Samajh gaya 👌 ab main tujhe **React Context API** ko simple **Hinglish** me samjhata hoon:

🌐 Context API kya hai?

Normally React me data parent → child via **props** pass hota hai.
Agar data ko deep nested components tak bhejna ho toh har level pe props dena padta hai (isko bolte hai **prop drilling**) → bahut irritating ho jata hai.

👉 **Context API** ek shortcut deta hai jisme tu directly kisi bhi component ko data de sakta hai bina har level pe props pass kiye.

⚙️ 3 Steps of Context API

1. **Create Context**
2. import { createContext } from "react";
3.
4. const MyContext = createContext();

Ye ek **context object** banata hai.

5. **Provider** (data dena)
   Apne component tree ko ek **Provider** ke andar wrap karna hota hai aur usme data dena hota hai.
6. <MyContext.Provider value={{ user: "Himanshu" }}>
7. <App />
8. </MyContext.Provider>
9. **Consumer** (data lena)
   Jaha data chahiye, waha use kar sakte ho:
   - **useContext Hook (modern way)**
   - import { useContext } from "react";
   -
   - const MyComponent = () => {
   - const { user } = useContext(MyContext);
   - return <h1>Hello {user}</h1>;

- };
- Ya **Context.Consumer** (old style, rarely used).

📖 **Chhota Example**

```
// Context.js
import { createContext } from "react";
export const ThemeContext = createContext();

// index.js
import React from "react";
import ReactDOM from "react-dom";
import App from "./App";
import { ThemeContext } from "./Context";

ReactDOM.render(
 <ThemeContext.Provider value={{ theme: "dark" }}>
  <App />
 </ThemeContext.Provider>,
 document.getElementById("root")
);

// App.js
import React, { useContext } from "react";
import { ThemeContext } from "./Context";

function App() {
 const { theme } = useContext(ThemeContext);
 return <h1>Current Theme: {theme}</h1>;
```

```
}

export default App;
```

✅ **Kab use karein Context API?**

- Jab ek hi data ko multiple jagah use karna ho (auth, theme, language, user info).
- Jab **prop drilling** avoid karna ho.
- Small/medium project ke liye best (large projects me Redux/Zustand zyada flexible hote hain).

Good question 👍
Chal, simple **Hinglish** me samjhata hoon:

🌐 **Provider kya hota hai (Context API me)?**

Jab tu createContext() use karke ek **Context** banata hai, to uske sath React automatically ek **Provider** deta hai.

- **Provider** ek special **component** hai.
- Ye component apne andar wrapped components ko **data (state, values, functions)** supply karta hai.
- Data dene ke liye iske paas ek prop hoti hai → value

📌 **Example**

```
import React, { createContext } from "react";

// 1. Context create kiya
const UserContext = createContext();

function App() {
  return (
```

```
  // 2. Provider banaya aur value di
  <UserContext.Provider value={{ name: "Himanshu", age: 23 }}>
    <Dashboard />
  </UserContext.Provider>
 );
}

function Dashboard() {
 return <Profile />;
}

function Profile() {
 // 3. Kahin bhi consume kar sakte ho
 return (
  <UserContext.Consumer>
   {(value) => <h1>Hello {value.name}, age {value.age}</h1>}
  </UserContext.Consumer>
 );
}
```

🔑 **Key Points**

- Provider data ko **share** karta hai puri component tree me.
- Jo bhi components Provider ke andar hote hain, wo sab uska data access kar sakte hain.
- Ek hi Context ke multiple Providers ho sakte hain with **different values**.
- Usually hum useContext hook use karte hain data consume karne ke liye, taaki code clean rahe.

👉 So, **Provider = Data distributor**
Jo bhi value tu value prop me deta hai, wo pura tree ke andar available ho jati hai.
