TRƯỜNG ĐẠI HỌC THỦY LỢI

**KHOA CÔNG NGHỆ THÔNG TIN**

A blue and white logo

AI-generated content may be incorrect.

**NỀN TẢNG PHÁT TRIỂN WEB**

***Đề tài:***

**Khởi tạo dự án & Cấu trúc Components**

**Nhóm sinh viên thực hiện: Nhóm 14**

**Thành viên nhóm: Khuất Văn Trường**

**Giảng viên phụ trách môn học: Tạ Chí Hiếu**

# **MỤC LỤC**

[**MỤC LỤC** 2](#_Toc193318515)

[**LỜI MỞ ĐẦU** 3](#_Toc193318516)

[**NỘI DUNG BÁO CÁO** 4](#_Toc193318517)

[**I. KHỞI TẠO DỰ ÁN** 4](#_Toc193318518)

[**1.1. Khái niệm React** 4](#_Toc193318519)

[**1.2. Nguyên lý Virtual DOM** 4](#_Toc193318520)

[**1.3. Cú pháp JSX** 5](#_Toc193318521)

[**1.4. Cách khởi tạo dự án** 6](#_Toc193318522)

[**II. CẤU TRÚC COMPONENTS** 6](#_Toc193318523)

[**1.1. Khái niệm Components** 6](#_Toc193318524)

[**1.2. Tính tái sử dụng của Components** 7](#_Toc193318525)

[**III. DỰ ÁN MINH HỌA** 8](#_Toc193318526)

# **LỜI MỞ ĐẦU**

Bài báo cáo nhằm xác định giải thích **React là gì**, nguyên lý **Virtual DOM**, cú pháp JSX, cách **khởi tạo dự án**, khái niệm Component (function component vs class component), tính tái sử dụng của component. Bên cạnh đó cũng Demo chức năng liên quan đến đề tài này.

# **NỘI DUNG BÁO CÁO**

# **I. KHỞI TẠO DỰ ÁN**

## **1.1. Khái niệm React**

* React là thư viện JavaScript mã nguồn mở, được phát triển bởi Facebook (Meta) vào năm 2013, với mục tiêu dùng công nghệ để tạo ra giao diện người dùng dễ dàng hơn. Với nền tảng này, các lập trình viên có thể tạo ra các thành phần mà dùng lại được và tăng khả năng tương tác. React chủ yếu dùng để xây dựng giao diện người dùng (UI) theo mô hình component.
* **Dễ học, dễ sử dụng:** React có cú pháp gần giống với JavaScript thuần, dễ tiếp cận.  
   **Tái sử dụng code:** Giao diện được chia nhỏ thành các **component** có thể dùng lại nhiều lần.  
   **Hiệu suất cao:** Nhờ cơ chế **Virtual DOM**, React cập nhật UI nhanh hơn các thư viện khác.  
   **Hỗ trợ mạnh từ cộng đồng:** Là một trong những thư viện UI phổ biến nhất thế giới.
* React cũng áp dụng mô hình "Virtual DOM" để tối ưu hiệu suất và cải thiện trải nghiệm người dùng. React có khả năng tích hợp với các thư viện và công cụ phổ biến như Redux, giúp quản lý dữ liệu của ứng dụng một cách dễ dàng và hiệu quả hơn.

## **1.2. Nguyên lý Virtual DOM**

DOM là viết tắt của chữ Document Object Model, dịch ra là mô hình các đối tượng trong tài liệu HTML. Thông qua mô hình DOM ta có thể truy xuất đến các thẻ HTML một cách dễ dàng. Khi có thay đổi trong UI, trình duyệt phải cập nhật lại toàn bộ DOM, gây tốn tài nguyên.

**Virtual DOM giúp cải thiện hiệu suất như thế nào?**

**Virtual DOM** là một bản sao của DOM thật, được lưu trong bộ nhớ.  
 Khi UI thay đổi, React sẽ tạo một bản Virtual DOM mới và so sánh với bản cũ (**diffing algorithm**).  
 Chỉ những phần bị thay đổi mới được cập nhật trên DOM thật (**reconciliation process**), giúp cải thiện hiệu suất.

Cách hoạt động:

1. **Khởi tạo:** Ban đầu, React tạo một cây Virtual DOM hoàn chỉnh dựa trên trạng thái ban đầu của ứng dụng.
2. **Render Component:** Khi trạng thái của ứng dụng thay đổi (ví dụ: người dùng nhấn nút), React tạo ra một cây Virtual DOM mới cho toàn bộ ứng dụng.
3. **So sánh:** React so sánh cây Virtual DOM mới với cây Virtual DOM cũ bằng cách sử dụng thuật toán diffing. Diffing là quá trình tìm ra sự khác biệt giữa hai cây và xác định những thay đổi cần phải thực hiện trên DOM thực tế để cập nhật nó.
4. **Cập nhật DOM:** Sau khi xác định được sự khác biệt, React chỉ cập nhật các phần tử DOM thực tế mà cần thay đổi. Thay vì cập nhật toàn bộ trang web, React chỉ cập nhật những phần tử thay đổi. Điều này giúp tối ưu hóa hiệu suất và làm giảm thời gian cần thiết để vẽ lại trang

**Ví dụ:** Nếu người dùng nhập văn bản vào một ô input, thay vì cập nhật toàn bộ trang, React chỉ cập nhật ô input đó.

## **1.3. Cú pháp JSX**

JSX (**JavaScript XML**) là một cú pháp mở rộng của JavaScript, cho phép viết HTML trong file JavaScript.

Giúp code dễ đọc hơn, viết gọn gàng hơn.  
 Kết hợp HTML và JavaScript dễ dàng.  
 JSX không phải HTML, mà được **biên dịch** thành JavaScript thuần trước khi chạy.

## **1.4. Cách khởi tạo dự án**

**a) Sử dụng Create React App (CRA)**

Đây là cách truyền thống và dễ sử dụng nhất để tạo ứng dụng React:

npx create-react-app my-app

cd my-app

npm start

**b) Sử dụng Vite (nhẹ hơn CRA, phổ biến hiện nay)**

npm create vite@latest my-app

cd my-app

npm install

npm run dev

**c) Sử dụng Next.js (dành cho dự án lớn, hỗ trợ SSR)**

npx create-next-app@latest my-app

cd my-app

npm run dev

# **II. CẤU TRÚC COMPONENTS**

## **1.1. Khái niệm Components**

* Component là thành phần cơ bản giúp chia nhỏ giao diện người dùng thành các phần nhỏ có thể tái sử dụng

**Function Component**

* Function Component được viết bằng cú pháp hàm thông thường của JavaScript.
* Là một hàm JavaScript thông thường, không có this.
* Trả về các phần tử JSX ngay từ bên trong hàm.
* Không có state mặc định, nhưng có thể sử dụng useState từ React Hooks để quản lý trạng thái.
* Thường được sử dụng cho các thành phần đơn giản, không có nhiều logic.  
  **Class Component**
* Class Component được viết bằng cú pháp của ES6 class.
* Yêu cầu phải kế thừa từ lớp React.Component.
* Sử dụng phương thức render() để trả về các phần tử JSX.
* Có thể sử dụng các tính năng bổ sung như state và lifecycle methods.
* Thường được sử dụng cho các thành phần có trạng thái phức tạp và cần quản lý nhiều logic.

## **1.2. Tính tái sử dụng của Components**

* Dễ bảo trì: Sửa một nơi, thay đổi khắp nơi.
* Tổ chức code tốt hơn: Phân chia giao diện thành các phần nhỏ, dễ quản lý.
* Tối ưu hiệu suất: Hạn chế việc render lại không cần thiết

# **III. DỰ ÁN MINH HỌA**

A screenshot of a computer

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.