**VITE + REACT**

# Vite

Vite là một công cụ phát triển web nhanh và hiệu quả, tập trung vào việc cung cấp môi trường phát triển cực kỳ nhanh chóng cho các ứng dụng web hiện đại. Tên "Vite" được lấy từ cụm từ tiếng Pháp "vitesse" có nghĩa là "tốc độ".

Vite được xây dựng bởi Evan You, người cũng là tác giả của framework JavaScript Vue.js. Nó chủ yếu được sử dụng cho việc phát triển các ứng dụng web sử dụng các ngôn ngữ như JavaScript hoặc TypeScript.

1. **Nhanh chóng**: Vite sử dụng ES module native và hậu tố (PostCSS) để cung cấp một môi trường phát triển vô cùng nhanh chóng.
2. **Tích hợp**: Hỗ trợ nhiều loại ứng dụng web, bao gồm cả Vue, React, và cả vanilla JavaScript.
3. **Hot Module Replacement (HMR)**: Cho phép bạn xem các thay đổi ngay lập tức mà không cần phải làm mới trình duyệt.
4. **Hỗ trợ TypeScript**: Tích hợp sẵn với TypeScript và cung cấp hỗ trợ tốt cho việc phát triển TypeScript.

Tóm lại, Vite là một công cụ phát triển web mạnh mẽ, nhẹ nhàng và dễ sử dụng, giúp tăng tốc quá trình phát triển ứng dụng web của bạn.

Cú pháp tạo: (Cần cài Node.js >= 14.18 version)

npm create vite@latest > Đặt tên project > React > TypeScript + SWC (có thể chọn cái khác)

cd vite-project

npm install

npm run dev

Tại sao phải dùng thêm Vite ?

- Có thể do nó đơn giản và nhanh chóng để tạo ra mấy dự án như React, Vue, …..

# React

# TypeScript

TypeScript là một ngôn ngữ lập trình mã nguồn mở được phát triển bởi Microsoft. Nó là một phần mở rộng của JavaScript, nơi nó thêm vào các tính năng tĩnh kiểu dữ liệu, khai báo biến, các lớp và interfaces để giúp kiểm tra lỗi và tăng tính bảo mật trong quá trình phát triển phần mềm.

Các tính năng chính của TypeScript bao gồm:

1. **Tính tĩnh kiểu dữ liệu**: TypeScript cho phép bạn xác định kiểu dữ liệu của biến, tham số hàm, giá trị trả về hàm và các thành phần khác trong mã nguồn của bạn. Điều này giúp kiểm tra lỗi ngay từ giai đoạn biên dịch, giúp tránh được một số lỗi phổ biến trong quá trình chạy ứng dụng.
2. **Khai báo kiểu**: TypeScript cho phép bạn định nghĩa các kiểu dữ liệu tùy chỉnh, bao gồm cả kiểu đối tượng, kiểu mảng, kiểu hàm và nhiều hơn nữa.
3. **Hỗ trợ ECMAScript mới nhất**: TypeScript hỗ trợ các tính năng mới của ECMAScript, cho phép bạn sử dụng các tính năng của các phiên bản mới của JavaScript mà không cần phải lo lắng về việc hỗ trợ trên trình duyệt.
4. **Kiểm tra kiểu dữ liệu tại biên dịch**: TypeScript kiểm tra kiểu dữ liệu của mã nguồn của bạn tại thời điểm biên dịch, giúp bạn phát hiện và sửa lỗi một cách dễ dàng trước khi chạy ứng dụng.
5. **IDE hỗ trợ tốt**: TypeScript được tích hợp tốt với nhiều IDE phổ biến như Visual Studio Code, Sublime Text, và IntelliJ IDEA, cung cấp các tính năng như gợi ý mã, kiểm tra lỗi, và tự động hoàn thành mã.

Tóm lại, TypeScript là một ngôn ngữ lập trình phổ biến được sử dụng rộng rãi trong cộng đồng phát triển phần mềm để tăng tính bảo mật, tính năng và hiệu suất của ứng dụng JavaScript.

React Hooks là một tính năng được giới thiệu trong React phiên bản 16.8, cho phép bạn sử dụng các tính năng của React mà không cần sử dụng lớp component (class component). Các Hooks là các hàm JavaScript đặc biệt cho phép bạn sử dụng các tính năng như state và vòng đời trong các hàm function component của React.

Dưới đây là một số hooks phổ biến và tính năng chính của chúng:

1. **useState**: Hook này cho phép bạn sử dụng state trong function component. Bằng cách này, bạn có thể thay đổi giá trị của state và gây ra việc render lại component.
2. **useEffect**: Hook này cho phép bạn thực hiện các side effects trong function component. Điều này có thể là việc gọi API, thay đổi DOM, hoặc thực hiện các tác vụ sau khi component được render.
3. **useContext**: Cho phép bạn truy cập vào context trong function component. Context là một cách để truyền dữ liệu qua các cây component mà không cần truyền dữ liệu qua các props cấp cao.
4. **useReducer**: Tương tự như **useState**, nhưng thay vì sử dụng state đơn giản, hook này sử dụng một reducer để quản lý state với các trạng thái phức tạp hơn.
5. **useCallback và useMemo**: Đây là các hook giúp tối ưu hóa hiệu suất của component bằng cách tránh việc render lại không cần thiết.
6. **useRef**: Cho phép bạn lưu trữ một tham chiếu đến một phần tử DOM hoặc một biến trong function component mà không gây ra việc render lại component.

Sử dụng React Hooks giúp code của bạn trở nên ngắn gọn, dễ đọc hơn và dễ bảo trì hơn so với việc sử dụng class components trước đây.