Episode 6 - Exploring the world

If there is any change React can quickly re-render everything in the component.

Microservices

In olden days, there used to be only one big project to handle all the operations through Apis using UI which is known as Monolith Architecture

By now, in newer startups, instead of having one big large project we have small-small different projects

This allows teams to choose best frameworks / libraries for the individual service

Example:
UI - React Js
Backend / Apis - Java
Notification - Python
Authentication - Python
Logs - Golang

Even the databases are different. All the pages are deployed at different ports but same domain name.
This is known as Microservice Architecture.

Advantages of Microservice Architecture

1) Easier to test

2) Develop, Deploy and scale individual component of your application independently

3) Easier to fix bugs

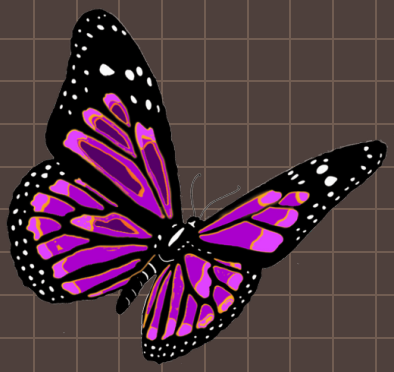
4) Won't crush entire app on one fault like in Monolith Architecture

To make network calls with other or outside

Fetch function is a browser function which is used to make api calls.

As in when my component loads it used to call the Api & fill the data

2 Ways of loading data



1) Loads → API calls → render page
(300 ms) (Total time = 500 ms)

2) Loads → Render UI → API calls → Update
(Total time = 100 ms)

This approach is better for User Experience

useEffect Hook → It has a callback function which will not be called immediately but when the useEffects wants it called. It is called after the component renders.
First it will render everytime the component renders or re-renders.

If you don't want to call it after every render, use a dependancy array.

[] → Dependancy array

When nothing is passed it will be called only one time

```
useEffect(() => {
  console.log("This is useEffect Hook");
}, [])

console.log("render");
```

Output:

render
This is useEffect Hook

Anything inside useEffect will be called later as it is a callback function

** Browser doesn't let us talk to local host to some external api directly.
Make the api calls (api fetching) in useEffect.

** When the component is rendered for the first it uses the default data and after the api call when data has been fetched it re-renders the component with new data.

** Whenever the state variable updates, react triggers a reconciliation cycle (re-renders the component)

Optional Chaining

It allows you to access properties of an object without worrying if the object or any of its nested properties are null or undefined.
It is denoted by '?.'

It is useful while dealing with asynchronous data fetching, where you're not sure if some data is present or not

```
// without optional chaining
setRestaurants(json.data.cards[2].data.cards)

// with optional chaining
setRestaurants(json?.data?.cards[2]?.data?.cards)
```

Conditional Rendering

It is rendering a peice of code / component based on some condition.
It is like 'if else' but using Ternary Operator (Short Hand if else)

Example:

```
{
  restaurant.length === 0 ? <Shimmer /> : <div>hello</div>
}
```

Early Return

if (!allRestaurants) return null

Only javaScript expression & statement can be rendered inside {} inside a component