

  
  
**Báo Cáo học phần  
Lập trình giao diện Web nâng cao**

***ĐỀ TÀI***

**<<Lập trình giao diện web nâng cao>>**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | **Th.S Trịnh Văn Chung** |
| **Sinh viên thực hiện:** | **Nguyễn Trực Kiên** |
| **Mã sinh viên:** | **2210900033** |
| **Các thành viên:** |  |
| **Lớp:** |  |
| **Khóa:** | **CNTT** |

**@FITNTU-** 09/2025

**MỤC LỤC**

Contents

[**DANH MỤC CÁC TỪ VIẾT TẮT VÀ THUẬT NGỮ** 4](#_Toc209303604)

[**CHƯƠNG 1: NGHIÊN CỨU CƠ SỞ LÝ THUYẾT** 1](#_Toc209303605)

[1.1. Tổng quan về Reactjs 1](#_Toc209303606)

[Tổng quan về ngôn ngữ lập trình Javascript 1](#_Toc209303607)

[1.1.1. Thư viện Jquery 1](#_Toc209303608)

[1.1.2. Framework Bootstrap 2](#_Toc209303609)

[1.2. Một số khái niệm quan trọng trong ReactJS 2](#_Toc209303610)

[1.2.1. Virtual DOM 2](#_Toc209303611)

[1.2.2. JSX 2](#_Toc209303612)

[1.2.3. Components 2](#_Toc209303613)

[1.2.4. Props và State 2](#_Toc209303614)

[1.2.5. React Lifecycle 3](#_Toc209303615)

[1.2.6. Event – Form – List & Key 3](#_Toc209303616)

[1.2.7. React Hooks 3](#_Toc209303617)

[1.2.8. Class Components 3](#_Toc209303618)

[1.2.9. Function Components 4](#_Toc209303619)

[1.2.10. React Router 4](#_Toc209303620)

[**CHƯƠNG 2 CHƯƠNG TRÌNH ỨNG DỤNG** 7](#_Toc209303621)

[2.1. Đặt vấn đề bài toán 7](#_Toc209303622)

[2.2. Hệ thống đề xuất 7](#_Toc209303623)

[2.3. Chức năng của hệ thống 7](#_Toc209303624)

[2.4. Xây dựng phần mềm 7](#_Toc209303625)

[2.4.1. Một số mã nguồn chương trình 7](#_Toc209303626)

[2.4.2. Một số giao diện người dùng và chức năng cụ thể 7](#_Toc209303627)

[1. Giao diện trang chủ 7](#_Toc209303628)

[2. Giao diện trang quản trị 7](#_Toc209303629)

[3. ….. 7](#_Toc209303630)

[**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN** 8](#_Toc209303631)

[3.1. Kết luận 8](#_Toc209303632)

[3.2. Hướng phát triển: 9](#_Toc209303633)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 10](#_Toc209303634)

# **DANH MỤC CÁC TỪ VIẾT TẮT VÀ THUẬT NGỮ**

# **CHƯƠNG 1: NGHIÊN CỨU CƠ SỞ LÝ THUYẾT**

## 1.1. Tổng quan về Reactjs

ReactJS là một thư viện JavaScript mã nguồn mở, được phát triển bởi Facebook từ năm 2013, dùng để xây dựng giao diện người dùng (User Interface – UI). React cho phép lập trình viên phát triển ứng dụng web theo hướng **component-based** (hướng thành phần), giúp tái sử dụng mã nguồn, dễ dàng bảo trì và mở rộng.

Đặc điểm nổi bật của ReactJS:

* **Virtual DOM**: Cơ chế so sánh và cập nhật thay đổi trong cây DOM ảo, giúp hiệu suất cao hơn so với việc thao tác trực tiếp trên DOM thật.
* **JSX (JavaScript XML)**: Cú pháp mở rộng cho phép viết HTML trong JavaScript, giúp mã nguồn trực quan và dễ đọc.
* **Tính chất component**: Chia ứng dụng thành nhiều thành phần nhỏ, độc lập, có thể tái sử dụng.
* **Quản lý state và props**: Giúp điều khiển luồng dữ liệu trong ứng dụng rõ ràng, hạn chế lỗi khi phát triển.
* **Hệ sinh thái mạnh mẽ**: Tích hợp nhiều thư viện hỗ trợ như Redux, React Router, Material UI,…

Nhờ các đặc điểm trên, ReactJS hiện là một trong những thư viện front-end phổ biến nhất, được sử dụng bởi nhiều công ty công nghệ lớn như Facebook, Instagram, Netflix, Airbnb,…

## Tổng quan về ngôn ngữ lập trình Javascript

JavaScript (JS) là ngôn ngữ lập trình kịch bản phía client, được Brendan Eich phát triển năm 1995. Ban đầu JavaScript chỉ dùng để xử lý các tác vụ đơn giản trên trình duyệt, nhưng đến nay đã trở thành ngôn ngữ đa năng, ứng dụng ở cả **front-end** và **back-end** (Node.js).

Đặc điểm chính của JavaScript:

* Hỗ trợ **lập trình hướng sự kiện** (event-driven).
* Có thể chạy trên nhiều nền tảng khác nhau.
* Cú pháp dễ học, tương tự C/Java.
* Kết hợp với HTML, CSS để tạo ra các ứng dụng web tương tác, hiện đại.

### 1.1.1. Thư viện Jquery

jQuery là thư viện JavaScript nổi tiếng, ra mắt năm 2006 bởi John Resig. Nó cung cấp cú pháp ngắn gọn, dễ hiểu để thao tác DOM, sự kiện, hiệu ứng và AJAX.

* **Ưu điểm**: Viết ít mã hơn, tăng tốc phát triển ứng dụng web.
* **Nhược điểm**: Dần trở nên lỗi thời khi các trình duyệt hiện đại đã hỗ trợ nhiều tính năng gốc mà jQuery cung cấp.

Khẩu hiệu nổi tiếng của jQuery là:   
👉 “Write less, do more” – Viết ít, làm nhiều.

### 1.1.2. Framework Bootstrap

Bootstrap là một framework front-end mã nguồn mở, do Twitter phát triển, lần đầu ra mắt năm 2011. Bootstrap giúp lập trình viên xây dựng giao diện nhanh chóng nhờ:

* Hệ thống **grid layout** (lưới chia cột).
* Các **thành phần UI dựng sẵn**: nút, form, menu, modal,…
* Tính năng **responsive design**: giao diện tự thích ứng trên nhiều kích thước màn hình.

Bootstrap đã trở thành công cụ phổ biến trong thiết kế web hiện đại, đặc biệt khi kết hợp với các thư viện JavaScript như ReactJS.

## 1.2. Một số khái niệm quan trọng trong ReactJS

### 1.2.1. Virtual DOM

Virtual DOM là bản sao của DOM thật, được React quản lý. Khi dữ liệu thay đổi, React sẽ so sánh Virtual DOM mới với bản cũ, từ đó chỉ cập nhật những phần tử thay đổi lên DOM thật. Điều này giúp tối ưu hiệu năng, giảm số lần thao tác DOM vốn rất tốn tài nguyên.

### 1.2.2. JSX

JSX là cú pháp mở rộng cho JavaScript, cho phép lập trình viên viết code giống HTML ngay trong file JS. JSX được trình biên dịch Babel dịch sang JavaScript thuần.

Ví dụ:

const element = <h1>Hello, React!</h1>;

### 1.2.3. Components

Component là khối xây dựng cơ bản trong React, có thể hiểu là một “mảnh” giao diện độc lập. Có hai loại:

* **Class Components**: Được khai báo dưới dạng class, có hỗ trợ lifecycle methods.
* **Function Components**: Viết dưới dạng hàm, thường kết hợp với Hooks để quản lý state và lifecycle.

### 1.2.4. Props và State

* **Props**: Thuộc tính được truyền từ component cha xuống component con (dữ liệu chỉ đọc).
* **State**: Trạng thái cục bộ bên trong component, có thể thay đổi bằng hàm setState hoặc useState.

### 1.2.5. React Lifecycle

Vòng đời của component trong React bao gồm các giai đoạn:

* Mounting (Khởi tạo)
* Updating (Cập nhật)
* Unmounting (Hủy bỏ)

### 1.2.6. Event – Form – List & Key

* **Event**: React cung cấp cách xử lý sự kiện tương tự DOM, nhưng sử dụng camelCase (ví dụ: onClick).
* **Form**: Quản lý dữ liệu nhập vào thông qua state hoặc controlled components.
* **List & Key**: Khi hiển thị danh sách, mỗi phần tử nên có key duy nhất để React tối ưu việc render.

### 1.2.7. React Hooks

Hooks là tính năng mới từ React 16.8, cho phép dùng state và các tính năng khác trong Function Component. Các hook thường dùng:

* useState (quản lý trạng thái)
* useEffect (quản lý vòng đời)
* useContext (quản lý dữ liệu toàn cục)

### 1.2.8. ****Class Components****

* Được viết dưới dạng **class** (ES6).
* Có thể sử dụng **state** và **lifecycle methods** (componentDidMount, componentDidUpdate, componentWillUnmount,…).
* Phải kế thừa từ React.Component.

**Ví dụ:**

import React, { Component } from "react";

class Welcome extends Component {

constructor(props) {

super(props);

this.state = { count: 0 };

}

componentDidMount() {

console.log("Component đã được render lần đầu");

}

render() {

return (

<div>

<h1>Xin chào, {this.props.name}</h1>

<p>Bạn đã bấm {this.state.count} lần</p>

<button onClick={() => this.setState({ count: this.state.count + 1 })}>

Bấm tôi

</button>

</div>

);

}

}

export default Welcome;

### 1.2.9. Function Components

* Được viết dưới dạng **hàm JavaScript**.
* Trước đây chỉ dùng để hiển thị UI (stateless).
* Từ React 16.8 trở đi, có thể sử dụng **Hooks** (như useState, useEffect) → nên Function Components ngày nay thay thế dần Class Components.

**Ví dụ:**

import React, { useState, useEffect } from "react";

function Welcome(props) {

const [count, setCount] = useState(0);

useEffect(() => {

console.log("Component đã render hoặc cập nhật");

}, [count]);

return (

<div>

<h1>Xin chào, {props.name}</h1>

<p>Bạn đã bấm {count} lần</p>

<button onClick={() => setCount(count + 1)}>Bấm tôi</button>

</div>

);

}

export default Welcome;

### 1.2.10. React Router

**Khái niệm:**  
React Router là một thư viện định tuyến dành cho ReactJS, giúp xây dựng các ứng dụng web đơn trang (Single Page Application – SPA). Với React Router, việc điều hướng giữa các trang trong ứng dụng sẽ diễn ra **mượt mà, không cần tải lại toàn bộ trang web**, từ đó nâng cao trải nghiệm người dùng.

**Chức năng chính:**

* **Routing**: Điều hướng giữa các trang trong ứng dụng (Home, About, Contact,…).
* **Dynamic Routing**: Cho phép sử dụng tham số động trong đường dẫn, ví dụ /products/:id.
* **Nested Routing**: Hỗ trợ lồng các route bên trong nhau, tổ chức ứng dụng theo cấu trúc module.
* **Navigation**: Cung cấp các component <Link>, <NavLink> để tạo menu điều hướng thay cho thẻ <a>.
* **Protected Route**: Cho phép kiểm soát quyền truy cập (ví dụ: chỉ người đã đăng nhập mới vào được trang quản trị).

**Cài đặt:**

npm install react-router-dom --save

**Ví dụ sử dụng cơ bản (React Router v6):**

import React from "react";

import { BrowserRouter as Router, Routes, Route, Link } from "react-router-dom";

function Home() {

return <h2>Trang chủ</h2>;

}

function About() {

return <h2>Giới thiệu</h2>;

}

function Contact() {

return <h2>Liên hệ</h2>;

}

export default function App() {

return (

<Router>

<div>

<nav>

<ul>

<li><Link to="/">Trang chủ</Link></li>

<li><Link to="/about">Giới thiệu</Link></li>

<li><Link to="/contact">Liên hệ</Link></li>

</ul>

</nav>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/about" element={<About />} />

<Route path="/contact" element={<Contact />} />

</Routes>

</div>

</Router>

);

}

**Ưu điểm:**

* Giúp tạo ứng dụng SPA mượt mà, không tải lại toàn trang.
* Dễ sử dụng, hỗ trợ đầy đủ tính năng điều hướng hiện đại.
* Có thể mở rộng với lazy loading để tối ưu hiệu suất.
* Tích hợp tốt với các tính năng khác của React như Hooks, Context API.

**Ứng dụng thực tế:**  
React Router được sử dụng rộng rãi trong các website và ứng dụng thương mại điện tử, hệ thống quản trị, blog, mạng xã hội,… nơi có nhiều trang và yêu cầu trải nghiệm điều hướng nhanh chóng.

# **CHƯƠNG 2 CHƯƠNG TRÌNH ỨNG DỤNG**

## Đặt vấn đề bài toán

<< Trình bày nội dung >>

## Hệ thống đề xuất

<< Trình bày nội dung >>

## Chức năng của hệ thống

* **Xem sản phẩm**
* ………….
* …………….
* ………………..
* ……………..
* **Đăng ký thành viên**

## Xây dựng phần mềm

## Một số mã nguồn chương trình

<< Các mã nguồn cơ bản copy code vào đây>>

## Một số giao diện người dùng và chức năng cụ thể

## Giao diện trang chủ

## Giao diện trang quản trị

## …..

# **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## Kết luận

Quá trình làm đồ án chuyên đề với đề tài "……" đã giúp em có cơ hội để trau dồi các kiến thức được học từ thầy cô trên giảng đường, đồng thời cũng tiếp thu và tìm hiểu được thêm nhiều kiến thức, kinh nghiệm mới, cụ thể là:

* + - Nắm được các quy trình nghiệp vụ trong phân tích và thiết kế một sản phẩm phần mềm.
    - Hiểu biết được một số phương pháp phân tích thiết kế hệ thống, từ đó đã áp dụng các hiểu biết của mình để phân tích thiết kế hệ thống bài toán theo mô hình ba lớp.
    - Tiếp thu được các kinh nghiệm thực tế khi thực hành xây dựng một sản phẩm cụ thể, áp dụng được các kiến thức đã học vào thực tiễn.
    - Củng cố các kiến thức về ngôn ngữ lập trình cũng như các thư viện – framework,…

**Ưu điểm:**

Hệ thống cơ bản đã hoàn thiện đầy đủ các tính năng đã được thiết kế và đưa vào sử dụng phục vụ các yêu cầu của người dùng. Quan trọng hơn, hệ thống đảm bảo duy trì được sự ổn định, tính bảo mật và tính tiện dụng cho người sử dụng. Các tính năng đã hoàn thành:

* + - ….
    - ….

**Nhược điểm:**

Trong những phiên bản tiếp theo, ứng dụng có thể được mở rộng để đáp ứng nhiều hơn những yêu cầu từ người sử dụng như:

* + - Giao diện của hệ thống do bản thân tự thiết kế nên vẫn chưa được hoàn thiện và đẹp mắt nhất.
    - Nhiều chức năng hoạt động chưa được mượt mà tạo hiệu ứng sử dụng không tốt với người dùng.
    - Trong thời gian tới em sẽ tiếp tục hoàn thiện các chức năng hiện thời và bổ sung thêm các chức năng mới nhằm mang lại sự tiện dụng nhất cho người dùng.

## Hướng phát triển:

Trong những phiên bản tiếp theo, ứng dụng có thể được mở rộng để đáp ứng nhiều hơn những yêu cầu từ người sử dụng như:

* + - Cải thiện, bảo trì những lỗi phát sinh ở các tính năng hiện tại.
    - Cải thiện giao diện thân thiện hơn với người dùng.
    - Phát triển thêm tính năng …..

# **DANH MỤC TÀI LIỆU THAM KHẢO**

#### **Tài liệu:**

#### **Website:**

1. W3C – *HTML5: A vocabulary and associated APIs for HTML and XHTML* (2021). Truy cập tại: <https://www.w3.org/TR/html5>; *Ngày truy cập: 25/08/2025*
2. Mozilla Developer Network (MDN) – *Introduction to HTML5*. Truy cập tại: <https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML>; *Ngày truy cập: 25/08/2025*