TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

BÁO CÁO GR1

Học tập và xây dựng trang web bán hàng bằng REACT

Sinh viên:
Vũ Thường Đạt
Mã số sinh viên:
20215031
Giảng viên hướng dẫn:
TS. Bùi Quốc Trung

1. Giới thiệu chung

Sau khi tìm hiểu về lập trình web frontend, em quyết định học React – một thư viện JavaScript phổ biến được sử dụng rộng rãi trong các dự án thực tế. Mục tiêu của em là nắm vững cách xây dựng giao diện người dùng hiện đại, tối ưu hóa trải nghiệm sử dụng, và rèn luyện tư duy phát triển phần mềm theo hướng component-based.

Em chọn làm trang web bán hàng vì đây là một dự án có tính thực tế cao, bao gồm nhiều chức năng cơ bản mà một ứng dụng web cần có như hiển thị danh sách dữ liệu, tương tác với người dùng, quản lý trạng thái, điều hướng giữa các trang, và xử lý các hành động như thêm vào giỏ hàng hay đặt hàng. Qua đó, em có thể áp dụng hầu hết những kiến thức đã học từ React vào một sản phẩm cụ thể.

Thời gian học và phát triển dự án

- **Thời gian học React:** khoảng 10 tuần, thông qua các khóa học trực tuyến, tài liệu chính thức và tự học qua tài nguyên trên internet. Cụ thể:

  Tuần 1–2: Ôn tập kiến thức HTML, CSS, JavaScript cơ bản và nâng cao (ES6+). Học các khái niệm như DOM, promise, async/await.

  Tuần 3–4: Làm quen với React:
  Hiểu cách cài đặt môi trường React bằng Vite và Create React App.
  Học về JSX, component (function component), props và state.

  Tuần 5–6:
  Tìm hiểu sâu về Hooks: `useState`, `useEffect`, `useRef`,...
  Học cách xử lý sự kiện, vòng đời component.

  Tuần 7–8:
  Làm việc với React Router để tạo nhiều trang (routing).
  Học cách quản lý dữ liệu

Tuần 9–10:
Làm quen với việc gọi API (sử dụng `fetch` hoặc `axios`).
Tìm hiểu cách tổ chức project và quản lý nhiều component.

- **Thời gian xây dựng dự án:** khoảng 6 tuần sau khi đã nắm kiến thức cơ bản.

  Tuần 1:

  - Phân tích yêu cầu chức năng của trang web bán hàng.
  - Thiết kế bố cục (UI), xác định các component cần dùng.
  - Tạo cấu trúc thư mục, khởi tạo project React.
  - Thiết kế database và kết nối phía backend

  Tuần 2 - 3:

  - Xây dựng trang chủ
  - Viết các api để lấy danh sách sản phẩm
  - Chức năng đăng nhập, đăng ký người dùng

  Tuần 4:

  - Thêm xóa sản phẩm với admin
  - Thêm xóa sản phẩm với user

  Tuần 5 + 6 +...:

  - Thống kê với admin
  - Bổ sung và hoàn thiện các chức năng
  - Fixbug

Tổng quan về sản phẩm

Sản phẩm em xây dựng là một **trang web bán hàng đơn giản**, cho phép người dùng:

- Xem danh sách sản phẩm.
- Lọc sản phẩm theo danh mục
- Thêm sản phẩm vào giỏ hàng
- Thêm/xóa sản phẩm vào giỏ hàng.
- Tính tiền và thanh toán(mô phỏng)
- Đăng ký và đăng nhập tài khoản

Với admin ngoài các chắc năng cơ bản có thể thêm, xóa sản phẩm, thống kê dữ liệu

Dự án sử dụng các công nghệ như

- **ReactJS**: Thư viện chính để xây dựng giao diện.
- **React Router DOM**: Điều hướng giữa các trang.
- **Zustand**: Quản lý trạng thái người dùng và giỏ hàng.
- **TailwindCSS**: Thiết kế giao diện bằng utility class.
- **Axios**: Gửi và nhận dữ liệu với backend thông qua API.

2. Dự án

2.1. Cấu trúc thư mục

frontend:

frontend/

├── public/ # Chứa các tài nguyên tĩnh

├── src/ # Thư mục chính chứa mã nguồn frontend

│ ├── assets/ # Hình ảnh, biểu tượng, font, v.v.

│ ├── components/ # Các component tái sử dụng như Navbar, Card,...

│ ├── lib/ # Chứa config chung axiosInstance

│ ├── pages/ # Chứa các trang chính

│ ├── stores/ # Trạng thái toàn cục dùng Zustand

│ ├── App.jsx # Thành phần gốc, định nghĩa route và layout chính

│ ├── main.jsx # Điểm bắt đầu render React

│ ├── App.css, index.css # CSS chung cho toàn ứng dụng

├── index.html # File HTML gốc (template chính)

├── tailwind.config.js # Cấu hình TailwindCSS

├── vite.config.js # Cấu hình Vite (trình build)

backend:

backend/

├── controllers/ # Chứa các hàm xử lý logic cho từng route (sản phẩm, người dùng, đơn hàng, v.v.)

├── middleware/ # Chứa middleware xác thực (auth)

├── models/ # Định nghĩa schema MongoDB bằng Mongoose (User, Product,...)

├── routes/ # Khai báo route cho API (vd: /api/products, /api/users,...)

├── uploads/ # Chứa tạm các file ảnh người dùng tải lên

├── .env # Biến môi trường: chuỗi kết nối DB, cổng chạy server,...

├── server.js # Điểm khởi chạy ứng dụng backend

2.2. Mô tả luồng hoạt động và các chức năng

Người dùng truy cập:

1. Truy cập trang chủ
2. Xem danh sách sản phẩm, danh mục. Khi ấn vào danh mục sẽ hiển thị các sản phẩm thuộc danh mục đó
3. Chọn thêm vào giỏ hàng
    - Nếu chưa đăng nhập -> Đăng nhập->3
    - Nếu chưa có tài khoản -> Đăng ký -> Đăng nhập -> 3
    - Đã đăng nhập -> Thêm sản phẩm vào giỏ hàng
4. Truy cập vào giỏ hàng
   -> Điều chỉnh, cập nhật giỏ hàng
5. Bấm "Thanh toán":
    - Nếu backend trả lỗi →quay về trang chủ
    - Nếu thành công → hiển thị "thanh toán thành công" và quay về trang chủ

Admin:

- Tương tự như user nhưng admin có thể truy cập dashboard ẩn
- Có thể thêm sản phẩm với các trường như tên, mô tả, giá, danh mục, hình ảnh
- Xem danh sách các sản phẩm hiện có, đánh dấu sản phẩm nổi bật, xóa sản phẩm
- Thống kê sẽ hiển thị số người dùng, số sản phẩm, số lượng đã bán, doanh thu

2.3. Quản lý state

Dự án sử dụng thư viện **Zustand** để quản lý trạng thái toàn cục (global state) cho các phần quan trọng như người dùng, giỏ hàng và sản phẩm. Zustand giúp viết code ngắn gọn, dễ đọc và dễ kiểm soát hơn so với Redux, đồng thời không yêu cầu boilerplate phức tạp.

a. Quản lý người dùng – useUserStore

Store useUserStore đảm nhiệm toàn bộ quá trình **xác thực người dùng**, gồm: đăng ký, đăng nhập, đăng xuất và kiểm tra phiên đăng nhập.

- **user**: Lưu thông tin người dùng hiện tại.
- **signup()**: Gửi yêu cầu đăng ký người dùng mới đến API /auth/signup.
- **login()**: Gửi email, mật khẩu tới API /auth/login, lưu kết quả vào user.
- **logout()**: Gửi POST /auth/logout, xoá user khỏi store.
- **checkAuth()**: Kiểm tra phiên đăng nhập hiện tại bằng GET /auth/profile.

b. Quản lý giỏ hàng – useCartStore

Store useCartStore quản lý trạng thái giỏ hàng cho người dùng, bao gồm các sản phẩm đã chọn, số lượng và tổng giá trị đơn hàng.

- **cart**: Danh sách sản phẩm trong giỏ hàng.
- **addToCart(product)**: Gửi POST /cart và cập nhật local state. Nếu sản phẩm đã tồn tại → tăng số lượng.

- **removeFromCart(productId)**: Gửi DELETE /cart, xoá sản phẩm khỏi cart.
- **updateQuantity(productId, quantity)**: Gửi PUT /cart/:id, cập nhật số lượng.
- **getCartItems()**: Gọi GET /cart khi người dùng đăng nhập để lấy dữ liệu từ server.
- **calculateTotals()**: Tính tổng tiền dựa vào price * quantity.

c. Quản lý sản phẩm – useProductStore

Store useProductStore chịu trách nhiệm quản lý danh sách sản phẩm, lọc theo danh mục, thêm/xóa sản phẩm (đặc biệt với tài khoản admin).

- **products**: Danh sách sản phẩm hiển thị trên giao diện.
- **createProduct(formData)**: Gửi POST /products, cho phép admin thêm sản phẩm (hỗ trợ ảnh).
- **fetchAllProducts()**: Gọi GET /products, tải tất cả sản phẩm từ server.
- **fetchProductsByCategory(category)**: Lọc sản phẩm theo danh mục.
- **deleteProduct(productId)**: Xoá sản phẩm khỏi hệ thống bằng DELETE.
- **toggleFeaturedProduct(productId)**: Chuyển trạng thái nổi bật của sản phẩm.
- **fetchFeaturedProducts()**: Lấy danh sách sản phẩm nổi bật (hiển thị trên trang chủ).

2.4. Các router

/ – Trang chủ

/category/:category – Lọc sản phẩm theo danh mục

/signup, /login – Đăng ký, đăng nhập

/cart – Giỏ hàng

/secret-dashboard – Trang quản trị (admin)

2.5. Backend

- Cơ sở dữ liệu sử dụng MongoDB
- Xác thực thông qua cookie
- Các api được viết có chức năng tương ứng với mục 2.3 ngoài ra có một số api phụ trợ khác hỗ trợ xác thực, xử lý ảnh, thanh toán, thống kê.

3. Những khó khăn gặp phải

Trong quá trình học và làm dự án, em gặp một số khó khăn như:

- Tìm kiếm tài liệu, nguồn học tập phù hợp khi mỗi kênh lại có lộ trình học, phiên bản khác nhau do react liên tục cập nhật, có nhiều các thư viện mới tiện lợi hơn
- Mất nhiều thời gian khi bắt đầu xây dựng dự án do không biết bắt đầu đầu từ đâu? Xây dựng như như thế nào?
- Ban đầu dự định sử dụng html và css là chính nhưng code bị rối khó code nên quyết định chuyển sang sử dụng tailwind đơn giản và hiệu quả hơn
- Mất nhiều thời gian để tìm lỗi, khi sửa lỗi này thì lỗi khác lại xuất hiện

4. Kết quả đạt được

Sau quá trình học tập và thực hành, em đã đạt được:

- Nắm được kiến thức cơ bản và nâng cao của React (component, state, props, hooks, routing).
- Áp dụng được Zustand để quản lý dữ liệu toàn cục thay cho Context API.
- Xây dựng được một trang web bán hàng với các chức năng cơ bản từ frontend đến backend.
- Cải thiện kỹ năng làm việc với API, Axios, và xử lý dữ liệ.
- Biết cách tổ chức cấu trúc project rõ ràng, phân chia component hợp lý.

- Rèn luyện kỹ năng thiết kế UI bằng TailwindCSS.

- Làm quen với tư duy xây dựng sản phẩm thực tế và kỹ năng debug

5. Hướng phát triển trong tương lai

Nếu tiếp tục phát triển dự án, em dự định:

- Tối ưu UI/UX để giao diện đẹp và chuyên nghiệp hơn
- Tích hợp hệ thống thanh toán thật (VNPay, PayPal) thay vì mô phỏng.
- Tách frontend và backend ra độc lập, deploy trên các dịch vụ
- Bổ sung thêm các chức năng ví dụ như thanh toán sau khi nhận hàng, nhắn tin với admin, bộ lọc, quản lý đơn hàng và lịch sử mua hàng,...