

**TRƯỜNG ĐẠI HỌC TRÀ VINH**  
**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN KẾT THÚC HỌC PHẦN**  
**CÔNG NGHỆ PHẦN MIỀM**

**TÊN ĐỀ TÀI**  
**XÂY DỰNG HỆ THỐNG QUẢN LÝ KÝ**  
**CỦA HÀNG BÁNH NGỌT**

**Sinh viên thực hiện:**

110122094 Nguyễn Đinh Tuấn Khoa	DA22TTD
110122243 Phạm Duy Tân	DA22TTD
110122025 Nguyễn Nhựt Trường	DA22TTB

**Giáo viên hướng dẫn:** Ts.Nguyễn Bảo Ân

Vĩnh Long, tháng 7 năm 2025

**TRƯỜNG ĐẠI HỌC TRÀ VINH**  
**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN KẾT THÚC HỌC PHẦN**  
**CÔNG NGHỆ PHẦN MIỀM**

**TÊN ĐỀ TÀI**  
**XÂY DỰNG HỆ THỐNG QUẢN LÝ**  
**CỦA HÀNG BÁNH NGỌT**

**Sinh viên thực hiện:**

110122094 Nguyễn Đinh Tuấn Khoa	DA22TTD
110122243 Phạm Duy Tân	DA22TTD
110122025 Nguyễn Nhựt Trường	DA22TTB

**Giáo viên hướng dẫn:** Ts.Nguyễn Bảo Ân

**Vĩnh Long , tháng 7 năm 2025**

## **NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ..... tháng ..... năm .....*  
**Giáo viên hướng dẫn**  
(Ký tên và ghi rõ họ tên)

## **NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ..... tháng ..... năm .....*

**Thành viên hội đồng**  
**(Ký tên và ghi rõ họ tên)**

## MỤC LỤC

CHƯƠNG 1 TỔNG QUAN.....	9
1.1 Lý do chọn đề tài .....	9
1.2 Mục tiêu của đề tài .....	9
1.3 Bảng phân công công việc cụ thể:.....	10
1.4 PHÂN TÍCH YÊU CẦU .....	11
1.4.1 Chức Năng Chính .....	11
1.5 Các yêu cầu chức năng .....	12
1.5.1 Yêu cầu chức năng quản lý kho hàng:.....	12
1.5.2 Yêu cầu chức năng quản lý sản phẩm: .....	12
1.5.3 Yêu cầu chức năng quản lý đơn hàng:.....	12
1.5.4 Yêu cầu chức năng quản lý khách hàng: .....	12
1.5.5 Yêu cầu chức năng báo cáo và thống kê: .....	13
1.5.6 Yêu cầu chức năng quản lý người dùng: .....	13
1.6 Các nhóm người dùng của hệ thống .....	14
1.6.1 Quản lý/Chủ cửa hàng: .....	14
1.6.2 Nhân viên bán hàng: .....	14
1.6.3 Nhân viên kho:.....	14
1.6.4 Nhân viên sản xuất/Thợ làm bánh: .....	15
1.6.5 Khách hàng (nếu có module online):.....	15
CHƯƠNG 2 CƠ SỞ LÝ THUYẾT .....	16
2.1 React .....	16
2.2 Node.js .....	18
2.3 RESTful API .....	20
2.4 Docker .....	21
2.5 Postman .....	23
2.6 GitHub .....	23
2.7 Jira .....	25
CHƯƠNG 3 THIẾT KẾ HỆ THỐNG .....	26
3.1 Tổng quan kiến trúc.....	26
3.2 Frontend Architecture.....	26
3.3 Backend Architecture .....	26

3.4 Database Design .....	27
3.5 Security & Deployment.....	27
3.6 Tính năng chính.....	28
3.7 Thiết kế API .....	28
3.7.1 API Đăng Nhập.....	28
3.7.2 Api Lấy danh sách tài khoản .....	28
3.7.3 API QUẢN LÝ SẢN PHẨM PRODUCT APIs.....	28
3.7.4 API QUẢN LÝ DANH MỤC CATEGORY APIs .....	28
3.7.5 API QUẢN LÝ ĐÓN HÀNG ORDER APIs .....	29
3.7.6 Tạo và quản lý mã khuyến mãi COUPON APIs .....	29
3.8 Thiết kế giao diện (UI/UX) .....	30
3.9 Thiết kế giao diện người dùng.....	30
3.9.1 Giao Diện trang chủ.....	30
3.9.2 Giao Diện trang tạo tài khoản .....	30
3.9.3 Giao Diện trang cửa hàng .....	31
3.9.4 Giao Diện trang ghở hàng.....	31
3.9.5 Giao Diện trang liên hệ.....	32
3.10 Thiết kế giao diện ADMIN .....	32
3.10.1 Thiết kế giao diện đăng nhập.....	32
3.10.2 Giao Diện Trang Tổng Quan .....	33
3.10.3 Giao Diện Trang Quản Lý Đơn Hàng .....	33
3.10.4 Giao Diện Trang Quản Lý Khách Hàng .....	34
3.10.5 Giao Diện Trang Quản Lý Sản phẩm .....	34
3.10.6 Giao Diện Trang Quản Lý Danh Mục .....	35
3.10.7 Giao Diện Trang Quản Lý Mã Giảm Giá .....	35
3.10.8 Giao Diện Trang Quản Lý Tin Nhắn Liên Hệ .....	36
3.10.9 Giao Diện Trang Quản Lý Cài Đặt.....	36
3.10.10 Giao Diện Trang Báo cáo Thông Kê .....	37
CHƯƠNG 4 TRIỂN KHAI CÔNG NGHỆ .....	38
4.1 Công Nghệ.....	38
4.2 Docker .....	38
4.3 Triển khai dự án lên hosting.....	39

CHƯƠNG 5 Quản lý dự án bằng jira.....	41
5.1 Jira .....	41
CHƯƠNG 6 KIỂM THỬ.....	42
6.1 Kiểm tra api bằng postman.....	42
6.1.1 Kiểm thử Api đăng nhập.....	42
6.1.2 Kiểm thử Api Sản Phẩm .....	43
6.2 Kiểm tra dorker.....	44
CHƯƠNG 7 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	45
7.1 TỔNG KẾT DỰ ÁN .....	45
7.2 KẾT LUẬN .....	46
7.3 HƯỚNG PHÁT TRIỂN .....	46

## DANH MỤC HÌNH ẢNH

Hình Reactjs .....	17
Hình 3.2 node.js .....	19
Hình 3.3 RESTful.....	20
Hình3.4 dorker .....	21
Hình 2.6 githup.....	24
hình 2.7 jira .....	25
Hình 3.4 diagram.....	27
Hình 3.9.1 Giao Diện trang chủ .....	30
Hình 3.9.2 Giao Diện trang tạo tài khoản .....	30
Hình 3.9.3 Giao Diện trang cửa hàng .....	31
Hình 3.9.4 Giao Diện trang ghở hàng .....	31
Hình 3.9.5 Giao Diện trang liên hệ .....	32
Hình 3.10.1 trang đăng nhập .....	32
Hình 3.10.2 Giao Diện Trang Tổng Quan .....	33
Hình 3.10.3 Giao Diện Trang Quản Lý Đơn Hàng.....	33
Hình 3.10.4 Giao Diện Trang Quản Lý Khách Hàng .....	34
Hình 3.10.5 Giao Diện Trang Quản Lý Sản phẩm.....	34
Hình 3.10.6 Giao Diện Trang Quản Lý Danh Mục .....	35
Hình 3.10.7 Giao Diện Trang Quản Lý Mã Giảm Giá .....	35
3.10.8 Giao Diện Trang Quản Lý Tin Nhắn Liên Hệ .....	36
Hình 3.10.9 Giao Diện Trang Quản Lý Cài Đặt .....	36
Hình 3.10.10 Giao Diện Trang Báo cáo Thống Kê .....	37
Hình 4.2 Chạy dự án bằng dorker .....	39
Hình 4.3 Triển khai dự án lên hosting.....	40
Hình 5.1 hoàn thành .....	41
Hình 6.1.1 Kiểm thử Api đăng nhập .....	43
Hình 6.1.2 Kiểm thử Api Sản Phẩm .....	44

## LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn chân thành đến Nguyễn Bảo Ân , giảng viên bộ môn Công nghệ Thông tin, vì sự hướng dẫn và hỗ trợ tận tình trong quá trình thực hiện đề tài “Hệ thống quản Lý Cửa hàng bánh Ngọt .” Sự tận tâm của thầy đã giúp nhóm chúng tôi hoàn thành đồ án một cách suôn sẻ.

Trong suốt quá trình làm việc, thầy không chỉ chia sẻ những kiến thức chuyên môn quý báu mà còn cung cấp những kinh nghiệm thực tiễn, giúp nhóm chúng tôi có cơ hội áp dụng kiến thức đã học vào thực tế. Những góp ý và nhắc nhở của thầy đã đóng góp lớn vào sự thành công của nhóm

Chúng tôi cũng muốn bày tỏ lòng biết ơn đối với Bộ môn Công nghệ Thông tin, Khoa Kỹ thuật và Công nghệ, nơi đã tạo ra môi trường học tập lý tưởng và trang bị cho chúng tôi những kiến thức cần thiết để hoàn thành đồ án.

Xin chân thành cảm ơn!

## CHƯƠNG 1 TỔNG QUAN

### 1.1 Lý do chọn đề tài

Trong bối cảnh ngành bánh kẹo đang phát triển mạnh mẽ tại Việt Nam, việc ứng dụng công nghệ thông tin vào quản lý kinh doanh trở thành nhu cầu cấp thiết. Nhiều tiệm bánh nhỏ lẻ hiện tại vẫn đang sử dụng phương pháp quản lý thủ công, dẫn đến những khó khăn trong việc theo dõi kho hàng, quản lý đơn hàng, chăm sóc khách hàng và báo cáo tài chính. Đề tài "Hệ thống quản lý tiệm bánh ngọt" được lựa chọn nhằm giải quyết những vấn đề thực tiễn này thông qua việc xây dựng một hệ thống quản lý toàn diện, từ quản lý nguyên liệu, thành phẩm, theo dõi hạn sử dụng, xử lý đơn hàng đến lưu trữ thông tin khách hàng và tạo báo cáo doanh thu. Đề tài có tính khả thi cao với phạm vi phù hợp, cho phép áp dụng các kiến thức về cơ sở dữ liệu và lập trình đã học, đồng thời mang lại giá trị thực tiễn khi có thể triển khai cho các cửa hàng bánh thực tế. Việc thực hiện đề tài này không chỉ giúp nâng cao hiệu quả quản lý cho các tiệm bánh mà còn góp phần thúc đẩy quá trình số hóa trong lĩnh vực kinh doanh thực phẩm.

### 1.2 Mục tiêu của đề tài

Đề tài "Hệ thống quản lý tiệm bánh ngọt" được thực hiện với mục tiêu chính là xây dựng một hệ thống quản lý toàn diện, hiện đại nhằm nâng cao hiệu quả hoạt động kinh doanh cho các tiệm bánh ngọt. Cụ thể, hệ thống hướng đến việc tự động hóa các quy trình quản lý từ khâu nhập kho nguyên liệu, theo dõi tồn kho và hạn sử dụng, quản lý thông tin sản phẩm, xử lý đơn hàng từ khách hàng đến việc tạo báo cáo doanh thu và lợi nhuận. Hệ thống cũng tập trung vào việc xây dựng cơ sở dữ liệu khách hàng hoàn chỉnh, giúp các tiệm bánh có thể theo dõi lịch sử mua hàng, sở thích và thực hiện các chương trình chăm sóc khách hàng hiệu quả. Bên cạnh đó, đề tài còn hướng đến mục tiêu cung cấp giao diện người dùng thân thiện, dễ sử dụng để nhân viên có thể nhanh chóng làm quen và vận hành hệ thống một cách hiệu quả. Cuối cùng, hệ thống được thiết kế với khả năng mở rộng và tích hợp, cho phép bổ sung thêm các tính năng mới trong tương lai như quản lý nhân viên, tích hợp thanh toán trực tuyến hay kết nối với các nền tảng thương mại điện tử, từ đó góp phần hiện đại hóa và nâng cao năng lực cạnh tranh cho các doanh nghiệp trong lĩnh vực bánh kẹo.

### 1.3 Bảng phân công công việc cụ thể:

Thời gian	NỘI DUNG CÔNG VIỆC	NGƯỜI THỰC HIỆN
Tuần 1	<ul style="list-style-type: none"> <li>- Xác định đề tài, mục tiêu, kế hoạch</li> <li>- Khảo sát, thu thập yêu cầu</li> <li>phân tích, xác định yêu cầu cơ bản</li> <li>-Xác định ràng buộc, nhóm người dùng, quyền cơ bản</li> <li>- Lập kế hoạch scrum trên jira</li> </ul>	Cả nhóm
Tuần 2	<ul style="list-style-type: none"> <li>- Thiết kế mô hình dữ liệu</li> <li>- Thiết kế FIGMA</li> <li>- Cài đặt môi trường node.js react</li> </ul>	Cả nhóm
Tuần 3	<ul style="list-style-type: none"> <li>-thiết kế giao diện frontend</li> <li>Các trang người dùng</li> <li>-thiết kế trang admin</li> <li>-Thiết kế cơ sở dữ liệu</li> </ul>	Cả nhóm
Tuần 4	<ul style="list-style-type: none"> <li>- Backend</li> <li>Sử lý api phía frontend</li> <li>-kiểm tra api bằng postman</li> </ul>	Cả nhóm
Tuần 5	<ul style="list-style-type: none"> <li>- kiểm tra lại wed</li> <li>- viết báo cáo</li> </ul>	Cả nhóm

## 1.4 PHÂN TÍCH YÊU CẦU

### 1.4.1 Chức Năng Chính

Hệ thống quản lý tiệm bánh ngọt được thiết kế với các chức năng chính nhằm đáp ứng toàn diện nhu cầu quản lý kinh doanh.

Chức năng quản lý kho hàng cho phép theo dõi chi tiết thông tin nguyên liệu đầu vào, bao gồm tên nguyên liệu, số lượng tồn kho, giá nhập, nhà cung cấp và đặc biệt là hạn sử dụng để tránh lãng phí và đảm bảo chất lượng sản phẩm.

Chức năng quản lý sản phẩm hỗ trợ lưu trữ thông tin chi tiết về các loại bánh, bao gồm tên sản phẩm, mô tả, giá bán, công thức chế biến và hình ảnh minh họa

Hệ thống tích hợp chức năng quản lý đơn hàng từ khâu tiếp nhận yêu cầu của khách hàng, tính toán tổng tiền, theo dõi trạng thái đơn hàng từ đang xử lý đến hoàn thành, cùng với việc quản lý thông tin giao hàng.

Chức năng quản lý khách hàng cho phép lưu trữ thông tin cá nhân, lịch sử mua hàng, sở thích và điểm tích lũy để phục vụ các chương trình khuyến mãi và chăm sóc khách hàng.

Hệ thống còn cung cấp chức năng báo cáo và thống kê toàn diện, tự động tạo các báo cáo về doanh thu theo ngày, tháng, năm, thống kê sản phẩm bán chạy, phân tích xu hướng kinh doanh và tính toán lợi nhuận.

Cuối cùng, chức năng quản lý người dùng đảm bảo mật thông tin thông qua hệ thống phân quyền, cho phép quản lý tài khoản nhân viên với các mức độ truy cập khác nhau tùy theo vị trí công việc.

## 1.5 Các yêu cầu chức năng

### 1.5.1 Yêu cầu chức năng quản lý kho hàng:

Thêm, sửa, xóa thông tin nguyên liệu và sản phẩm trong kho

Cập nhật số lượng tồn kho khi nhập/xuất hàng

Theo dõi và cảnh báo hạn sử dụng của nguyên liệu

Tạo phiếu nhập/xuất kho tự động

Kiểm tra và cảnh báo khi hàng tồn kho dưới mức tối thiểu

### 1.5.2 Yêu cầu chức năng quản lý sản phẩm:

Tạo danh mục sản phẩm với thông tin chi tiết (tên, mô tả, giá, hình ảnh)

Quản lý công thức chế biến và nguyên liệu cần thiết cho từng sản phẩm

Phân loại sản phẩm theo danh mục (bánh sinh nhật, bánh mì, bánh ngọt...)

Cập nhật giá bán và trạng thái có sẵn của sản phẩm

### 1.5.3 Yêu cầu chức năng quản lý đơn hàng:

Tạo đơn hàng mới với thông tin khách hàng và sản phẩm

Tính toán tổng tiền bao gồm thuế và phí giao hàng

Theo dõi trạng thái đơn hàng (chờ xử lý, đang làm, hoàn thành, đã giao)

In hóa đơn và phiếu giao hàng

Hủy hoặc chỉnh sửa đơn hàng khi cần thiết

### 1.5.4 Yêu cầu chức năng quản lý khách hàng:

Lưu trữ thông tin cá nhân khách hàng (tên, địa chỉ, số điện thoại, email)

Theo dõi lịch sử mua hàng và tổng chi tiêu

Quản lý điểm tích lũy và chương trình khuyến mãi

Phân loại khách hàng theo mức độ thân thiết

### **1.5.5 Yêu cầu chức năng báo cáo và thống kê:**

Tạo báo cáo doanh thu theo ngày, tuần, tháng, năm

Thống kê sản phẩm bán chạy và ít bán

Báo cáo tình hình tồn kho và nguyên liệu sắp hết hạn

Phân tích xu hướng bán hàng và lợi nhuận

Xuất báo cáo dưới định dạng PDF hoặc Excel

### **1.5.6 Yêu cầu chức năng quản lý người dùng:**

Đăng nhập/đăng xuất hệ thống với xác thực bảo mật

Phân quyền truy cập theo vai trò (quản lý, nhân viên bán hàng, thủ kho)

Quản lý thông tin tài khoản và thay đổi mật khẩu

Ghi log hoạt động của người dùng trong hệ thống

## 1.6 Các nhóm người dùng của hệ thống

### 1.6.1 Quản lý/Chủ cửa hàng:

Có quyền truy cập toàn bộ chức năng của hệ thống

Quản lý thông tin cửa hàng, sản phẩm và giá cả

Xem tất cả các báo cáo doanh thu, lợi nhuận và thống kê kinh doanh

Quản lý tài khoản nhân viên và phân quyền truy cập

Thiết lập các chính sách khuyến mãi và chương trình tích điểm

Theo dõi hiệu suất kinh doanh tổng thể và đưa ra quyết định chiến lược

### 1.6.2 Nhân viên bán hàng:

Tiếp nhận và xử lý đơn hàng từ khách hàng

Tạo hóa đơn bán hàng và tính toán tổng tiền

Quản lý thông tin khách hàng và cập nhật điểm tích lũy

Kiểm tra tình trạng sản phẩm có sẵn trong kho

Xử lý thanh toán và in hóa đơn cho khách hàng

Xem báo cáo doanh thu cá nhân và thống kê bán hàng

### 1.6.3 Nhân viên kho:

Quản lý việc nhập/xuất nguyên liệu và sản phẩm

Cập nhật số lượng tồn kho và theo dõi hạn sử dụng

Tạo phiếu nhập/xuất kho và kiểm kê định kỳ

Cảnh báo khi nguyên liệu sắp hết hạn hoặc tồn kho thấp

Quản lý thông tin nhà cung cấp và giá nhập hàng

Xem báo cáo tình hình kho hàng và xuất nhập tồn

#### **1.6.4 Nhân viên sản xuất/Thợ làm bánh:**

Xem danh sách đơn hàng cần sản xuất theo thứ tự ưu tiên

Truy cập công thức chế biến và danh sách nguyên liệu cần thiết

Cập nhật trạng thái sản xuất của từng đơn hàng

Báo cáo tình trạng hoàn thành sản phẩm

Kiểm tra tình trạng nguyên liệu có sẵn trước khi sản xuất

Ghi nhận thời gian sản xuất và số lượng thành phẩm

#### **1.6.5 Khách hàng (nếu có module online):**

Xem danh sách sản phẩm và thông tin chi tiết

Đặt hàng trực tuyến và theo dõi trạng thái đơn hàng

Quản lý thông tin cá nhân và địa chỉ giao hàng

Xem lịch sử mua hàng và điểm tích lũy

Nhận thông báo về khuyến mãi và sản phẩm mới

Đánh giá và phản hồi về sản phẩm đã mua

## CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

### 2.1 React

React là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook (nay là Meta) vào năm 2013, chuyên dụng cho việc xây dựng giao diện người dùng (User Interface) cho các ứng dụng web và mobile. React được thiết kế theo mô hình component-based architecture, cho phép các nhà phát triển chia nhỏ giao diện thành các thành phần độc lập, có thể tái sử dụng và dễ dàng quản lý. Điểm nổi bật của React là Virtual DOM (Document Object Model ảo), một kỹ thuật tối ưu hóa hiệu suất bằng cách tạo ra một bản sao của DOM thực trong bộ nhớ, so sánh sự thay đổi và chỉ cập nhật những phần cần thiết trên DOM thực, giúp giảm thiểu thời gian render và nâng cao trải nghiệm người dùng.

React sử dụng JSX (JavaScript XML), một cú pháp mở rộng cho phép viết HTML-like code bên trong JavaScript, giúp việc tạo ra các component trở nên trực quan và dễ hiểu hơn. Thư viện này hỗ trợ mạnh mẽ việc quản lý state (trạng thái) thông qua useState Hook và useEffect Hook, cho phép các component phản ứng với sự thay đổi dữ liệu một cách tự động. React cũng cung cấp Context API để quản lý state toàn cục, giúp chia sẻ dữ liệu giữa các component mà không cần truyền props qua nhiều cấp độ. Với hệ sinh thái phong phú bao gồm React Router cho routing, Redux cho state management phức tạp, và hàng nghìn thư viện hỗ trợ khác, React đã trở thành một trong những công nghệ frontend phổ biến nhất hiện nay.



### *Hình Reactjs*

Trong dự án hệ thống quản lý tiệm bánh ngọt, React phiên bản 19.0.0 được sử dụng để xây dựng toàn bộ giao diện người dùng, từ trang chủ khách hàng đến dashboard quản trị. Các tính năng chính như hiển thị danh sách sản phẩm, quản lý giỏ hàng, xử lý đơn hàng, và báo cáo thống kê đều được triển khai thông qua các React component độc lập. React Router DOM được tích hợp để tạo ra single-page application với nhiều trang khác nhau, trong khi Context API được sử dụng để quản lý trạng thái giỏ hàng và thông tin đăng nhập người dùng. Việc sử dụng React không chỉ giúp tạo ra giao diện động và tương tác cao mà còn đảm bảo khả năng bảo trì và mở rộng hệ thống trong tương lai.

Dự án "Hệ thống quản lý tiệm bánh ngọt" được xây dựng dựa trên kiến trúc web hiện đại với mô hình 3 tầng (3-tier architecture) bao gồm tầng giao diện người dùng, tầng xử lý logic nghiệp vụ và tầng cơ sở dữ liệu. Hệ thống sử dụng công nghệ containerization thông qua Docker và Docker Compose để đảm bảo tính nhất quán trong môi trường phát triển và triển khai, cho phép các thành phần của hệ thống hoạt động độc lập và dễ dàng mở rộng.

Về phía frontend, hệ thống được phát triển bằng React phiên bản 19.0.0, một trong những thư viện JavaScript phổ biến nhất hiện nay, kết hợp với Vite 6.3.5 làm build tool để tối ưu hóa quá trình phát triển và đóng gói ứng dụng. React Router DOM 7.5.3 được sử dụng để quản lý điều hướng trong ứng dụng single-page, trong khi React Context API đảm nhận việc quản lý trạng thái toàn cục như giỏ hàng và thông tin người dùng. Giao diện được thiết kế responsive với CSS3 và các quy tắc styling tùy chỉnh, đảm bảo trải nghiệm người dùng tối ưu trên mọi thiết bị.

## 2.2 Node.js

Node.js là một runtime environment mã nguồn mở được xây dựng trên JavaScript V8 engine của Google Chrome, cho phép thực thi mã JavaScript phía server thay vì chỉ giới hạn trong trình duyệt web. Được phát triển bởi Ryan Dahl vào năm 2009, Node.js đã cách mạng hóa cách thức phát triển ứng dụng web bằng cách cho phép các nhà phát triển sử dụng cùng một ngôn ngữ JavaScript cho cả frontend và backend. Điểm mạnh nổi bật của Node.js là kiến trúc non-blocking I/O và event-driven, cho phép xử lý hàng nghìn kết nối đồng thời mà không cần tạo ra nhiều thread, giúp tối ưu hóa hiệu suất và tiết kiệm tài nguyên hệ thống.

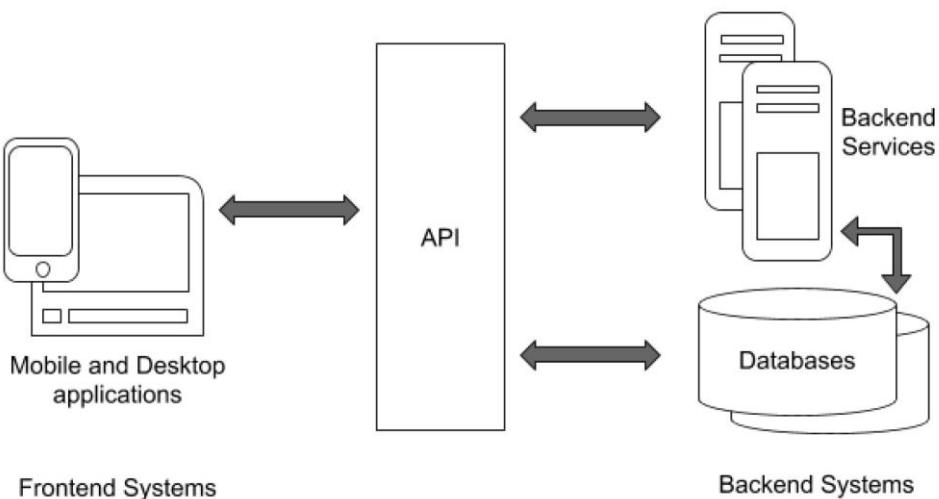
Node.js sử dụng mô hình single-threaded event loop, cho phép xử lý các tác vụ bất đồng bộ một cách hiệu quả thông qua callback functions, Promises và async/await. Hệ sinh thái NPM (Node Package Manager) của Node.js là kho thư viện lớn nhất thế giới với hàng triệu package, cung cấp các giải pháp sẵn có cho mọi nhu cầu phát triển từ web framework, database connector, authentication, đến các công cụ build và testing. Node.js đặc biệt phù hợp cho việc xây dựng các ứng dụng real-time như chat applications, collaborative tools, gaming platforms, và các RESTful API services nhờ khả năng xử lý I/O operations nhanh chóng và hiệu quả.



*Hình 3.2 node.js*

Trong dự án hệ thống quản lý tiệm bánh ngọt, Node.js đóng vai trò là nền tảng backend chính, kết hợp với Express.js framework để tạo ra các RESTful API endpoints phục vụ cho frontend React. Server Node.js xử lý các chức năng quan trọng như xác thực người dùng thông qua JWT, quản lý session, xử lý upload file với Multer, tương tác với cơ sở dữ liệu MySQL thông qua Sequelize ORM, và triển khai các middleware bảo mật như CORS, Helmet, và rate limiting. Khả năng xử lý bất đồng bộ của Node.js cho phép hệ thống xử lý đồng thời nhiều yêu cầu từ khách hàng và admin mà không bị block, đảm bảo hiệu suất cao ngay cả khi có nhiều người dùng truy cập cùng lúc. Việc sử dụng Node.js cũng giúp đội ngũ phát triển có thể sử dụng cùng một ngôn ngữ JavaScript cho toàn bộ stack, giảm thiểu complexity và tăng tốc độ phát triển.

## 2.3 RESTful API



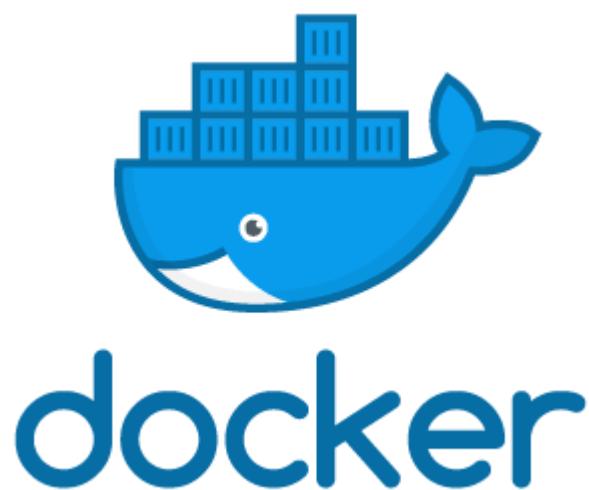
Hình 3.3 RESTful

RESTful API (Representational State Transfer Application Programming Interface) là một kiểu kiến trúc phần mềm được sử dụng rộng rãi để thiết kế các web service, cho phép các ứng dụng khác nhau giao tiếp với nhau thông qua giao thức HTTP. REST được Roy Fielding giới thiệu vào năm 2000 như một tập hợp các nguyên tắc thiết kế để tạo ra các hệ thống phân tán có khả năng mở rộng cao. RESTful API hoạt động dựa trên các HTTP methods chuẩn bao gồm GET (lấy dữ liệu), POST (tạo mới), PUT (cập nhật toàn bộ), PATCH (cập nhật một phần), và DELETE (xóa), kết hợp với các URL endpoints có cấu trúc rõ ràng để thực hiện các thao tác CRUD (Create, Read, Update, Delete) trên tài nguyên.

Các nguyên tắc cốt lõi của RESTful API bao gồm stateless (không lưu trạng thái), nghĩa là mỗi request phải chứa đầy đủ thông tin cần thiết để server xử lý mà không phụ thuộc vào các request trước đó; uniform interface với việc sử dụng các HTTP methods và status codes chuẩn; client-server architecture tách biệt rõ ràng giữa frontend và backend; và cacheable để tối ưu hiệu suất. RESTful API thường trả về dữ liệu dưới định dạng JSON (JavaScript Object Notation) do tính nhẹ và dễ xử lý, đồng thời sử dụng các HTTP status codes như 200 (thành công), 201 (tạo mới thành công), 400 (lỗi client), 401 (chưa xác thực), 403 (không có quyền), 404 (không tìm thấy), và 500 (lỗi server) để thông báo kết quả xử lý.

Trong dự án hệ thống quản lý tiệm bánh ngọt, RESTful API được triển khai thông qua Express.js framework trên Node.js, cung cấp các endpoints để frontend React có thể tương tác với backend. Các API endpoints được thiết kế theo chuẩn REST như /api/products (GET để lấy danh sách sản phẩm, POST để tạo sản phẩm mới), /api/products/:id (GET để lấy chi tiết, PUT để cập nhật, DELETE để xóa), /api/orders cho quản lý đơn hàng, /api/customers cho quản lý khách hàng, và /api/auth cho xác thực người dùng. Hệ thống sử dụng middleware để xử lý authentication với JWT token, validation dữ liệu đầu vào, error handling, và CORS policy. Các response được chuẩn hóa với format JSON bao gồm status code, message, và data, giúp frontend dễ dàng xử lý và hiển thị thông tin phù hợp cho người dùng. RESTful API architecture này đảm bảo hệ thống có khả năng mở rộng, dễ bảo trì, và có thể tích hợp với các ứng dụng mobile hoặc third-party services trong tương lai.

## 2.4 Docker



*Hình 3.4 dorker*

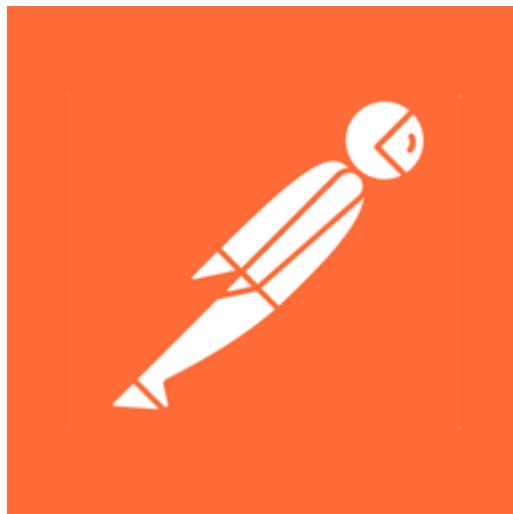
Docker là một nền tảng containerization mã nguồn mở được phát triển bởi Docker Inc., cho phép đóng gói ứng dụng cùng với tất cả các dependencies, thư viện và cấu hình cần thiết vào trong một container nhẹ, portable và có thể chạy trên bất kỳ môi trường nào có cài đặt Docker. Được ra mắt vào năm 2013, Docker đã cách mạng hóa cách thức triển khai và quản lý ứng dụng bằng cách giải quyết vấn đề "it works on my machine" thông qua việc tạo ra môi trường thực thi nhất quán từ development đến production. Container khác với virtual machine ở chỗ nó chia sẻ kernel của hệ điều hành host, do đó nhẹ hơn, khởi động nhanh hơn và sử dụng ít tài nguyên hơn.

Docker hoạt động dựa trên kiến trúc client-server với Docker daemon chạy trên host machine và Docker client giao tiếp thông qua REST API. Dockerfile là file text chứa các instruction để build Docker image, định nghĩa base image, copy source code, cài đặt dependencies, expose ports và set entry point. Docker image là template read-only được sử dụng để tạo container, trong khi Docker container là instance chạy của image. Docker Hub là registry công cộng lớn nhất chứa hàng triệu pre-built images cho các ứng dụng và services phổ biến. Docker Compose là công cụ để định nghĩa và chạy multi-container applications thông qua file YAML, cho phép orchestrate nhiều services cùng lúc với các cấu hình network, volume và environment variables.

Trong dự án hệ thống quản lý tiệm bánh ngọt, Docker được sử dụng để containerize toàn bộ application stack bao gồm ba services chính: MySQL database, Node.js backend, và React frontend. File docker-compose.yml định nghĩa cấu hình cho từng service với các environment variables, port mappings, volume mounts, và dependencies. MySQL container sử dụng official MySQL 8.0 image với persistent data storage thông qua Docker volume, automatic database initialization từ init.sql file, và health check để đảm bảo database sẵn sàng trước khi start các services khác. Backend và frontend containers được build từ custom Dockerfiles, với volume mounting cho development environment để enable hot-reload. Docker network được tạo để các containers có thể giao tiếp với nhau một cách an toàn và isolated từ host network. Việc sử dụng Docker không chỉ đảm bảo consistency across different

environments mà còn simplify deployment process, enable horizontal scaling, và facilitate CI/CD pipeline implementation.

## 2.5 Postman



Hình 2.5 postman

Postman là một công cụ phát triển API (Application Programming Interface) mạnh mẽ và phổ biến nhất hiện nay, được thiết kế để hỗ trợ các nhà phát triển trong việc thiết kế, testing, documenting và monitoring các RESTful API và GraphQL API. Ra mắt vào năm 2012 bởi Abhinav Asthana, Postman ban đầu là một Chrome extension đơn giản nhưng đã phát triển thành một platform toàn diện với ứng dụng desktop và web-based interface. Postman cung cấp giao diện trực quan cho phép gửi HTTP requests với các methods khác nhau (GET, POST, PUT, DELETE, PATCH), thiết lập headers, parameters, authentication, và request body một cách dễ dàng mà không cần viết code.

## 2.6 GitHub



*Hình 2.6 githup*

GitHub là một nền tảng hosting và collaboration service dựa trên hệ thống quản lý phiên bản Git, được thành lập vào năm 2008 bởi Tom Preston-Werner, Chris Wanstrath, và PJ Hyett. Được Microsoft mua lại vào năm 2018 với giá 7.5 tỷ USD, GitHub đã trở thành platform lớn nhất thế giới cho việc lưu trữ và quản lý source code với hơn 100 triệu repositories và 83 triệu developers. GitHub cung cấp giao diện web thân thiện cho Git, một distributed version control system, cho phép developers track changes, collaborate với team members, và manage project history một cách hiệu quả. Platform này hỗ trợ cả public repositories (miễn phí và open source) và private repositories cho các dự án thương mại.

Các tính năng chính của GitHub bao gồm Repositories để lưu trữ source code và project files, Branches để phát triển features độc lập, Pull Requests để review và merge code changes, Issues để track bugs và feature requests, và Projects để quản lý workflow. GitHub Actions cung cấp CI/CD (Continuous Integration/Continuous Deployment) capabilities, cho phép automate testing, building, và deployment processes. GitHub Pages enable static website hosting trực tiếp từ repository. Collaboration features bao gồm code review tools, inline comments, team management, và access control với granular permissions. GitHub cũng cung cấp comprehensive API để integrate với third-party tools và services.

## 2.7 Jira



*hình 2.7 jira*

Jira là một công cụ quản lý dự án và theo dõi vấn đề (issue tracking) được phát triển bởi Atlassian, được sử dụng rộng rãi trong các tổ chức phát triển phần mềm để quản lý quy trình làm việc, theo dõi bugs, và triển khai các phương pháp Agile như Scrum và Kanban. Ra mắt vào năm 2002, Jira ban đầu được thiết kế như một bug tracking system nhưng đã phát triển thành một platform toàn diện hỗ trợ project management, requirement management, test case management, và release planning. Jira cung cấp giao diện web-based cho phép teams tạo, assign, prioritize, và track các tasks, user stories, bugs, và epics thông qua customizable workflows và dashboards.

Các tính năng chính của Jira bao gồm Issue Types để phân loại công việc (Story, Task, Bug, Epic, Subtask), Custom Fields để capture thông tin specific cho từng dự án, Workflows để define quy trình xử lý từ creation đến completion, và Boards (Scrum/Kanban) để visualize work progress. Jira hỗ trợ Agile methodologies với Sprint planning, Backlog management, Burndown charts, và Velocity tracking. Advanced search với JQL (Jira Query Language) cho phép filter và report data theo nhiều criteria khác nhau. Integration capabilities với các tools khác như Confluence (documentation), Bitbucket (source control), Bamboo (CI/CD), và third-party applications thông qua REST APIs và marketplace add-ons.

## CHƯƠNG 3 THIẾT KẾ HỆ THỐNG

### 3.1 Tổng quan kiến trúc

Hệ thống được xây dựng theo mô hình 3-tier architecture bao gồm Frontend (React), Backend (Node.js) và Database (MySQL). Toàn bộ hệ thống được containerized bằng Docker để đảm bảo tính nhất quán và dễ dàng triển khai.

### 3.2 Frontend Architecture

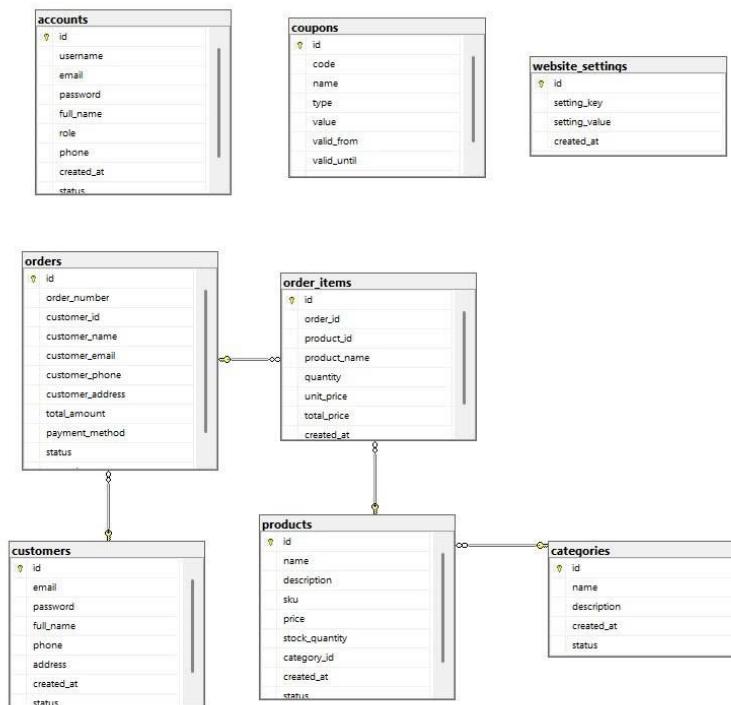
Frontend sử dụng React 19.0.0 với Vite làm build tool, React Router DOM để quản lý routing và Context API cho state management. Giao diện được thiết kế responsive với CSS modules, hỗ trợ đa vai trò (Admin/Staff/Customer) với các tính năng như quản lý giỏ hàng, tạo hóa đơn PDF, upload hình ảnh và cập nhật real-time.

### 3.3 Backend Architecture

Backend được phát triển trên Node.js với Express.js framework, sử dụng Sequelize ORM để tương tác với database. Hệ thống authentication dựa trên JWT token với bcryptjs để mã hóa mật khẩu. API được thiết kế RESTful với các endpoint chính: /auth, /accounts, /products, /categories, /orders, /customers, /coupons, /messages, /reports và /settings.

### 3.4 Database Design

Database MySQL 8.0 với các bảng chính: accounts (quản lý admin/staff), customers (khách hàng), products (sản phẩm), categories (danh mục), orders (đơn hàng), order\_items (chi tiết đơn hàng), coupons (mã giảm giá), messages (tin nhắn) và website\_settings (cấu hình hệ thống). Thiết kế database tuân thủ chuẩn normalization với các foreign key relationships.



Hình 3.4 diagram

### 3.5 Security & Deployment

Hệ thống tích hợp các biện pháp bảo mật như JWT authentication, password hashing, role-based access control, CORS protection, rate limiting và input validation. Triển khai thông qua Docker Compose với 3 services: database (port 3309), backend (port 5000) và frontend (port 5173), hỗ trợ cả development và production environment.

### 3.6 Tính năng chính

Hệ thống hỗ trợ quản lý tài khoản đa cấp, quản lý sản phẩm và danh mục, xử lý đơn hàng, quản lý khách hàng, hệ thống mã giảm giá, báo cáo thống kê và cấu hình website. Giao diện thân thiện với người dùng, responsive trên mọi thiết bị và có khả năng mở rộng cao cho tương lai.

### 3.7 Thiết kế API

#### 3.7.1 API Đăng Nhập

`POST /api/auth/login`

Request Body:

```
{
  "username": "admin",
  "password": "admin123"
}
```

Response thành công:

```
{
  "message": "Đăng nhập thành công!",
  "user": {
    "id": 1000,
    "username": "admin",
    "email": "admin@tiembanh.com",
    "role": "admin"
  }
}
```

#### 3.7.2 API Lấy danh sách tài khoản

<code>GET /api/accounts</code>	// Lấy danh sách tài khoản (Admin only)
--------------------------------	---

#### 3.7.3 API QUẢN LÝ SẢN PHẨM PRODUCT APIs

<code>GET /api/products</code>	// Lấy danh sách sản phẩm (Public)
<code>GET /api/products/:id</code>	// Lấy chi tiết sản phẩm (Public)
<code>POST /api/products</code>	// Tạo sản phẩm mới (Admin only)
<code>PUT /api/products/:id</code>	// Cập nhật sản phẩm (Admin only)
<code>DELETE /api/products/:id</code>	// Xóa sản phẩm (Admin only)
<code>PATCH /api/products/:id/stock</code>	// Cập nhật tồn kho (Admin/Staff)

#### 3.7.4 API QUẢN LÝ DANH MỤC CATEGORY APIs

<code>GET /api/categories</code>	// Lấy danh sách danh mục (Public)
<code>GET /api/categories/:id</code>	// Lấy chi tiết danh mục (Public)
<code>POST /api/categories</code>	// Tạo danh mục mới (Admin only)
<code>PUT /api/categories/:id</code>	// Cập nhật danh mục (Admin only)
<code>DELETE /api/categories/:id</code>	// Xóa danh mục (Admin only)

### 3.7.5 API QUẢN LÝ ĐƠN HÀNG ORDER APIs

```
// Admin/Staff routes
GET /api/orders           // Lấy tất cả đơn hàng (Admin/Staff)
GET /api/orders/:id        // Lấy chi tiết đơn hàng (Admin/Staff)
PUT /api/orders/:id/status // Cập nhật trạng thái đơn hàng (Admin/Staff)
DELETE /api/orders/:id     // Xóa đơn hàng (Admin only)

// Customer routes
POST /api/orders           // Tạo đơn hàng mới (Customer)
GET /api/orders/customer/my-orders // Lấy đơn hàng của khách hàng (Customer)
GET /api/orders/customer/:id   // Lấy chi tiết đơn hàng khách hàng (Customer)
```

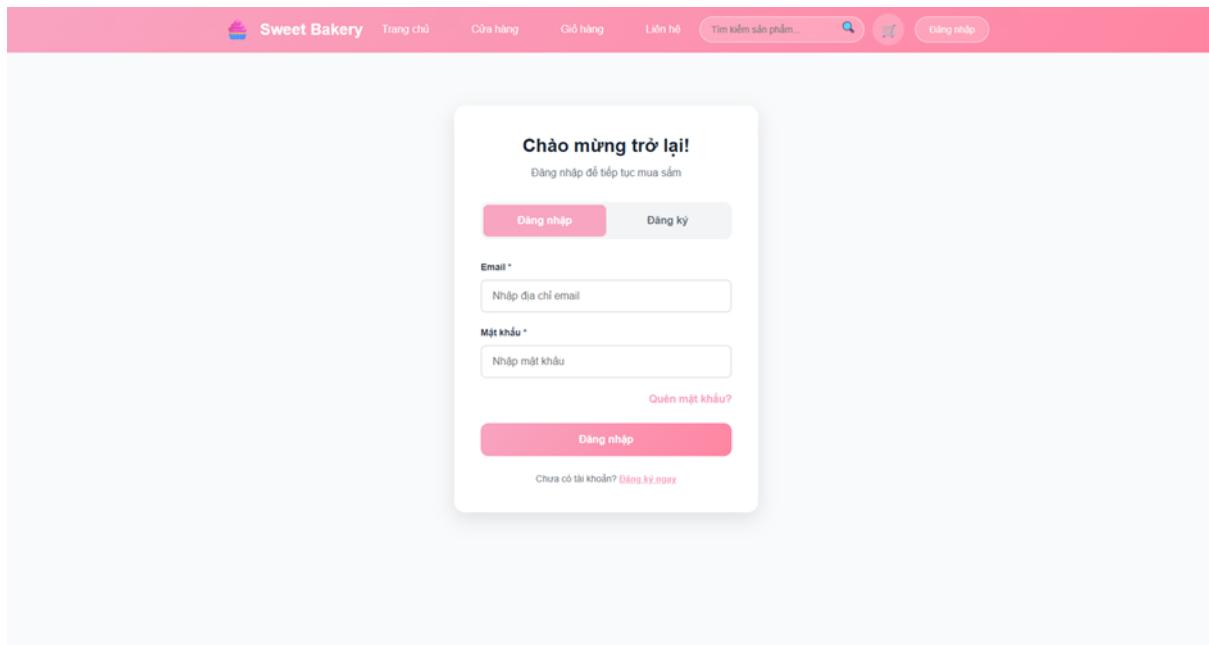
### 3.7.6 Tạo và quản lý mã khuyến mãi COUPON APIs

```
POST /api/coupons/validate    // Kiểm tra mã giảm giá (Public)
GET /api/coupons/active        // Lấy mã giảm giá đang hoạt động (Public)
GET /api/coupons               // Lấy tất cả mã giảm giá (Admin only)
GET /api/coupons/:id           // Lấy chi tiết mã giảm giá (Admin only)
POST /api/coupons              // Tạo mã giảm giá mới (Admin only)
PUT /api/coupons/:id           // Cập nhật mã giảm giá (Admin only)
DELETE /api/coupons/:id         // Xóa mã giảm giá (Admin only)
PUT /api/coupons/:id/status    // Cập nhật trạng thái mã giảm giá (Admin only)
GET /api/coupons/:id/usage      // Xem thống kê sử dụng mã giảm giá (Admin only)
```

### 3.8 Thiết kế giao diện (UI/UX)

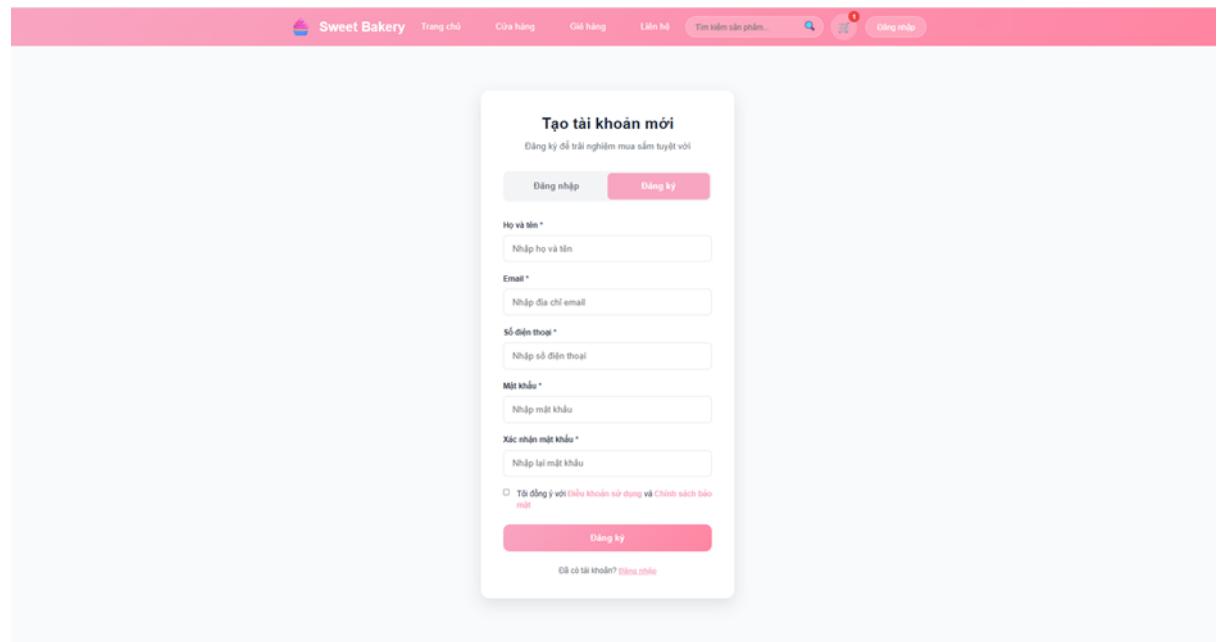
#### 3.9 Thiết kế giao diện người dùng

##### 3.9.1 Giao Diện trang chủ



Hình 3.9.1 Giao Diện trang chủ

##### 3.9.2 Giao Diện trang tạo tài khoản



Hình 3.9.2 Giao Diện trang tạo tài khoản

### 3.9.3 Giao Diện trang cửa hàng

The screenshot shows the homepage of the Sweet Bakery website. At the top, there is a pink header bar with the logo "Sweet Bakery", a search bar containing "Tim kiếm sản phẩm...", and a user profile icon for "Nguyễn Đình Tuấn Khoa". Below the header, a banner reads "Cửa Hàng Bánh Ngọt" with the subtext "Khám phá bộ sưu tập bánh ngọt tươi ngon được làm thủ công hàng ngày". A sidebar on the left lists categories: Tất cả (All), Bánh Gạo, and Bánh sinh nhật. The main content area displays three products: Bánh Gạo (5,000đ), Bánh Sinh Nhật Màu Xanh (30,000đ), and Bánh Sinh Nhật Trái Cây (50,000đ). Each product card includes a "Thêm vào giỏ hàng" button.

*Hình 3.9.3 Giao Diện trang cửa hàng*

### 3.9.4 Giao Diện trang ghở hàng

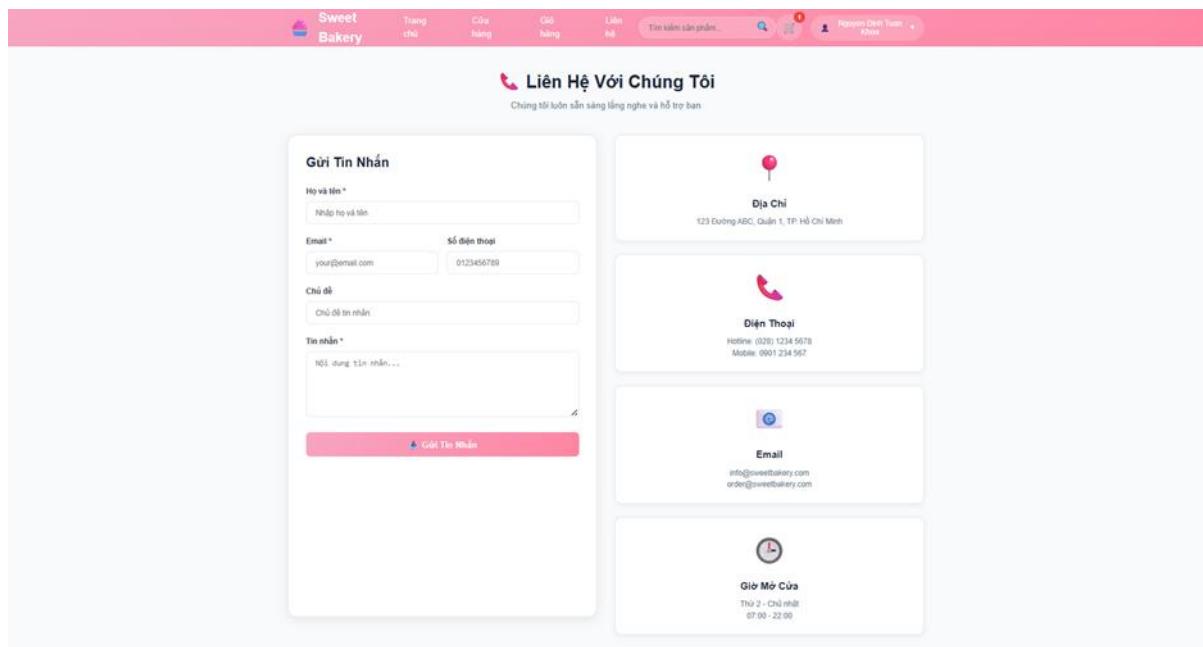
The screenshot shows the checkout page for a single product, "Bánh Gạo", which costs 5,000đ. The page includes a summary table for the order details:

Tên tính:	5.000 đ
Phi vận chuyển:	30.000 đ
<b>Tổng cộng:</b>	<b>35.000 đ</b>

Below the table, there are buttons for "Tiến hành thanh toán" (Proceed to payment) and "Tiếp tục mua sắm" (Continue shopping). A note at the bottom right says "Cam kết của chúng tôi" (Our commitment) with three items: "Giao hàng nhanh chóng", "Đảm bảo chất lượng", and "Hỗ trợ 24/7".

*Hình 3.9.4 Giao Diện trang ghở hàng*

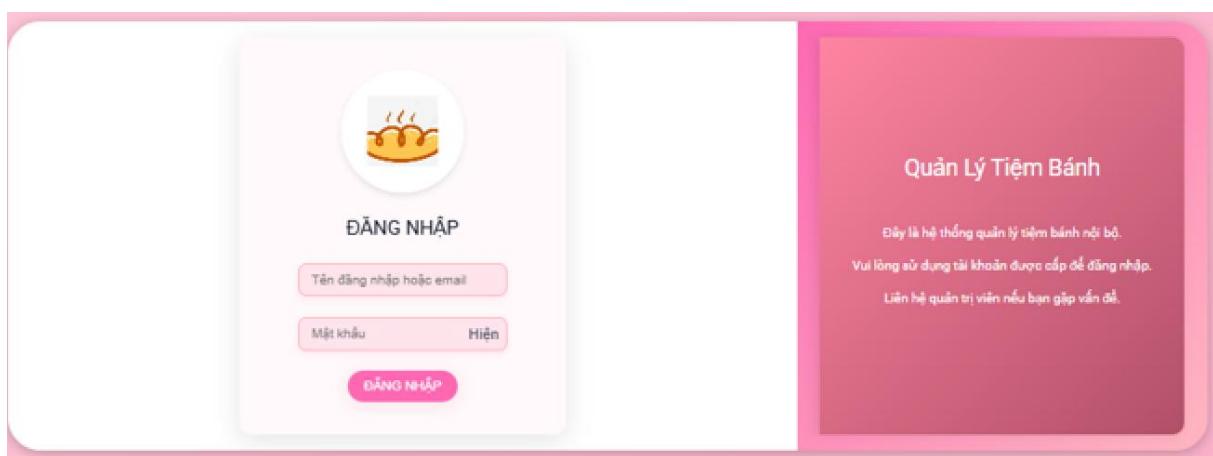
### 3.9.5 Giao Diện trang liên hệ



Hình 3.9.5 Giao Diện trang liên hệ

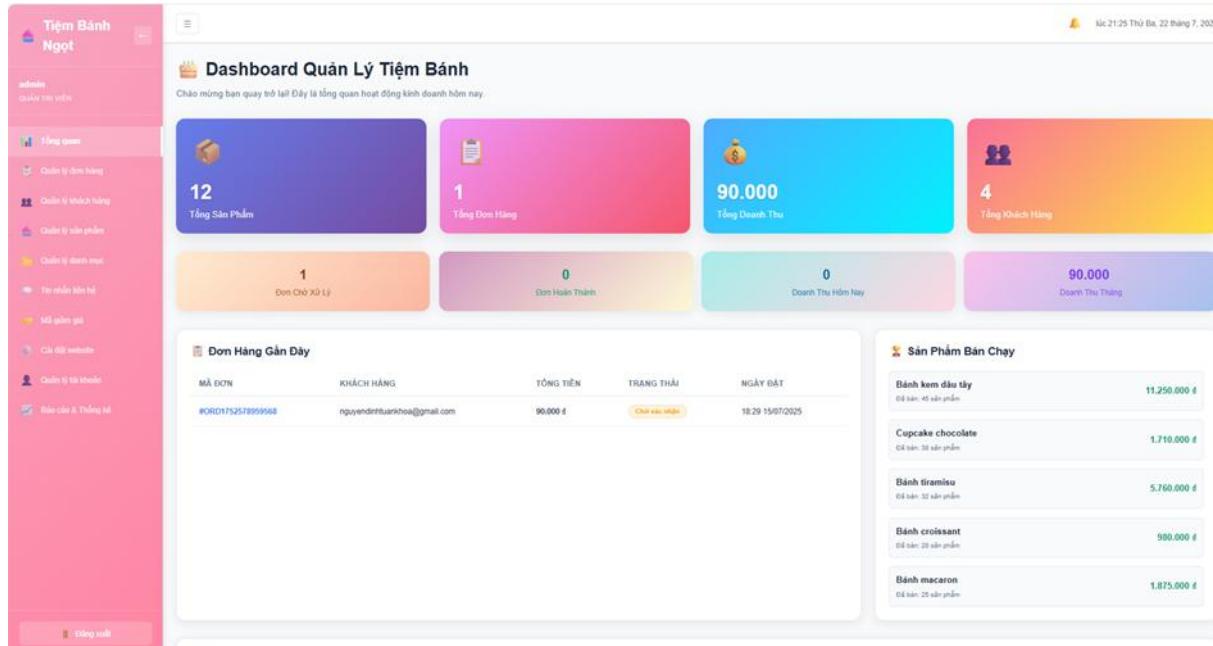
### 3.10 Thiết kế giao diện ADMIN

#### 3.10.1 Thiết kế giao diện đăng nhập



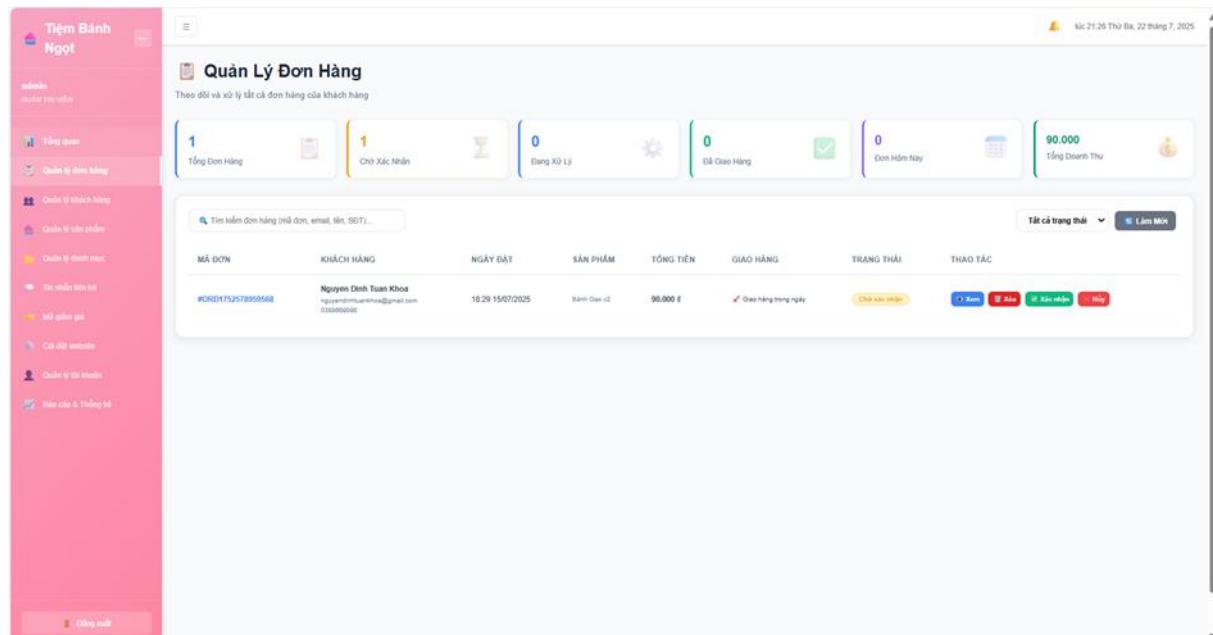
Hình 3.10.1 trang đăng nhập

### 3.10.2 Giao Diện Trang Tổng Quan



Hình 3.10.2 Giao Diện Trang Tổng Quan

### 3.10.3 Giao Diện Trang Quản Lý Đơn Hàng



Hình 3.10.3 Giao Diện Trang Quản Lý Đơn Hàng

### 3.10.4 Giao Diện Trang Quản Lý Khách Hàng

The screenshot shows the 'Quản Lý Khách Hàng' (Customer Management) section of the application. On the left sidebar, under 'Quản lý khách hàng', there are several other menu items: Tổng quan, Quản lý đơn hàng, Quản lý khách hàng, Quản lý sản phẩm, Quản lý danh mục, Tin nhắn liên hệ, Mật khẩu già, Cài đặt website, Quản lý tài khoản, and Báo cáo & Thống kê. The main area displays a summary bar with counts: 4 Khách Hàng, 1 Khách Hàng Hoạt Động, 2 Mới Tháng Nay, and 1 Tổng Đơn Hàng. Below this is a table listing 4 customers:

KHÁCH HÀNG	LIÊN HỆ	NGÀY THAM GIA	CẬP NHẬT CUỐI	DƠN HÀNG	TỔNG CHI TIỀU	DƠN CUỐI	TRẠNG THÁI	THAO TÁC
Nguyễn Văn A ID: customer1@email.com...	customer1@email.com 0912345678	15/05/2025	Chưa cập nhật	0	0 ₫	Chưa có	Không hoạt động	<button>Xem</button> <button>Xóa</button>
Trần Thị B ID: customer2@email.com...	customer2@email.com 0912345678	08/06/2025	Chưa cập nhật	0	0 ₫	Chưa có	Không hoạt động	<button>Xem</button> <button>Xóa</button>
Lê Văn C ID: customer3@email.com...	customer3@email.com 0912345678	03/07/2025	Chưa cập nhật	0	0 ₫	Chưa có	Không hoạt động	<button>Xem</button> <button>Xóa</button>
Nguyễn Đình Tuấn Khoa ID: nguyendinhthuankhoa@gmail.com...	nguyendinhthuankhoa@gmail.com 0912345678	15/07/2025	15/07/2025 Cập nhật tháng 7	1	90.000 ₫	15/07/2025	Hoạt động	<button>Xem</button> <button>Xóa</button>

Hình 3.10.4 Giao Diện Trang Quản Lý Khách Hàng

### 3.10.5 Giao Diện Trang Quản Lý Sản phẩm

The screenshot shows the 'Quản Lý Sản Phẩm' (Product Management) section of the application. On the left sidebar, under 'Quản lý sản phẩm', there are several other menu items: Quản lý danh mục, Tin nhắn liên hệ, Mật khẩu già, Cài đặt website, Quản lý tài khoản, and Báo cáo & Thống kê. The main area displays a summary bar with counts: 3 Sản Phẩm, 3 Còn Hàng, 1 Sắp hết Hàng, and 1 Hết Hàng. Below this is a table listing 3 products:

SẢN PHẨM	GIÁ	TỒN KHỐI	THAO TÁC
Bánh Gạo	6,000 ₫	Tồn kho: 20	<button>Sửa</button> <button>Xóa</button>
Bánh Sinh Nhật Màu Xanh	30,000 ₫	Tồn kho: 23	<button>Sửa</button> <button>Xóa</button>
Bánh Sinh Nhật Trái Cây	60,000 ₫	Tồn kho: 44	<button>Sửa</button> <button>Xóa</button>

Hình 3.10.5 Giao Diện Trang Quản Lý Sản phẩm

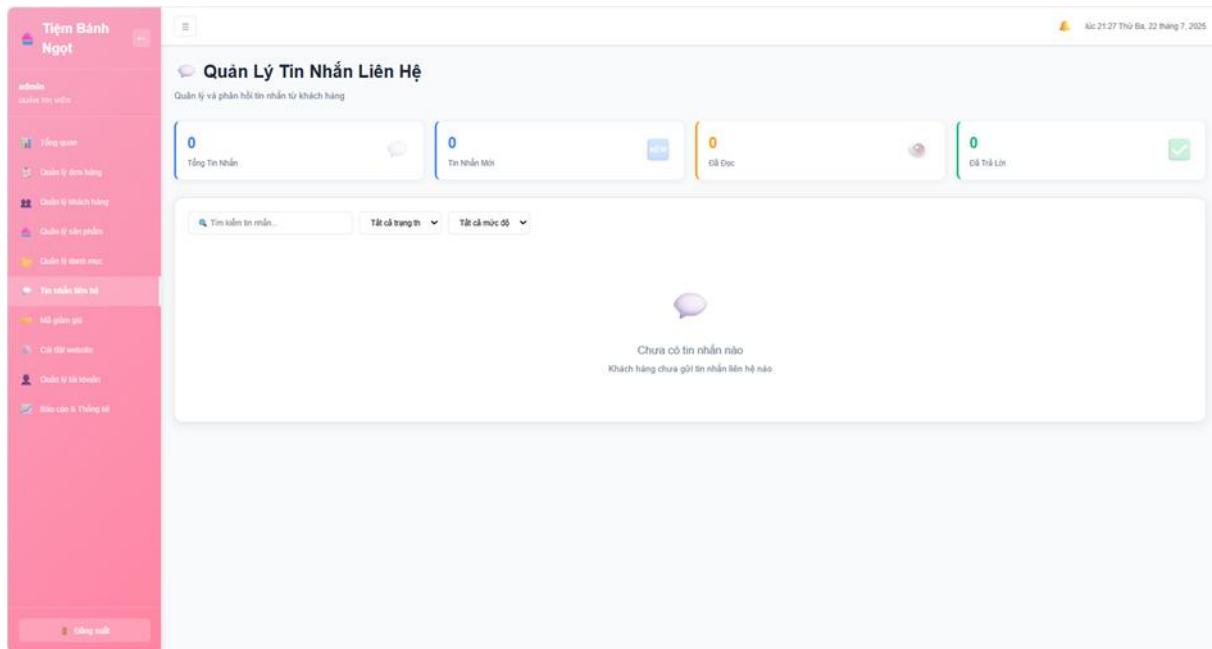
### 3.10.6 Giao Diện Trang Quản Lý Danh Mục

Hình 3.10.6 Giao Diện Trang Quản Lý Danh Mục

### 3.10.7 Giao Diện Trang Quản Lý Mã Giảm Giá

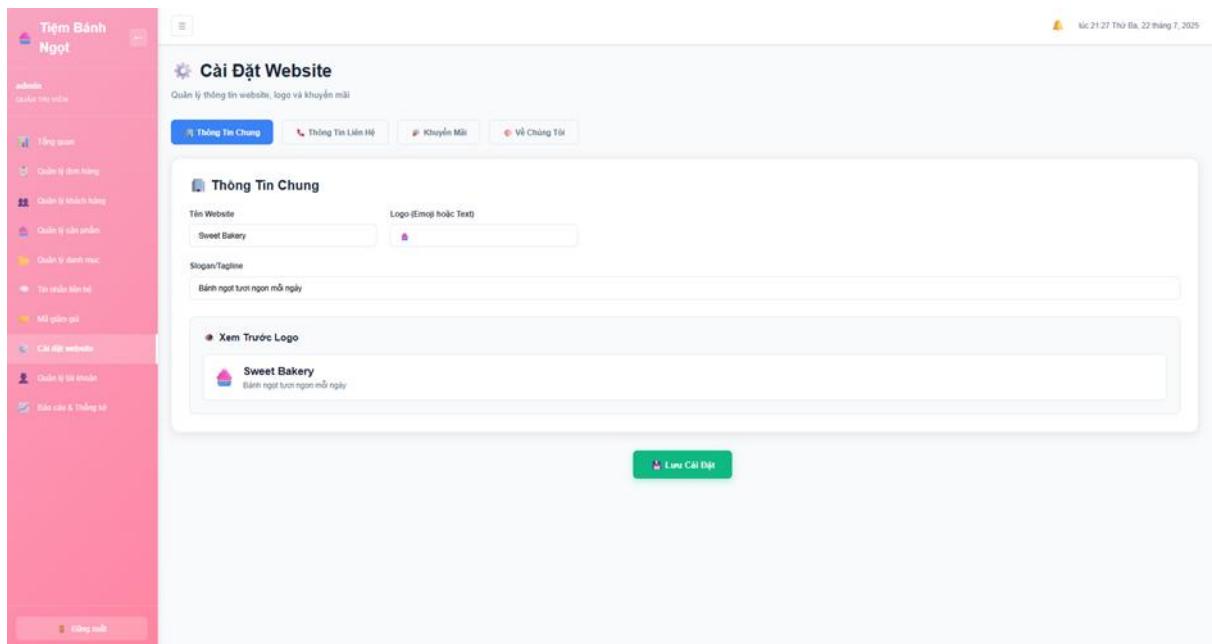
Hình 3.10.7 Giao Diện Trang Quản Lý Mã Giảm Giá

### 3.10.8 Giao Diện Trang Quản Lý Tin Nhắn Liên Hệ



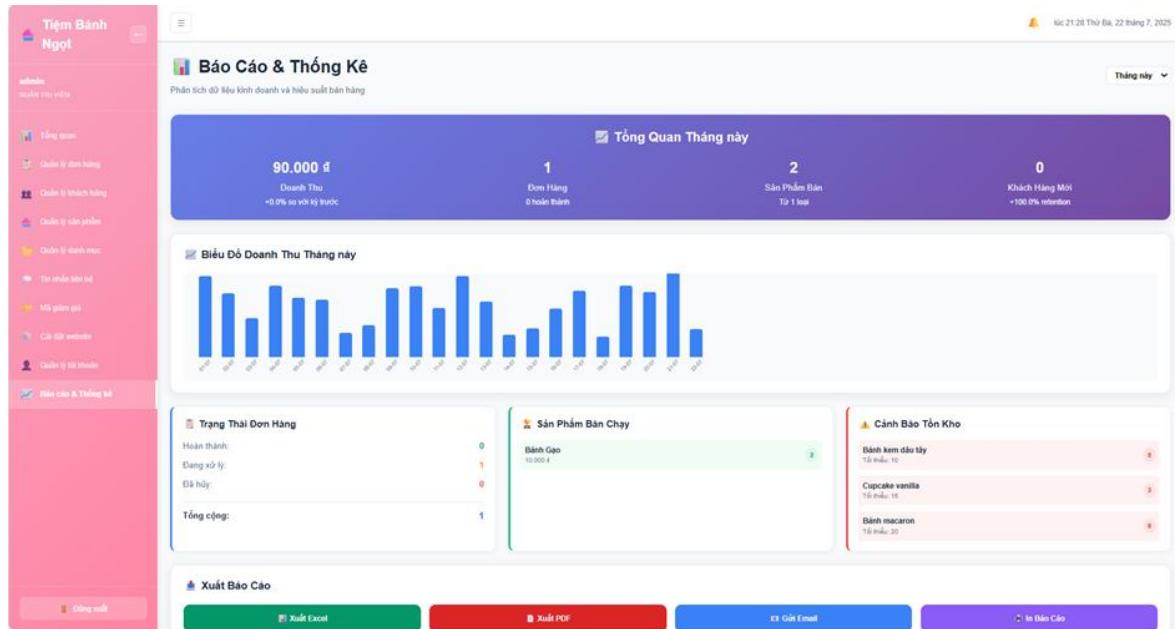
3.10.8 Giao Diện Trang Quản Lý Tin Nhắn Liên Hệ

### 3.10.9 Giao Diện Trang Quản Lý Cài Đặt



Hình 3.10.9 Giao Diện Trang Quản Lý Cài Đặt

### 3.10.10 Giao Diện Trang Báo cáo Thông Kê



*Hình 3.10.10 Giao Diện Trang Báo cáo Thông Kê*

## CHƯƠNG 4 TRIỂN KHAI CÔNG NGHỆ

### 4.1 Công Nghệ

Dự án hệ thống quản lý tiệm bánh ngọt được xây dựng trên nền tảng công nghệ hiện đại với kiến trúc microservices. Frontend sử dụng React 19 kết hợp với Vite 6.3.5 để tạo giao diện người dùng tương tác mượt mà và hiệu suất cao.

Backend được phát triển bằng Node.js và Express framework, cung cấp các **API RESTful** để xử lý logic nghiệp vụ. Hệ thống sử dụng MySQL 8.0 làm cơ sở dữ liệu chính với Sequelize ORM để quản lý dữ liệu hiệu quả.

Bảo mật được đảm bảo thông qua JWT authentication, bcrypt password hashing, và các middleware security như Helmet và CORS.

Toàn bộ hệ thống được containerized bằng Docker với Docker Compose, bao gồm 3 container độc lập cho frontend (port 5173), backend (port 5000), và database (port 3309), đảm bảo tính cô lập và khả năng triển khai linh hoạt.

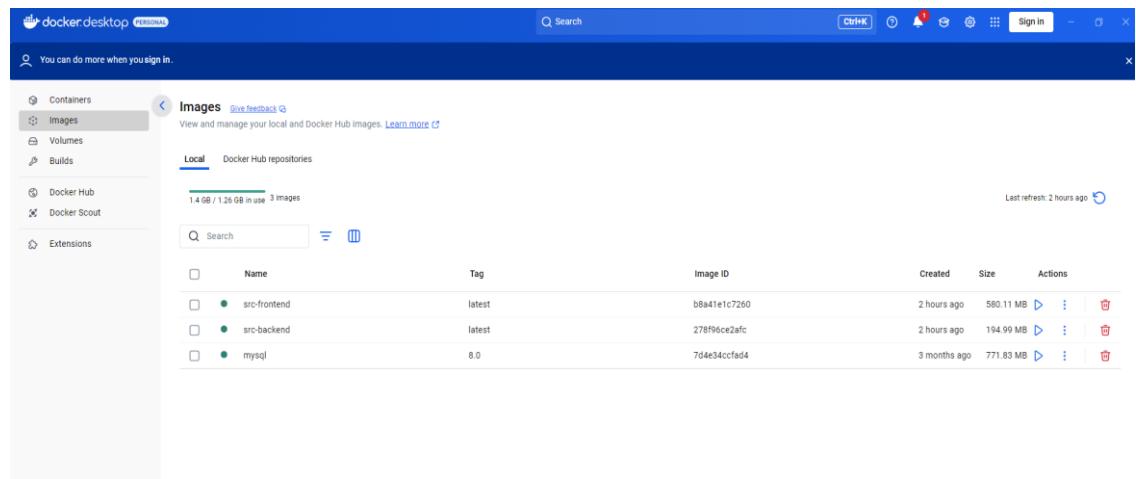
Kiến trúc này cho phép hệ thống dễ dàng mở rộng, bảo trì và triển khai trong môi trường production.

### 4.2 Docker

Frontend: Chạy độc lập trong container với port riêng, phục vụ giao diện người dùng.

Backend: Container riêng có vai trò xử lý toàn bộ logic nghiệp vụ và tương tác với cơ sở dữ liệu.

Database: Container MySQL được cấu hình volume để đảm bảo dữ liệu không mất khi container bị xoá hoặc khởi động lại.



Hình 4.2 Chạy dự án bằng docker

### 4.3 Triển khai dự án lên hosting

Dự án "Quản lý Tiệm bánh" đã được triển khai thành công trên nền tảng đám mây hiện đại, sử dụng kiến trúc tách biệt để đảm bảo hiệu suất và khả năng mở rộng. Cụ thể, quy trình triển khai được thực hiện qua các bước chính sau:

**Thiết lập Cơ sở dữ liệu:** Cơ sở dữ liệu PostgreSQL của dự án được khởi tạo trên nền tảng **Supabase**. Cấu trúc schema và các bảng dữ liệu ban đầu được thiết lập thông qua việc thực thi script SQL, đảm bảo tính toàn vẹn và sẵn sàng cho hệ thống.

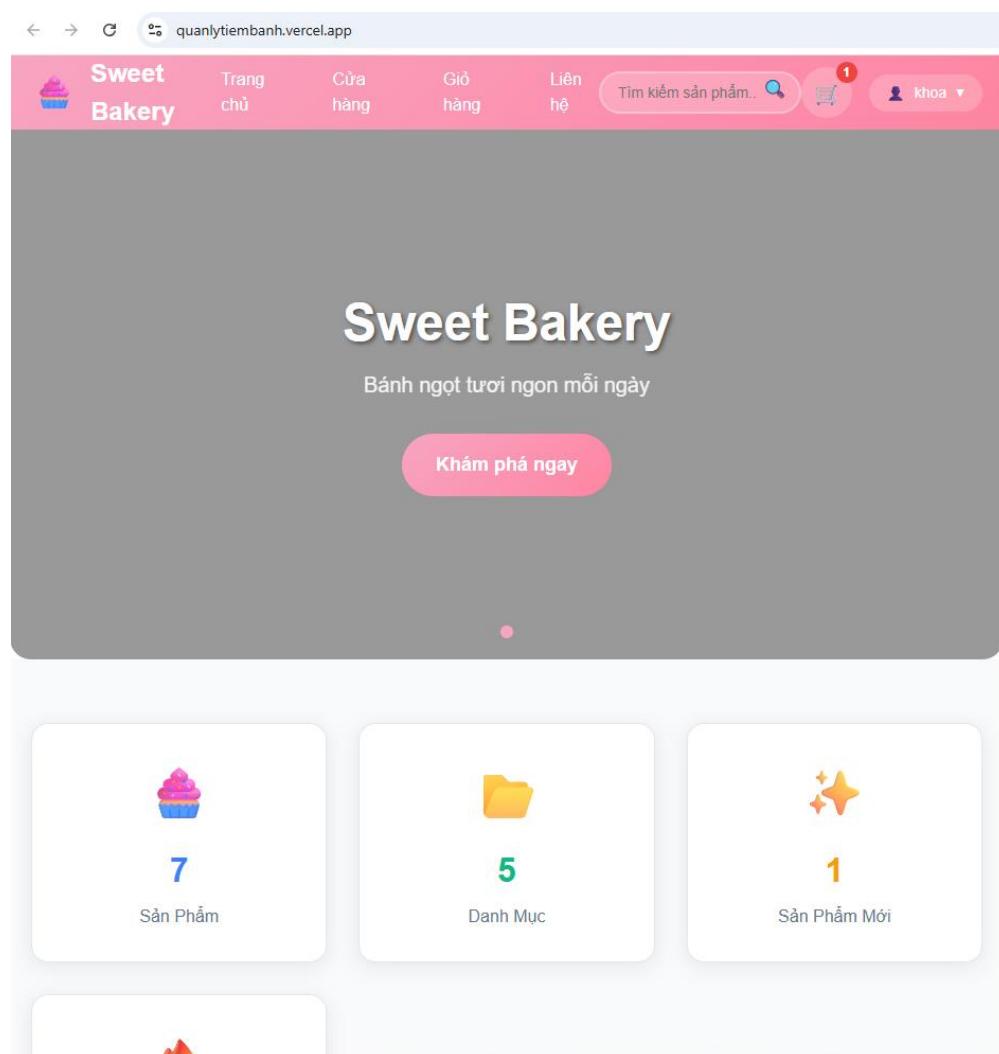
**Triển khai Backend:** Phần backend, xây dựng bằng **Node.js** và **Express**, đã được triển khai lên **Vercel** dưới dạng một Serverless Function. Các biến môi trường quan trọng như chuỗi kết nối đến database Supabase và các khóa bí mật cho xác thực (JWT) đã được cấu hình an toàn trên dashboard của Vercel để kết nối với cơ sở dữ liệu và bảo mật API.

**Triển khai Frontend:** Giao diện người dùng, được phát triển bằng **React** và **Vite**, cũng được triển khai trên **Vercel**. Quá trình này bao gồm việc cấu hình biến môi trường để trỏ đến địa chỉ API của backend, đảm bảo giao tiếp dữ liệu thông suốt.

**Tích hợp và Kiểm thử:** Để đảm bảo sự tương tác liền mạch và an toàn giữa các thành phần, chính sách CORS (Cross-Origin Resource Sharing) trên backend đã được cập nhật để chỉ cho phép các yêu cầu từ tên miền của frontend. Một quy trình kiểm thử toàn diện đã được thực hiện, bao gồm việc xác minh kết nối cơ sở dữ liệu, kiểm tra tất cả các API endpoint, và thử nghiệm các luồng chức năng người dùng trên giao diện frontend như đăng ký, đăng nhập và đặt hàng.

Kết quả là một hệ thống hoàn chỉnh, hoạt động ổn định trên môi trường production, với dữ liệu được lưu trữ an toàn trên cloud. Hệ thống tận dụng các ưu điểm của Vercel và Supabase như tự động mở rộng theo lưu lượng truy cập, cung cấp SSL/HTTPS mặc định, và cho phép giám sát hiệu năng qua các công cụ tích hợp, đảm bảo một giải pháp chuyên nghiệp và bền vững.

### Kết quả sau khi triển khai



Hình 4.3 Triển khai dự án lên hosting

## CHƯƠNG 5 Quản lý dự án bằng jira

### 5.1 Jira

Nhóm sử dụng Jira để quản lý Sprint, tạo User Story và chia task

Có bảng phân công và backlog rõ ràng.

The screenshot shows the Jira Backlog interface. At the top, there's a search bar and a 'Create' button. Below that, a navigation bar with links like Summary, Timeline, Backlog (which is selected), Board, Calendar, List, Forms, Goals, All work, Code, Archived work items, Pages, Shortcuts, Reports, and a plus sign for creating new items. The main area displays four sprints:

- QUAN Sprint 1** (19 Jul – 22 Jul): Contains one task, QUAN-59, labeled 'User Story 1: fronten Đăng nhập / Đăng ký / Đăng xuất'. Status: DONE.
- QUAN Sprint 2** (20 Jul – 22 Jul): Contains two tasks, QUAN-55 ('Là người dùng, tôi muốn xem sản phẩm của cửa hàng và mua hàng') and QUAN-96 ('về giao diện người dùng figma'). Status: DONE.
- QUAN Sprint 3** (21 Jul – 4 Aug): Contains one task, QUAN-72 ('về giao diện figma đăng nhập đăng ký'). Status: DONE.
- QUAN Sprint 6** (21 Jul – 22 Aug): Contains one task, QUAN-119 ('thiết kế trang người dùng bằng react.js'). Status: DONE.

Hình 5.1 phân chia công việc

The screenshot shows the Jira Summary page for the 'QUANLYTIEMBANHNGOT' project. At the top, there's a search bar and a 'Create' button. Below that, a navigation bar with links like Summary (selected), Timeline, Backlog, Board, Calendar, List, Forms, Goals, All work, Code, Archived work items, Pages, Shortcuts, Reports, and a plus sign for creating new items. On the left, there's a sidebar with a 'HN...' button. The main area displays several cards:

- Status overview:** Shows 103 completed work items in the last 7 days. A large pink circle indicates a total of 103 work items.
- Recent activity:** Shows 103 updated work items in the last 7 days and 0 due soon.
- Priority breakdown:** Shows a chart with a single point at 103.
- Types of work:** Shows a distribution of work items by type: Epic (0) and Task (103).

Hình 5.1 hoàn thành

## CHƯƠNG 6 KIỂM THỬ

### 6.1 Kiểm tra api bằng postman

#### 6.1.1 Kiểm thử Api đăng nhập

Kiểm thử chức năng đăng nhập trên endpoint /api/auth/login bằng công cụ Postman, nhằm xác minh quy trình xác thực người dùng và tính toàn vẹn của dữ liệu trả về.

##### Cấu hình Kiểm thử:

- Công cụ: Postman
- Môi trường: Server local (<http://localhost:5000>) được khởi chạy bằng docker-compose.
- Thông số Request:
- Phương thức: POST
- URL: <http://localhost:5000/api/auth/login>
- Headers: Content-Type: application/json

Body (JSON):

JSON

```
{  
    "username": "admin",  
    "password": "admin123"  
}
```

### Kết quả:

The screenshot shows a POST request to `http://localhost:5000/api/auth/login`. The Body tab contains the following JSON payload:

```
1 {  
2   "username": "admin",  
3   "password": "admin123"  
4 }  
5
```

The response status is 200 OK, with a response time of 420 ms and a response size of 807 B. The response body is:

```
1 {  
2   "message": "Đăng nhập thành công!",  
3   "user": {  
4     "id": 1000,  
5     "username": "admin",  
6     "email": "admin@tiembanh.com",  
7     "role": "admin"  
8   }  
9 }
```

Hình 6.1.1 Kiểm thử Api đăng nhập

### 6.1.2 Kiểm thử Api Sản Phẩm

- GET - Lấy tất cả sản phẩm (Public)
- Method: GET
- URL: `http://localhost:5000/api/products`
- Headers: Không cần
- Body: Không cần
- GET - Lấy sản phẩm theo ID (Public)
- Method: GET
- URL: `http://localhost:5000/api/products/1`
- Headers: Không cần
- Body: Không cần

- POST - Tạo sản phẩm mới (Admin only)
- Method: POST
- URL: http://localhost:5000/api/products
  - Headers:
  - Content-Type: application/json
  - Authorization: Bearer YOUR\_TOKEN

## Kết quả

The screenshot shows a Postman request to `http://localhost:5000/api/categories`. The 'Body' tab is selected, showing a JSON payload for creating a new category. The response tab shows a successful JSON response with a message about fetching category data and a list of categories, including one for 'Banh Ngot'.

```
1 {  
2   "success": true,  
3   "message": "Lấy danh sách danh mục thành công",  
4   "data": [  
5     {  
6       "id": 1,  
7       "name": "Bánh Ngót",  
8       "slug": "banh-ngot",  
9       "description": "Các loại bánh ngọt thơm ngon",  
10      "image": null,  
11      "parent_id": null,  
12      "sort_order": 1,  
13      "is_featured": 1,  
14      "created_at": "2025-07-23T04:16:24.000Z",  
15      "updated_at": "2025-07-23T04:16:24.000Z",  
16      "status": "active",  
17      "product_count": 2  
18    },  
19  ]  
}
```

Hình 6.1.2 Kiểm thử API Sản Phẩm

## 6.2 Kiểm tra docker

Docker Health Checks: Monitoring tự động tình trạng containers

Docker Compose Testing: Kiểm tra inter-service communication

Volume Persistence Testing: Đảm bảo dữ liệu không bị mất khi restart

## CHƯƠNG 7 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 7.1 TỔNG KẾT DỰ ÁN

Dự án hệ thống quản lý tiệm bánh ngọt đã được hoàn thành thành công với đầy đủ các chức năng cốt lõi phục vụ cho việc quản lý hoạt động kinh doanh của một tiệm bánh. Hệ thống được xây dựng trên nền tảng công nghệ hiện đại với kiến trúc microservices, sử dụng React 19, Node.js/Express, MySQL và Docker, đảm bảo tính ổn định, bảo mật và khả năng mở rộng cao.

#### Về mặt kỹ thuật

Xây dựng thành công hệ thống với kiến trúc 3-tier được containerized hoàn toàn

Triển khai authentication và authorization system bảo mật với JWT

Phát triển RESTful API đầy đủ với documentation và testing

Tích hợp thành công các tính năng upload file, export Excel

Đạt được hiệu suất cao với Vite build tool và React 19

#### Về mặt chức năng

Hoàn thành đầy đủ các module: quản lý sản phẩm, đơn hàng, khách hàng, danh mục

Xây dựng giao diện người dùng thân thiện cho cả admin và customer

Triển khai hệ thống giờ hàng và thanh toán hoàn chỉnh

Phát triển tính năng báo cáo và thống kê chi tiết

Tích hợp hệ thống thông báo real-time

#### Về mặt vận hành

Thiết lập môi trường development và production với Docker

Xây dựng quy trình testing toàn diện với Postman và automated tests

Tạo documentation chi tiết cho API và hướng dẫn sử dụng

Thiết lập backup và recovery procedures cho database

## 7.2 KẾT LUẬN

Dự án hệ thống quản lý tiệm bánh ngọt đã được hoàn thành thành công với đầy đủ các chức năng cốt lõi phục vụ quản lý hoạt động kinh doanh. Hệ thống được xây dựng trên nền tảng công nghệ hiện đại với React 19, Node.js/Express, MySQL và Docker, đảm bảo tính ổn định, bảo mật và khả năng mở rộng.

Các module chính bao gồm quản lý sản phẩm, đơn hàng, khách hàng và báo cáo đã hoạt động ổn định với hiệu suất cao. Hệ thống authentication/authorization được triển khai bảo mật với JWT, giao diện người dùng responsive và thân thiện, cùng với quy trình testing toàn diện đảm bảo chất lượng sản phẩm. Tuy nhiên, hệ thống vẫn còn một số hạn chế như chưa tích hợp cổng thanh toán thực tế, thiếu tính năng real-time notifications và chưa có ứng dụng mobile native.

## 7.3 HƯỚNG PHÁT TRIỂN

Dựa trên nền tảng vững chắc đã có, định hướng phát triển dự án sẽ tập trung vào việc giải quyết các hạn chế và gia tăng giá trị. Trước mắt, hệ thống sẽ ưu tiên nâng cao trải nghiệm khách hàng và tối ưu hóa vận hành thông qua việc tích hợp cổng thanh toán trực tuyến và hệ thống thông báo real-time. Giai đoạn tiếp theo sẽ mở rộng quy mô tiếp cận thị trường bằng việc phát triển ứng dụng di động, đồng thời khai thác dữ liệu kinh doanh thông minh qua các báo cáo phân tích sâu và chương trình khách hàng thân thiết. Về dài hạn, dự án sẽ được hiện đại hóa kiến trúc kỹ thuật để đảm bảo khả năng mở rộng bền vững, hướng tới mục tiêu trở thành một giải pháp quản lý toàn diện và có tính cạnh tranh cao.

Một trong những ưu tiên hàng đầu trong lộ trình phát triển sắp tới là tích hợp các cổng thanh toán trực tuyến. Hệ thống sẽ được kết nối với các dịch vụ phổ biến tại Việt Nam như MoMo, ZaloPay, VNPay, cùng với các loại thẻ ngân hàng (Visa/Mastercard), nhằm mang lại sự tiện lợi, nhanh chóng và an toàn cho khách hàng trong quá trình mua sắm. Việc tích hợp này không chỉ giúp tự động hóa khâu xác nhận đơn hàng, giảm thiểu sai sót và gánh nặng vận hành thủ công, mà còn nâng cao tỷ lệ chốt đơn thành công và xây dựng hình ảnh chuyên nghiệp cho tiệm bánh.

## DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Trần Đình Quê, *Giáo trình phân tích và thiết kế hệ thống thông tin*, Đại học Công nghệ Sài Gòn.
- [2] Phạm Minh Đương, *Tài liệu giảng dạy môn phân tích và thiết kế hệ thống thông tin*, Đại học Trà Vinh.
- [3] Phan Thị Phương Nam, Tài liệu giảng dạy môn hệ quản trị cơ sở dữ liệu, Đại học Trà Vinh.