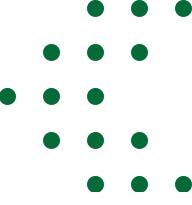




Eethal Nad
IT Solutions

MERN STACK DEVELOPMENT

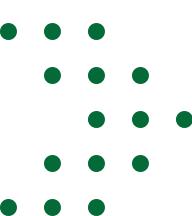
DAY-32

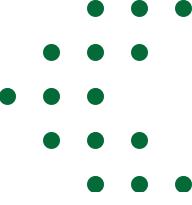


CONTEXT API – OVERVIEW

Context API is a built-in React feature used to share data globally across components without passing props manually at every level.

- Solves prop drilling
- Useful for global state (auth, theme, user, settings)





CONTEXT API – OVERVIEW

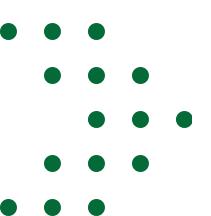
Why Context API ?

Problems without Context

- Props passed through many components
- Hard to maintain large apps

Context API Solution:

- Create one central store
- Any component can access data directly



CONTEXT API – OVERVIEW

When to Use Context API ?

Use Context when :

- Authentication data (user, token)
- Theme (dark / light)
- Language settings
- Global app configuration

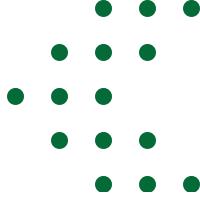
CORE CONCEPTS

Context API has 3 main parts:

- `createContext` – Create a context
- `Provider` – Provide global state
- `useContext` – Consume state anywhere

CREATING A CONTEXT

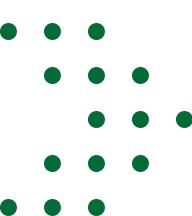
```
import { createContext } from "react";  
  
export const AuthContext = createContext();
```

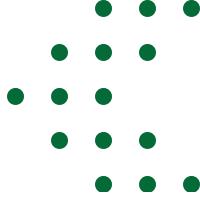


CONTEXT PROVIDER (GLOBAL STORE)

```
import { useState } from "react";
import { AuthContext } from "./AuthContext";

export const AuthProvider = ({ children }) => {
  const [user, setUser] = useState("Rocky");
```

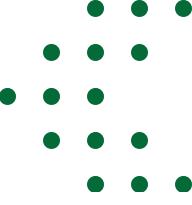




CONTEXT PROVIDER (GLOBAL STORE)

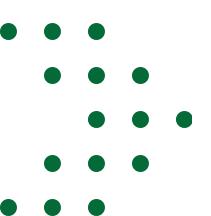
```
return (  
  <AuthContext.Provider value={{ user, setUser }}>  
    {children}  
  </AuthContext.Provider>  
);  
};
```

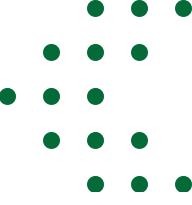
- ✓ Holds global state
- ✓ value = data shared to all components



WRAPPING APP WITH PROVIDER

```
import React from "react";
import ReactDOM from "react-dom";
import App from "./App";
import { AuthProvider } from "./AuthProvider";
ReactDOM.createRoot(document.getElementById("root")).render(
  <AuthProvider>
    <App />
  </AuthProvider>
);
```

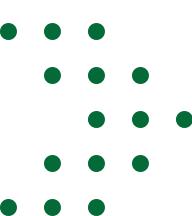


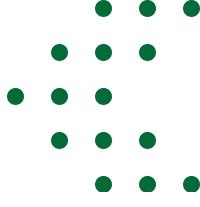


CONSUMING CONTEXT USING USECONTEXT

```
import { useContext } from "react";
import { AuthContext } from "./AuthContext";

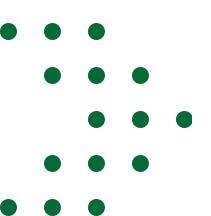
const Dashboard = () => {
  const { user, setUser } = useContext(AuthContext);
```





CONSUMING CONTEXT USING USECONTEXT

```
return (  
  <>  
  <h1>Welcome {user}</h1>  
  <button onClick={() => setUser("Spidey")}> Change User </button>  
  </>  
);  
};  
export default Dashboard;
```



CONTEXT API FLOW

- Create Context
- Wrap App with Provider
- Store global state in Provider
- Access using useContext

ADVANTAGES

- Avoids prop drilling
- Clean and readable code
- Built-in (no extra library)
- Perfect for medium-scale apps