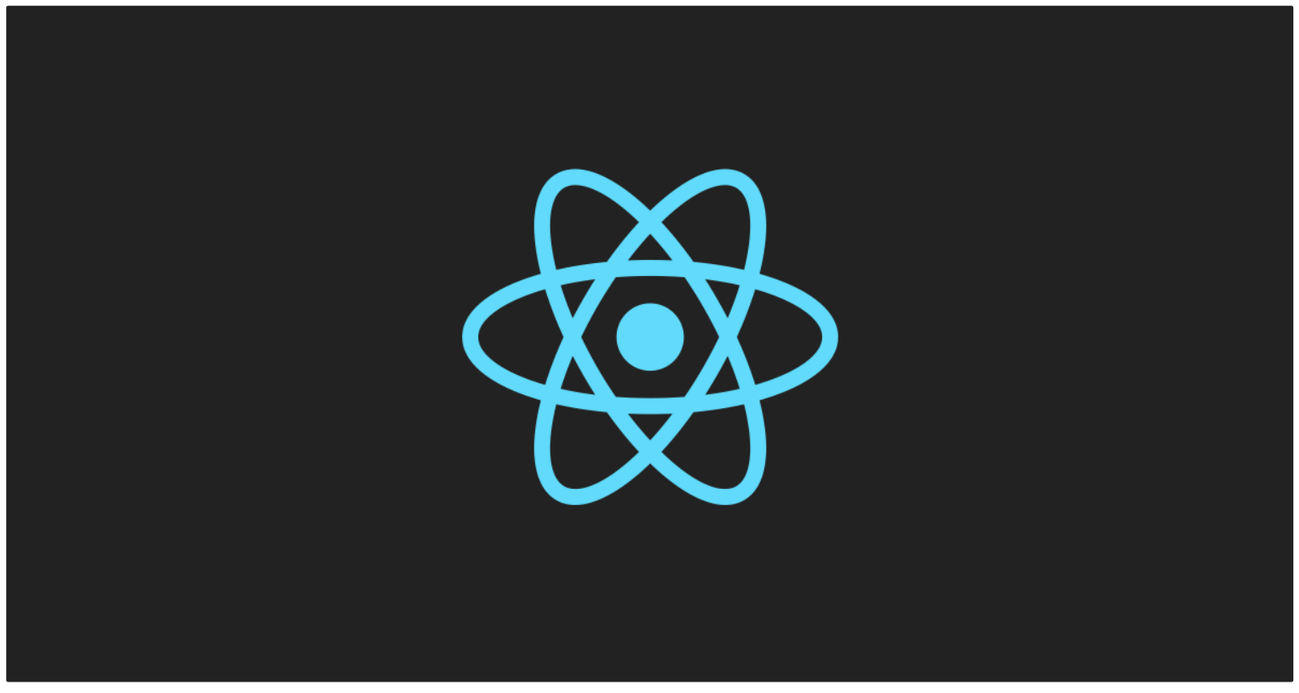
# Lesson 3: Introduction to React

| | |
|---|---|
| Created | March 22, 2022 9:55 AM |
| Tags | Empty |
| Related to Untitle... | Empty |

💡 Hiểu cơ bản React là gì, tại sao sử dụng React và cách tạo ra một ứng dụng React.

Trong khoá này, chúng ta sẽ tìm hiểu về React, thư viện phổ biến nhất hiện nay để xây dựng ứng dụng web. Hãy cùng tìm hiểu xem React là gì và tại sao lại sử dụng React.



## 1. ReactJS là gì?

**ReactJS là thư viện dùng để xây dựng giao diện người dùng.** ReactJS được tạo ra bởi Facebook vào năm 2013. ReactJS và các thư viện khác xung quanh nó được sử dụng để xây dựng các ứng dụng web hiện đại, giàu tính năng và nhiều tương tác. React tiếp cận theo hướng **component**, chia nhỏ giao diện thành nhiều thành phần và kết hợp chúng lại với nhau để tạo thành một ứng dụng đầy đủ.

## 2. Tại sao lại chọn ReactJS?

Các ứng dụng web càng ngày càng có nhiều tính năng phức tạp hơn. Người dùng cũng muốn có những trải nghiệm tốt hơn với khả năng phản hồi nhanh chóng. Javascript đã có thể giúp chúng ta làm được việc đó. Tuy nhiên, với những ứng dụng rất phức tạp và nhiều tính năng, việc chỉ sử dụng Javascript thuần tuý sẽ nhanh chóng khiến ứng dụng rất khó bảo trì và nâng cấp sau này. Các thư viện JS ra đời nhằm giúp lập trình viên có thể tạo ra các trang web nhanh chóng và dễ bảo trì sau này. React là một trong số các thư viện như vậy.

React có một vài đặc điểm đáng chú ý giúp nó trở nên phổ biến và được quan tâm nhiều ở thời điểm hiện nay:

- Code được viết theo hướng declarative: Thay vì quá tập trung vào các thao tác chỉnh sửa DOM (DOM manipulations), React giúp lập trình viên tập trung vào các thành phần logic và nó sẽ chịu trách nhiệm việc chỉnh sửa DOM tương ứng với các dữ liệu.
- React có hiệu năng tốt khi nó sử dụng DOM ảo (Virtual DOM). React sẽ tính toán các sự thay đổi trên DOM ảo để lấy được kết quả sau cùng. Sau đó mới tiến hành việc cập nhập lại giao diện người dùng sao cho những phần thay đổi là ít nhất.
- React được phát triển và duy trì bởi Facebook. Điều này khiến cho vòng đời của React có thể sẽ được kéo dài lâu.
- React có hệ sinh thái các plugin phong phú. Lập trình viên có thể dễ dàng tìm kiếm các thư viện mà không cần phải tự làm.
- React có hỗ trợ viết ứng dụng mobile với ReactNative. Như vậy trong team chỉ cần nắm được React là đã có thể hoàn thành các phần frontend.

---

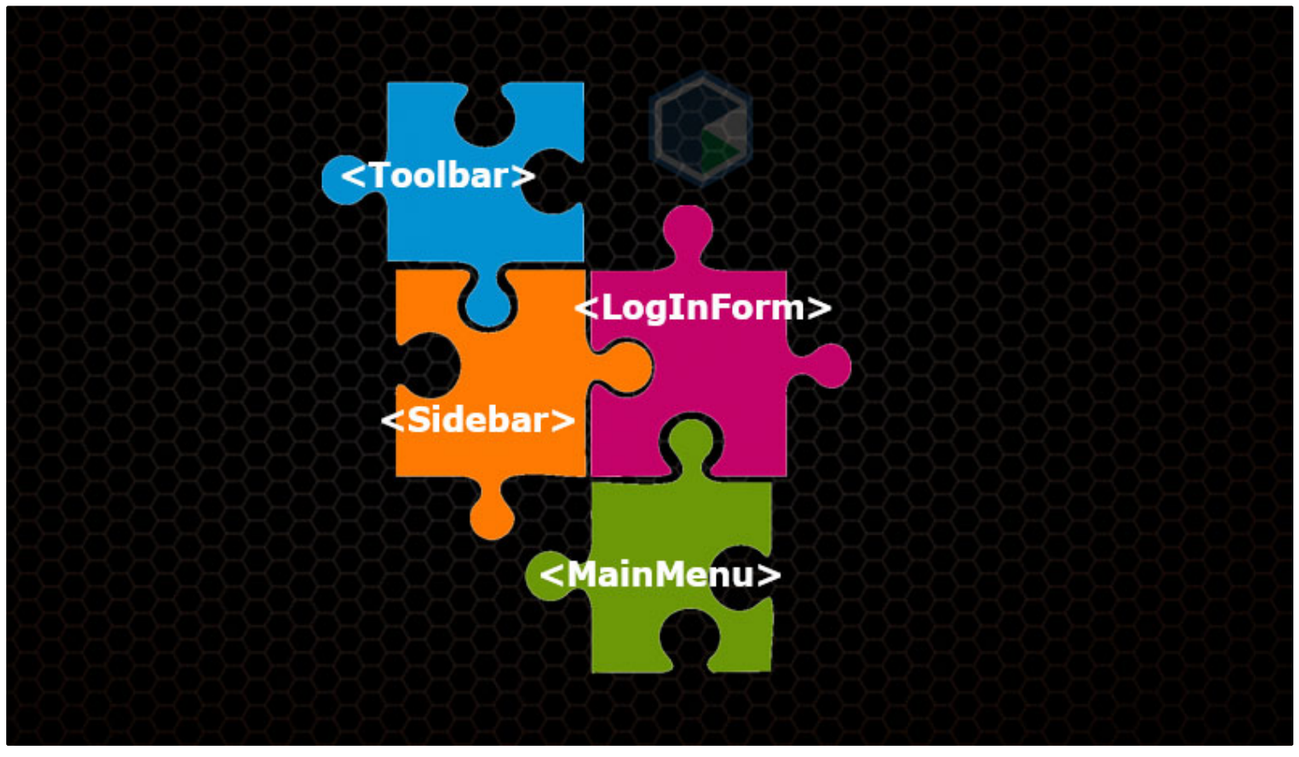
## 3. Những ai đang sử dụng ReactJS?

Có rất nhiều công ty nổi tiếng đang sử dụng React ở trong sản phẩm của mình. Có thể kể đến vài cái tên như Facebook, Netflix, Tesla... Những công ty trên đều là các công ty rất lớn với quy mô toàn cầu. Nghĩa là React đã được chứng minh trong thực tế là một thư viện giúp giải quyết được các vấn đề lớn.


> 💡 Ngoài ra thì cũng có nhiều đối thủ của React trên thị trường. Một vài cái tên có thể là Angular, Vue, Svelte hoặc thậm chí là jQuery. Mỗi thư viện đều có những ý tưởng giống nhau và có những đặc điểm khác nhau. Nếu có thời gian, các bạn nên tìm hiểu thêm về các thư viện đó, và so sánh nó với React.

---

## 4. Component là gì?

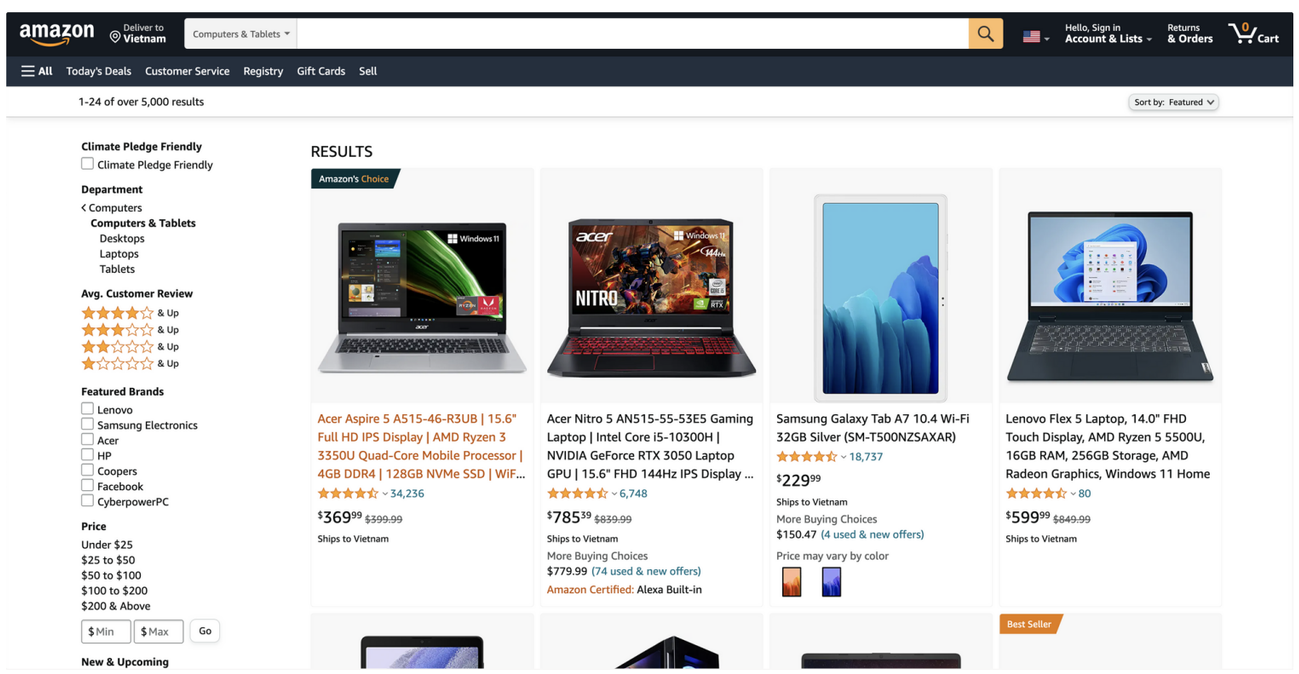



Component hiểu đơn giản là các thành phần riêng lẻ. Tuy nhiên chúng có thể kết hợp lại với nhau để tạo thành một hệ thống hoàn chỉnh.

Tư duy "chia để trị" này là một tư duy rất phổ biến trong các ngành kĩ thuật nói chung. Lấy ví dụ là một chiếc ô tô, ta thấy lốp xe và cửa kính chắn gió là hai bộ phận hoàn toàn riêng biệt. Tuy nhiên chúng phải tuân thủ những đặc điểm kĩ thuật nhất định (đường kính bánh, độ dày của kính, ...)
Với việc chia nhỏ các thành phần ra và sản xuất chúng riêng rẽ, ta có được những lợi ích sau:

- **Dễ sửa chữa**: Kĩ sư có thể tháo chiếc lốp xe ra và sửa chữa nó mà không làm ảnh hưởng tới toàn bộ cấu trúc của chiếc xe.
- **Tái sử dụng**: Khi đã có một bản thiết kế cho thành phần đó hoàn chỉnh, các kĩ sư có thể dễ dàng dùng lại bản thiết kế đó để tạo ra một sản phẩm khác tương tự.
- **Dễ thay thế:** Người dùng có thể thực hiện nâng cấp một phần của chiếc xe. Họ có thể tìm một chiếc lốp xe với các đặc tính khác, miễn sao nó tuân thủ những đặc điểm kĩ thuật nhất định để có thể khớp với các thành phần khác của xe.
- **Chuyên môn hoá cao**: Các nhà sản xuất chỉ cần tập trung vào một mảng nhất định, có thể xây dựng những dây chuyền lớn để sản xuất hàng loạt một thành phần nào đó.

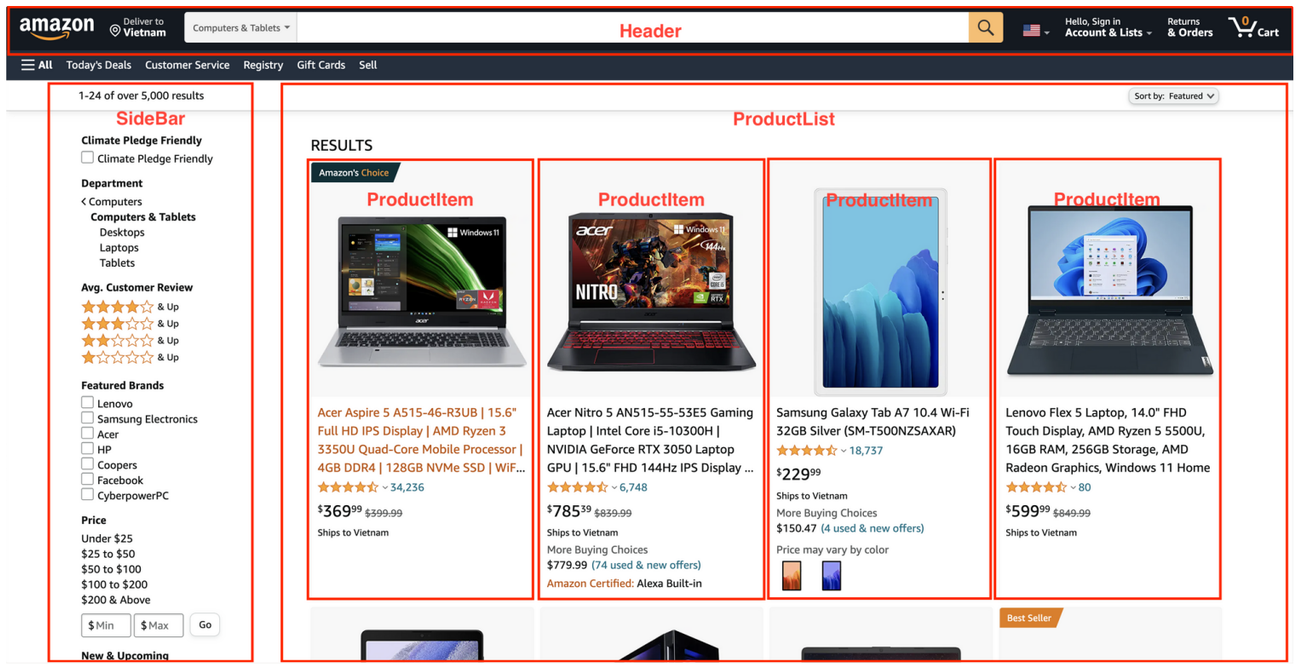
Trong việc xây dựng các ứng dụng web cũng vậy. Đây là hình ảnh của một trang web thương mại điện tử nổi tiếng:




Ứng dụng trên có nhiều tính năng. Tuy nhiên, chúng ta cũng có thể phân chia thành một vài thành phần như sau:

- Phần header có ô tìm kiếm và các menu điều hướng

- Phần sidebar với các bộ lọc
- Danh sách chi tiết các sản phẩm




Trong ứng dụng có nhiều san phẩm với các nội dung khác nhau. Tuy nhiên mỗi sản phẩm đều có cấu trúc giống nhau, bao gồm các thành phần: tên sản phẩm, đánh giá & reviews, giá, các phiên bản màu sắc, ... Như vậy, mỗi một sản phẩm là một đơn vị nhỏ. Khi chúng ta tái sử dụng các đơn vị đó nhiều lần, kết quả sẽ được một khối tổng thể là danh sách các sản phẩm như trên hình.

Hiểu được tư duy phân chia thành các component là cực kỳ quan trọng để có thể viết được các ứng dụng tốt, có tính tái sử dụng cao, dễ bảo trì sau này.

🤔 Trong ví dụ trên, một component như ProductItem còn có thể được phân tách thành các component nhỏ hơn nữa. Các bạn có thể thử chia nhỏ các thành phần khác thành các component nhỏ hơn.

---

## 5. Thử nghiệm đầu tiên với React Component

💡 Truy cập vào ứng dụng https://codepen.io/pen/ để thực hiện live code

Ta có một ứng dụng cơ bản như sau:

```
<div id="app"> <div class="card"> <div class="name">Name: Alice</div> <div
class="age">Age: 20</div> </div> <div class="card"> <div class="name">Name:
Bob</div> <div class="age">Age: 20</div> </div> <div class="card"> <div
class="name">Name: Cris</div> <div class="age">Age: 20</div> </div> </div>
```

HTML

```
.card { width: 200px; border: 1px solid black; padding: 20px; border-radius:
10px; font-family: sans-serif; margin: 5px } .name { font-size: 20px; font-
weight: bold; } .age { font-size: 14px; font-style: italic; }
```

CSS

Khi chạy ứng dụng, chúng ta sẽ nhận được kế quả như hình dưới đây:


Đây là cách tiếp cận ứng dụng theo cách thông thường. Ba phần tử trong danh sách là ba phần khác nhau trong HTML. Với các ứng dụng đơn giản thì hoàn toàn bình thường. Tuy nhiên nó cũng bộc lộ một vài điểm yếu sau:

- Khi muốn thêm một phần tử mới vào danh sách, chúng ta cần copy lại HTML và sửa lại nội dung tên, tuổi tương ứng
- Khi thêm trường “Address”, chúng ta sẽ cần phải thêm vào cả 3 phần tử trong danh sách.
- Tương tự với việc sửa dữ liệu, chúng ta sẽ phải làm với cả 3 phần tử

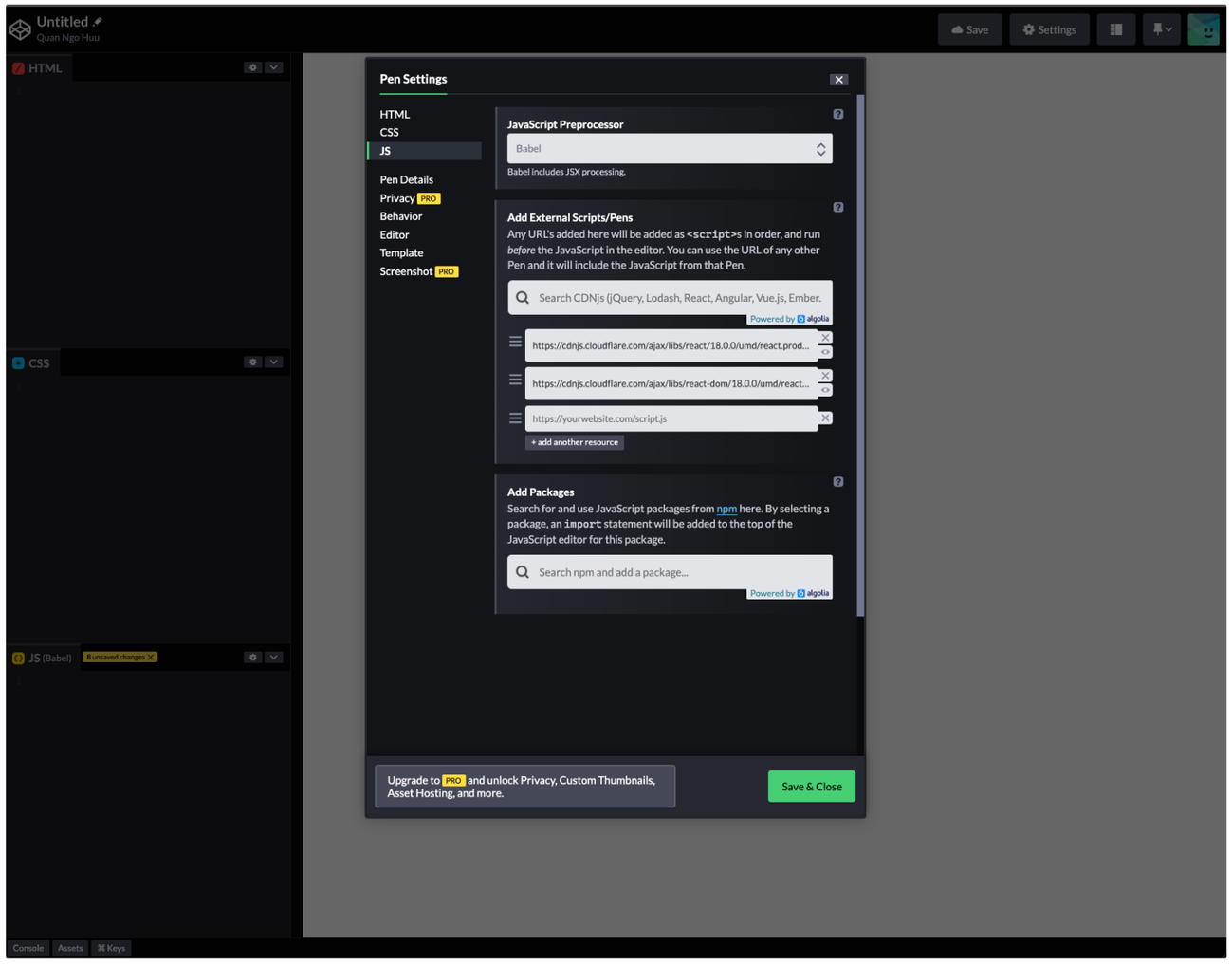
Và vấn đề sẽ trở nên phức tạp hơn khi chúng ta có nhiều phần tử hơn!
Với ReactJS, ta có đoạn code như sau:

```
function NameCard(props) { return ( <div className="card"> <div
className="name">Name: {props.name}</div> <div className="age">Age:
{props.age}</div> </div> ) } ReactDOM.render(<div> <NameCard name="Alice"
age="20" /> <NameCard name="Bob" age="20" /> <NameCard name="Cris" age="20" />
</div>, document.getElementById("app"))
```

Với ReactJS ta có thể thấy các đặc điểm sau:

- Function `NameCard` nhận vào một tham số đầu vào là “props”, và trả ra kết quả là một đoạn “HTML”? Là một cú pháp không tồn tại trong Javascript thông thường
- `class` được thay thế bằng `className`
- Một cú pháp đặc biệt `{}` để chèn nội dung từ JS vào trong HTML

📌 Để có thể chạy được đoạn code trên, chúng ta cần thay đổi một vài cài đặt trong codepen:
- Click vào biểu tượng gear icon của phần JS
- Phần ***JavaScript Preprocessor*:** chọn ***Babel***
**-** Phần **Add External Scripts/Pens,** chọn thêm 2 thư viện là `react` và `react-dom`




NameCard chính là một component trong React. Và sau khi có component đó, chúng ta có thể bắt đầu "tái sử dụng" nó bằng việc tạo ra 3 `NameCard` với các thuộc tính khác nhau. Cú pháp để sử dụng là `<NameCard ... />`, giống như việc sử dụng HTML. Đó cũng chính là cách React hoạt động: viết ra các component cũng chính là tạo ra các "thẻ HTML" tuỳ chỉnh và tái sử dụng chúng.

👷 Với kiến thức về component trong React, hãy thử tạo ra giao diện ứng dụng todo list đơn giản với. Chi tiết xem thêm tại Workbook

---

## 6. Khởi tạo ứng dụng React ở trên máy tính

Để có thể cài đặt được ứng dụng ReactJS trên máy tính, máy tính sẽ cần phải được cài đặt các công cụ sau:

- NodeJS (https://nodejs.org/en/)
- npm  & npx (sẽ được cài đặt cùng với NodeJS)

💡 Để kiểm tra NodeJS & npx đã được cài đặt ở trên máy tính hay chưa, chúng ta có thể sử dụng câu lệnh `node --version` và `npx --version`. Máy tính sẽ hiển thị phiên bản được cài đặt. Nếu nhìn thấy có lỗi xảy ra, không hiển thị phiên bản, hãy thử khởi động lại ứng dụng / máy tính.

Các bước để khởi tạo một ứng dụng React có tên là "hello-world":

1. Tạo một thư mục trên máy tính
2. Mở cửa sổ dòng lệnh bên trong thư mục đó
3. Chạy câu lệnh `npx create-react-app hello-world`. Sau đó đợi để ứng dụng được khởi tạo
4. Sau khi đã khởi tạo thành công, di chuyển vào bên trong thư mục "hello-world" vừa được tạo bằng câu lệnh `cd hello-world`.

5. Khởi chạy ứng dụng bằng câu lệnh `npm start` .
6. Trình duyệt có thể sẽ tự động mở ra, và ứng dụng đã được khởi tạo thành công

Trong quá trình phát triển ứng dụng, cửa sổ dòng lệnh cần phải được giữ nguyên và không được tắt ứng dụng đang chạy trên cửa sổ đó. Để tắt ứng dụng đó trên cửa sổ dòng lệnh, có thể sử dụng tổ hợp phím `Ctrl + C` .

Để chạy lại ứng dụng, hãy mở cửa sổ dòng lệnh trong thư mục "hello-world" và sử dụng câu lệnh `npm start`

## Cấu trúc dự án của React

Trong dự án React, chúng ta cần quan tâm một vài file và thư mục sau:

- `package.json`
- `node_modules`
- `public`
- `src`

File `package.json` là file cấu hình cho dự án. File này đóng vai trò quan trọng, giúp chúng ta tuỳ chỉnh các cấu hình, cài và xoá các thư viện khác nhau. Các thành phần cần chú ý là `dependencies` và `scripts`

Thư mục `node_modules` là thư mục chứa các thư viện được tải xuống từ bên ngoài được cấu hình bên trong `package.json`. Thư mục này có thể được cài lại thông qua câu lệnh `npm install`

Thư mục `public` là thư mục chứa file HTML duy nhất của dự án React, nơi mà các component sẽ được render.

Thư mục `src` là thư mục chứa toàn bộ source code của ứng dụng. Đây sẽ là thư mục chúng ta viết các components bên trong.

Hãy thử viết lại ứng dụng `NameCard` ở trên trong ứng dụng React vừa được khởi tạo.