Roadmap de ReactJS

Iniciante a intermediário

1. **Introdução ao React:**
   - [ ] Compreenda os conceitos básicos do React, como componentes, `props` e `state`.
   - [ ] Instale o Node.js e o **npm** (Node Package Manager) para criar um ambiente de desenvolvimento.
2. **JSX (JavaScript XML):**
   - [ ] Aprenda a sintaxe **JSX**, que é uma extensão do JavaScript usada para definir a estrutura dos componentes React.
3. **Componentização:**
   - [ ] Aprofunde-se na criação e reutilização de componentes.
   - [ ] Compreenda os conceitos de componentes funcionais e de classe.
4. **State e Lifecycle:**
   - [ ] Entenda como o `state` funciona em um componente.
   - [ ] Saiba quando usar componentes baseados em classe e quando usar componentes funcionais com `hooks`.
5. **Props e Prop Drilling:**
   - [ ] Aprenda a passar dados para componentes filhos através de `props`.
   - [ ] Compreenda o conceito de **prop drilling** e suas limitações.
6. **Hooks em React:**
   - [ ] Domine os conceitos básicos de `useState`, `useEffect`, `useContext` e `useReducer`.
   - [ ] Aprenda a criar seus próprios hooks personalizados.
7. **Context API e `useContext`:**
   - [ ] Compreenda o que é a **Context API** e como ela pode ser usada para compartilhar estados entre componentes sem *prop drilling*.
   - [ ] Aprenda a utilizar o `useContext` para consumir o contexto em componentes funcionais.

Avançado

8. **Redux e Ggerenciamento de estado:**
   - [ ] Introdução ao **Redux** como uma biblioteca para gerenciamento de estado global.

- [ ] Crie actions, reducers e utilize o store para gerenciar o estado da aplicação.

9. **React Router:**
   - [ ] Aprenda a navegar entre diferentes componentes usando **React Router**.
   - [ ] Configure rotas aninhadas e autenticação de rotas.
10. **Testes em React:**
    - [ ] Aprenda a escrever testes unitários e de integração para seus componentes React usando **Jest** e **React Testing Library**.
    - [ ] Entenda os conceitos de *mocks* e *stubs* para testar componentes dependentes.
11. **Otimização de performance em React:**
    - [ ] Compreenda a importância do **memoization** e como usar o `React.memo` para otimizar a renderização de componentes.
    - [ ] Aprenda sobre o uso do `shouldComponentUpdate` em componentes de classe e o `useMemo` e `useCallback` em componentes funcionais.
12. **Hooks avançados e custom hooks:**
    - [ ] Aprofunde-se em hooks avançados como `useReducer`, `useLayoutEffect`, `useRef`, entre outros.
    - [ ] Crie seus próprios hooks customizados para lógicas reutilizáveis.
13. **Integração com APIs e Axios:**
    - [ ] Aprenda a fazer requisições HTTP usando **Axios** e integre APIs externas em sua aplicação React.
    - [ ] Trate erros e estados de carregamento ao fazer requisições assíncronas.