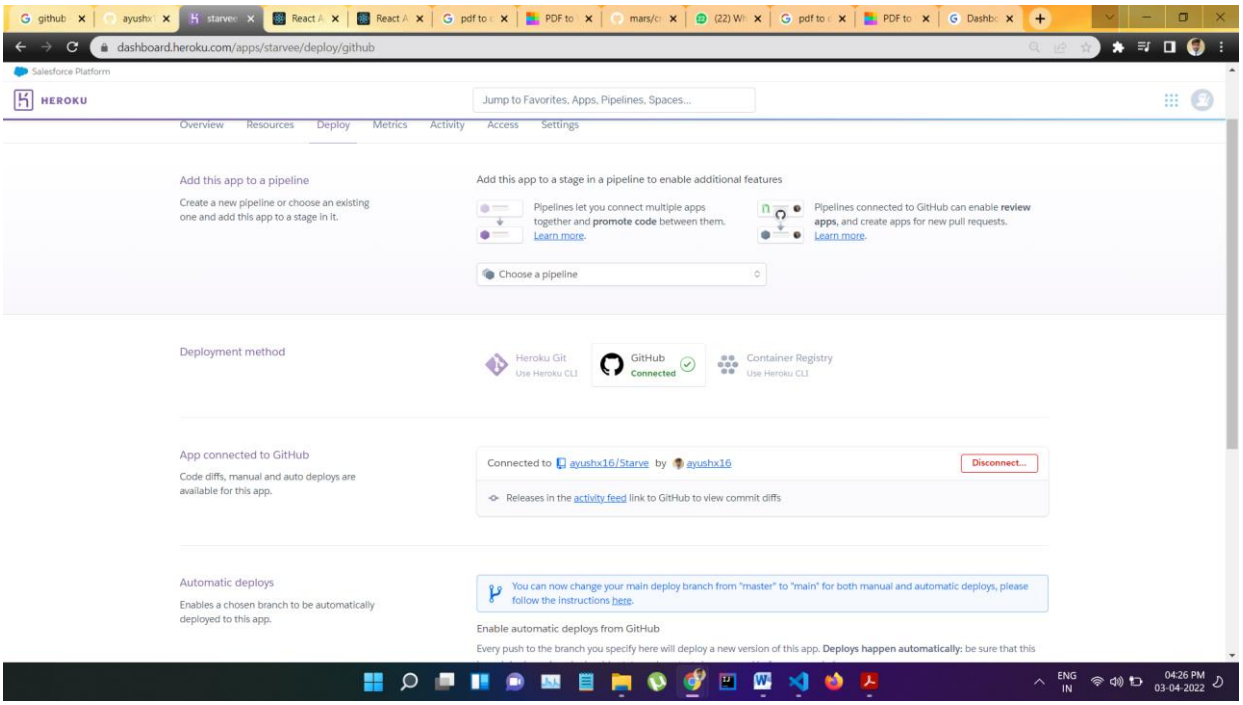
After that we added the build pack from this website as you can see https://github.com/mars/create-react-app-buildpack

Buildpacks

Buildpacks are scripts that are run when your app is deployed. They are used to install dependencies for your app and configure your environmen



After that go to Deploy then go to connect to GitHub and search for the repository and the hit connect



After that choose a branch to deploy and hit deploy branch it will automatically deploy.

And after successfully getting deployed your code is now ready to go as you can see https://starvee.herokuapp.com/
from the given link we can navigate to it.

I. Research Methodology

Video hosting platforms include a large style of tools for hosting, storing, and managing video content at many alternative price points. Some hosting features that are valuable for creating an OTT platform include HTML5 video players, white-label capabilities, brand customization, and API access. some of the most innovative OTT platform technology is available via Dacast, Vimeo OTT, and Brightcove.

- **Technology in OTT**

For starters, all content across OTT platforms is stored within the cloud. The cloud is formed of many physical data centres spread across the planet, rife with cables connecting heavy-duty equipment. These are founded by major tech giants like Google, Amazon and Microsoft and are widely called Google Cloud Platform, Amazon Webservices and AZURE, respectively.

- **Storage for OTT**

OTT players rent space within the cloud for storing their files. On AWS, as an example, the fundamental monthly fee for storing 1,000 GB of content per hour is $125. the speed changes looking on several factors, including the region of cloud storage.

- **Encryption of Data**

Every time a streaming platform creates or acquires content, it uses a third-party encoding software to compress the first file which will run in terabytes (TBs) into a streaming-friendly lower file size.

- **Content Delivery Network**

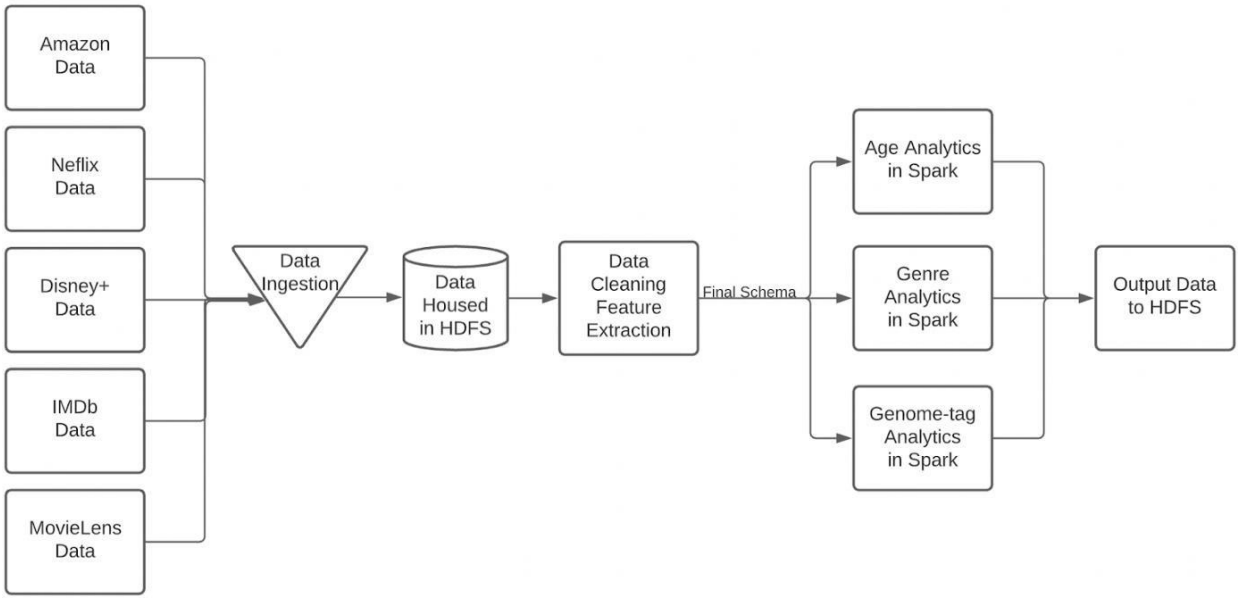
Content delivery network (CDN) is precisely what it sounds like: a network of servers that delivers content. Most online video platforms partner with professional CDNs, so this isn't typically the most concern for broadcasters. You'll want to form sure that

your chosen video hosting platform uses a CDN that has servers stationed round the globe. this can help maintain the standard of your stream for viewers in every corner of the world. it'll also help avoid lagging and buffering. However, if you're self- hosting, you may have to choose a CDN of your own. thanks to complicated CDN pricing structures, this may get expensive and confusing, so choosing a video hosting platform with a built-in CDN is that the thanks to go.

- **Video Licensing**
  Video licensing is strictly what it looks like — it's the legal process of paying to license video content for the other project. Copyright laws (and the overall concept of intellectual property) exist to shield creatives. So, licensing videos works rather like finding images, fonts, music, or sound effects. If you're using something that somebody else created, you'll must license it. Video licensing is simply the method of filling out and paying for a license, usually from a stock video website or maybe a content creator's online marketplace.

- **DRM**
  DRM stands for "Digital Rights Management". It's a technology that in effect pairs a digital eBook to the device that it's downloaded to. This prevents the file from being shared with someone who didn't purchase their own copy, because the DRM on the shared file won't match with the device the opposite person is using. The result's a file that's useful only to the first purchaser when it's run on a licensed device.

So now the complete operations that are carried out here are



References

- Albarran, A. B. (2013). Media management and economics research in a transmedia environment. New York: Routledge. Ashrafi, D. (2014).
- Forcing the connection: The antitrust concerns with broadband data caps and data discrimination in the wake of the new Internet television age. Santa Clara Law Review, 54, 465. Banerjee, A., Alleman, J., & Rappaport, P. (2013).
- Video-viewing behaviour in the era of connected devices. Communications & Strategies, (92), 19. Bury, R. (2005). Cyberspaces of their own: Female fandoms online (Vol. 25). New York: Peter Lang. Bury, R., & Li, J. (2013).
- Is it live or is it timeshifted, streamed or downloaded? Watching television in the era of multiple screens. New Media & Society, 6(0), 119. Dimmick, J., & Rothenbuhler, E. (1984). The theory of the niche: Quantifying competition among media industries.