Roteiro de Atividade Prática

Nome: Turma: .

Nome: Turma: .

**Atividade: UserTracker – Ciclo de Vida de Componentes React**

Você é responsável por desenvolver um painel de controle para uma aplicação web que monitora a atividade de usuários em tempo real. Neste painel, você precisa implementar um componente React que exibe a lista de usuários on-line.

Crie um projeto em React chamado **“UserTracker”** que simule o monitoramento de usuários em tempo real. Neste projeto, você implementará um componente que exibe a lista de usuários on-line, atualiza essa lista sempre que um novo usuário se conecta e limpa os dados de conexão quando o componente é desmontado.

**Requisitos:**

1. **ComponentDidMount:** carregue a lista de usuários conectados ao montar o componente.
2. **ComponentDidUpdate:** atualize a lista de usuários conectados sempre que houver uma mudança nos dados de conexão.
3. **ComponentWillUnmount:** limpe os dados e desconecte os usuários quando o componente for desmontado.

**Tutorial: Implementação do Projeto “UserTracker”**

**Passo 1: *Setup* do Projeto**

Abra o terminal e crie um novo projeto React:

npx create-react-app user-tracker

|  |
| --- |
| Navegue até a pasta do projeto:  cd user-tracker  Abra o projeto no seu editor de código favorito (por exemplo, VSCode):  code.  **Passo 2: Criação do Componente Principal**  Dentro da pasta src, crie um novo arquivo chamado UserList.js.  No UserList.js, importe o React e defina a classe do componente dessa forma:  import React, { Component } from 'react';  class UserList extends Component {  constructor(props) {  super(props);  this.state = {  users: [] // Inicializa a lista de usuários como um array vazio  };  }  // Simula a obtenção de dados de usuários ao montar o componente  componentDidMount() {  this.setState({ users: ['Alice', 'Bob', 'Charlie'] });  }  // Atualiza a lista de usuários ao receber novos dados (simulado)  componentDidUpdate(prevProps, prevState) {  if (prevState.users !== this.state.users) {  console.log('Usuários atualizados:', this.state.users);  }  }  // Limpa dados ao desmontar o componente  componentWillUnmount() {  console.log('Desmontando componente, limpando dados...');  }  render() {  return (  <div>  <h1>Usuários Online:</h1>  <ul>  {this.state.users.map((user, index) => (  <li key={index}>{user}</li>  ))}  </ul>  </div>  );  }  }  export default UserList;  **Passo 3: Implementação do Componente no App Principal**  No arquivo src/App.js, substitua o conteúdo com o código abaixo para importar e usar o componente UserList:  import React from 'react';  import './App.css';  import UserList from './UserList';  function App() {  return (  <div className="App">  <UserList />  </div>  );  }  export default App;  **Passo 4: Executando o Projeto**  No terminal, inicie o servidor de desenvolvimento:  npm start  Abra o navegador e acesse http://localhost:3000 para ver o componente UserList em ação.  Verifique se a lista de usuários é carregada corretamente, se atualiza ao mudar alguma *prop*/estado relevante, e se qualquer processo em segundo plano é limpo ao desmontar o componente.  **Explicação do funcionamento**   * **ComponentDidMount:** assim que o componente UserList é montado, a lista de usuários é carregada, simulando uma conexão inicial. * **ComponentDidUpdate:** sempre que a lista de usuários muda, o componente registra essa mudança no console. * **ComponentWillUnmount:** antes do componente ser desmontado, ele limpa os dados, simulando a desconexão dos usuários. |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |