## Tuần 7: React Router: Tìm hiểu và điều hướng giữa các trang trong ứng dụng React

### Tìm hiểu về React Router

#### React Router để làm gì?

* Nếu React giúp bạn xây dựng giao diện, thì React Router lại giúp bạn quản lý các đường dẫn (URL) trong ứng dụng của bạn. Nhờ có React Router, bạn có thể tạo ra các trang web đa trang (multi-page application)[[1]](#footnote-1) hoặc các ứng dụng một trang (single-page application)[[2]](#footnote-2) một cách dễ dàng. Nó cho phép bạn định nghĩa các route (đường dẫn)[[3]](#footnote-3) khác nhau và hiển thị nội dung tương ứng khi người dùng truy cập vào các route đó.

#### Về React Router

* Nó là một thư viện (library) được xây dựng trên nền tảng của React. Nói cách khác, React Router được thiết kế đặc biệt để làm việc cùng với React, cung cấp các công cụ và tính năng giúp bạn quản lý các đường dẫn (URL) trong ứng dụng React của mình.
* React Router không phải là một tính năng mặc định của React. Bạn cần phải cài đặt và cấu hình nó vào dự án React của mình để sử dụng.
* React không có tính năng định tuyến (routing)[[4]](#footnote-4) tích hợp sẵn. Để tạo ra các ứng dụng web đa trang hoặc các ứng dụng một trang phức tạp, bạn cần sử dụng một thư viện định tuyến như React Router.
* React Router thường được sử dụng như một phần của các framework[[5]](#footnote-5) lớn hơn. Nhiều framework React như Next.js hay Gatsby[[6]](#footnote-6), đã tích hợp sẵn React Router hoặc cung cấp các cấu hình mặc định để bạn dễ dàng sử dụng.

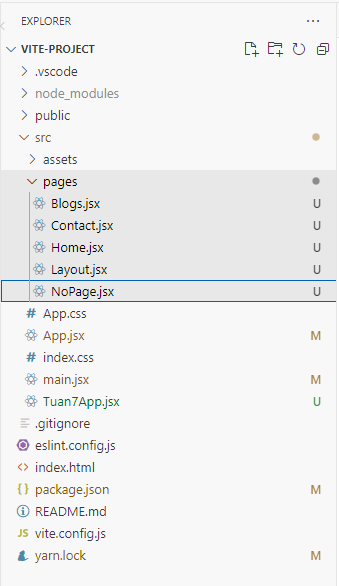
### Cài đặt thư viện React Router

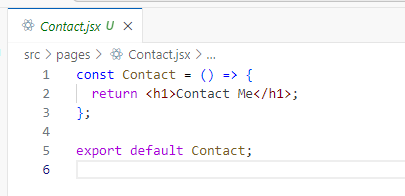
* Bởi vì React Router là một thư viện (library) được xây dựng trên nền tảng của React, nó không có sẵn trong dự án mà cần phải được cài đặt.
* Đảm bảo bạn đã có một dự án Vite đang chạy để tiếp tục.

#### Cài đặt

1. Cài đặt React Router:
   * Bắt đầu với một dự án Yarn mới được tạo
   * Mở terminal trong thư mục gốc của dự án và chạy lệnh sau:
   * 

#### Cấu trúc thư mục

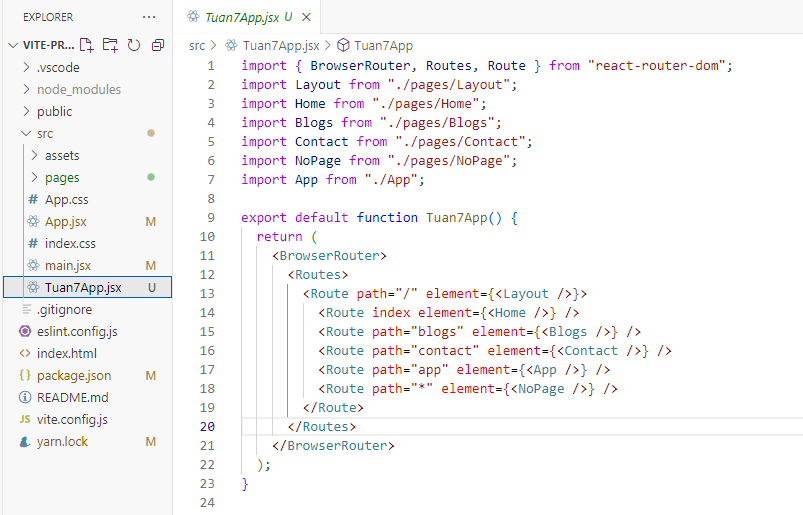
* + Trong thư mục src, chúng ta sẽ tạo một thư mục có tên pages gồm nhiều tệp:
    1. Layout.js
    2. Home.js
    3. Blogs.js
    4. Contact.js
    5. NoPage.js
  + 
  + Mỗi tệp sẽ chứa một thành phần React rất cơ bản.
  + A screenshot of a computer

    Description automatically generated
  + 
  + A screenshot of a computer program

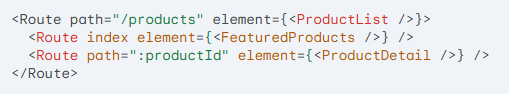
    Description automatically generated
  + A screenshot of a computer code

    Description automatically generated

#### Sử dụng cơ bản

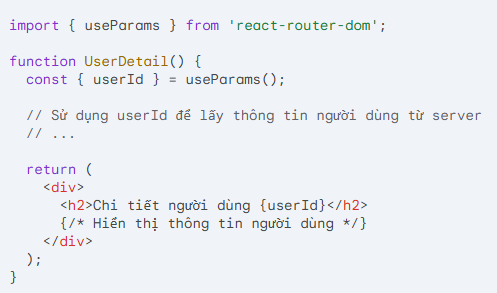
* + Tạo một component Tuan7App sẽ làm một App mới để không phải sửa App mặc định
  + 
    1. Đầu tiên gói nội dung của mình bằng <BrowserRouter>.
    2. Sau đó chúng ta định nghĩa <Routes>. Một ứng dụng có thể có nhiều <Routes>. Ví dụ cơ bản của chúng ta chỉ sử dụng một .
    3. Các <Route> có thể được lồng nhau. <Route> đầu tiên có đường dẫn là / và hiển thị thành phần <Layout>.
    4. Các <Route> lồng nhau kế thừa và thêm vào tuyến đường cha. Vì vậy, đường dẫn cho <Blogs> được kết hợp với tuyến đường cha và trở thành /blogs.
    5. Route <Home> không có đường dẫn nhưng có một thuộc tính index. Thuộc tính này chỉ định tuyến này là tuyến mặc định cho tuyến cha, đó là /.
    6. Route có path là \* hoạt động như một giải pháp chung cho bất kỳ URL nào không xác định. Điều này rất tuyệt vời cho trang lỗi 404.
  + <BrowserRouter> lưu trữ vị trí hiện tại trên thanh địa chỉ của trình duyệt bằng cách sử dụng URL và cho phép điều hướng bằng cách sử dụng các nút tiến và lùi lịch sử của trình duyệt.
  + <Routes>
    1. Là một container chứa tất cả các route của ứng dụng.
    2. Quản lý việc khớp các URL với các route tương ứng.
    3. Bất cứ khi nào vị trí thay đổi, <Routes> sẽ tìm kiếm tất cả các tuyến con của nó để tìm tuyến phù hợp nhất và hiển thị nhánh đó của UI.

#### <Route>

* + Route là một thành phần cốt lõi trong React Router, đóng vai trò định nghĩa một đường dẫn cụ thể trong ứng dụng của bạn. Khi người dùng truy cập vào một đường dẫn nào đó, React Router sẽ so sánh đường dẫn đó với các Route đã được định nghĩa và hiển thị component tương ứng.
  + Các thuộc tính chính của Route
    1. path: thuộc tính bắt buộc, xác định đường dẫn URL mà route sẽ khớp. Ví dụ: /, /about, /products/:id.
    2. element: thuộc tính bắt buộc, xác định component React sẽ được hiển thị khi đường dẫn khớp.
    3. exact: Nếu được đặt là true, route chỉ khớp chính xác với đường dẫn đã định nghĩa.
       1. Đảm bảo rằng một route chỉ khớp chính xác với đường dẫn đã định.
       2. Ngăn chặn các route con (nested routes) vô tình khớp với route cha.
       3. 
       4. Với thuộc tính exact, route này chỉ khớp với đường dẫn /about chính xác, không khớp với các đường dẫn con như /about/contact.
    4. index: Khi một route có index={true}, nó sẽ được hiển thị khi đường dẫn khớp với đường dẫn cha của nó.
       1. Thường được sử dụng để hiển thị component mặc định cho một route.
       2. 
       3. Trong ví dụ này, khi người dùng truy cập /products, component FeaturedProducts (có index={true}) sẽ được hiển thị.
    5. children: Cho phép bạn định nghĩa các route con, tạo ra cấu trúc route lồng nhau.
       1. A computer screen shot of a code

          Description automatically generated
       2. Trong ví dụ này, /users và /products là các route con của /admin.
       3. /admin/users và /admin/products

#### Dynamic Routing

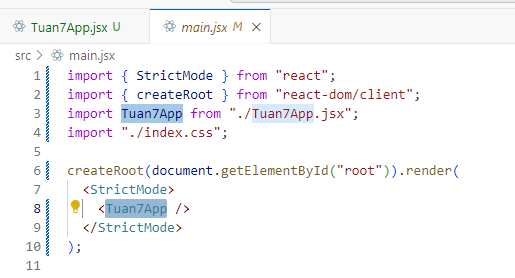
* + Dynamic routing cho phép bạn tạo các đường dẫn động, tức là các đường dẫn có chứa các phần thay đổi được. Điều này rất hữu ích khi bạn cần hiển thị thông tin khác nhau dựa trên các tham số trong URL.
  + 
  + Trong ví dụ này, :userId là một tham số động. Khi người dùng truy cập vào /users/123, tham số userId sẽ có giá trị là 123 và component UserDetail sẽ được hiển thị với thông tin của người dùng có ID là 123.
  + Cách sử dụng Dynamic Routing
    1. Định nghĩa route: Sử dụng dấu : để đánh dấu một tham số động trong path của route.
    2. Truy xuất tham số: Sử dụng hook useParams để truy xuất giá trị của các tham số động từ URL.
    3. 

#### Các trang/ Component

* + Thành phần Layout có các phần tử <Outlet> và <Link>.
    1. <Outlet> sẽ render Route hiện tại được chọn.
    2. <Link> được sử dụng để đặt URL và theo dõi lịch sử duyệt web.
  + Bất cứ khi nào cần liên kết đến một đường dẫn, sử dụng <Link> thay cho <a href="">.
  + Layout Route là một component được chia sẻ và được chèn vào tất cả các trang, chẳng hạn như menu điều hướng.
  + A screenshot of a computer program

    Description automatically generated
  + <Layout> sẽ là trang được render, <Outlet> sẽ là các trang con được điều hướng

#### Hoàn tất ứng dụng thử

* + Đổi root component từ App thành Tuan7App
  + 
  + Chạy server
  + A screenshot of a computer

    Description automatically generated
  + Bởi vì <Route> Home có thuộc tính index, nên nó được hiển thị khi truy cập <Route> cha, ở đây là /
  + Bởi vì <Outlet> nằm dưới list các link, nên nội dung trang nằm ở dưới menu
  + A screenshot of a computer program

    Description automatically generated
  + Điều hướng đến các trang Blogs, Contact, Default App
  + 
  + A screenshot of a computer

    Description automatically generated
  + A screenshot of a computer

    Description automatically generated
  + Trang NoPage được hiển thị khi Route đang điều hướng đến không được xác định
  + A screenshot of a computer

    Description automatically generated
  + Ví dụ này truy cập đường dẫn /404 không được xác định trước
  + A screenshot of a computer code

    Description automatically generated

1. Loại website truyền thống, mỗi khi bạn click vào một liên kết, trình duyệt sẽ tải lại toàn bộ trang mới. Nghĩa là mỗi trang đều độc lập và được tải riêng biệt. [↑](#footnote-ref-1)
2. Loại website hiện đại, chỉ tải một trang HTML duy nhất khi bạn truy cập lần đầu. Sau đó, khi bạn tương tác với website, chỉ có các phần nội dung cần thiết được cập nhật mà không cần tải lại toàn bộ trang. [↑](#footnote-ref-2)
3. Một cấu hình xác định một đường dẫn. Nó giống như một bản đồ chỉ đường, giúp ứng dụng biết nên hiển thị gì khi người dùng nhập một URL cụ thể vào trình duyệt. [↑](#footnote-ref-3)
4. Quá trình xác định và hiển thị các thành phần UI khác nhau của ứng dụng dựa trên URL mà người dùng truy cập. [↑](#footnote-ref-4)
5. Một cấu trúc nền tảng cung cấp các công cụ và quy tắc để xây dựng ứng dụng một cách nhanh chóng và hiệu quả. [↑](#footnote-ref-5)
6. Ví dụ về các framework React phổ biến, cung cấp các tính năng bổ sung như tối ưu hóa hiệu suất, tạo trang tĩnh (static site generation), giúp quá trình phát triển ứng dụng React trở nên dễ dàng hơn. [↑](#footnote-ref-6)