

**BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC ĐIỆN LỰC
KHOA CÔNG NGHỆ THÔNG TIN**



BÁO CÁO KẾT THÚC HỌC PHẦN

NGÔN NGỮ KỊCH BẢN

WEBSITE BÁN ĐỒ ĂN NHANH FASTFOOD SHOP

Giảng viên hướng dẫn	:Đỗ Đức Cường
Sinh viên thực hiện	:Nguyễn Tiên Thanh Phạm Ngọc Huyền
Ngành	:Công nghệ thông tin
Chuyên ngành	:Công nghệ phần mềm
Lớp	:D17CNPM6
Khóa	:2022 - 2027

Hà Nội, tháng 11 năm 2025

LỜI CẢM ƠN

Trong bài báo cáo chuyên đề này, chúng em xin gửi lời cảm ơn sâu sắc và lòng biết ơn chân thành đến những người đã đóng góp và hỗ trợ chúng em trong quá trình nghiên cứu và viết bài báo cáo này.

Đầu tiên, chúng em muốn bày tỏ lòng biết ơn đến giảng viên hướng dẫn của chúng em, thầy Đỗ Đức Cường. Sự tận tâm và kiến thức sâu rộng của thầy đã giúp định hình và chỉ dẫn cho chúng em trong quá trình nghiên cứu. Những ý kiến xây dựng và hướng dẫn của thầy đã cung cấp cho chúng em một cơ sở vững chắc để phát triển ý tưởng và triển khai nghiên cứu của mình.

Chúng em cũng muốn gửi lời cảm ơn đến các thành viên trong ban giảng dạy và các chuyên gia đã dành thời gian để đọc và đánh giá bài báo cáo chuyên đề của chúng em. Sự đánh giá chuyên môn và phản hồi xây dựng từ thầy cô đã giúp chúng em nắm bắt những khía cạnh quan trọng và cải thiện chất lượng của bài viết.

Một lần nữa, chúng em xin chân thành cảm ơn tất cả những người đã đóng góp và hỗ trợ chúng em trong quá trình nghiên cứu và viết bài báo cáo chuyên đề này. Sự giúp đỡ của các bạn đã làm cho dự án này trở thành hiện thực và mang lại những kết quả mà chúng em tự hào.

Chúng em xin chân thành cảm ơn!

PHIẾU CHẤM ĐIỂM

Danh sách sinh viên tham gia:

Họ và Tên	Nội dung thực hiện	Điểm	Chữ ký	Ghi chú
Nguyễn Tiến Thanh 22810310382	Xây dựng admin, backend, deploy website			
Phạm Ngọc Huyền 22810310376	Xây dựng cơ sở dữ liệu, thiết kế frontend			

Danh sách giảng viên chấm thi:

Họ và Tên	Chữ ký	Ghi chú

MỤC LỤC

Trang

LỜI CẢM ƠN	
PHIẾU CHẤM ĐIỂM.....	
MỤC LỤC.....	
DANH MỤC HÌNH ẢNH, BẢNG BIỂU.....	
PHẦN MỞ ĐẦU	1
CHƯƠNG 1: TỔNG QUAN VỀ NGÔN NGỮ KỊCH BẢN	2
1.1 Khái niệm về ngôn ngữ kịch bản	2
1.1.1 Định nghĩa.....	2
1.1.2 Vai trò trong phát triển Web	2
1.2 Giới thiệu về JavaScript.....	3
1.2.1 Lịch sử và đặc điểm	3
1.2.2 Mô hình Bất đồng bộ (Asynchronous Model).....	5
CHƯƠNG 2: CÁC CÔNG NGHỆ CỐT LÕI TRONG HỆ THỐNG.....	6
2.1 Node.js và Express.js	6
2.1.1 Node.js: Môi trường Runtime JavaScript	6
2.1.2 Express.js: Web Framework cho Node.js	7
2.2 Cơ sở dữ liệu MongoDB.....	8
2.3 React.js.....	9
2.3.1 Kiến trúc dựa trên Component (Component-Based Architecture) ...	9
2.3.2 Virtual DOM (Document Object Model Ảo).....	9
2.3.3 Quản lý trạng thái (State Management)	10
2.4 Tương tác giữa các thành phần (Data Flow).....	10
CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG.....	12
3.1 Phân tích chức năng	12

3.1.1 Chức năng ở Frontend.....	12
3.1.2 Chức năng Admin	13
3.2 Thiết kế kiến trúc hệ thống.....	13
3.2.1 Sơ đồ Kiến trúc	13
3.2.2 Sơ đồ Luồng Dữ liệu.....	14
3.3 Thiết kế cơ sở dữ liệu.....	15
3.3.1 Collection Users	15
3.3.2 Collection Categories	15
3.3.3 Collection Foods	16
3.3.4 Collection Orders	16
3.4 Thiết kế API RESTful	16
CHƯƠNG 4: THIẾT KẾ GIAO DIỆN	18
4.1 Thiết kế giao diện người dùng	18
4.1.1 Trang đăng ký, đăng nhập.....	18
4.1.2 Trang chủ.....	19
4.1.3 Trang About Us	19
4.1.4 Trang Menu	20
4.1.5 Trang Cart.....	20
4.1.6 Trang Checkout	21
4.1.7 Trang Orders.....	21
4.2 Thiết kế giao diện Admin.....	22
4.2.1 Trang đăng nhập.....	22
4.2.2 Trang thêm sản phẩm	22
4.2.3 Trang danh sách sản phẩm	23
4.2.4 Trang quản lý danh mục.....	23
4.2.5 Trang quản lý đơn hàng.....	24
4.2.6 Trang phân tích thống kê.....	24

PHẦN KẾT LUẬN	25
DANH MỤC TÀI LIỆU THAM KHẢO	26

DANH MỤC HÌNH ẢNH, BẢNG BIỂU

	Trang
Hình 3.1: Collection Users	15
Hình 3.2: Collection Categories	15
Hình 3.3: Collection Foods	16
Hình 3.4: Collection Orders	16
Hình 4.1: Trang đăng nhập.....	18
Hình 4.2: Trang đăng ký.....	18
Hình 4.3: Trang chủ.....	19
Hình 4.4: Trang About Us	19
Hình 4.5: Trang Menu	20
Hình 4.6: Trang Cart	20
Hình 4.7: Trang Checkout.....	21
Hình 4.8: Trang Orders	21
Hình 4.9: Trang đăng nhập.....	22
Hình 4.10: Trang thêm sản phẩm	22
Hình 4.11: Trang danh sách sản phẩm	23
Hình 4.12: Trang quản lý danh mục.....	23
Hình 4.13: Trang quản lý đơn hàng	24
Hình 4.14: Trang phân tích thống kê.....	24
Bảng 1.1: Đặc điểm của JavaScript	4
Bảng 2.1: Đặc điểm của Node.js.....	6
Bảng 2.2: Đặc điểm của Express.js.....	7
Bảng 2.3: Đặc điểm của MongoDB	8
Bảng 2.4: Quản lý trạng thái	10

Bảng 3.1: Chức năng ở Frontend	12
Bảng 3.2: Chức năng Admin	13
Bảng 3.3: Thiết kế API RESTful.....	17

PHẦN MỞ ĐẦU

Trong bối cảnh đô thị hóa nhanh chóng và sự phát triển mạnh mẽ của thương mại điện tử, nhu cầu đặt đồ ăn trực tuyến của người tiêu dùng ngày càng tăng cao, đòi hỏi các cửa hàng Fast Food phải chuyển đổi số để tối ưu hóa dịch vụ. Việc sở hữu một nền tảng website chuyên nghiệp, tiện lợi đã trở thành yếu tố then chốt để duy trì và mở rộng kinh doanh trong ngành ẩm thực hiện đại.

Đề tài "Website Bán Đồ Ăn Nhanh FastFood Shop" được triển khai nhằm xây dựng một hệ thống E-commerce hoàn chỉnh dựa trên bộ công nghệ MERN Stack (MongoDB, Express.js, React.js, Node.js). Ứng dụng này sẽ cung cấp một giải pháp công nghệ toàn diện, bao gồm: giao diện đặt hàng trực quan cho khách hàng (Front-end) và hệ thống quản lý đơn hàng/sản phẩm hiệu quả cho người bán (Back-end).

Mục tiêu chính là tạo ra một nền tảng linh hoạt, giúp tối ưu hóa quy trình bán hàng, nâng cao trải nghiệm mua sắm và gia tăng khả năng cạnh tranh cho doanh nghiệp Fast Food.

CHƯƠNG 1: TỔNG QUAN VỀ NGÔN NGỮ KỊCH BẢN

1.1 Khái niệm về ngôn ngữ kịch bản

1.1.1 Định nghĩa

Ngôn ngữ kịch bản (Scripting Language) là một loại ngôn ngữ lập trình được thiết kế để điều khiển hoạt động của một chương trình ứng dụng hoặc hệ điều hành. Khác biệt cơ bản so với các ngôn ngữ biên dịch truyền thống (như C++ hay Java) là các ngôn ngữ kịch bản thường được thông dịch (interpreted) ngay khi chạy thay vì được biên dịch thành mã máy (machine code) trước đó.

- **Tính linh hoạt:** Cho phép phát triển nhanh chóng, dễ dàng thay đổi và thử nghiệm.

- **Mục đích:** Thường được sử dụng để tự động hóa các tác vụ, quản lý ứng dụng, hoặc tạo ra các chức năng động (dynamic) cho giao diện người dùng, đặc biệt trong lĩnh vực phát triển Web.

1.1.2 Vai trò trong phát triển Web

Trong phát triển Web, ngôn ngữ kịch bản đóng vai trò thiết yếu trong việc tạo ra sự tương tác và tính động cho các trang web. Ban đầu, các ngôn ngữ kịch bản như JavaScript được sử dụng chủ yếu ở phía máy khách (Client-side) để thao tác với mô hình đối tượng tài liệu (DOM) và phản hồi lại hành động của người dùng. Ngày nay, với sự ra đời của các nền tảng như Node.js, ngôn ngữ

kịch bản đã mở rộng sang phía máy chủ (Server-side), cho phép sử dụng cùng một ngôn ngữ cho cả hai môi trường.

1.2 Giới thiệu về JavaScript

1.2.1 Lịch sử và đặc điểm

JavaScript (JS) là ngôn ngữ kịch bản phổ biến nhất và là ngôn ngữ cốt lõi của Web. Ban đầu được thiết kế để thêm tính tương tác vào các trang web tĩnh, JS đã phát triển vượt bậc, trở thành một ngôn ngữ lập trình đa mô hình (Multi-paradigm), hỗ trợ Lập trình hướng đối tượng (Object-Oriented Programming - OOP), Lập trình chức năng (Functional Programming) và Lập trình dựa trên nguyên mẫu (Prototype-based Programming).

Bảng 1.1: Đặc điểm của JavaScript

Đặc điểm	Mô tả
Tính thông dịch (Interpreted)	JavaScript thường được thông dịch ngay khi chạy bởi Engine của trình duyệt (ví dụ: V8 của Chrome) hoặc môi trường runtime (ví dụ: Node.js). Điều này giúp quá trình phát triển và gỡ lỗi nhanh chóng hơn.
Ngôn ngữ động (Dynamically Typed)	Kiểu dữ liệu của biến không cần khai báo tường minh và có thể thay đổi trong quá trình thực thi. Ví dụ: một biến có thể giữ giá trị số và sau đó giữ giá trị chuỗi, mang lại tính linh hoạt cao nhưng cũng đòi hỏi cẩn trọng hơn trong quản lý kiểu dữ liệu.
Tiêu chuẩn ECMAScript (ES)	JavaScript được chuẩn hóa bởi tổ chức ECMA International với tên gọi ECMAScript. Các bản cập nhật hàng năm (như ES6/ES2015 và các phiên bản tiếp theo) đã bổ sung nhiều cú pháp và tính năng hiện đại như let, const, mũi tên hàm (Arrow Functions), và async/await, giúp cải thiện đáng kể khả năng viết code sạch và hiệu quả.

1.2.2 Mô hình Bất đồng bộ (Asynchronous Model)

Một trong những đặc điểm quan trọng nhất của JavaScript, đặc biệt là trong môi trường Node.js (Back-end), là mô hình hoạt động đơn luồng (Single-threaded) và bất đồng bộ (Non-blocking I/O).

- **Single Threaded:** JavaScript thực thi trên một luồng duy nhất (main thread).

- **Non-blocking I/O:** Để tránh bị tắc nghẽn khi thực hiện các tác vụ tốn thời gian như truy vấn cơ sở dữ liệu (Database access) hay gọi API (Input/Output - I/O), JavaScript sử dụng:

- + **Callback Functions:** Các hàm được gọi lại sau khi một tác vụ bất đồng bộ hoàn thành.

- + **Promises:** Một đối tượng đại diện cho một giá trị có thể có trong tương lai, cho phép xử lý chuỗi các hoạt động bất đồng bộ một cách có tổ chức hơn.

- + **Async/Await:** Cú pháp hiện đại cho phép viết code bất đồng bộ nhìn giống như code đồng bộ, giúp dễ đọc và quản lý hơn.

Mô hình này cho phép Node.js xử lý hàng nghìn yêu cầu đồng thời (Concurrency) mà không cần tạo ra nhiều luồng, làm cho nó trở nên lý tưởng để xây dựng các API hiệu suất cao cho ứng dụng.

CHƯƠNG 2: CÁC CÔNG NGHỆ CỐT LÕI TRONG HỆ THỐNG

2.1 Node.js và Express.js

2.1.1 Node.js: Môi trường Runtime JavaScript

Node.js không phải là một ngôn ngữ hay Framework, mà là một môi trường thực thi (Runtime Environment) cho JavaScript ở phía máy chủ (Server-side).

Bảng 2.1: Đặc điểm của Node.js

Đặc điểm	Mô tả
Được xây dựng trên V8	Node.js sử dụng Engine V8 JavaScript của Google Chrome, nổi tiếng về tốc độ thực thi code JavaScript.
Mô hình I/O Bất đồng bộ, Không chặn (Non-blocking I/O)	Đây là đặc điểm then chốt giúp Node.js nổi bật. Thay vì chờ đợi các thao tác I/O (như truy vấn cơ sở dữ liệu, đọc file) hoàn tất, Node.js sẽ tiếp nhận yêu cầu mới, và sử dụng vòng lặp sự kiện (Event Loop) để xử lý kết quả I/O khi chúng sẵn sàng. Điều này giúp Node.js xử lý một lượng lớn kết nối đồng thời một cách hiệu quả, rất phù hợp

	cho ứng dụng FastFood có lưu lượng truy cập cao.
Hệ sinh thái NPM (Node Package Manager)	NPM là kho lưu trữ phần mềm lớn nhất thế giới, cung cấp hàng trăm nghìn module và thư viện JavaScript (ví dụ: bcrypt để mã hóa mật khẩu, jsonwebtoken để tạo token xác thực) giúp tăng tốc độ phát triển.

2.1.2 Express.js: Web Framework cho Node.js

Express.js là một Web Application Framework tối giản và linh hoạt, được xây dựng trên Node.js. Nó đóng vai trò là xương sống cho việc xây dựng API (Application Programming Interface) của dự án.

Bảng 2.2: Đặc điểm của Express.js

Đặc điểm	Mô tả
Xây dựng RESTful API	Express.js cung cấp các công cụ mạnh mẽ để định tuyến (Routing), cho phép định nghĩa các điểm cuối (Endpoints) theo tiêu chuẩn REST (ví dụ: GET /api/products, POST /api/orders).
Middleware	Express.js sử dụng các hàm Middleware để xử lý các yêu cầu HTTP. Middleware có thể thực hiện

	các tác vụ như ghi log, kiểm tra xác thực người dùng, phân tích cú pháp dữ liệu yêu cầu (body-parser), trước khi yêu cầu đó đến được logic nghiệp vụ chính.
--	---

2.2 Cơ sở dữ liệu MongoDB

MongoDB là cơ sở dữ liệu NoSQL (Not Only SQL) mã nguồn mở, được chọn cho dự án này nhờ vào tính linh hoạt và khả năng mở rộng cao.

Bảng 2.3: Đặc điểm của MongoDB

Đặc điểm	Mô tả
Mô hình Tài liệu (Document Model)	MongoDB lưu trữ dữ liệu dưới dạng các tài liệu BSON (Binary JSON), thay vì bảng và hàng cố định như cơ sở dữ liệu quan hệ (SQL). Điều này cho phép lưu trữ các cấu trúc dữ liệu phức tạp (như một đơn hàng với nhiều mục sản phẩm) một cách tự nhiên và linh hoạt.
Linh hoạt về Schema	Không cần định nghĩa cấu trúc (Schema) cố định trước. Điều này rất hữu ích trong một ứng dụng thương mại điện tử như FastFood Shop, nơi cấu trúc

	sản phẩm hoặc đơn hàng có thể thay đổi nhanh chóng.
Đồng nhất với JavaScript	Việc sử dụng định dạng JSON/BSON giúp dữ liệu truyền tải giữa MongoDB (Lưu trữ), Express.js (Server) và React.js (Client) diễn ra liền mạch mà không cần chuyển đổi kiểu dữ liệu phức tạp.

2.3 React.js

React.js là một thư viện JavaScript được phát triển bởi Facebook, dùng để xây dựng giao diện người dùng (User Interface - UI). Nó là trái tim của giao diện khách hàng trong dự án FastFood Shop.

2.3.1 Kiến trúc dựa trên Component (Component-Based Architecture)

- **Tái sử dụng (Reusability):** React khuyến khích chia giao diện thành các thành phần độc lập, có thể tái sử dụng gọi là Components (ví dụ: Component <ProductCard>, Component <Header>, Component <ShoppingCart>). Điều này giúp mã nguồn dễ quản lý, bảo trì và mở rộng hơn.

- **Logic đóng gói:** Mỗi Component chứa cả logic (JavaScript) và giao diện (JSX), tạo thành một đơn vị khép kín.

2.3.2 Virtual DOM (Document Object Model Ảo)

- **Tối ưu hiệu suất:** Thay vì thao tác trực tiếp với DOM thật (thường chậm), React sử dụng một bản sao trong bộ nhớ gọi là Virtual DOM. Khi dữ

liệu (State) thay đổi, React sẽ tạo ra một Virtual DOM mới và so sánh nó với bản trước đó để tìm ra sự khác biệt tối thiểu.

- **Cập nhật hiệu quả:** Chỉ phần bị thay đổi tối thiểu đó mới được áp dụng vào DOM thật, giúp tối ưu hóa hiệu suất và mang lại trải nghiệm người dùng nhanh chóng, mượt mà.

2.3.3 Quản lý trạng thái (State Management)

React sử dụng cơ chế State và Props để quản lý dữ liệu.

Bảng 2.4: Quản lý trạng thái

Trạng thái	Mô tả
State	Dữ liệu nội bộ của một Component, có thể thay đổi theo thời gian (ví dụ: số lượng món ăn trong giỏ hàng).
Props	Dữ liệu được truyền từ Component cha xuống Component con (để hiển thị thông tin sản phẩm).
Hooks (Ví dụ: useState, useEffect)	Các hàm đặc biệt trong React cho phép sử dụng các tính năng của React như State mà không cần viết dưới dạng class Component, giúp code gọn gàng và dễ hiểu hơn.

2.4 Tương tác giữa các thành phần (Data Flow)

Kiến trúc MERN Stack hoạt động theo mô hình Client-Server điển hình.

1. **Người dùng tương tác:** Người dùng sử dụng giao diện React.js để thực hiện hành động (ví dụ: nhấn nút "Thêm vào giỏ hàng").
 2. **Yêu cầu API (Request):** Component React sử dụng các thư viện như Axios hoặc fetch để gửi yêu cầu HTTP (GET, POST, PUT, DELETE) đến Server Node.js/Express.js.
 3. **Xử lý Server:** Express.js nhận yêu cầu, định tuyến nó đến hàm xử lý phù hợp.
 4. **Truy vấn DB:** Hàm xử lý (Controller) sử dụng thư viện Mongoose (một thư viện ODM - Object Data Modeling cho Node.js và MongoDB) để tương tác với MongoDB.
 5. **Phản hồi DB:** MongoDB trả về dữ liệu (dưới dạng BSON, được Express.js chuyển thành JSON).
 6. **Phản hồi API (Response):** Express.js gửi phản hồi JSON chứa dữ liệu hoặc thông báo trạng thái trở lại cho Front-end React.
 7. **Cập nhật UI:** React nhận phản hồi JSON, cập nhật State của Component, và sử dụng Virtual DOM để cập nhật giao diện người dùng một cách hiệu quả, hiển thị kết quả cho người dùng.
- Mô hình tạo thành một chu trình khép kín, đảm bảo tính động và khả năng tương tác cao.

CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

3.1 Phân tích chức năng

Hệ thống Website Bán Đồ Ăn Nhanh được chia thành hai nhóm người dùng chính: Khách hàng (Client) và Quản trị viên (Admin).

3.1.1 Chức năng ở Frontend

Bảng 3.1: Chức năng ở Frontend

Mã	Chức năng	Mô tả chi tiết
C1	Quản lý Tài khoản	Đăng ký, Đăng nhập/Đăng xuất (sử dụng JWT), xem và chỉnh sửa thông tin cá nhân.
C2	Duyệt Sản phẩm	Xem danh sách sản phẩm theo danh mục (Burgers, Drinks, Sides), tìm kiếm và lọc sản phẩm.
C3	Quản lý Giỏ hàng	Thêm/Bớt/Cập nhật số lượng sản phẩm trong giỏ hàng.
C4	Thanh toán & Đặt hàng	Xem tổng đơn hàng, nhập thông tin giao hàng, thanh toán đơn hàng
C5	Theo dõi đơn hàng	Xem lịch sử đơn hàng đã đặt và theo dõi trạng thái hiện tại (Đang xử lý, Đang giao, Đã hoàn thành, Hủy).

3.1.2 Chức năng Admin

Bảng 3.2: Chức năng Admin

Mã	Chức năng	Mô tả
A1	Quản lý Sản phẩm	Thêm, Chỉnh sửa, Xóa sản phẩm; Cập nhật giá, mô tả và hình ảnh.
A2	Quản lý Danh mục	Thêm, Chỉnh sửa, Xóa các danh mục sản phẩm.
A3	Quản lý Đơn hàng	Xem chi tiết tất cả đơn hàng, cập nhật trạng thái đơn hàng.
A4	Thống kê Báo cáo	Xem các số liệu cơ bản như doanh thu, số lượng đơn hàng, sản phẩm bán chạy.

3.2 Thiết kế kiến trúc hệ thống

3.2.1 Sơ đồ Kiến trúc

Hệ thống tuân thủ kiến trúc ba lớp cơ bản (Presentation, Logic, Data Access), được hiện thực hóa thông qua MERN Stack và giao tiếp qua API RESTful.

- **Lớp Trình bày (Presentation Layer):** Được đảm nhiệm bởi React.js. Chịu trách nhiệm hiển thị giao diện, quản lý tương tác người dùng, và gửi yêu cầu đến Back-end.

- **Lớp Logic Ứng dụng (Application Logic Layer):** Được đảm nhiệm bởi Express.js (Controllers). Xử lý nghiệp vụ chính (kiểm tra hàng tồn kho, tính toán chiết khấu, xác thực thanh toán, cập nhật trạng thái).

- **Lớp Truy cập Dữ liệu (Data Access Layer):** Được đảm nhiệm bởi Mongoose (ODM) trên Express.js. Quản lý việc kết nối, truy vấn, và cập nhật dữ liệu trong MongoDB.

3.2.2 Sơ đồ Luồng Dữ liệu

- **React.js:** Người dùng nhấn nút "Đặt hàng" trên trang giỏ hàng.
- **API Call:** React gửi yêu cầu POST đến endpoint `/api/orders` kèm theo dữ liệu đơn hàng (list sản phẩm, địa chỉ giao hàng, phương thức thanh toán).
- **Express.js (Routing):** Express nhận yêu cầu POST, chuyển tới Controller OrderController.
- **OrderController:**
 - + Kiểm tra tính hợp lệ của dữ liệu (validation).
 - + Sử dụng Mongoose để tạo một Document mới trong Collection Order trên MongoDB.
 - + Cập nhật các thông tin liên quan (ví dụ: giảm số lượng hàng tồn kho).
- **MongoDB:** Lưu trữ đơn hàng mới và trả về trạng thái thành công.
- **Express.js (Response):** Server trả về mã trạng thái 201 Created và thông tin đơn hàng mới dưới dạng JSON.

- **React.js (Update UI):** Front-end nhận phản hồi, hiển thị thông báo "Đặt hàng thành công", và chuyển hướng người dùng đến trang "Theo dõi đơn hàng".

3.3 Thiết kế cơ sở dữ liệu

3.3.1 Collection Users

```
_id: ObjectId('6908cba784a1f7bf9e09584d')
name: "Nguyễn Tiến Thanh"
email: "thanhtk123xx@gmail.com"
password: "$2b$10$jxVmALuZ0oL.zM0uR9sT00Vup71tjCpBGhNeqCn9puNKXkELQ1TJ6"
provider: "local"
▶ cartData: Object
  __v: 7
▶ customPermissions: Array (4)
  isActive: true
  updatedAt: 2025-11-09T14:57:01.936+00:00
  role: ObjectId('69106981ea606036a3fe77c7')
▶ addresses: Array (1)
  avatar: "http://localhost:4000/images/avatar_1762961826712IMG_1611.JPG"
```

Hình 3.1: Collection Users

3.3.2 Collection Categories

```
_id: ObjectId('6904e8c4c03ceb09048e8a22')
name: "Chicken"
description: "Chicken"
image: "1761931772268hÃ~nh á°fnh_2025-11-01_002923110.png"
__v: 0
```

Hình 3.2: Collection Categories

3.3.3 Collection Foods

```
_id: ObjectId('68f819f9a72621a04b54e66d')
name: "Veg Salad"
description: "1x Veg Salad"
price: 1.5
image: "1761090041192food_2.png"
category: "Salad"
__v: 0
```

Hình 3.3: Collection Foods

3.3.4 Collection Orders

```
_id: ObjectId('6900fe7f18780598aa8c853c')
userId: "68f821973974294eb95183b8"
▶ items: Array (4)
  amount: 45
▶ address: Object
  status: "Out for Delivery"
  date: 2025-10-28T17:27:19.169+00:00
  payment: false
__v: 0
```

Hình 3.4: Collection Orders

3.4 Thiết kế API RESTful

Các API được thiết kế theo tiêu chuẩn RESTful, sử dụng các động từ HTTP (GET, POST, PUT, DELETE) để thực hiện các thao tác CRUD (Create, Read, Update, Delete) với tài nguyên.

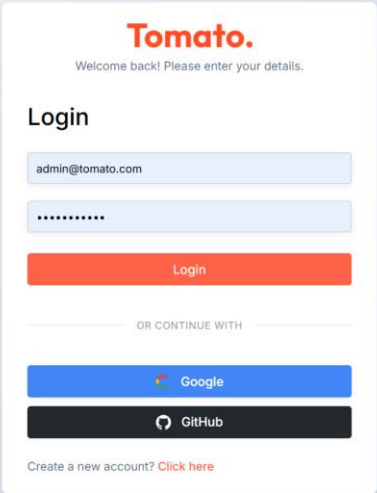
Bảng 3.3: Thiết kế API RESTful

Tài nguyên	Method	Endpoint	Mô tả
Sản phẩm	GET	/api/products	Lấy tất cả sản phẩm.
		/api/products/:id	Lấy chi tiết một sản phẩm.
	POST	/api/products	Thêm sản phẩm mới (Chỉ Admin).
Đơn hàng	POST	/api/orders	Tạo đơn hàng mới.
	GET	/api/orders/myorders	Lấy lịch sử đơn hàng của người dùng hiện tại.
	PUT	/api/orders/:id/status	Cập nhật trạng thái đơn hàng (Chỉ Admin).
Người dùng	POST	/api/users/login	Đăng nhập và nhận JWT Token.
	GET	/api/users/profile	Xem thông tin cá nhân.

CHƯƠNG 4: THIẾT KẾ GIAO DIỆN

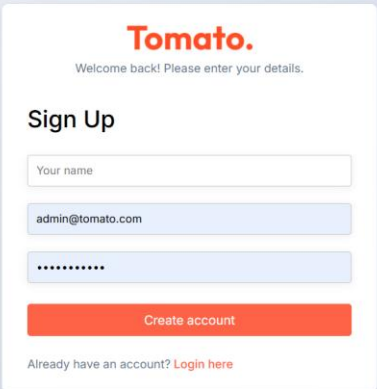
4.1 Thiết kế giao diện người dùng

4.1.1 Trang đăng ký, đăng nhập



The login form for 'Tomato.' features a white card on a light blue background. At the top is the 'Tomato.' logo in red, followed by the text 'Welcome back! Please enter your details.' Below this is the heading 'Login'. The form includes an email input field with 'admin@tomato.com' and a password input field with masked characters. A red 'Login' button is positioned below the password field. A separator line with the text 'OR CONTINUE WITH' is followed by two social login buttons: a blue 'Google' button and a dark grey 'GitHub' button. At the bottom, a link 'Create a new account? Click here' is displayed in red.

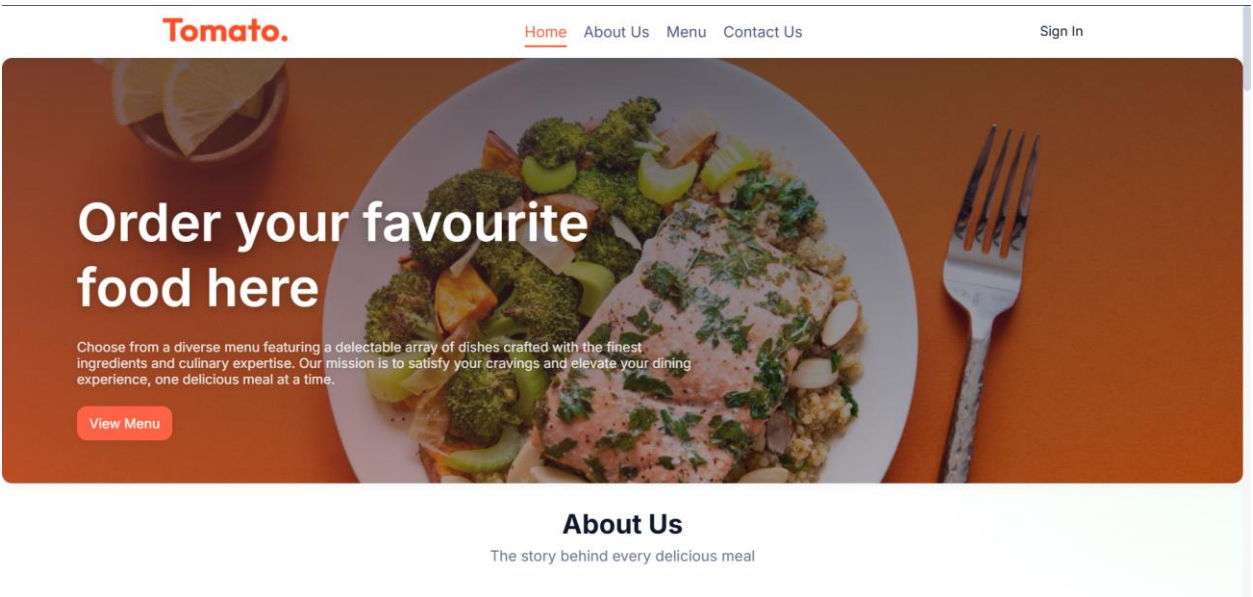
Hình 4.1: Trang đăng nhập



The sign-up form for 'Tomato.' features a white card on a light blue background. At the top is the 'Tomato.' logo in red, followed by the text 'Welcome back! Please enter your details.' Below this is the heading 'Sign Up'. The form includes three input fields: 'Your name', an email field with 'admin@tomato.com', and a password field with masked characters. A red 'Create account' button is positioned below the password field. At the bottom, a link 'Already have an account? Login here' is displayed in red.

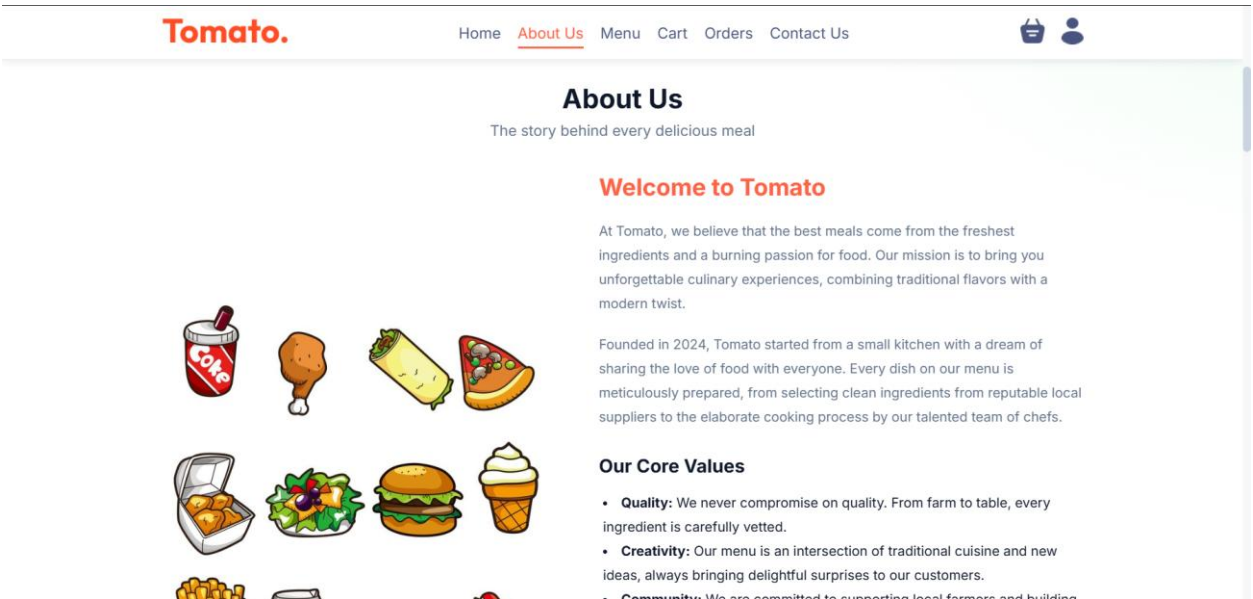
Hình 4.2: Trang đăng ký

4.1.2 Trang chủ



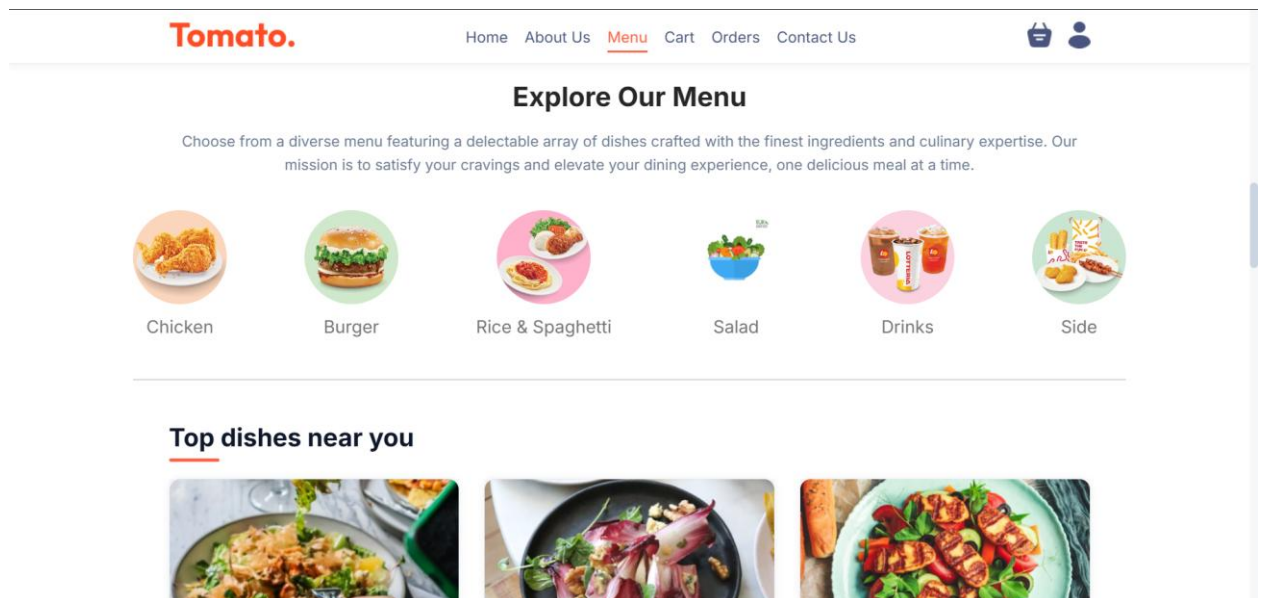
Hình 4.3: Trang chủ

4.1.3 Trang About Us



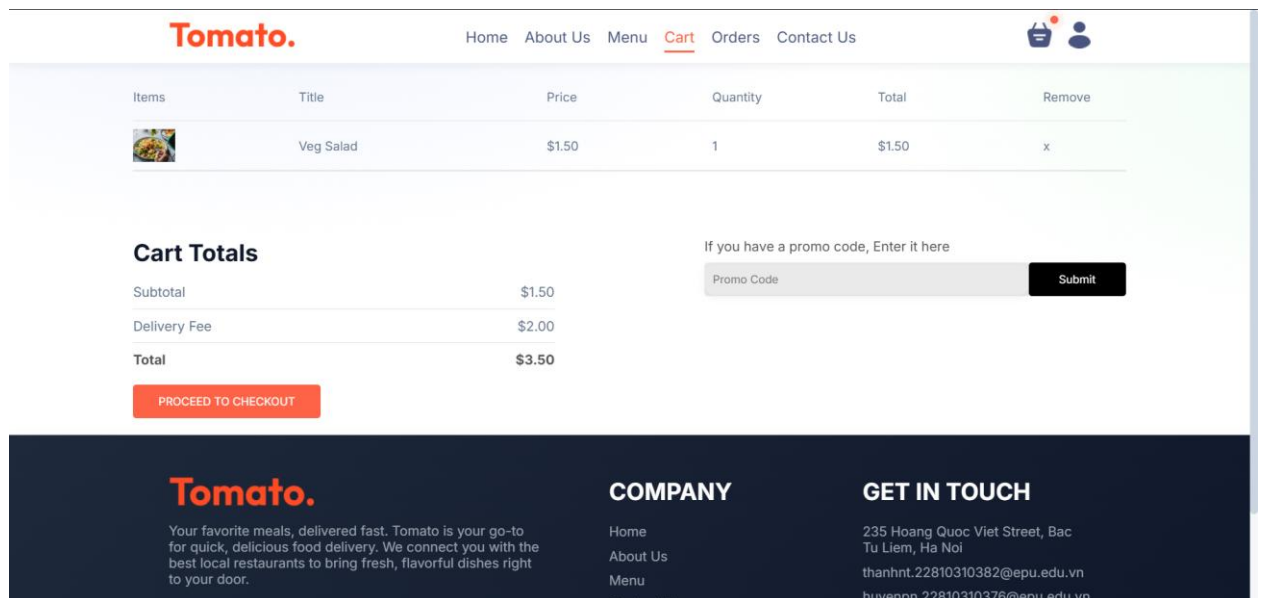
Hình 4.4: Trang About Us

4.1.4 Trang Menu



Hình 4.5: Trang Menu

4.1.5 Trang Cart



Hình 4.6: Trang Cart

4.1.6 Trang Checkout

← Environnement de test New business

Chọn một loại tiền tệ:

1 USD = 27.430,5714 VND

Veg Salad	41.146 đ
Delivery Charges	54.861 đ

Trả với

HOẶC

Email

Phương thức thanh toán

Thông tin thẻ

1234 1234 1234 1234

MM / YY CVC

Tên chủ thẻ

Quốc gia hoặc khu vực

☐ Lưu thông tin của tôi để thanh toán nhanh chóng hơn
Thanh toán an toàn tại Environnement de test New business và ở mọi nơi hỗ trợ [Link](#).

Hình 4.7: Trang Checkout

4.1.7 Trang Orders

Tomato. Home About Us Menu Cart Orders Contact Us

My Orders

	Veg Salad x 1	\$3.50	Items: 1	• Delivered	Track Order
	Veg Salad x 1	\$3.50	Items: 1	• Delivered	Track Order
	Crab Sticks x 1, Cheese Stick x 1, Squid Ring x 1	\$7.70	Items: 3	• Delivered	Track Order
	Veg Salad x 1	\$3.50	Items: 1	• Food Processing	Track Order

Tomato. Your favorite meals, delivered fast. Tomato is your go-to for quick, delicious food delivery. We connect you with the best local restaurants to bring fresh, flavorful dishes right to your door.

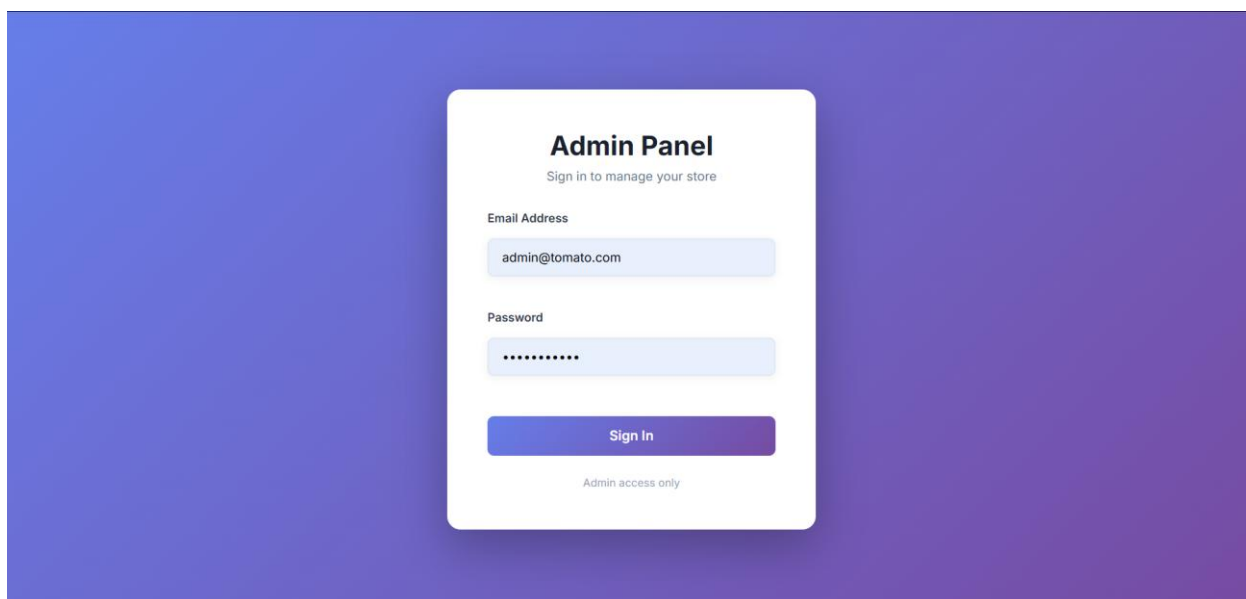
COMPANY
Home
About Us
Menu

GET IN TOUCH
235 Hoang Quoc Viet Street, Bac Tu Liem, Ha Noi
thanhtnt.22810310382@epu.edu.vn

Hình 4.8: Trang Orders

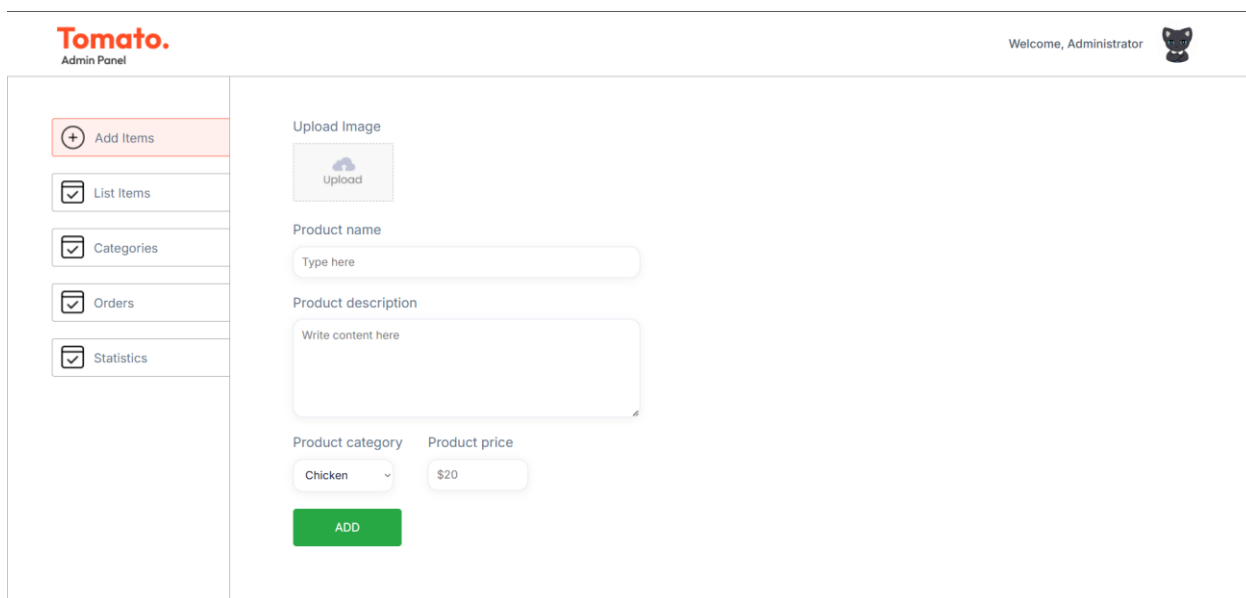
4.2 Thiết kế giao diện Admin

4.2.1 Trang đăng nhập



Hình 4.9: Trang đăng nhập

4.2.2 Trang thêm sản phẩm



Hình 4.10: Trang thêm sản phẩm

4.2.3 Trang danh sách sản phẩm

Tomato.
Admin Panel

Welcome, Administrator

+ Add Items

☒ List Items









☒ Categories

☒ Orders

☒ Statistics

All Food List

Filter by Category: All Categories

Image	Name	Category	Price	Actions	
	Veg Salad	Salad	\$1.50	Edit	Delete
	Clover Salad	Salad	\$1.50	Edit	Delete
	Chicken Salad	Salad	\$1.50	Edit	Delete
	Gochu Chicken	Chicken	\$2.20	Edit	Delete
	Garlic Chicken	Chicken	\$2.20	Edit	Delete
	K-Chicken	Chicken	\$2.20	Edit	Delete
	Carbo Buldak Shrimp Burger	Burger	\$1.80	Edit	Delete
	Shrimm Runner	Runner	\$1.80	Edit	Delete

Hình 4.11: Trang danh sách sản phẩm

4.2.4 Trang quản lý danh mục

Tomato.
Admin Panel

Welcome, Administrator

+ Add Items

☒ List Items







☒ Categories

☒ Orders

☒ Statistics

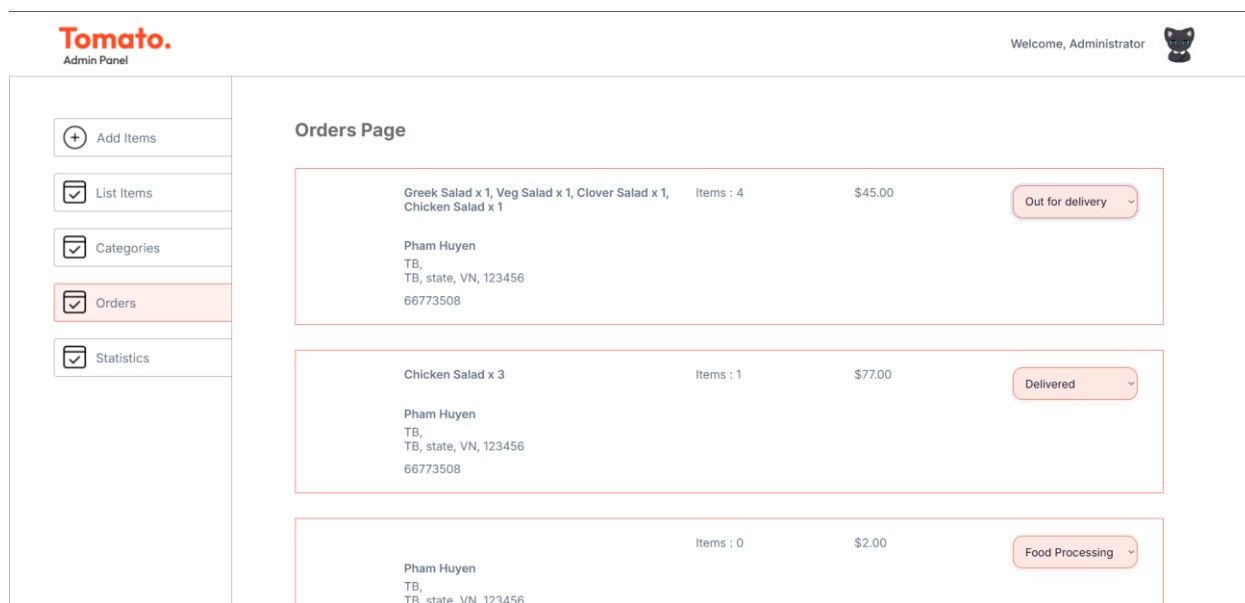
All Categories

Add Category

Image	Name	Description	Actions	
	Chicken	Chicken	Edit	Delete
	Burger	Burger	Edit	Delete
	Rice & Spaghetti	Rice & Spaghetti	Edit	Delete
	Salad	Salad	Edit	Delete
	Drinks	Drinks	Edit	Delete
	Side	Side	Edit	Delete

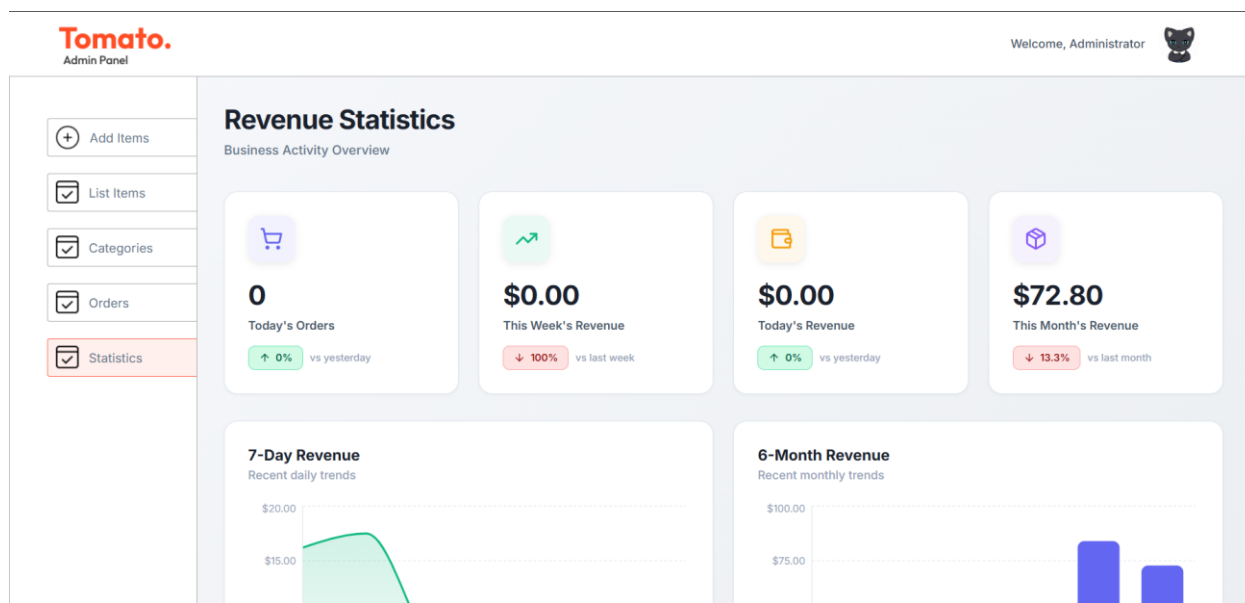
Hình 4.12: Trang quản lý danh mục

4.2.5 Trang quản lý đơn hàng



Hình 4.13: Trang quản lý đơn hàng

4.2.6 Trang phân tích thống kê



Hình 4.14: Trang phân tích thống kê

PHẦN KẾT LUẬN

Đề tài "Website Bán Đồ Ăn Nhanh FastFood Shop" đã hoàn thành xuất sắc mục tiêu đề ra là xây dựng một hệ thống thương mại điện tử ẩm thực hoàn chỉnh, sử dụng bộ công nghệ hiện đại MERN Stack (MongoDB, Express.js, React.js, Node.js). Hệ thống đã tạo ra được một kiến trúc Full-stack JavaScript đồng bộ, đảm bảo tính ổn định, tốc độ xử lý nhanh chóng nhờ mô hình I/O bất đồng bộ của Node.js, và giao diện người dùng (Front-end) mượt mà, thân thiện nhờ vào thư viện React.js với kiến trúc Component và Virtual DOM. Các chức năng nghiệp vụ cốt lõi, từ quản lý sản phẩm, xử lý giỏ hàng, đặt hàng thanh toán cho đến hệ thống quản trị (Admin Dashboard) đều được triển khai thành công, đáp ứng đầy đủ yêu cầu của một mô hình kinh doanh Fast Food trực tuyến. Dự án không chỉ là minh chứng cho khả năng ứng dụng hiệu quả các công nghệ Web tiên tiến mà còn cung cấp một giải pháp thiết thực, sẵn sàng triển khai cho các đơn vị kinh doanh ẩm thực.

Mặc dù đã triển khai thành công các chức năng cốt lõi, dự án còn hạn chế ở việc thiếu các tính năng nâng cao như thanh toán trực tuyến hoặc hệ thống khuyến mãi phức tạp, do đó, hướng phát triển tương lai sẽ tập trung vào việc hoàn thiện hệ thống toàn diện, bao gồm tối ưu hóa hiệu suất, tăng cường bảo mật, và mở rộng khả năng tích hợp với các dịch vụ bên ngoài để nâng cao tính cạnh tranh và khả năng mở rộng quy mô.

DANH MỤC TÀI LIỆU THAM KHẢO

1. Giáo trình môn học Ngôn ngữ kịch bản
2. Slide môn học Ngôn ngữ kịch bản