- Back End
- ReactJS (Facebook)
- Hệ thống cũ
    - Static (Website tĩnh, không sử được, chỉ có giao diện demo)
    - Dynamic (Website động, đã kết nối với Server Back End và Database)
- Quy trình làm việc

Hệ thống cũ:

- Front End sẽ xây dựng Website tĩnh
- Đưa toàn bộ mã nguồn (Source Code) cho Back End.
- Back End có nhiệm vụ tách mã nguồn ra phía server, kết nối database, ...
- User gửi một yêu cầu (Request) tới Server Back End
- Server nhận được yêu cầu và xử lý ..... Server sẽ trả về thông tin (Response)
- Request 1: Nhận được mã nguồn html (Bên trong HTML khai báo script, link, img, ...)
- Sau Request 1: Trình duyệt biên dịch mã nguồn HTML được nhận về

Hệ thống mới:

- Tách biệt phần Front End và Back End hoàn toàn.
- Front End không liên quan gì tới mã nguồn Back End
- Back End không liên quan tới mã nguồn Front End
- Không gửi mã nguồn cho Back End -> Nhiệm vụ ráp dữ liệu thực tế (dữ liệu dynamic) là nhiệm vụ của người Front End
- Khi truy cập vào một trang xây dựng hoàn toàn bằng ReactJs -> View page source -> Không thấy gì cả ngoài div rỗng và một số đoạn script -> Đó trang có tên là SPA (Single Page Application) -> Tất cả nội dung đều được quản lí bởi Javascript -> ReactJs là một thư viện được xây dựng hoàn toàn bằng Javascript

Nhưng khái niệm phải biết

- SPA
- React Developer Tools
    - Giúp mình tìm kiếm lỗi, debug dễ dàng hơn khi lập trình
    - Giúp mình biết được trang nào xây dựng bằng ReactJs
- Javascript ES6 (Phiên bản mới của Javascript, chức năng mới)
- Babel
    - Là một trình biên dịch cho Javasript
    - Biên dịch cú pháp Javascript ES6 (Cú pháp mới) -> Về cú pháp cũ

- Webpack
- Khi xây dựng một trang có dùng ReactJs
    - Toàn bộ trang 100% đều dùng ReactJs để làm
    - Chỉ dùng ReactJs trong một phần chức năng nhỏ của trang Web
- Create React App: Một công cụ được xây dựng bởi Facebook giúp lập trình trinh tạo ra một Project mới xây dựng hoàn toàn 100% là reactjs kết hớp Babel, Webpack, ... được tích hợp và cấu hình sẵn.
- Khi cài đặt xong `npx create-react-app ten-folder-project`
- `node_modules` là thư mục chứa tất cả thư viện khác cần dùng để chạy được project. được quản lý bởi nodejs (React, Webpack, Babel,)
- `node_modules` hoàn toàn có thể xoá đi. Để cài lại từ đầu (Mọi thông tin đều được khai báo trong package.json)
- `package.json` là file khai báo thông tin, tên thư viện cần dùng, phiên bản của nó, ...
- Không được xoá file `package.json`;
- Nếu xoá `node_modules` -> chạy lại câu lệnh `npm install` để npm đọc file package.json và cài tải lại các thư viện cho mình
- Mình dùng thư viện A. Thư viện A lại dùng thêm 10 thư viện khác. Đôi khi 10 thư viện khác lại dùng thêm n thư viện khác nữa -> Coi trong `node_modules` có rất nhiều folder chứa thư viện
- Khi gửi code cho người, nộp bài, ... KHÔNG BAO GIỜ nộp `node_modules`
- Từ khoá `render`: Ráp dữ liệu vào HTML -> Xuất ra cấu trúc DOM
- Từ khoá `component`: Những thành phần trong trang Web để gộp lại thành một Website hoàn chỉnh
- Không có bất kỳ giới hạn nào cho một `component`. Có thể là một file js chỉ chứa duy nhất thẻ

  . Thậm chí nó có thể là một file js chưa một mã nguồn html cực lớn.
- Từ khoá `JSX`: Có thể viết cú pháp của HTML bên trong Javascript (Biên dịch ngược lại Javascript thuần)
- Tools chuyển đổi cú pháp HTML -> JSX: https://magic.reactjs.net/htmltojsx.htm
- Folder `public` là nơi chứa tất cả những file, thư mục có thể truy xuất bằng đường dẫn
    - Khi gõ `http://localhost:3000/` -> Tự hiểu mình tìm một cái file gì đó ở trong folder public
- State: Trạng thái -> Có thể thay đổi được -> Chính dữ liệu riêng của mỗi component
- Props: Property -> Thuộc tính của component

```
<!--  title là attribute trong HTML -->
<p title="Tieu de cua the p">Noi dung cua the p</p>
```

```
<Header siteTitle="Tieu de Website" >Noi dung ben trong</Header>
<ComponentHeader>Noi dung ben trong</ComponentHeader>
```

- Cách Lưu trữ dữ liệu trong Component
- Cách chuyển đổi dữ liệu qua lại giữa các Component với nhau
- Chuyển đổi sự kiện qua lại giữa các Component với nhau
- Props Down - Event Up (Hàm xử lý click, ...)
- Khái niệm `render` và `rerender`
- `render` trích xuất HTML từ JSX và các biến liên quan -> DOM
- `rerender` làm lại công việc render một lần nữa với dữ liệu mới
- Làm sao React biết được khi nào mình thay đổi dữ liệu??
- Trong React, để thay đổi State -> Bắt buộc phải do chính nó nắm giữ mới thay đôi -> Tránh bị lỗi sau này