In React, the useContext hook is used to access the value of a context that has been created using the createContext function. Context provides a way to pass data through the component tree without having to pass props manually at every level. Here's an example of how to use useContext in a React application:

Suppose you have a simple application where you want to provide a theme (e.g., light or dark mode) throughout your component tree using context.

First, create a context for your theme:

```
// ThemeContext.js
import { createContext } from 'react';

const ThemeContext = createContext();

export default ThemeContext;
```

Next, create a provider component that will provide the theme to its children:

```
// ThemeProvider.js
import React, { useState } from 'react';
import ThemeContext from './ThemeContext';

const ThemeProvider = ({ children }) => {
  const [theme, setTheme] = useState('light');

  const toggleTheme = () => {
    setTheme(theme === 'light' ? 'dark' : 'light');
  };

  return (
    <ThemeContext.Provider value={{ theme, toggleTheme }}>
      {children}
    </ThemeContext.Provider>
  );
};

export default ThemeProvider;
```

In this example, ThemeProvider provides both the theme value and a function to toggle the theme to its children components.

Now, let's create a component that uses the useContext hook to access the theme:

```
// ThemedButton.js
import React, { useContext } from 'react';
import ThemeContext from './ThemeContext';

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

  return (
    <button onClick={toggleTheme} style={{ background: theme === 'light' ? 'white' : 'black', color:
theme === 'light' ? 'black' : 'white' }}>
      Toggle Theme
    </button>
  );
};

export default ThemedButton;
```

In this component, we're using useContext to access the theme and toggleTheme function from the context.

Finally, use the ThemeProvider at the top level of your application to wrap the components that need access to the theme:

```
// App.js
import React from 'react';
import ThemedButton from './ThemedButton';
import ThemeProvider from './ThemeProvider';

const App = () => {
  return (
    <ThemeProvider>
      <div>
        <h1>Theme Example</h1>
        <ThemedButton />
      </div>
    </ThemeProvider>
  );
};

export default App;
```

Now, when you click the "Toggle Theme" button inside ThemedButton, it will update the theme in the entire application because it's using the context provided by ThemeProvider.

This is a simple example of how to use useContext in React to share data or state across components without having to pass props down manually.