5 . Let's get Hooked!

Clearing mess

Creating separate file to for separate component
Components first letter should be Capital

1. Creating **src** folder
2. Create **Components** folder inside src folder
3. Put Component file code and **export** code from component file **import** it in App.js file
4. Create **utils** folder inside the src
5. Create **constants.js** file inside utils folder
6.create **mockdata.js** file for API data (resList)

We are using CDN URL that is Hardcoded data
So don't keep hardcoded data inside the Components file

For that create separate file like
constant.js
config.js
utils.js
So creating utils folder for that now
For footer links resting fie constant

Creating utils folder and constants file it's not mandatory we are creating just for
Good industry practices

Put CDN url inside constants.js

Create variable in capital letter

```
export const CDN_URL =
"https://media-assets.swiggy.com/swiggy/image/upload/fl_lossy,f_auto,q_auto,w_660/" ;

export const LOGO_URL=
"https://www.logodesign.net/logo/smoking-burger-with-lettuce-3624ld.png?nwm=1&nws=1&industry=restaurant-
```

Copy resList data and paste inside mockData

Export Component

```
Syntax
export default name of the Component;

eg . export default Header;
```

Import Component

```
Syntax
import component name from "path of file";

eg . import Header from "./Components/Header";
```

By default file only export one thing
If we have to export multiple things then we used **Name export**

Before import we need to export Component

Two types of Export / Import

- Default Export/ Import

export default component Name ;
import component from “path”;

- Named Export/ Import

export const component name;
import { component } from “path”;

```
export const CDN_URL =
"https://media-assets.swiggy.com/swiggy/image/upload/fl_loss

export const LOGO_URL=
"https://www.logodesign.net/logo/smoking-burger-with-lettuce
```

```
import { CDN_URL } from "../utils/constants";
```

Creating top rated restaurant filter button

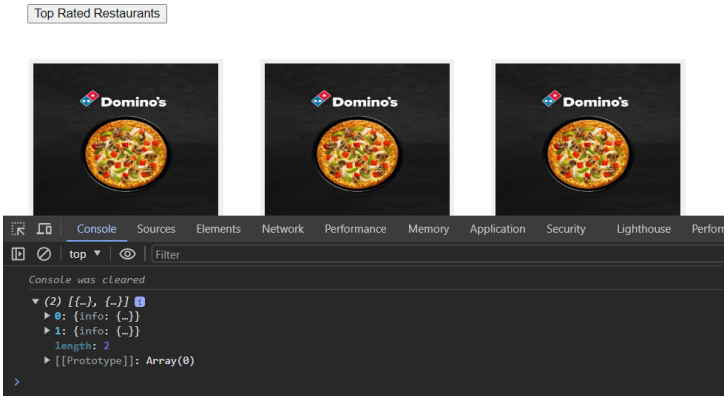
```
<div className="body">
      <div className="filter">
      <button className="filter-btn" onClick={()=>
     { //filter logic here
      ListOfRestaurants = ListOfRestaurants.filter((res)=>
      res.info.avgRating > 4
           );
          console.log(ListOfRestaurants)
      }} >Top Rated Restaurants</button>
      </div>
```

Creating new variable (ListOfRestauramts) Adding 3 restaurants data

```
let ListOfRestaurants=
[


    { ···
      },
      { ···
      },
      { ···
      },
    ]
```

When we click on button (Top rated restaurants) our data is filtered but UI not updated
So for we need to use Hooks



Hooks

(Normal JS Utility function)
- useState ()
- useEffects ()

useState

- import useState as Named import
  Eg . import {useState} from react

Usestate use to create state variable , it maintain the state of our application or maintain the state of component

```
//local state varible
const [Listofres]= useState([]);

// Normal Js variable
let listofres =[];
```

Here we passing empty array
Hooks keeps UI sink with data layer
If we want to modify we will use second parameter of this array (set…)

Whenever state variable update react will re-render the component

If here ListOfRestarant is normal variable so if we update UI will not update it only data update

Using Hooks we can update data + UI

1. Import useState hook from react

```
import {useState} from "react";
```

2. Syntax of useState

```
const [ListOfRestaurants, setListOfRestaurants]=useState([Initial value ])
```

3. Logic

```
<div className="filter">

<button className="filter-btn" onClick={()=>
{ //filter logic here
FilterList = ListOfRestaurants.filter((res)=>
res.info.avgRating > 4
    );
    setListOfRestaurants(FilterList);
}} >Top Rated Restaurants</button>

</div>
```

Here instead of ListOfRestaurants we using mockData

```
import RestaurantCard from "./RestaurantCard";
import resList from "../utils/mockData";
import {useState} from "react";

const Body =() =>
{

const [ListOfRestaurants, setListOfRestaurants]=useState(resList)
```

React fiber

React use **reconciliation** algorithm is also known as react fiber

Why react is fast ?

Because React doing efficient DOM Manipulation
How ?
Because it has virtual DOM it has Diff Algorithm it can do efficient DOM Manipulation it can find out diff and update the UI

React has one of the best render Mechanism
React render cycle is very fast

Virtual DOM

Virtual DOM is representation of actual DOM

```
const AppLayout = () =>
{
  console.log(<Body/>);
  return(


      <div className="app">

       <Header/>
       <Body/>

      </div>
  )
}
```

```
▼ Object i
    $$typeof: Symbol(react.element)
    key: null
  ▶ props: {}
    ref: null
  ▶ type: ()=> {…}
  ▶ _owner: FiberNode {tag: 0, key: null, stateNode: null, elementType: ƒ, type: ƒ, …}
  ▶ _store: {validated: false}
    _self: undefined
  ▶ _source: {fileName: 'App.js', lineNumber: 14, columnNumber: 15}
  ▶ [[Prototype]]: Object
```

This is virtual DOM

Previously we create React element is object so similarly when we have big structure it's all an object

Whenever state variable update React rerender the components

Diff Algo

Find out the difference between two virtual DOM

Before and after update , calculating that difference and actually update the DOM on every render cycle

React do fast DOM manipulation

It can do efficient DOm manipulation