# 1. Giới thiệu đề tài, mục tiêu, và phân chia bài báo cáo.

## 1.1 Tiêu Đề và Tóm Tắt:

*Tiêu đề*: "Khám phá React.js: Tạo ứng dụng web tương tác hiệu quả"

*Tóm tắt*: Bài báo cáo này sẽ giới thiệu về React.js, một thư viện JavaScript phát triển web phổ biến và mạnh mẽ. Chúng ta sẽ tìm hiểu về lý do tại sao React.js đã trở thành một công cụ quan trọng trong phát triển ứng dụng web và cách cài đặt nó. Bài báo cáo sẽ cung cấp hướng dẫn cụ thể về cách tạo các thành phần, quản lý trạng thái, thực hiện điều hướng, và xử lý sự kiện bằng React. Chúng ta cũng hứa hẹn rằng bạn sẽ ra khỏi buổi seminar này với kiến thức cơ bản về React.js và khả năng tạo các ứng dụng web tương tác đơn giản.

## 1.2 Mục Tiêu:

Mục tiêu của bài báo cáo này là:

* Giới thiệu React.js và lý do tại sao nó quan trọng trong phát triển web.
* Hướng dẫn cài đặt môi trường phát triển React.js.
* Cung cấp hướng dẫn cụ thể về tạo React components, quản lý trạng thái, điều hướng, và xử lý sự kiện.
* Đảm bảo rằng người tham dự buổi seminar sẽ có kiến thức cơ bản để bắt đầu phát triển ứng dụng React đơn giản.

## 1.3 Giới Thiệu Đề Tài:

React.js đã trở thành một công cụ phát triển web quan trọng bởi vì nó cung cấp một cách hiệu quả để xây dựng các ứng dụng web tương tác. React.js tập trung vào việc tạo lại giao diện người dùng khi trạng thái của ứng dụng thay đổi, giúp tạo ra trải nghiệm người dùng mượt mà và hiệu quả. Trong bài báo cáo này, chúng ta sẽ khám phá các khái niệm cơ bản của React.js và cách bắt đầu sử dụng nó để phát triển ứng dụng web.

## 1.4 Phân Chia Bài Báo Cáo:

Bài báo cáo này sẽ được chia thành các phần chính sau:

* Phần 1: Giới thiệu đề tài, mục tiêu, và phân chia bài báo cáo.
* Phần 2: Nghiên Cứu React.js.
* Phần 3: Chuẩn Bị Nội Dung về React.js.
* Phần 4: Cài Đặt React.js.
* Phần 5: Hướng Dẫn Thực Hiện Các Ví Dụ và Dự Án.

# 2. Nghiên cứu React.js.

## 2.1 Giới Thiệu về React.js

React.js, hay đơn giản là React, là một thư viện JavaScript phát triển web phổ biến được phát triển bởi Facebook. Nó đã trở thành một công cụ quan trọng trong phát triển ứng dụng web do các ưu điểm vượt trội của nó trong việc quản lý giao diện người dùng và cải thiện hiệu suất ứng dụng web.

Tại sao React.js quan trọng và có giá trị trong việc nghiên cứu và sử dụng?

***Lý Do Tại Sao React Quan Trọng:***

* **Thiết Kế Giao Diện Tương Tác**: React được thiết kế để tạo ra các ứng dụng web tương tác. Với khả năng quản lý giao diện người dùng và cập nhật nhanh chóng trạng thái của ứng dụng, React giúp tạo ra trải nghiệm người dùng mượt mà và hiệu quả.
* **Thư Viện, Không Phải Khung Làm Việc**: React không phải là một khung làm việc (framework), mà chỉ là một thư viện chuyên tập trung vào việc xây dựng giao diện người dùng. Điều này cho phép bạn linh hoạt lựa chọn các công nghệ khác để hoàn thiện ứng dụng.
* **Cộng Đồng Lớn**: Cộng đồng phát triển và hỗ trợ React rất lớn, điều này đảm bảo rằng bạn có sự hỗ trợ và tài liệu dồi dào khi làm việc với nó.
* **Sử Dụng Rộng Rãi**: React đã được sử dụng trong nhiều dự án lớn và bởi các công ty nổi tiếng như Facebook, Instagram, Airbnb, Uber, và nhiều khác. Điều này chứng tỏ tính ổn định và khả năng mở rộng của React.
* **Hiệu Suất Cao**: React cung cấp một cơ chế cập nhật giao diện người dùng hiệu quả thông qua việc sử dụng Virtual DOM. Điều này giúp giảm thiểu các thay đổi không cần thiết trong giao diện, dẫn đến hiệu suất tốt.

***Ví Dụ về Sử Dụng React:***

* **Facebook**: Facebook là một trong những công ty sáng lập React và sử dụng nó rộng rãi cho các phần của trang web của họ.
* **Instagram**: Instagram, một ứng dụng chia sẻ ảnh và video phổ biến, cũng sử dụng React để xây dựng giao diện người dùng.
* **Airbnb**: Airbnb, một dịch vụ đặt phòng trực tuyến, đã sử dụng React để tạo ra trang web của họ và ứng dụng di động.
* **Netflix**: Netflix đã sử dụng React để xây dựng một số phần của trang web và ứng dụng của họ để cung cấp trải nghiệm xem phim tốt hơn.
* **Uber**: Uber đã sử dụng React để xây dựng các phần giao diện người dùng của ứng dụng di động và trang web của họ để cung cấp dịch vụ đặt xe.

## 2.2 Lịch Sử và Phát Triển

React.js, phát triển bởi Facebook, đã trải qua một quá trình phát triển liên tục và nhanh chóng kể từ khi ra mắt ban đầu. Lịch sử và sự phát triển của React.js có thể được chia thành các giai đoạn chính, với mỗi giai đoạn đánh dấu bằng việc giới thiệu các phiên bản mới và tính năng quan trọng.

***Lịch Sử và Các Phiên Bản Chính:***

1. **Ban Đầu (2013)**: React.js đã được giới thiệu lần đầu vào năm 2013. Tại thời điểm này, React tập trung vào việc tạo ra các thành phần giao diện người dùng tái sử dụng, giúp cải thiện quản lý trạng thái của ứng dụng.
2. **React Native (2015)**: React Native, một phiên bản dành cho phát triển ứng dụng di động, đã được giới thiệu vào năm 2015. Điều này cho phép phát triển ứng dụng di động cho cả iOS và Android bằng cách sử dụng React.js.
3. **React 16 (2017)**: React 16 đã giới thiệu Virtual DOM cải tiến, làm tăng hiệu năng và khả năng xử lý các thành phần lớn. Các tính năng như Portals và Fragment cũng đã được giới thiệu tại phiên bản này.
4. **React 17 (2020)**: React 17 tập trung vào việc cải thiện khả năng tương thích ngược với phiên bản trước đó. Phiên bản này không đưa ra nhiều tính năng mới mà tập trung vào việc giảm thiểu sự thay đổi trong mã nguồn hiện có khi nâng cấp React.
5. **Phiên bản hiện tại của React (2023) là 18.2.0, được phát hành vào ngày 10 tháng 10 năm 2023. Một số tính năng và cải tiến mới của phiên bản này bao gồm: Concurrent Mode, Suspense, Server Components, Hooks, và Error Boundaries.**

***Các Tính Năng Quan Trọng:***

* **Virtual DOM**: React sử dụng Virtual DOM để cải thiện hiệu năng và làm cho việc cập nhật giao diện người dùng trở nên hiệu quả hơn.
* **JavaScript XML (JSX)**: JSX cho phép viết mã JavaScript và HTML cùng một lúc, giúp xây dựng giao diện dễ dàng hơn.
* **React Router**: Thư viện React Router giúp quản lý điều hướng trong ứng dụng React.
* **Redux**: Redux là một thư viện quản lý trạng thái (state management) thường được sử dụng cùng với React để quản lý dữ liệu ứng dụng.
* **Hooks**: React Hooks đã giới thiệu một cách mới để quản lý trạng thái và các tính năng trong các component hàm.
* **React Fiber**: Fiber là một phiên bản tái thiết của React, thiết kế lại cơ chế cập nhật giao diện người dùng để cải thiện hiệu suất và khả năng ứng phó với các tác vụ ưu tiên khác nhau.
* **Server-side Rendering (SSR)**: React hỗ trợ SSR, cho phép tạo ra các ứng dụng có khả năng tạo ra giao diện trên máy chủ trước khi gửi đến trình duyệt, cải thiện SEO và hiệu suất của ứng dụng.
* **Hỗ Trợ Lâu Dài**: React nhận được hỗ trợ lâu dài và cải tiến liên tục từ cộng đồng React và Facebook.

Thông qua việc tìm hiểu lịch sử và phát triển của React.js, bạn sẽ hiểu rõ hơn về cách nó đã phát triển thành một thư viện phát triển web mạnh mẽ và quan trọng trong ngành công nghệ.

## 2.3 Các Khái Niệm Cơ Bản

Trong phần này, chúng ta sẽ tìm hiểu về các khái niệm quan trọng trong React.js, bao gồm components, props, state, và JSX. Chúng ta cũng sẽ hiểu cách React hoạt động dựa trên mô hình "one-way data flow" (luồng dữ liệu một chiều), một trong những đặc điểm quan trọng của React.

***Components (Thành Phần):***

* **Giới thiệu**: Component là một khái niệm quan trọng trong React. Chúng là các phần tử độc lập của giao diện người dùng và có khả năng tự quản lý trạng thái và hiển thị dữ liệu.
* **Loại Component**: Có hai loại component chính trong React: functional components và class components. Functional components là hàm JavaScript, trong khi class components là các lớp JavaScript.
* **Render Method**: Mỗi component cần phải có một phương thức **render()**, trong đó chúng định nghĩa giao diện người dùng của component.

***Props (Thuộc Tính):***

* **Giới thiệu**: Props (short for "properties") là các tham số truyền vào một component. Chúng cho phép truyền dữ liệu từ component cha đến component con.
* **Sử dụng**: Để truyền props cho một component, bạn sử dụng cú pháp **<ComponentName propName={value} />**. Trong component con, bạn có thể truy cập giá trị của props thông qua **this.props.propName**.

***State (Trạng Thái):***

* **Giới thiệu**: State là một trong những trạng thái nội tại của một component. Nó biểu thị dữ liệu thay đổi theo thời gian và có ảnh hưởng đến việc hiển thị giao diện.
* **setState()**: Để cập nhật trạng thái, bạn sử dụng phương thức **setState()** để thông báo cho React cập nhật giao diện người dùng.
* **Quản Lý Trạng Thái**: React khuyến nghị quản lý trạng thái tại mức càng cao càng tốt, tránh lưu trữ trạng thái tại nhiều thành phần khác nhau.

***JSX (JavaScript XML):***

* **Giới thiệu**: JSX là một phần quan trọng của React. Nó cho phép viết mã HTML tương tự trong JavaScript, giúp xây dựng giao diện người dùng dễ dàng hơn.
* **Biên Dịch JSX**: JSX cần phải được biên dịch thành JavaScript thông qua trình biên dịch như Babel trước khi nó có thể chạy trên trình duyệt.

***Mô Hình "One-Way Data Flow" (Luồng Dữ Liệu Một Chiều):***

* **Giới thiệu**: Mô hình "one-way data flow" là nguyên tắc cơ bản của React. Nó đề xuất rằng dữ liệu chảy theo một hướng từ component cha đến component con, không có sự truyền ngược dữ liệu từ con lên cha.
* **Ưu Điểm**: Mô hình này giúp đơn giản hóa việc theo dõi dữ liệu và dễ dàng debug. Nó cũng làm cho ứng dụng dễ bảo trì và dễ mở rộng.
* **Lưu ý**: Trong mô hình này, khi trạng thái thay đổi, React cập nhật giao diện người dùng thay vì cập nhật trực tiếp DOM.

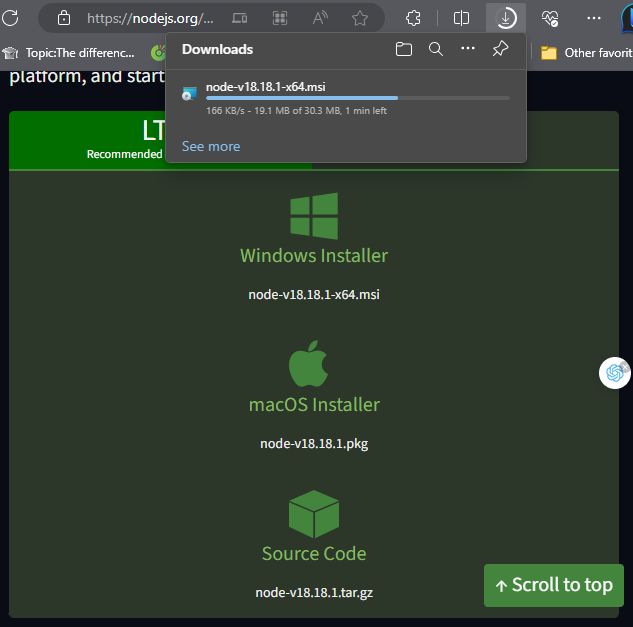
Các khái niệm này là nền tảng của việc làm việc với React.js, và hiểu chúng là quan trọng để xây dựng các ứng dụng web hiệu quả và dễ bảo trì.

# 4. Cài Đặt React.js.

## 4.1 Cài Đặt Node.js và npm

*Hướng dẫn chi tiết cách cài đặt Node.js và npm (Node Package Manager) trên máy tính của người dùng:*

* **Cài Đặt Node.js**: Node.js là môi trường chạy mã JavaScript phía máy chủ. Để cài đặt Node.js, bạn cần truy cập trang chính thức của Node.js tại [Download | Node.js (nodejs.org)](https://nodejs.org/en/download) và tải xuống phiên bản LTS (Long-Term Support) dành cho hệ điều hành của bạn (Windows, macOS, hoặc Linux).



A screenshot of a computer

Description automatically generatedA screenshot of a software license agreement

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated

**- Cài Đặt npm (Node Package Manager)**: npm là trình quản lý gói (package manager) dùng để cài đặt các thư viện và công cụ phát triển. Sau khi cài đặt Node.js, npm sẽ được cài đặt tự động. Để kiểm tra phiên bản npm, bạn có thể mở cửa sổ dòng lệnh và chạy lệnh **npm -v**.

A screenshot of a computer program

Description automatically generated

## 4.2 Sử Dụng Create React App

*Hướng dẫn cụ thể về cách sử dụng Create React App để tạo một ứng dụng React.js cơ bản:*

### Phần 1: Giải Thích Cách Create React App Hoạt Động.

1. **Giới Thiệu Create React App:**
   * CRA (Create React App) là một công cụ dòng lệnh được phát triển bởi cộng đồng React để tạo một môi trường phát triển React nhanh chóng và dễ dàng.
   * CRA giúp loại bỏ quá trình cấu hình ban đầu phức tạp và cho phép bạn tập trung vào việc phát triển ứng dụng React thay vì cấu hình công cụ.
2. **Cấu Hình Tự Động:**
   * CRA tự động cấu hình các công cụ cần thiết cho phát triển React như Webpack, Babel, và ESLint.
   * Webpack được sử dụng để đóng gói tất cả các tệp và phụ thuộc trong dự án thành một tệp JavaScript hoặc một số tệp tĩnh.
   * Babel được sử dụng để biên dịch mã nguồn JavaScript và JSX thành mã JavaScript chuẩn để chạy trên trình duyệt.
   * ESLint giúp kiểm tra và đảm bảo mã nguồn tuân thủ các quy tắc và tiêu chuẩn cụ thể.
3. **Cơ Chế Làm Việc:**
   * CRA sử dụng lệnh **npx create-react-app** để tạo một dự án React mới.
   * CRA tạo một thư mục dự án với cấu trúc và tệp cấu hình sẵn sàng.
   * CRA cung cấp một máy chủ phát triển để bạn có thể xem trang web React trong quá trình phát triển và tự động cập nhật khi bạn lưu thay đổi.
   * CRA quản lý các phụ thuộc và bao gồm một số script npm để biên dịch, chạy và triển khai ứng dụng.
4. **Tùy Chỉnh và Cấu Hình Bổ Sung:**
   * CRA cho phép bạn tùy chỉnh và cấu hình dự án của mình nếu cần thiết.
   * Bạn có thể thêm các plugin hoặc cấu hình riêng của mình để mở rộng tính năng của dự án.
   * CRA không bắt buộc bạn sử dụng cấu hình mặc định và cho phép bạn thích nghi với nhu cầu cụ thể của dự án.

Bằng cách sử dụng Create React App, bạn có thể bắt đầu phát triển ứng dụng React một cách nhanh chóng mà không cần phải lo lắng về cấu hình và cài đặt ban đầu.

Bây giờ chúng ta hãy đến với phần cài đặt nào!!!

* + Cài đặt Create React App trên window:

A screen shot of a computer

Description automatically generated

- Tạo dự án React bằng cách chạy lệnh sau:A screenshot of a computer program

Description automatically generated A screenshot of a computer program

Description automatically generated

- Cấu trúc thư mực dự án:

A computer screen with white text

Description automatically generated

- Khởi động máy chủ phát triển (develop server) bằng CRA bằng câu lệnh “npm start”, Lệnh trên sẽ mở trình duyệt và chạy ứng dụng React của bạn trên địa chỉ [http://localhost:3000](http://localhost:3000/)

A screenshot of a computer

Description automatically generated

**5.1 Tạo Component Đầu Tiên**

*Hướng dẫn tạo một ứng dụng React.js đơn giản với một component đầu tiên:*

* **Bước 1: Tạo Dự Án React Mới**
  + Sử dụng Create React App hoặc tạo dự án React từ đầu, tùy theo sự lựa chọn của bạn.
  + Nếu bạn sử dụng Create React App, mở cửa sổ dòng lệnh và chạy lệnh: **npx create-react-app my-first-app**.
* **Bước 2: Tạo Component**
  + Một component trong React có thể là một hàm (functional component) hoặc một lớp (class component). Dưới đây là một ví dụ về cách tạo một functional component:

A screenshot of a computer program

Description automatically generated**Bước 3: Sử Dụng Component**

* Để sử dụng component, bạn cần import nó vào file chính của ứng dụng. Ví dụ, nếu bạn muốn sử dụng **MyFirstComponent** ở file **App.js**, bạn cần import nó:

**A screenshot of a computer program

Description automatically generatedBước 4: Hiển Thị Component**

* + Component **MyFirstComponent** đã được sử dụng trong **App.js**, và nó sẽ hiển thị nội dung "Hello, World!" trong ứng dụng của bạn.

A screenshot of a computer

Description automatically generated

Hoặc bạn có thể giữ nguyên chương trình ban đầu của App.js sau đó thêm:

A black background with white text

Description automatically generated

Và <MyFirstComponent /> vào hàm Function App() ta sẽ có nội dung được hiển thị trong ứng dụng là:

A screenshot of a computer

Description automatically generated

Phần này sẽ giúp người đọc hiểu cách tạo một component đầu tiên trong React và cách sử dụng nó trong ứng dụng.

**5.2 Truyền Dữ Liệu thông qua Props**

*Hướng dẫn cách truyền dữ liệu từ một component cha đến component con bằng cách sử dụng props:*

**Truyền Props từ Cha đến Con**: Trong React, bạn có thể truyền dữ liệu từ component cha đến component con bằng cách sử dụng thuộc tính **props**. Bạn cần định nghĩa các props trong component cha và sau đó truyền chúng vào component con.

* + 1. Tạo Component Cha và Component Con.

A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

* + 1. Truyền Props từ Cha đến Con.

Trong component cha (**ParentComponent.js**), bạn định nghĩa dữ liệu bạn muốn truyền vào component con. Trong ví dụ trên, dữ liệu được đặt tên là **dataToPass** và được truyền vào component con (**ChildComponent.js**) thông qua props.

* + 1. Sử dụng Props trong Component Con.

Trong component con (**ChildComponent.js**), bạn sử dụng dữ liệu được truyền qua props. Trong ví dụ trên sử dụng **this.props.data** để truy cập dữ liệu từ component cha.

* + 1. Hiển thị Component Cha và Component Con.

A screen shot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

**5.3 Quản Lý Trạng Thái với State**

*Giới thiệu khái niệm về state trong React và cách quản lý trạng thái của một component:*

* **Trạng Thái trong React**: Trạng thái (state) là dữ liệu nội tại của một component. Nó biểu thị các giá trị có thể thay đổi theo thời gian và ảnh hưởng đến hiển thị của component.
* **Cập Nhật Trạng Thái với setState()**: Trong React, bạn sử dụng phương thức **setState()** để cập nhật trạng thái. Khi trạng thái thay đổi, React sẽ tự động cập nhật giao diện người dùng.
* **Ví dụ về Quản Lý Trạng Thái**:

A screen shot of a computer code

Description automatically generated

Kết quả sẽ là:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**5.4 Điều Hướng và Routing**

*Giới thiệu về React Router, một thư viện để quản lý điều hướng trang trong ứng dụng React:*

* **React Router**: React Router là một thư viện phổ biến trong cộng đồng React để quản lý điều hướng (routing) trong ứng dụng web. Nó cho phép bạn tạo các route và liên kết giữa các trang của ứng dụng.
* **Cài Đặt và Sử Dụng React Router**:

**1**. **Cài đặt React Router**:

A screenshot of a computer program

Description automatically generated

**2**. **Tạo 2 component Home và About**

A screen shot of a computer program

Description automatically generated

A screen shot of a computer code

Description automatically generated

**3**. **Tiếp theo, trong file App.js, có thể sử dụng React Router như sau:**

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

* + Trong ví dụ trên, chúng ta có hai route (“/” và “/about”) và hai liên kết tương ứng trong thanh điều hướng.

**5.5 Xử Lý Sự Kiện**

* **Xử Lý Sự Kiện trong React**: Trong React, xử lý sự kiện là quá trình xác định và đáp ứng các tương tác của người dùng với ứng dụng. Các sự kiện bao gồm như **onClick**, **onChange**, **onSubmit**, và nhiều loại khác. Điều này cho phép bạn tạo ứng dụng tương tác và đáp ứng.
* **Hướng Dẫn Cụ Thể**: Hướng dẫn cụ thể về cách xử lý các loại sự kiện phổ biến nhất trong React:
  + **Xử Lý Sự Kiện Click (onClick)**: Hướng dẫn cách tạo một sự kiện click cho một nút hoặc thành phần khác và thực hiện các hành động khi người dùng nhấn vào đó.

A computer screen with text and images

Description automatically generated

* + **Xử Lý Sự Kiện Thay Đổi (onChange)**: Hướng dẫn cách xử lý sự kiện thay đổi, chẳng hạn như nhập liệu từ người dùng trong các ô văn bản hoặc ô chọn.

A screen shot of a computer code

Description automatically generated

* + 2 SỰ KIỆN TRÊN MÌNH ĐÃ TỪNG DEMO Ở PHẦN STATE
  + **Xử Lý Sự Kiện Gửi Biểu Mẫu (onSubmit)**: Hướng dẫn cách xử lý sự kiện gửi biểu mẫu khi người dùng bấm nút Gửi trên biểu mẫu.

A computer screen shot of code

Description automatically generated

KẾT QUẢ

A screenshot of a computer

Description automatically generated