Single Responsibility Principal

Custom hooks

If we are creating custom hooks points to be remember

> it should be in separate folder

> name should start with use like useState useEffect. (react will consider as hook if you start with use )

>

import { useEffect, useState } from "react";

import SimmerUIResList from './SimmerUIResList';

import { MENU\_URL } from "../utils/constents";

import { useParams } from "react-router-dom";

import Error from './Error';

import useRestaurantMenu from "../utils/useRestaurantMenu";

const RestaurantMenu = () => {

const {id} = useParams();

const resData = useRestaurantMenu(id);

if (resData == null) return <SimmerUIResList />;

const resInfo = resData?.cards[2]?.card?.card?.info;

const menus = resData?.cards[4]?.groupedCard?.cardGroupMap?.REGULAR?.cards[2]?.card?.card?.itemCards;

return (<>

<h1>

{resInfo.name}

</h1>

<h3>{resInfo.cuisines.join(', ')}</h3>

<h3>{resInfo.costForTwoMessage}</h3>

<hr/>

<div className='res-container'>

{

menus.map((menu, indx)=>{

return(

<div className="card-container" key={indx}>

<div className="card">

<h4>{menu.card.info.name}</h4>

<p>{menu.card.info.description}</p>

<p>Veg: {menu.card.info.isVeg == 1?"Yes":"No"}</p>

<p>Rs. {(menu.card.info.price||menu.card.info.defaultPrice)/100}/-</p>

</div></div>

)

})

}

</div>

</>)

}

export default RestaurantMenu;

Custom hook

import { useEffect, useState } from "react";

import { MENU\_URL } from "./constents";

const useRestaurantMenu = (resId)=>{

const [resInfo, setResInfo] = useState(null);

useEffect(()=>{

fetchMenu();

},[]);

const fetchMenu = async () =>{

const data = await fetch(MENU\_URL+resId);

const json = await data.json();

await setResInfo(json.data);

}

return resInfo;

}

export default useRestaurantMenu;

Part 2

import { useEffect, useState } from "react";

const useOnlineStatus = () =>{

const [onlineStatus, setOnlineStatus] = useState(true);

useEffect(()=>{

window.addEventListener('offline', ()=>{

setOnlineStatus(false);

});

window.addEventListener('online', ()=>{

setOnlineStatus(true);

});

},[]);

return onlineStatus;

}

export default useOnlineStatus;

const onlineStatus = useOnlineStatus()

{onlineStatus === false ? 'N' :'Y' }

Part 5:

How to optimize

Known as

* App chunking
* App splitting
* Dynamic bundling
* Lazy loading
* On demand loading
* Dynamic import

As we know bundller make a single file of js for entire project.

But for large scale projects its not good

We have to split the code

With help of lazy() method of react

If we do like

import Grocery from './components/Grocery'; // this will load in index.js

But when we do like

const Grocery= lazy(()=> import('./components/Grocery') );

It will chunk it (create separate js file for it.

But as we know we are doing lazy loading here

So it takes some time to load the grocery js file

But react is very fast to it may show some error

For that we have method named Suspense from React

Which we can use like

{

path:"/grocery",

element:<Suspense> <Grocery /> </Suspense>,

}

But what to do when data is loading?

Fasllback method

{

path:"/grocery",

element:<Suspense fallback={<h2>Loading...</h2>}> <Grocery /> </Suspense>,

}

We can put any jsx here like loading or anything.