**React JS**

**Theory of React Context API**

React Context API is a state management tool that helps pass data through the component tree without having to manually pass props at every level. It is useful for global state management, such as themes, authentication, or user preferences.

**Why Use Context API?**

1. Avoid Prop Drilling: Eliminates the need to pass props manually through every component.
2. Global State Management: Efficient for managing app-wide states like user authentication, themes, or language settings.
3. Lightweight Alternative to Redux: Works well for small to medium-scale applications without adding extra dependencies.

**Core Components of Context API**

1. React.createContext() – Creates a new context.
2. Provider – Wraps around components to provide the state.
3. Consumer – Used to access context values in class components.
4. useContext() – A React hook that allows function components to consume the context directly.

**React Context API Implementation**

**Here’s how you use the Context API in a React application**.

Step 1: Create the Context

import { createContext } from "react";

export const ThemeContext = createContext(null);

Step 2: Create a Context Provider

import { useState } from "react";

import { ThemeContext } from "./ThemeContext";

const ThemeProvider = ({ children }) => {

const [theme, setTheme] = useState("light");

const toggleTheme = () => {

setTheme((prevTheme) => (prevTheme === "light" ? "dark" : "light"));

};

return (

<ThemeContext.Provider value={{ theme, toggleTheme }}>

{children}

</ThemeContext.Provider>

);

};

export default ThemeProvider;

Step 3: Use Context in Components

import { useContext } from "react";

import { ThemeContext } from "./ThemeContext";

const ThemeToggleButton = () => {

const { theme, toggleTheme } = useContext(ThemeContext);

return (

<button onClick={toggleTheme} style={{ background: theme === "light" ? "#fff" : "#333", color: theme === "light" ? "#000" : "#fff" }}>

Toggle Theme

</button>

);

};

export default ThemeToggleButton;

Step 4: Wrap the App with the Provider

import ThemeProvider from "./ThemeProvider";

import ThemeToggleButton from "./ThemeToggleButton";

const App = () => {

return (

<ThemeProvider>

<ThemeToggleButton />

</ThemeProvider>

);

};

export default App;

When to Use Context API?

✅ When you need to share state globally (e.g., authentication, theme).  
✅ When prop drilling becomes an issue.  
✅ When Redux/MobX is overkill for your project.

❌ Avoid using Context for frequently changing states, like form inputs or animations, as it may cause unnecessary re-renders.

This should give you a solid foundation for React Context API.