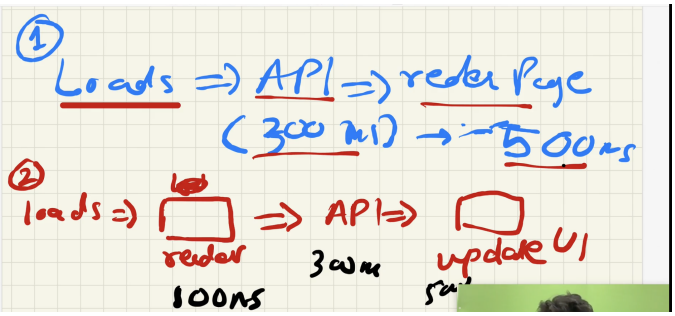
Chapter 06: Exploring the world

▼ Api Calls

- window.fetch builtin browser api helps us to make api call
  - Where do I call my API?
    - What happens if i call in component?
      - We will call api in every rerender
    - Usual cases →
      1. As and when page loads → call the api (~300ms)→ render page(~200ms)
      2. Render Initial Page (~100ms)→ Call api (300ms) → Update UI (100ms)
         a. This is good way, because of User experience. Page will be loading in 100ms
         b. This is also most preferred way in React
         c. React's strenght is reconciallition algorithm which updates DOM nodes faster using diff . So above way of renderred is not costly performance wise
         d. useEffect hook is useful here, using which we can call api after render



▼ useEffect()

- This is function.
- We call it by passing another function to it . Callback function.
- Callback function will be called whenever useEffect needs to be called
- React calls useEffect after every render

```
useEffect(() => {
  console.log("render")
})
```

- We can pass an empty dependency array to prevent it being calling after every rerender. It will be called just once after render

```
useEffect(() => {
  console.log("render")
}, []);      You, 39 sec
```

- Incase we want to call it on only certain state changes, add it to depencenvy array. Now it will be called once after initial render and then whenever state in dependency changes.

```
useEffect(() => {
  console.log("call this when dependency is changed");
}, [searchText]);      You, 35 seconds ago • Uncommitted changes
```

- So for api calls , we can use useEffect.

▼ Using Swiggy's public api

- When we try to call swiggy's api through localhost - browser doesnot allow. → Cors error. Use plugin to bypass cors → Allow cors
- Since this is Swiggy's public api, we can call it without any credentials

▼ Conditional renderring

Simple usecases

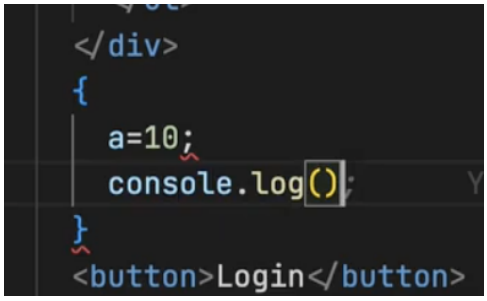
- Render shimmer UI or data UI

```
//Conditional Rendering
//if restraunt is empty ⇒ shimmer Ui
// if restraunt has data ⇒ actual data UI
```

- Login/LogOut button - render based on authentication flag.

Interesting facts:

- React will rerender component on every keystroke in input (two way binding- state update). But only updates diff element in DOM (Input element). That's why fast
- Inside {} in JSX: we cannot do this:



  - Any Javascript expression works over here, but not statement. This is expression now and it will work

  - **JavaScript Expressions:**
    - An expression is any piece of code that produces a value. It can be as simple as a literal value (e.g., 5 , 'hello') or a more complex combination of operators, variables, and function calls. Expressions can be used to

perform calculations, assign values to variables, or be passed as arguments to functions. Eg:

```
5 + 3
myVariable
2 * (x + 5)
Math.random()
```

- **JavaScript Statements:**
  - A statement is a complete instruction or action that performs a specific task. Unlike expressions, statements do not produce a value directly. Instead, they are executed for their side effects, such as modifying variables, controlling program flow, or interacting with the environment. Eg:

```
if (x > 5) {
  // code to execute if the condition is true
}

while (i < 10) {
  // code to execute repeatedly while the condition is true
  i++;
}

console.log("Hello, World!");
```

- React fibre is written in React library not in React-DOM. So diff algorithm is common in React-native too

Homework:

Error handle

Show Shimmer UI (Skeleton UI before loading)