React Context API Lab Report: Employee Management App

# Objectives

- Explain the need and benefits of React Context API  
- Working with createContext()  
- List the types of Router Components

# In this hands-on lab, you will learn how to:

- Create a context to be used by child components  
- Create a provider and consumer of the context

# Prerequisites

- Node.js  
- NPM  
- Visual Studio Code

# Step-by-Step Guide

## 1. Setup and Run the Application

- Unzip the provided Employee Management App.  
- Open the folder in Visual Studio Code.  
- In the terminal, run:  
 npm install  
 npm start

## 2. Explore the Files

- Open and explore: App.js, EmployeesList.js, EmployeeCard.js.  
- These currently use props to pass the theme.

## 3. Create ThemeContext.js

Inside the src folder, create ThemeContext.js with the following content:

import React from 'react';  
  
const ThemeContext = React.createContext('light');  
export default ThemeContext;

## 4. Modify App.js

Import and wrap the app in ThemeContext.Provider:

import React from 'react';  
import EmployeesList from './EmployeesList';  
import ThemeContext from './ThemeContext';  
  
function App() {  
 const theme = 'light'; // or 'dark'  
  
 return (  
 <ThemeContext.Provider value={theme}>  
 <div className="App">  
 <h1>Employee Management App</h1>  
 <EmployeesList />  
 </div>  
 </ThemeContext.Provider>  
 );  
}  
  
export default App;

## 5. Modify EmployeesList.js

Update the file to remove prop usage and keep it clean:

import React from 'react';  
import EmployeeCard from './EmployeeCard';  
  
const EmployeesList = () => {  
 const employees = [  
 { id: 1, name: 'Alice' },  
 { id: 2, name: 'Bob' }  
 ];  
  
 return (  
 <div>  
 <h2>Employees List</h2>  
 {employees.map(emp => (  
 <EmployeeCard key={emp.id} employee={emp} />  
 ))}  
 </div>  
 );  
}  
  
export default EmployeesList;

## 6. Modify EmployeeCard.js

Use useContext() to get the theme:

import React, { useContext } from 'react';  
import ThemeContext from './ThemeContext';  
  
const EmployeeCard = ({ employee }) => {  
 const theme = useContext(ThemeContext);  
  
 return (  
 <div className={`card card-${theme}`}>  
 <p>{employee.name}</p>  
 <button className={`btn-${theme}`}>View</button>  
 </div>  
 );  
}  
  
export default EmployeeCard;

## 7. Add Button Styles in App.css

.btn-light {  
 background-color: white;  
 color: black;  
 border: 1px solid black;  
}  
  
.btn-dark {  
 background-color: black;  
 color: white;  
 border: none;  
}  


