TRƯỜNG ĐẠI HỌC TÂY NGUYÊN

**KHOA KHOA HỌC TỰ NHIÊN VÀ CÔNG NGHỆ**



**HỌC PHẦN: LẬP TRÌNH DI ĐỘNG**

**TÊN ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG DI ĐỘNG ĐẶT MÓN ĂN TRỰC TUYẾN**

**Sinh viên thực hiện: 22103054 Nguyễn Mạnh Quân**

**22103060 Nguyễn Hoàng Thanh**

**22103080 Vi Văn Tuấn**

**Chuyên ngành : Công Nghệ Thông Tin**

**Lớp : CNTT - K22**

**Đắk Lắk, tháng 10 năm 2025**

TRƯỜNG ĐẠI HỌC TÂY NGUYÊN

**KHOA KHOA HỌC TỰ NHIÊN VÀ CÔNG NGHỆ**



**HỌC PHẦN: LẬP TRÌNH DI ĐỘNG**

**TÊN ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG DI ĐỘNG ĐẶT MÓN ĂN TRỰC TUYẾN**

**Sinh viên thực hiện: 22103054 Nguyễn Mạnh Quân**

**22103060 Nguyễn Hoàng Thanh**

**22103080 Vi Văn Tuấn**

**Chuyên ngành : Công Nghệ Thông Tin**

**Lớp : CNTT - K22**

**Người hướng dẫn: TS. Trương Thị Hương Giang**

**Đắk Lắk, tháng 10 năm 2025**

# **LỜI CẢM ƠN**

Để hoàn thành bài báo cáo này, nhóm chúng em đã trải qua một hành trình học tập và nghiên cứu đầy ý nghĩa về lập trình ứng dụng di động. Quá trình này không chỉ giúp chúng em củng cố và mở rộng kiến thức chuyên môn mà còn rèn luyện kỹ năng làm việc nhóm, khả năng phân tích vấn đề cũng như kinh nghiệm phát triển ứng dụng thực tế.

Trong suốt thời gian thực hiện đề tài "Xây dựng ứng dụng di động đặt món ăn trực tuyến", nhóm chúng em đã nhận được sự hướng dẫn tận tình và chu đáo từ **cô TS. Trương Thị Hương Giang**. Những lời góp ý, chỉ dẫn chuyên môn của cô đã giúp nhóm định hướng đúng đắn trong việc phân tích yêu cầu, thiết kế hệ thống và triển khai các tính năng của ứng dụng. Nhờ đó, chúng em không chỉ hiểu rõ hơn về quy trình xây dựng một sản phẩm phần mềm hoàn chỉnh mà còn biết cách vận dụng linh hoạt kiến thức lý thuyết vào thực tiễn.

Nhóm chúng em xin gửi lời cảm ơn sâu sắc đến cô vì đã luôn động viên, định hướng và hỗ trợ nhiệt tình trong suốt quá trình thực hiện đề tài. Sự tâm huyết của cô chính là động lực to lớn giúp chúng em hoàn thành tốt báo cáo này.

Chúng em xin chân thành cảm ơn!

.

# **MỤC LỤC**

[**LỜI CẢM ƠN** i](#_Toc212647698)

[**MỤC LỤC** ii](#_Toc212647699)

[**DANH MỤC BẢNG** iv](#_Toc212647700)

[**1. ĐẶT VẤN ĐỀ** 1](#_Toc212647701)

[1.1. Lý do chọn đề tài 1](#_Toc212647702)

[1.2. Mục tiêu và phạm vi đề tài 1](#_Toc212647703)

[1.2.1. Mục Tiêu 1](#_Toc212647704)

[1.2.2. Phạm vi đề tài 1](#_Toc212647705)

[1.3. Ý nghĩa 2](#_Toc212647706)

[1.4. Phương pháp nghiên cứu 2](#_Toc212647707)

[**2. TỔNG QUAN** 3](#_Toc212647708)

[2.1. Tổng quan tình hình nghiên cứu ứng dụng di động đặt món ăn trực tuyến 3](#_Toc212647709)

[2.2. Tổng quan về Flutter và Laravel 4](#_Toc212647710)

[**3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG** 6](#_Toc212647711)

[3.1. Đối tượng sử dụng 6](#_Toc212647712)

[3.2. Yêu cầu chức năng 6](#_Toc212647713)

[3.3 Yêu cầu phi chức năng 7](#_Toc212647714)

[3.4. Sơ đồ Usecase 7](#_Toc212647715)

[3.5. Sơ đồ luồng hoạt động 12](#_Toc212647716)

[3.6. Sơ đồ lớp 15](#_Toc212647717)

[3.7. Sơ đồ phân rã chức năng(Functional Decomposition Diagram) 16](#_Toc212647718)

[3.8. Sơ đồ tuấn tự 17](#_Toc212647719)

[3.9. Cơ sở dữ liệu 18](#_Toc212647720)

[3.10. Phân tích mô tả API 21](#_Toc212647721)

[**4. THIẾT KẾ VÀ TRIỂN KHAI VỚI LARAVEL VÀ FLUTTER** 24](#_Toc212647722)

[4.1. Cài đặt và cấu hình môi trường Laravel 24](#_Toc212647723)

[4.2. Thiết kế và xử lý các route API 26](#_Toc212647724)

[4.3. Kết nối và quản lý cơ sở dữ liệu 29](#_Toc212647725)

[4.3. Kết nối ứng dụng Flutter với API Laravel. 30](#_Toc212647726)

[**5. KẾT QUẢ** 32](#_Toc212647727)

[**6. ĐÁNH GIÁ VÀ HƯỚNG PHÁT TRIỂN** 46](#_Toc212647728)

[6.1. Ưu điểm của hệ thống 46](#_Toc212647729)

[6.2. Hạn chế của hệ thống 46](#_Toc212647730)

[6.3. Hướng phát triển 46](#_Toc212647731)

[**TÀI LIỆU THAM KHẢO** 47](#_Toc212647732)

# **DANH MỤC BẢNG**

[**Bảng 3.1. Bảng mô tả use case chi tiết chức năng quản lý giỏ hàng** 8](#_Toc212647739)

[**Bảng 3.2. Bảng mô tả use case chi tiết chức năng đăng nhập** 9](#_Toc212647740)

[**Bảng 3.3. Bảng mô tả use case chi tiết chức năng đăng ký tài khoản** 11](#_Toc212647741)

[**Bảng 3.4. Bảng mô tả cơ sở dữ liệu bảng users** 18](#_Toc212647742)

[**Bảng 3.5. Bảng mô tả cơ sở dữ liệu bảng categories** 19](#_Toc212647743)

[**Bảng 3.6 Bảng mô tả cơ sở dữ liệu bảng foods** 19](#_Toc212647744)

[**Bảng 3.7. Bảng mô tả cơ sở dữ liệu bảng orders** 20](#_Toc212647745)

[**Bảng 3.8. Bảng mô tả cơ sở dữ liệu bảng order\_items** 20](#_Toc212647746)

[**Bảng 3.9. Bảng mô tả cơ sở dữ liệu bảng cart\_items** 21](#_Toc212647747)

# **1. ĐẶT VẤN ĐỀ**

## 1.1. Lý do chọn đề tài

Trong thời đại công nghệ phát triển mạnh mẽ, các ứng dụng di động đang trở thành một phần không thể thiếu trong đời sống con người. Việc đặt món ăn thông qua ứng dụng di động đã trở nên phổ biến nhờ sự tiện lợi, nhanh chóng và thân thiện với người dùng.  
Đặc biệt trong môi trường sinh viên hoặc nhân viên văn phòng, nhu cầu đặt món ăn nhanh, theo dõi đơn hàng, thanh toán tiện lợi ngày càng tăng cao. Tuy nhiên, nhiều ứng dụng hiện nay còn phức tạp, nặng nề hoặc chưa tối ưu cho trải nghiệm người dùng.

Xuất phát từ thực tế đó, nhóm quyết định chọn đề tài **“Xây dựng ứng dụng di động đặt món ăn trực tuyến”** nhằm mục tiêu rèn luyện kỹ năng phát triển ứng dụng Android, đồng thời tạo ra một giải pháp tiện ích giúp người dùng dễ dàng đặt món, quản lý đơn hàng, và tương tác với quán ăn một cách nhanh chóng, hiệu quả.

## 1.2. Mục tiêu và phạm vi đề tài

### **1.2.1. Mục Tiêu**

**Mục tiêu chính**

- Xây dựng một **ứng dụng di động đặt món ăn trực tuyến** hoạt động trên nền tảng Android.

- Thiết kế giao diện thân thiện, dễ sử dụng, phù hợp với nhiều đối tượng người dùng.

- Hỗ trợ chức năng đặt món, giỏ hàng, theo dõi đơn hàng và thanh toán trực tuyến.

**Mục tiêu cụ thể**

- Cho phép người dùng xem danh sách món ăn, chọn món và thêm vào giỏ hàng.

- Cung cấp chức năng đăng ký, đăng nhập, cập nhật thông tin cá nhân.

- Cho phép chủ cửa hàng quản lý món ăn, đơn hàng và phản hồi người dùng**.**

- Sử dụng cơ sở dữ liệu thời gian thực (Firebase hoặc MySQL) để quản lý dữ liệu.

- Ứng dụng được xây dựng bằng **Android Studio**

### **1.2.2. Phạm vi đề tài**

Hệ thống hoạt động trong lĩnh vực đặt món ăn trực tuyến, cho phép:

- Người dùng tìm kiếm, đặt món ăn từ nhiều nhà hàng.

- Thanh toán trực tuyến hoặc COD.

- Theo dõi trạng thái đơn hàng theo thời gian thực.

- Nhà hàng quản lý menu, đơn hàng và đánh giá.

- Quản trị viên giám sát, phân quyền, đảm bảo vận hành ổn định.

## 1.3. Ý nghĩa

Việc phát triển ứng dụng đặt món ăn có tích hợp trí tuệ nhân tạo mang lại nhiều ý nghĩa thực tiễn cả về mặt công nghệ lẫn đời sống xã hội:

**Đối với người dùng:**

Ứng dụng giúp tiết kiệm thời gian, hỗ trợ tìm kiếm và lựa chọn món ăn nhanh chóng hơn. Việc gợi ý món ăn dựa trên sở thích và lịch sử đặt món giúp trải nghiệm của người dùng trở nên cá nhân hóa và tiện lợi.

**Đối với cửa hàng, nhà hàng:**

Hệ thống quản lý đơn hàng và phân tích dữ liệu tiêu thụ giúp nhà hàng tối ưu thực đơn, kiểm soát nguyên liệu, dự đoán nhu cầu và đưa ra chiến lược kinh doanh phù hợp.

**Về mặt xã hội:**

Đề tài góp phần thúc đẩy chuyển đổi số trong lĩnh vực ẩm thực – dịch vụ, đồng thời khuyến khích việc ứng dụng các kỹ thuật AI, Machine Learning vào thực tiễn đời sống.

**Về mặt học Thuật:**

Sinh viên có cơ hội áp dụng kiến thức về trí tuệ nhân tạo, học máy, và phát triển ứng dụng thực tế qua đó củng cố kỹ năng nghiên cứu, lập trình, và triển khai hệ thống thông minh.

## 1.4. Phương pháp nghiên cứu

**Frontend (Mobile App):** Flutter framework - cho phép phát triển ứng dụng đa nền tảng trên cả iOS và Android từ một mã nguồn duy nhất.

**Backend (Server-side):** Laravel framework - xử lý logic nghiệp vụ, quản lý cơ sở dữ liệu, xác thực người dùng và cung cấp RESTful API.

**Kiến trúc:** Client-Server với RESTful API để kết nối giữa ứng dụng mobile và server.

# **2. TỔNG QUAN**

## 2.1. Tổng quan tình hình nghiên cứu ứng dụng di động đặt món ăn trực tuyến

**2.1.1. Bối cảnh phát triển**

Trong những năm gần đây, sự phát triển mạnh mẽ của công nghệ di động và internet đã thay đổi đáng kể thói quen tiêu dùng của người dân, đặc biệt là trong lĩnh vực ẩm thực. Xu hướng đặt món ăn trực tuyến ngày càng trở nên phổ biến nhờ vào sự tiện lợi, nhanh chóng và đa dạng mà nó mang lại. Người dùng có thể dễ dàng tìm kiếm, so sánh và đặt món ăn từ nhiều nhà hàng, quán ăn khác nhau chỉ với vài thao tác đơn giản trên smartphone.

Đại dịch COVID-19 đã tạo ra một bước ngoặt lớn, thúc đẩy mạnh mẽ nhu cầu sử dụng các dịch vụ giao đồ ăn trực tuyến. Theo các nghiên cứu thị trường, quy mô thị trường giao đồ ăn trực tuyến toàn cầu đã tăng trưởng đáng kể và dự kiến sẽ tiếp tục tăng trong những năm tới. Tại Việt Nam, các ứng dụng như Grab Food, ShopeeFood, Baemin, GoFood đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày của người dân đô thị.

**2.1.2. Tình hình nghiên cứu trong nước và quốc tế**

Trên thế giới, nhiều nghiên cứu đã được thực hiện nhằm phát triển và cải thiện các ứng dụng đặt món ăn trực tuyến. Các công ty công nghệ lớn như Uber (với Uber Eats), DoorDash, Deliveroo, Just Eat đã đầu tư mạnh mẽ vào việc nghiên cứu và phát triển công nghệ, tối ưu hóa trải nghiệm người dùng thông qua các thuật toán gợi ý thông minh, hệ thống định vị GPS chính xác, và tích hợp các phương thức thanh toán đa dạng.

Các nghiên cứu học thuật tập trung vào nhiều khía cạnh như phân tích hành vi người dùng, tối ưu hóa thuật toán giao hàng, ứng dụng trí tuệ nhân tạo (AI) và machine learning để cá nhân hóa trải nghiệm, và nghiên cứu về bảo mật thông tin thanh toán. Nhiều nghiên cứu cũng đề cập đến việc tích hợp các công nghệ mới như chatbot AI, hệ thống đánh giá và review thông minh, và tối ưu hóa logistics để giảm thời gian giao hàng.

Tại Việt Nam, lĩnh vực này cũng đang thu hút sự quan tâm của các nhà nghiên cứu và doanh nghiệp công nghệ. Nhiều đề tài nghiên cứu sinh viên, luận văn thạc sĩ đã tập trung vào việc xây dựng các ứng dụng đặt món ăn với các tính năng phù hợp với thị trường địa phương. Các nghiên cứu này thường tập trung vào việc áp dụng các công nghệ mới như Flutter, React Native cho mobile app, và Node.js, Laravel, Django cho backend.

**2.1.3. Các ứng dụng tiêu biểu và bài học kinh nghiệm**

Một số ứng dụng đặt món ăn trực tuyến tiêu biểu trên thế giới và Việt Nam đã đạt được thành công lớn:

***Trên thế giới:***

* **Uber Eats**: Tận dụng hệ sinh thái Uber sẵn có, tích hợp tốt với dịch vụ đi xe, giao diện đơn giản và trải nghiệm người dùng mượt mà.
* **DoorDash**: Tập trung vào thị trường Mỹ với mạng lưới nhà hàng đa dạng và dịch vụ giao hàng nhanh chóng.
* **Deliveroo**: Nổi bật với thiết kế giao diện hiện đại và hệ thống đánh giá nhà hàng chi tiết.

***Tại Việt Nam:***

* **Grab Food**: Ứng dụng phổ biến nhất với mạng lưng rộng khắp, tích hợp thanh toán đa dạng và chương trình khuyến mãi hấp dẫn.
* **ShopeeFood**: Tận dụng hệ sinh thái Shopee, có nhiều ưu đãi và voucher giảm giá.
* **Baemin**: Giao diện trẻ trung, thu hút người dùng trẻ tuổi với các chương trình marketing sáng tạo.

Từ các ứng dụng này, một số bài học kinh nghiệm có thể rút ra bao gồm: giao diện người dùng phải đơn giản, trực quan và dễ sử dụng; hệ thống tìm kiếm và lọc món ăn phải nhanh chóng và chính xác; cần có nhiều phương thức thanh toán linh hoạt; hệ thống đánh giá và phản hồi giúp nâng cao chất lượng dịch vụ; và chương trình khuyến mãi, tích điểm là yếu tố quan trọng để giữ chân người dùng.

## 2.2. Tổng quan về Flutter và Laravel

**2.2.1. Tổng quan về Flutter**

Flutter là framework mã nguồn mở do Google phát triển và ra mắt chính thức vào năm 2018. Đây là một bộ công cụ phát triển giao diện người dùng (UI toolkit) cho phép xây dựng ứng dụng đa nền tảng từ một mã nguồn duy nhất, hỗ trợ iOS, Android, Web, Windows, macOS và Linux.

Flutter sử dụng ngôn ngữ lập trình Dart, cũng được phát triển bởi Google. Điểm đặc biệt của Flutter là khả năng rendering giao diện độc lập thông qua engine đồ họa Skia, không phụ thuộc vào các thành phần giao diện gốc của nền tảng, giúp đảm bảo giao diện nhất quán và hiệu suất cao trên mọi thiết bị.

Các ưu điểm nổi bật của Flutter bao gồm tính năng Hot Reload giúp xem thay đổi code ngay lập tức, thư viện widget phong phú và có thể tùy chỉnh cao, hiệu suất gần như native app nhờ biên dịch trực tiếp sang mã máy, và cộng đồng phát triển năng động với hệ sinh thái package đa dạng trên pub.dev.

**1.2.2. Tổng quan về Laravel**

Laravel là framework PHP mã nguồn mở được phát triển bởi Taylor Otwell, phát hành lần đầu năm 2011. Đây là framework backend phổ biến nhất trong cộng đồng PHP, được xây dựng dựa trên kiến trúc MVC (Model-View-Controller), giúp tổ chức code có cấu trúc và dễ bảo trì.

Laravel cung cấp nhiều tính năng mạnh mẽ như Eloquent ORM cho phép thao tác với cơ sở dữ liệu một cách trực quan thông qua các model, hệ thống routing linh hoạt, middleware để xử lý các yêu cầu HTTP, Blade template engine, và hệ thống authentication & authorization tích hợp sẵn. Framework này còn hỗ trợ tốt việc xây dựng RESTful API, queue system, task scheduling và nhiều tính năng khác phục vụ phát triển ứng dụng hiện đại.

Các ưu điểm của Laravel bao gồm cú pháp rõ ràng và elegant, tài liệu đầy đủ chi tiết, cộng đồng lớn mạnh với hàng nghìn package hỗ trợ, bảo mật cao với các cơ chế chống CSRF, SQL Injection, XSS, và công cụ Artisan CLI mạnh mẽ giúp tự động hóa các tác vụ phát triển. Laravel đặc biệt phù hợp cho việc xây dựng backend API cho các ứng dụng di động và web.

**1.2.3. Tính kết hợp giữa Flutter và Laravel trong dự án**

Trong dự án Fooddgo, sự kết hợp giữa Flutter và Laravel tạo nên một kiến trúc Client-Server hiện đại và hiệu quả. Mô hình này tách biệt rõ ràng giữa phần giao diện người dùng (Frontend) và phần xử lý nghiệp vụ (Backend), mang lại nhiều lợi ích về mặt phát triển và vận hành.

Laravel đóng vai trò là backend server, chịu trách nhiệm xử lý toàn bộ logic nghiệp vụ của ứng dụng bao gồm xác thực người dùng, quản lý cơ sở dữ liệu, xử lý đơn hàng và cung cấp các RESTful API endpoint. Dữ liệu được trao đổi giữa client và server dưới định dạng JSON thông qua giao thức HTTP/HTTPS, đảm bảo an toàn và hiệu quả.

Flutter đóng vai trò frontend mobile app, giao tiếp với Laravel thông qua các API call để lấy dữ liệu, gửi yêu cầu và hiển thị thông tin cho người dùng. Ứng dụng Flutter sử dụng các package như http hoặc dio để thực hiện các request đến server, nhận response dưới dạng JSON và parse thành các object Dart để sử dụng trong ứng dụng.

Kiến trúc này mang lại nhiều lợi ích: dễ dàng mở rộng và bảo trì khi logic nghiệp vụ được tập trung ở backend, có thể phát triển thêm các nền tảng khác (web, desktop) sử dụng chung API mà không cần viết lại backend, cho phép phân chia công việc hiệu quả trong nhóm phát triển, và đảm bảo bảo mật tốt hơn khi logic quan trọng được xử lý ở phía server. Sự phối hợp này tạo nên một hệ thống vừa mạnh mẽ, vừa linh hoạt, đáp ứng tốt yêu cầu của một ứng dụng đặt món ăn trực tuyến hiện đại

# **3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

## 3.1. Đối tượng sử dụng

- Mọi người đều có thể sử dụng dễ dàng vì giao diện thân thiện, thao tác đơn giản.

- Sinh viên, nhân viên văn phòng hoặc người bận rộn cần đặt món nhanh.

- Chủ quán ăn muốn quản lý món và đơn hàng tiện lợi.

- Người dùng có nhu cầu mua đồ ăn, thức uống trực tuyến qua điện thoại.

## 3.2. Yêu cầu chức năng

- Người dùng có thể đăng ký, đăng nhập và quản lý tài khoản cá nhân.

- Hiển thị danh sách món ăn, cho phép tìm kiếm và xem chi tiết từng món.

- Cho phép người dùng thêm món vào giỏ hàng, đặt hàng và thanh toán.

- Người dùng có thể xem lịch sử đơn hàng và theo dõi trạng thái giao hàng.

- Quản trị viên hoặc chủ quán có thể thêm, sửa, xóa món ăn.

- Tìm kiếm món ăn, nhà hàng theo từ khóa, danh mục.

- Quản lý đơn hàng (xác nhận, xử lý, hủy, theo dõi giao hàng).

- Đánh giá và phản hồi món ăn, dịch vụ.

- Tùy chỉnh giao diện: ngôn ngữ, font chữ, theme sáng/tối.

## 3.3 Yêu cầu phi chức năng

- Ứng dụng hoạt động ổn định, phản hồi nhanh, không bị treo khi thao tác.

- Giao diện đơn giản, dễ sử dụng, hiển thị rõ ràng trên mọi kích thước màn hình..

- Bảo mật: Mã hóa mật khẩu (bcrypt), xác thực JWT, phân quyền truy cập.

- Hiệu năng: Tối ưu API, cache dữ liệu món ăn phổ biến.

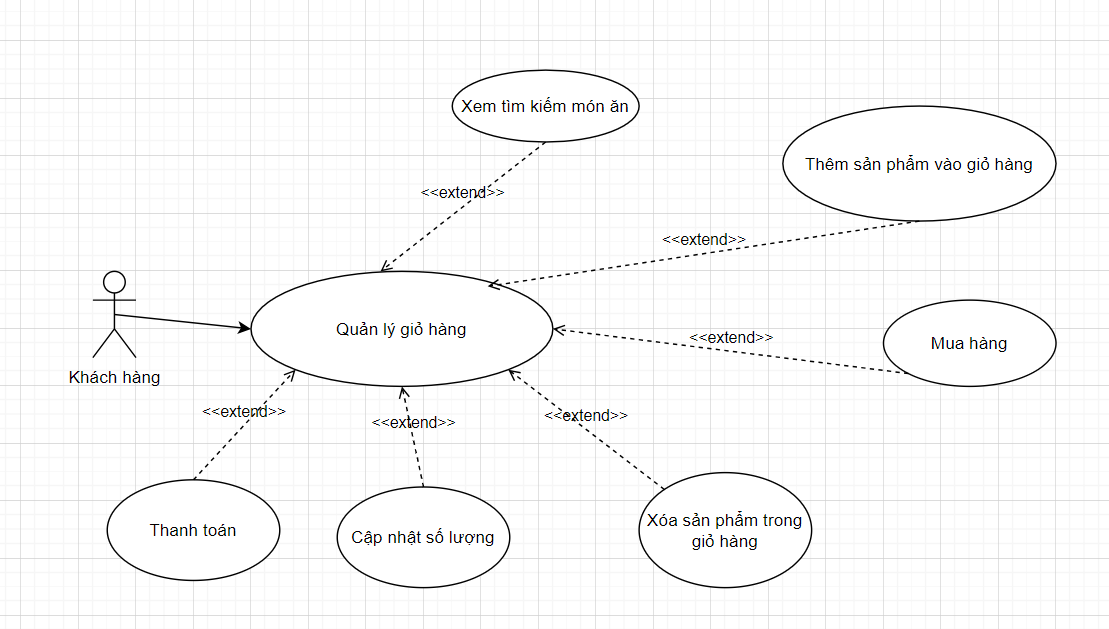
- Khả năng mở rộng: Thiết kế module hóa để dễ thêm tính năng mới.

- Đa ngôn ngữ: Sử dụng thư viện i18n trong Flutter và Laravel.

- Tùy chỉnh giao diện: Cho phép người dùng thay đổi font chữ, theme sáng/tối.

## 3.4. Sơ đồ Usecase

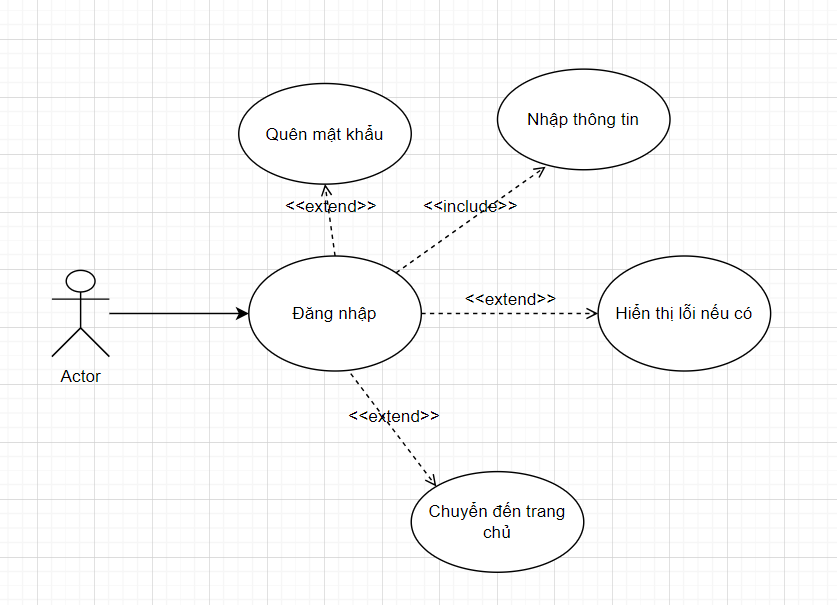
**3.4.1. Sơ đồ use case: Quản lý giỏ hàng**

****

**Bảng 3.1. Bảng mô tả use case chi tiết chức năng quản lý giỏ hàng**

|  |  |
| --- | --- |
| **Tác nhân** | khách hàng sử dụng ứng dụng để đặt món ăn. |
| **Sự kiện đầu vào** | Người dùng đã đăng nhập vào hệ thống  Người dùng đang duyệt món ăn và muốn thêm, sửa, xóa hoặc thanh toán giỏ hàng |
| **Các luồng sự kiện** | Luồng chính:   1. Người dùng mở giỏ hàng từ giao diện chính. 2. Hệ thống hiển thị danh sách món ăn đã thêm. 3. Người dùng thực hiện các thao tác:   + Thêm sản phẩm vào giỏ hàng: chọn món từ thực đơn → thêm vào giỏ.  + Xem và tìm kiếm món ăn: lọc theo tên, loại món, giá…  + Cập nhật số lượng: thay đổi số lượng món ăn trong giỏ.  + Xóa sản phẩm khỏi giỏ hàng: loại bỏ món không muốn mua.  + Mua hàng: xác nhận đơn → chuyển sang bước thanh toán.  + Thanh toán: chọn phương thức thanh toán → hoàn tất đơn hàng.  **Luồng thay thế**:  - Nếu món ăn đã hết hàng → hệ thống thông báo và đề xuất món khác.  - Nếu người dùng chưa đăng nhập → chuyển hướng đến màn hình đăng nhập.  - Nếu thanh toán thất bại → hiển thị lỗi và yêu cầu chọn lại phương thức. |
| **Kết quả trả về** | - Giỏ hàng được cập nhật theo thao tác của người dùng.  - Nếu người dùng chọn “Mua hàng” và thanh toán thành công:  Đơn hàng được ghi nhận.  - Hệ thống chuyển trạng thái sang “Đang chuẩn bị” hoặc “Đang giao”.  - Người dùng nhận được thông báo xác nhận đơn hàng. |

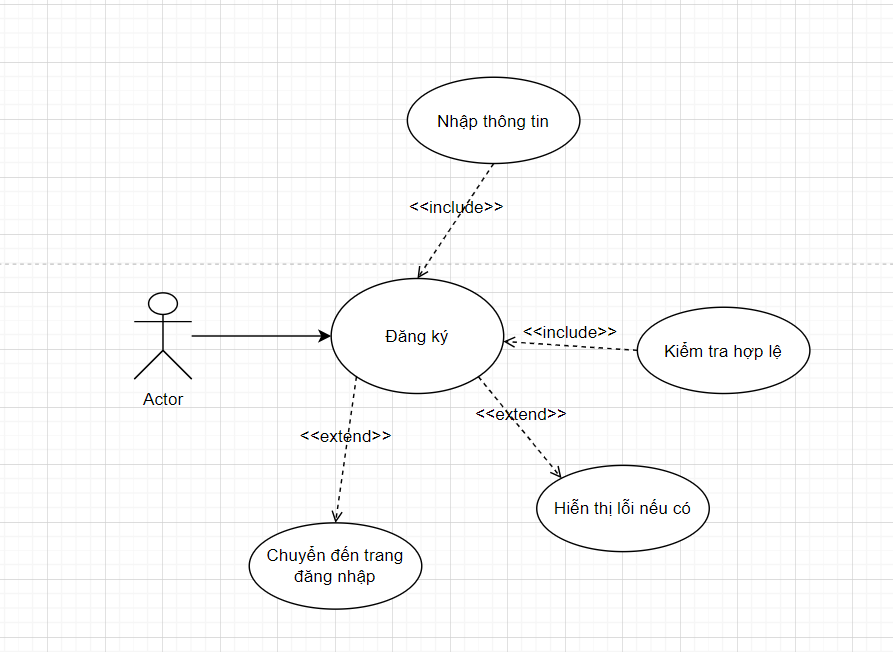
**3.4.2. Sơ đồ use case đăng nhập tài khoảncủa người dùng**



**Bảng 3.2. Bảng mô tả use case chi tiết chức năng đăng nhập**

|  |  |
| --- | --- |
| **Tác nhân** | Người dùng: là khách hàng đã có tài khoản và muốn truy cập vào hệ thống. |
| **Sự kiện đầu vào** | - Người dùng truy cập vào màn hình đăng nhập.  - Nhập thông tin: email (hoặc tên đăng nhập) và mật khẩu. |
| **Các luồng sự kiện** | * **Luồng chính:**  1. Người dùng mở giao diện đăng nhập. 2. Hệ thống hiển thị biểu mẫu yêu cầu nhập thông tin. 3. Người dùng nhập email và mật khẩu → nhấn nút “Đăng nhập”. 4. Hệ thống kiểm tra thông tin:   + Nếu đúng → chuyển đến trang chủ hoặc trang trước đó.  + Nếu sai → hiển thị lỗi.   * **Luồng thay thế:**   - Nếu người dùng nhập sai thông tin:  + Hệ thống hiển thị thông báo lỗi: “Sai mật khẩu” hoặc “Tài khoản không tồn tại”.  - Nếu người dùng chọn “Quên mật khẩu”:  - Hệ thống chuyển đến giao diện khôi phục mật khẩu.  - Người dùng nhập email để nhận hướng dẫn đặt lại mật khẩu. |
| **Kết quả trả về** | - Thành công: Người dùng được đăng nhập vào hệ thống và chuyển đến trang chính.  - Thất bại: Hiển thị lỗi và cho phép người dùng thử lại hoặc chọn “Quên mật khẩu”. |

**3.4.3. Sơ đồ use case đăng ký tài khoản**

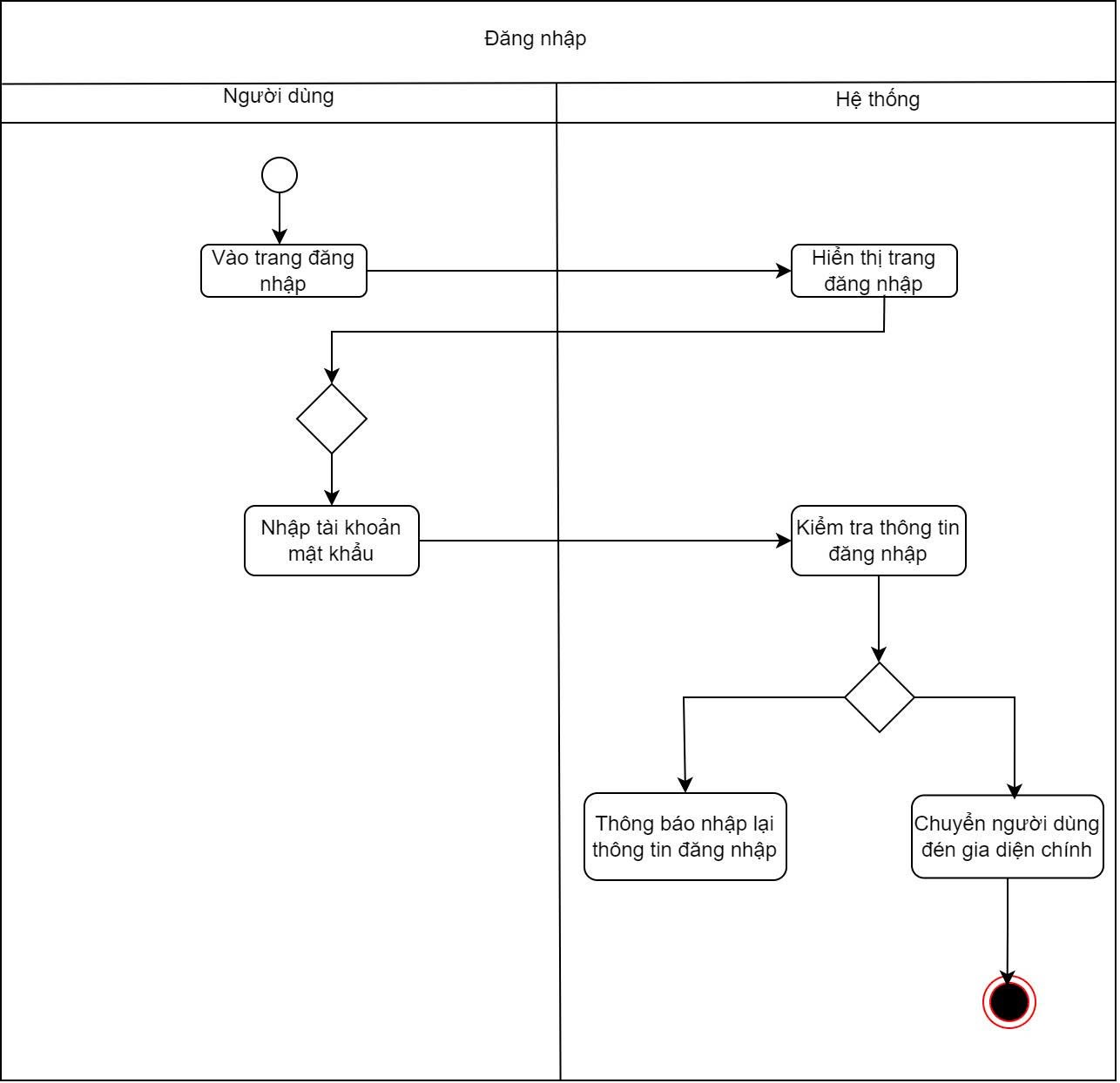


**Bảng 3.3. Bảng mô tả use case chi tiết chức năng đăng ký tài khoản**

|  |  |
| --- | --- |
| **Tác nhân** | Người dùng mới: là người chưa có tài khoản, muốn tạo tài khoản để sử dụng hệ thống. |
| **Sự kiện đầu vào** | - Người dùng truy cập vào giao diện đăng ký.  - Nhập thông tin cá nhân: tên, email, số điện thoại, mật khẩu. |
| **Các luồng sự kiện** | **Luồng chính:**   1. Người dùng mở màn hình “Đăng ký”. 2. Hệ thống hiển thị biểu mẫu yêu cầu “Nhập thông tin”. 3. Người dùng nhập đầy đủ thông tin → nhấn nút “Đăng ký”. 4. Hệ thống thực hiện “Kiểm tra hợp lệ”:   Kiểm tra định dạng email, độ mạnh mật khẩu, số điện thoại…   1. Nếu hợp lệ → hệ thống tạo tài khoản. 2. Sau khi đăng ký thành công → “Chuyển đến trang đăng nhập”.   **Luồng thay thế:**  Nếu thông tin không hợp lệ:  + Hệ thống “Hiển thị lỗi nếu có” (ví dụ: email đã tồn tại, mật khẩu quá yếu…).  + Người dùng được yêu cầu chỉnh sửa và thử lại. |
| **Kết quả trả về** | **Thành công**: Tài khoản được tạo, người dùng được chuyển đến trang đăng nhập.  **Thất bại**: Hiển thị lỗi cụ thể, người dùng cần sửa thông tin để đăng ký lại. |

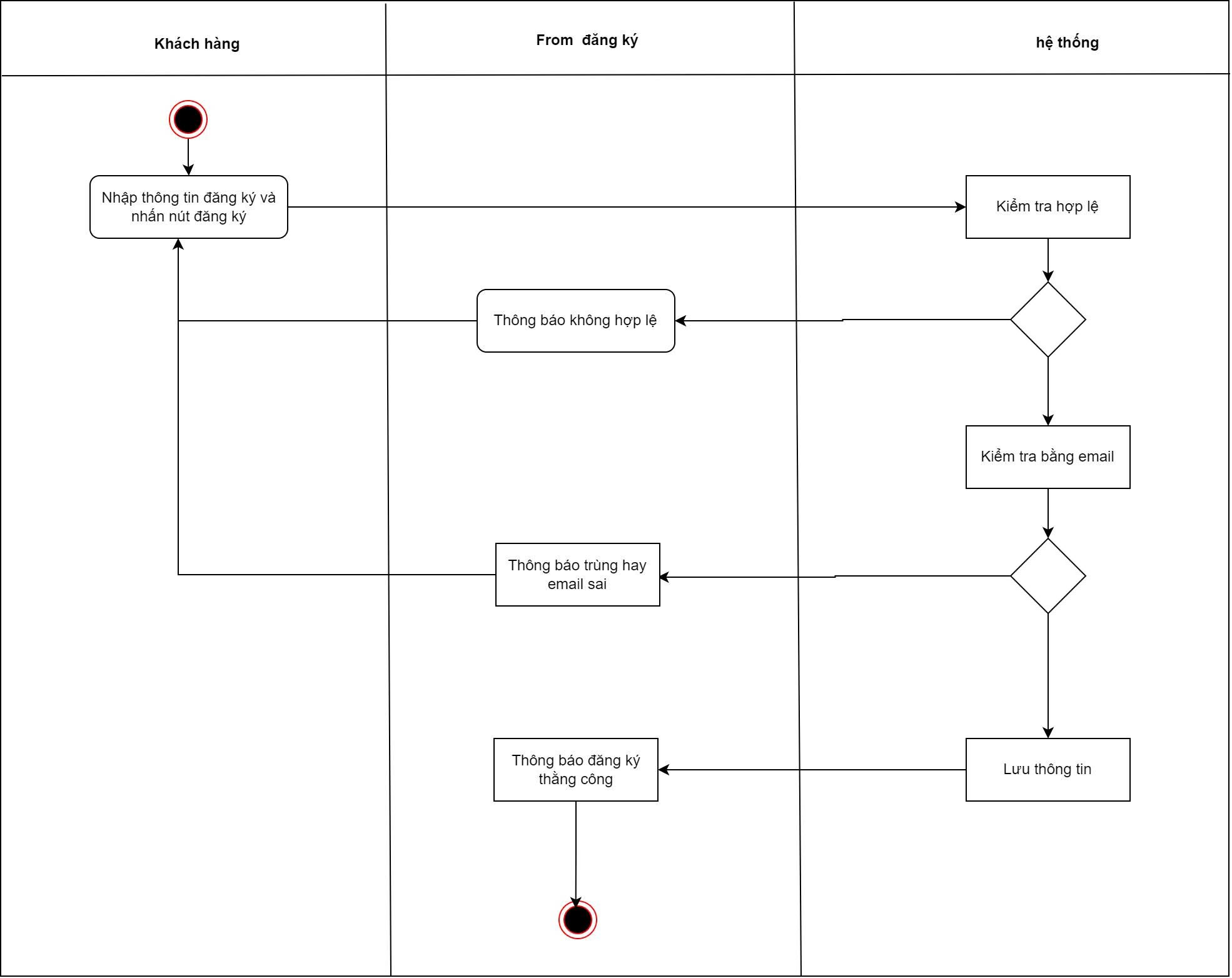
## 3.5. Sơ đồ luồng hoạt động

**3.5.1. Sơ đồ luồng đăng nhập tài khoản**

****

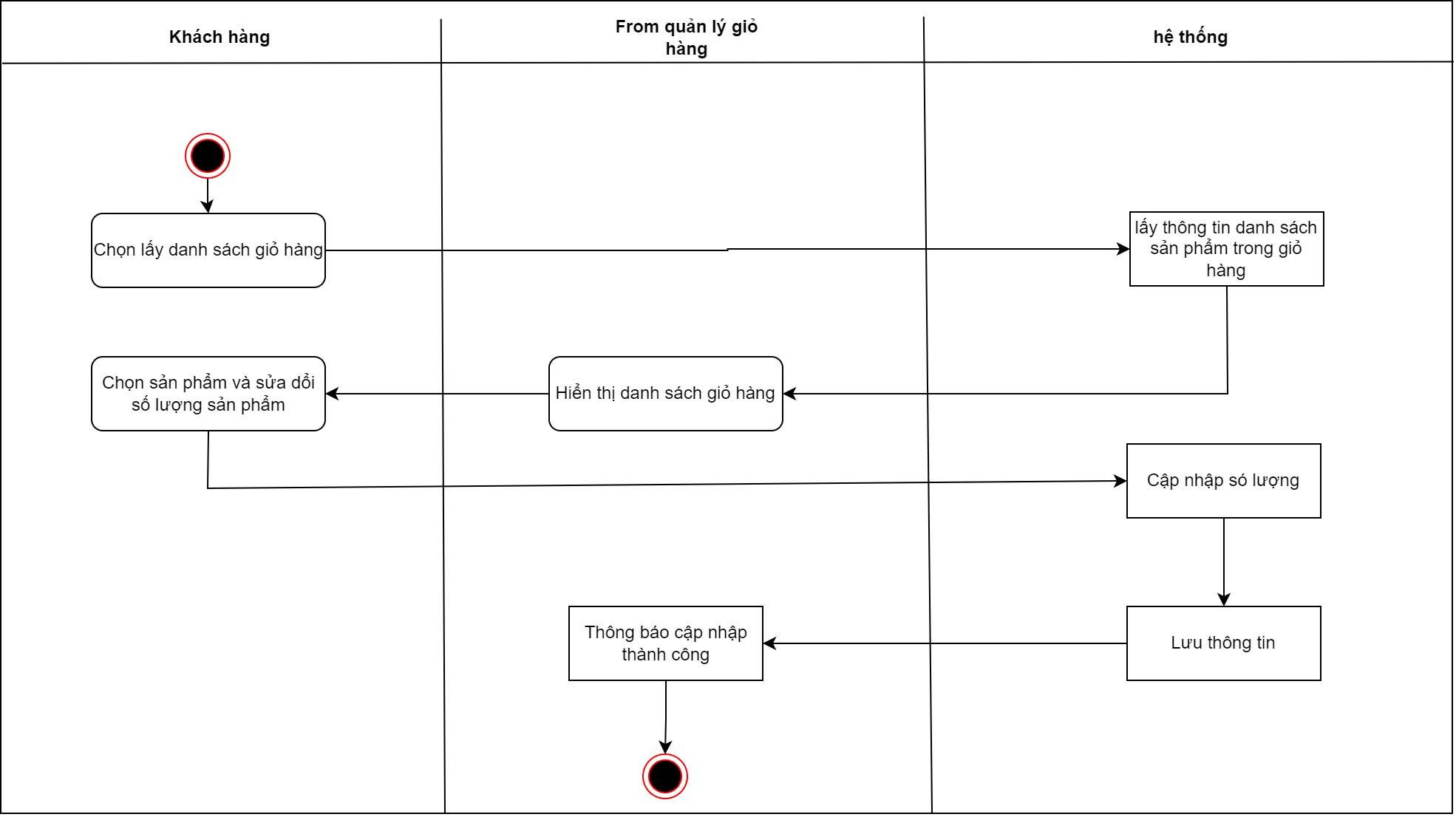
Quy trình đăng nhập của người dùng bao gồm các bước sau: Người dùng mở ứng dụng và truy cập vào màn hình đăng nhập, sau đó nhập thông tin tài khoản (email/số điện thoại) và mật khẩu. Hệ thống sẽ kiểm tra tính hợp lệ của thông tin đăng nhập. Nếu thông tin chính xác, người dùng được chuyển đến màn hình chính của ứng dụng. Ngược lại, nếu thông tin sai, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu người dùng nhập lại.

**3.5.2. Sơ đồ luồng đăng ký tài khoản**

****

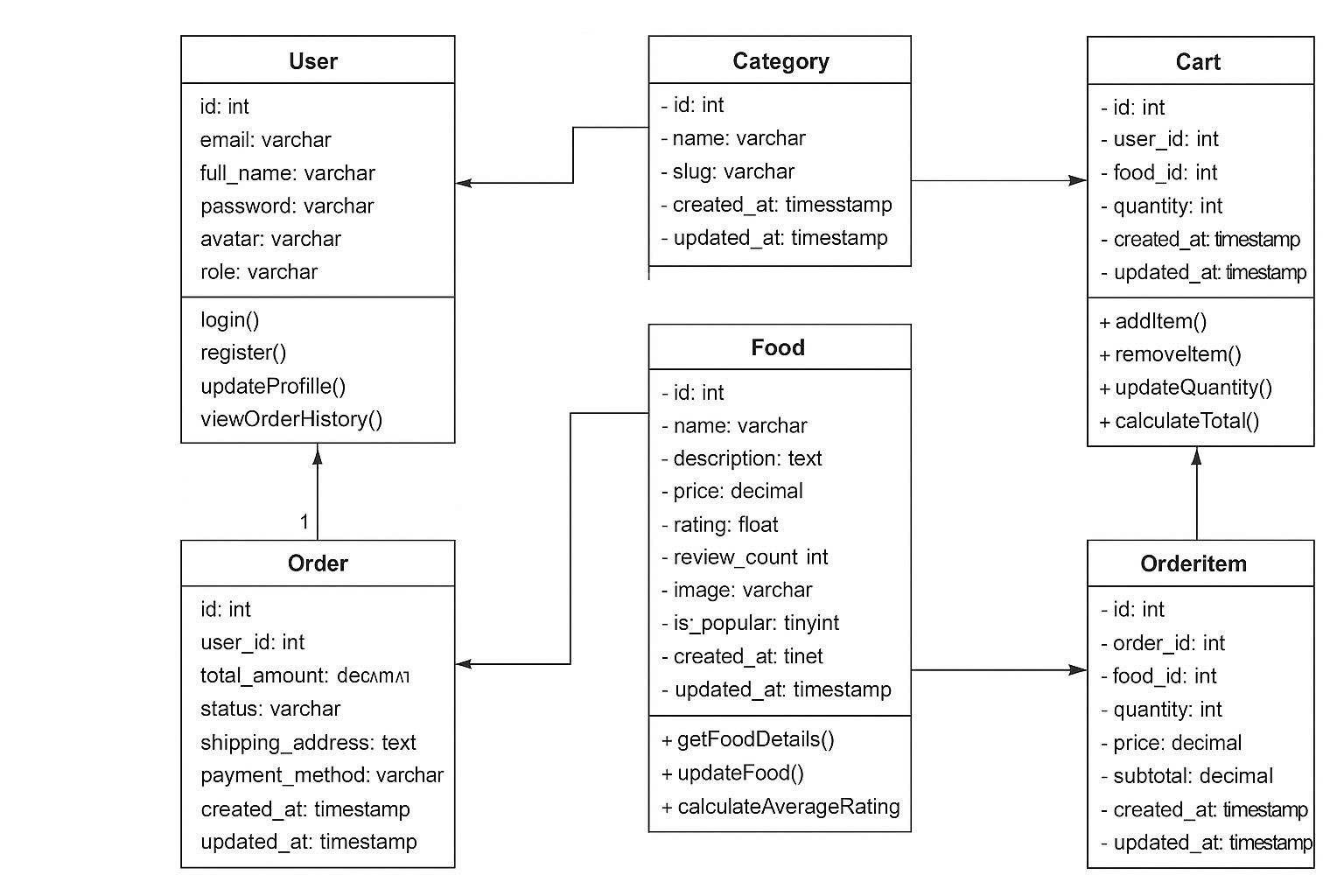
Quy trình đăng ký tài khoản mới gồm các bước: Người dùng truy cập màn hình đăng ký và điền đầy đủ thông tin cá nhân bao gồm họ tên, email, số điện thoại, mật khẩu và xác nhận mật khẩu. Hệ thống kiểm tra tính hợp lệ của dữ liệu đầu vào như định dạng email, độ mạnh mật khẩu và sự trùng khớp giữa mật khẩu với xác nhận mật khẩu. Nếu thông tin hợp lệ và email chưa tồn tại trong hệ thống, tài khoản mới sẽ được tạo thành công và người dùng có thể đăng nhập. Nếu có lỗi như email đã tồn tại hoặc thông tin không hợp lệ, hệ thống sẽ thông báo để người dùng chỉnh sửa.

**3.5.3. Sơ đồ Lường quản lí giỏ hàng**

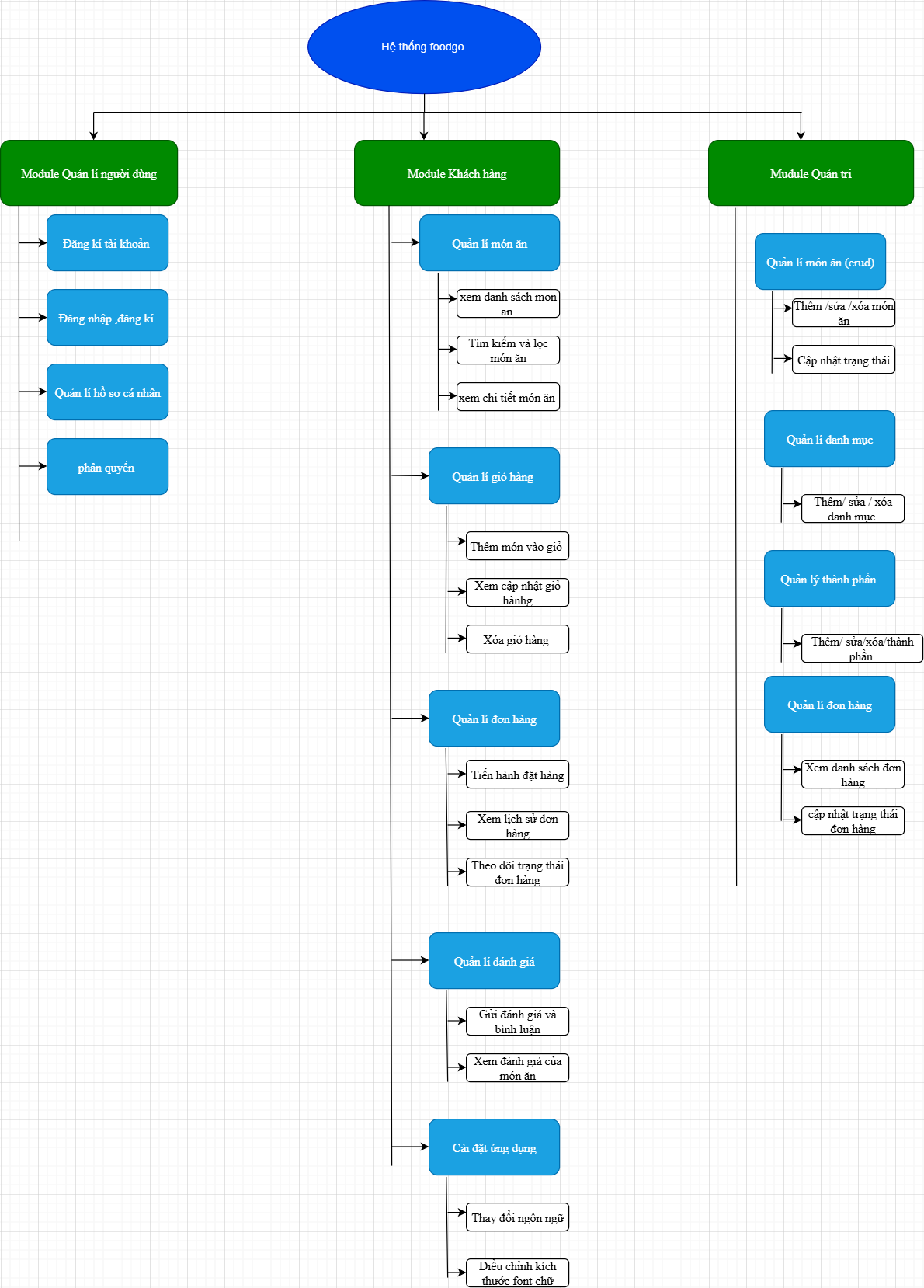
****

Quy trình quản lý giỏ hàng cho phép người dùng thực hiện các thao tác: Người dùng xem danh sách các món ăn đã thêm vào giỏ hàng với thông tin chi tiết về tên món, số lượng và giá tiền. Người dùng có thể điều chỉnh số lượng món ăn (tăng hoặc giảm), xóa món khỏi giỏ hàng nếu không muốn đặt nữa. Hệ thống tự động cập nhật tổng giá trị đơn hàng sau mỗi thay đổi. Khi đã hài lòng với giỏ hàng, người dùng có thể tiến hành đặt hàng bằng cách xác nhận thông tin giao hàng và phương thức thanh toán. Sau khi xác nhận, đơn hàng được gửi đi và người dùng nhận được thông báo xác nhận đơn hàng thành công

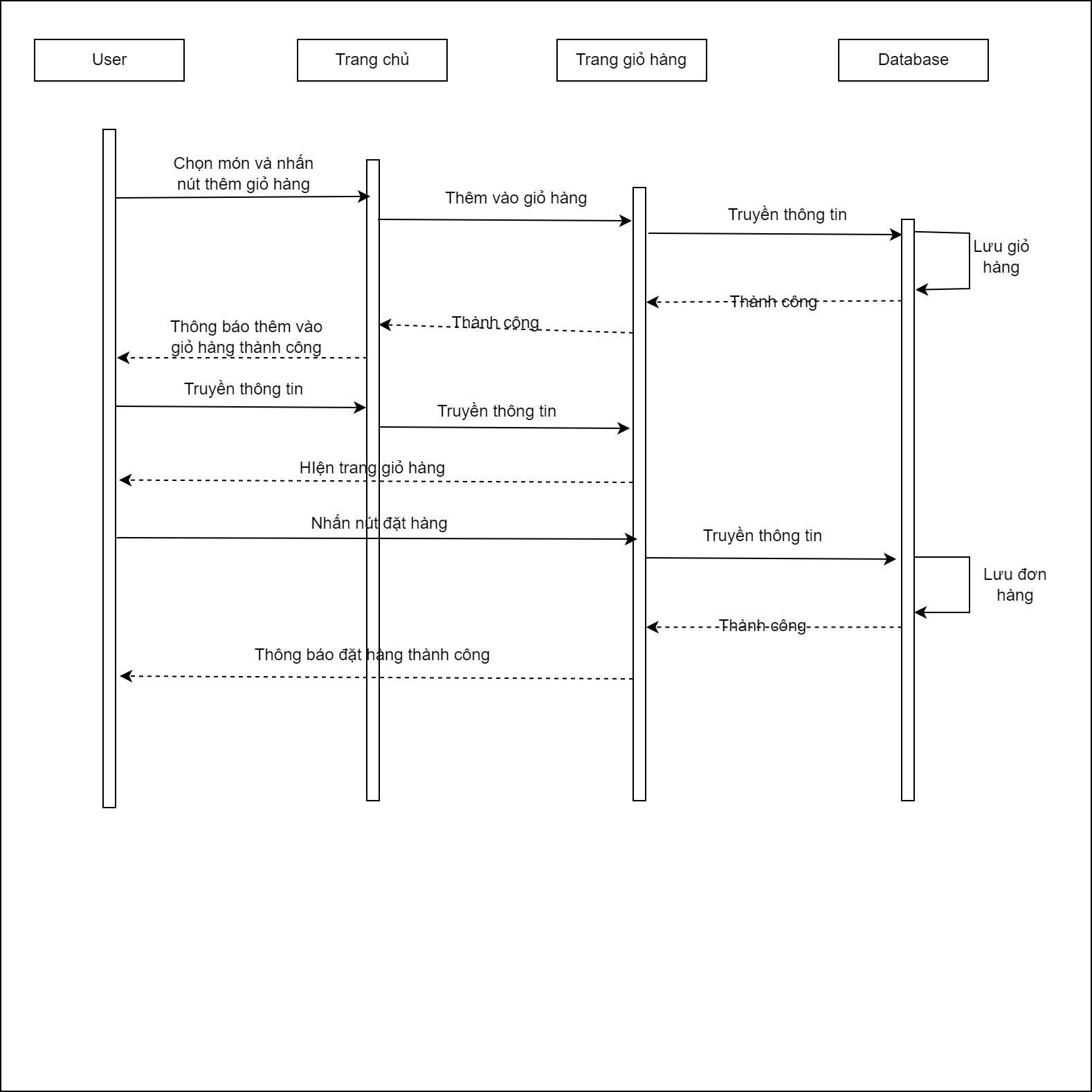
## 3.6. Sơ đồ lớp



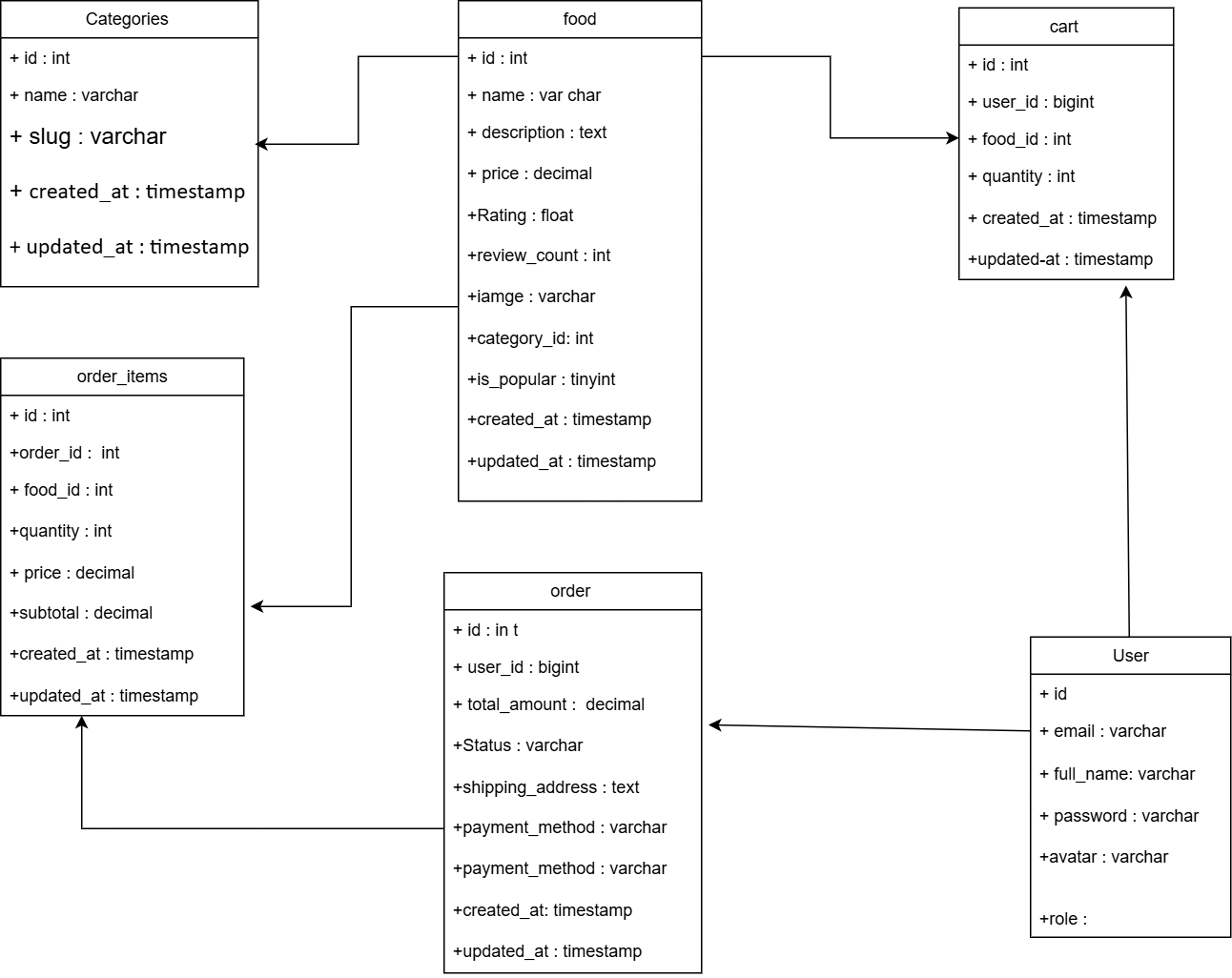
## 3.7. Sơ đồ phân rã chức năng(Functional Decomposition Diagram)

****

## 3.8. Sơ đồ tuấn tự

****

## 3.9. Cơ sở dữ liệu



**Bảng 3.4. Bảng mô tả cơ sở dữ liệu bảng users**

|  |  |  |
| --- | --- | --- |
| **Trường** | **Kiểu dữ liệu** | **Ý nghĩa** |
| id | bigint (PK, AI) | Khóa chính, tự động tăng |
| full\_name | varchar | Họ và tên người dùng |
| email | varchar | Email duy nhất |
| password | varchar | Mật khẩu đã mã hóa |
| address | text | Địa chỉ người dùng |
| avatar | varchar (nullable) | Đường dẫn ảnh đại diện |
| role | enum('customer','seller','admin') | Vai trò người dùng |

**Bảng 3.5. Bảng mô tả cơ sở dữ liệu bảng categories**

|  |  |  |
| --- | --- | --- |
| **Trường** | **Kiểu dữ liệu** | **Ý nghĩa** |
| id | int (PK, AI) | Khóa chính, tự động tăng |
| name | varchar | Tên danh mục |
| slug | varchar | Đường dẫn thân thiện SEO |
| created\_at | timestamp | Thời gian tạo |
| updated\_at | timestamp | Thời gian cập nhật |

**Bảng 3.6 Bảng mô tả cơ sở dữ liệu bảng foods**

|  |  |  |
| --- | --- | --- |
| **Trường** | **Kiểu dữ liệu** | **Ý nghĩa** |
| id | int (PK, AI) | Khóa chính |
| name | varchar | Tên món ăn |
| description | text | Mô tả món ăn |
| price | decimal | Giá món ăn |
| rating | float | Điểm đánh giá trung bình |
| review\_count | int | Số lượng đánh giá |
| image | varchar | Đường dẫn hình ảnh |
| category\_id | int (FK) | Liên kết bảng categories |
| is\_popular | tinyint (0/1) | Đánh dấu món ăn phổ biến |
| created\_at | timestamp | Thời gian tạo |
| updated\_at | timestamp | Thời gian cập nhật |

**Bảng 3.7. Bảng mô tả cơ sở dữ liệu bảng orders**

|  |  |  |
| --- | --- | --- |
| **Trường** | **Kiểu dữ liệu** | **Ý nghĩa** |
| id | int (PK, AI) | Khóa chính |
| user\_id | bigint (FK) | Liên kết bảng users |
| total\_amount | decimal | Tổng số tiền đơn hàng |
| status | varchar | Trạng thái đơn hàng |
| shipping\_address | text | Địa chỉ giao hàng |
| payment\_method | varchar | Phương thức thanh toán |
| payment\_status | varchar | Trạng thái thanh toán |
| created\_at | timestamp | Thời gian tạo |
| updated\_at | timestamp | Thời gian cập nhật |

**Bảng 3.8. Bảng mô tả cơ sở dữ liệu bảng order\_items**

|  |  |  |
| --- | --- | --- |
| **Trường** | **Kiểu dữ liệu** | **Ý nghĩa** |
| id | int (PK, AI) | Khóa chính |
| order\_id | int (FK) | Liên kết bảng orders |
| food\_id | int (FK) | Liên kết bảng foods |
| quantity | int | Số lượng món ăn |
| price | decimal | Giá tại thời điểm đặt |
| subtotal | decimal | Tổng tiền (quantity × price) |
| created\_at | timestamp | Thời gian tạo |
| updated\_at | timestamp | Thời gian cập nhật |

**Bảng 3.9. Bảng mô tả cơ sở dữ liệu bảng cart\_items**

|  |  |  |
| --- | --- | --- |
| **Trường** | **Kiểu dữ liệu** | **Ý nghĩa** |
| id | int (PK, AI) | Khóa chính |
| user\_id | bigint (FK) | Liên kết bảng users |
| food\_id | int (FK) | Liên kết bảng foods |
| quantity | int | Số lượng món ăn trong giỏ |
| created\_at | timestamp | Thời gian thêm vào giỏ |
| updated\_at | timestamp | Thời gian cập nhật |

## 3.10. Phân tích mô tả API

**3.10.1.** **Authentication & User**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Bảng liên quan | Chức năng |
| POST | /register | User | Đăng ký tài khoản mới |
| POST | /login | User | Đăng nhập, lấy token |
| POST | /logout | User | Đăng xuất, hủy token |
| GET | /user | User | Lấy thông tin user hiện tại |
| PUT | /profile | User | Cập nhật thông tin cá nhân |
| PUT | /password | User | Đổi mật khẩu |

**3.10.2. Categories**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Bảng liên quan | Chức năng |
| GET | /categories | Categories | Lấy danh sách danh mục món ăn |
| GET | /categories/{id} | Categories | Chi tiết danh mục theo ID |
| GET | /categories/{slug}/foods | Categories, Food | Lấy món ăn theo slug danh mục |

**3.10.3. Foods**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Bảng liên quan | Chức năng |
| GET | /foods | Food | Lấy danh sách tất cả món ăn |
| GET | /foods/{id} | Food | Chi tiết món ăn theo ID |
| GET | /foods/category/{categoryId} | Food, Categories | Lấy món ăn theo category |
| GET | /foods/featured | Food | Lấy danh sách món ăn nổi bật |

**3.10.4. Cart**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Bảng liên quan | Chức năng |
| GET | /cart | Cart, Food | Lấy giỏ hàng của user |
| POST | /cart/add | Cart | Thêm món ăn vào giỏ |
| PUT | /cart/update/{foodId} | Cart | Cập nhật số lượng món trong giỏ |
| DELETE | /cart/remove/{foodId} | Cart | Xóa 1 món khỏi giỏ |
| DELETE | /cart/clear | Cart | Xóa toàn bộ giỏ hàng |
| GET | /cart/count | Cart | Lấy số lượng món trong giỏ |

**3.10.5. Orders**

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Endpoint | Bảng liên quan | Chức năng |
| GET | /orders | Order, Order\_items | Lấy danh sách đơn hàng |
| POST | /orders | Order, Order\_items, Cart | Tạo đơn hàng mới từ giỏ |
| GET | /orders/{id} | Order, Order\_items | Chi tiết đơn hàng |
| PUT | /orders/{id}/cancel | Order | Hủy đơn hàng |

# **4. THIẾT KẾ VÀ TRIỂN KHAI VỚI LARAVEL VÀ FLUTTER**

## 4.1. Cài đặt và cấu hình môi trường Laravel

**4.1.1.Yêu cầu hệ thống**

Để triển khai hệ thống backend sử dụng Laravel Framework, môi trường phát triển cần đáp ứng các yêu cầu tối thiểu sau:

• PHP: Phiên bản 8.1 hoặc cao hơn (để tận dụng các tính năng mới như enum và cải thiện hiệu năng).

• Composer: Công cụ quản lý dependencies cho PHP, phiên bản 2.0 trở lên.

• MySQL: Hệ quản trị cơ sở dữ liệu quan hệ, phiên bản 8.0 trở lên (hỗ trợ JSON và indexing nâng cao).

• Web Server: Apache (phiên bản 2.4+) hoặc Nginx (phiên bản 1.18+); có thể sử dụng Laravel built-in server (php artisan serve) trong quá trình phát triển.

• Git: Công cụ quản lý phiên bản mã nguồn, phiên bản 2.0 trở lên

Các yêu cầu này đảm bảo hệ thống hoạt động ổn định và tương thích với các phiên bản mới nhất của Laravel.

**4.1.2. Các bước cài đặt Laravel**

**Bước 1: Khởi tạo dự án**

+ Sử dụng Composer để tạo mới dự án Laravel với tên foodgo

+ Chạy lệnh sau trong terminal:

+ Copy code

+ composer create-project laravel/laravel foodgo-backend

+ cd foodgo-backend

+ Lệnh này sẽ tải xuống và cài đặt Laravel phiên bản mới nhất, tạo cấu trúc thư mục chuẩn.

**Bước 2: Cấu hình file môi trường (.env)**

File .env chứa các thông số cấu hình quan trọng của ứng dụng. Sao chép file .env.example thành .env và chỉnh sửa các thông số cần thiết, bao gồm:

+ Tên ứng dụng và môi trường: APP\_NAME=FoodGo, APP\_ENV=local.

+ Thông tin kết nối cơ sở dữ liệu: DB\_CONNECTION=mysql, DB\_DATABASE=food\_order, DB\_HOST=127.0.0.1, DB\_USERNAME=root, DB\_PASSWORD=your\_password.

+ Khóa bí mật JWT\_SECRET để mã hóa token xác thực (sẽ được tạo ở bước sau).

Việc cấu hình này đảm bảo ứng dụng kết nối đúng với cơ sở dữ liệu và hoạt động an toàn trong môi trường phát triển.

**Bước 3: Khởi tạo cơ sở dữ liệu**

Tạo cơ sở dữ liệu có tên foodgo\_db trong MySQL thông qua công cụ quản lý như phpMyAdmin hoặc lệnh SQL. Sau đó, thực thi lệnh:

+ Copy code

+ php artisan migrate

Lệnh này sẽ tạo các bảng dữ liệu theo cấu trúc định nghĩa trong các file migration, đảm bảo tính toàn vẹn dữ liệu.

**Bước 4: Cài đặt các package bổ sung**

Cài đặt các package cần thiết bằng Composer để mở rộng chức năng:

+ JWT Auth: **composer require tymon/jwt-auth** – Xử lý xác thực người dùng bằng JSON Web Token (JWT).

+ CORS Support: **composer require fruitcake/laravel-cors** – Cho phép ứng dụng Flutter gọi API từ domain khác.

+ Laravel Sanctum: **composer require laravel/sanctum** – Hỗ trợ xác thực API và quản lý token cho client.

Sau khi cài đặt, chạy **php artisan vendor:publish** để xuất bản các file cấu hình tương ứng.

**4.1.3. Cấu hình JWT Authentication**

Hệ thống sử dụng JWT (JSON Web Token) để xác thực người dùng, đảm bảo bảo mật và stateless. Các bước cấu hình bao gồm:

+ Xuất bản provider: **php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"**.

+ Tạo khóa bí mật: **php artisan jwt:secret** – Lệnh này tự động thiết lập giá trị JWT\_SECRET trong file .env.

+ Quy định thời gian sống của token: Mặc định 24 giờ cho access token và 7 ngày cho refresh token (có thể tùy chỉnh trong config/jwt.php).

+ Cập nhật model User để implement interface JWTSubject, giúp tương thích với JWT.

**4.1.4. Cấu hình CORS (Cross-Origin Resource Sharing)**

Để đảm bảo ứng dụng Flutter có thể giao tiếp với API từ domain khác, cấu hình CORS được thực hiện trong config/cors.php:

+ Cho phép tất cả các route có tiền tố /api/\*.

+ Hỗ trợ tất cả các HTTP methods: GET, POST, PUT, DELETE.

+ Cho phép mọi origin trong môi trường phát triển ở môi trường production, giới hạn cụ thể (ví dụ: chỉ cho phép domain của app).

+ Cho phép các headers cần thiết: Authorization, Content-Type, Accept.

Cấu hình này ngăn chặn lỗi CORS trong quá trình phát triển, đồng thời đảm bảo bảo mật khi triển khai.

## 4.2. Thiết kế và xử lý các route API

**4.2.1. Cấu trúc tổng quan routing**

Toàn bộ các API routes được định nghĩa trong file routes/api.php, được chia thành hai nhóm chính để tối ưu hóa bảo mật:

+ Public Routes: Các API công khai, không yêu cầu xác thực (ví dụ: đăng ký, đăng nhập, danh sách món ăn).

+ Protected Routes: Các API yêu cầu JWT token hợp lệ (ví dụ: giỏ hàng, đơn hàng), được bảo vệ bởi middleware.

Cấu trúc này tuân thủ nguyên tắc least privilege, chỉ cho phép truy cập khi cần thiết.

**4.2.2. Authentication Routes**

**Đăng ký (Register)**: Nhận dữ liệu full\_name, email, password từ người dùng. Kiểm tra hợp lệ (validation rules trong Laravel), mã hóa mật khẩu bằng bcrypt và lưu vào cơ sở dữ liệu. Trả về thông tin người dùng sau khi đăng ký thành công, kèm mã trạng thái 201.

**Đăng nhập (Login)**: Xác thực thông tin đăng nhập. Nếu thành công, tạo JWT token và trả về cùng thông tin người dùng (ví dụ: **{ "token": "jwt\_token\_here", "user": {...} }**). Client lưu token để gửi kèm trong header Authorization.

**Đăng xuất (Logout)**: Vô hiệu hóa token hiện tại bằng cách thêm vào blacklist, đảm bảo an toàn và ngăn chặn tái sử dụng.

Các route này sử dụng AuthController để xử lý logic.

**4.2.3. Categories API**

Các API phục vụ quản lý danh mục món ăn (public routes):

- Lấy danh sách danh mục: Trả về tất cả categories (id, name, slug) với mã 200.

- Chi tiết danh mục: Trả về thông tin cụ thể theo id.

- Lấy món ăn theo danh mục: Trả về danh sách món ăn thuộc slug tương ứng, hỗ trợ phân trang.

**4.2.4. Foods API**

Cung cấp các API liên quan đến món ăn (public routes):

+ Lấy danh sách món ăn với thông tin chi tiết (tên, mô tả, giá, hình ảnh, rating).

+ Lấy chi tiết món ăn theo id.

+ Lọc theo category\_id (ví dụ: query parameter **?category\_id=1**).

+ Lấy danh sách món ăn nổi bật (is\_popular = 1).

**4.2.5. Cart API (Protected)**

Các API phục vụ chức năng giỏ hàng (chỉ dành cho user đã đăng nhập):

+ Xem giỏ hàng, thêm món ăn, cập nhật số lượng, xóa món hoặc xóa toàn bộ giỏ.

+ Trả về tổng số lượng món trong giỏ để hiển thị badge trên UI.

**4.2.6. Orders API (Protected)**

Chức năng quản lý đơn hàng:

* Lấy danh sách đơn hàng của người dùng hiện tại.
* Tạo đơn hàng mới từ giỏ hàng (bao gồm tính tổng tiền, địa chỉ giao hàng, phương thức thanh toán).
* Xem chi tiết đơn hàng và cho phép hủy nếu đơn chưa giao (trạng thái pending).

**4.2.7. Middleware xác thực**

Middleware JWT được áp dụng cho tất cả Protected Routes, thực hiện:

* Lấy token từ header Authorization (định dạng: Bearer <token>).
* Kiểm tra tính hợp lệ và hạn sử dụng của token.
* Trả về lỗi 401 Unauthorized nếu token không hợp lệ hoặc hết hạn.

Middleware này được đăng ký trong app/Http/Kernel.php.

**4.2.8. Controllers**

* AuthController: Quản lý xác thực người dùng và JWT token (ví dụ: validate input, generate token).
* CategoryController: Quản lý danh mục và các món ăn trong danh mục (sử dụng Eloquent để query).
* FoodController: Xử lý logic lấy danh sách, lọc, tìm kiếm món ăn (hỗ trợ pagination và sorting).
* CartController: Quản lý giỏ hàng, tính tổng tiền tự động (sử dụng relationships trong Eloquent).
* OrderController: Xử lý tạo, hiển thị, cập nhật và hủy đơn hàng (bao gồm validation và transaction để đảm bảo atomicity).

## 4.3. Kết nối và quản lý cơ sở dữ liệu

**4.3.1. Migration Files**

Các migration định nghĩa cấu trúc bảng trong cơ sở dữ liệu foodgo\_db:

* Users: Lưu thông tin người dùng (id, name, email, password).
* Categories: Lưu danh mục món ăn (id, name, slug).
* Foods: Lưu thông tin chi tiết món ăn (id, name, description, price, image, category\_id).
* Orders: Lưu thông tin đơn hàng (id, user\_id, total, status).
* Order\_Items: Lưu chi tiết món trong từng đơn hàng (order\_id, food\_id, quantity).
* Cart\_Items: Lưu giỏ hàng của người dùng (user\_id, food\_id, quantity).

Tất cả foreign key được thiết lập ràng buộc với chế độ onDelete cascade để đảm bảo toàn vẹn dữ liệu.

**4.3.2. Eloquent Models và Relationships**

Sử dụng Eloquent ORM để tương tác với cơ sở dữ liệu một cách an toàn:

* User Model: Implement JWTSubject, có quan hệ hasMany với orders và cart\_items.
* Category Model: Quan hệ hasMany với foods.
* Food Model: Quan hệ belongsTo với category, hasMany với order\_items và cart\_items.
* Order Model: Quan hệ belongsTo user, hasMany order\_items.
* OrderItem Model: Quan hệ belongsTo order và food.
* CartItem Model: Quan hệ belongsTo user và food.

Các relationships này giúp truy vấn dữ liệu hiệu quả và tránh N+1 query problem.

**4.3.3. Database Seeding**

Seeder được dùng để thêm dữ liệu mẫu phục vụ kiểm thử:

* CategorySeeder: Thêm các danh mục như Pizza, Burger, Sushi, Pasta, Dessert.
* FoodSeeder: Thêm các món ăn mẫu cho từng danh mục (ví dụ: dữ liệu JSON với hình ảnh giả).
* UserSeeder: Tạo tài khoản admin và một số tài khoản khách hàng mẫu.

Chạy lệnh **php artisan db:seed** để khởi tạo dữ liệu mẫu.

## 4.3. Kết nối ứng dụng Flutter với API Laravel.

**4.4.1. Khởi tạo dự án Flutter**

Ứng dụng frontend được phát triển bằng Flutter, sử dụng:

* Flutter SDK ≥ 3.0.
* Dart SDK (tích hợp sẵn).
* Android Studio hoặc VS Code.
* Android SDK (cho Android) và Xcode (cho iOS).

Khởi tạo project foodgo\_app bằng lệnh: **flutter create foodgo\_app**, với cấu trúc thư mục chuẩn: lib/, android/, ios/, pubspec.yaml.

**4.4.2. Cài đặt Dependencies**

Các thư viện chính trong pubspec.yaml:

* dio (^5.0.0), http (^1.0.0): Gửi yêu cầu HTTP.
* provider (^6.0.0): Quản lý state.
* shared\_preferences (^2.0.0), flutter\_secure\_storage (^8.0.0): Lưu trữ dữ liệu cục bộ an toàn.
* cached\_network\_image (^3.0.0): Cache hình ảnh để tối ưu hiệu năng.
* flutter\_localizations (^0.0.0), intl (^0.18.0): Hỗ trợ đa ngôn ngữ.

**4.4.3. Cấu trúc thư mục Flutter**

Ứng dụng được tổ chức theo Clean Architecture:

* config/: Cấu hình API (base URL, endpoints).
* models/: Data models (User, Food, Order,...).
* services/: Giao tiếp API (ApiService, AuthService,...).
* providers/: Quản lý state (AuthProvider, CartProvider,...).
* screens/: Các màn hình UI (HomeScreen, CartScreen,...).
* widgets/: Các thành phần UI tái sử dụng (CustomButton,...).

**4.4.4. API Configuration**

Class ApiConfig quản lý tập trung các endpoint (ví dụ: loginEndpoint = '/api/login', foodsEndpoint = '/api/foods'). Điều này giúp dễ dàng thay đổi base URL khi triển khai ở các môi trường (development, staging, production).

**4.4.5. API Service Layer**

Class ApiService sử dụng Dio để gửi request:

* Tự động thêm JWT token vào header Authorization nếu có.
* Xử lý lỗi 401 (Unauthorized) bằng cách redirect đến màn hình đăng nhập, timeout (30 giây), và các exception khác (ví dụ: network error).

**4.4.6. AuthService**

Xử lý đăng nhập, đăng ký, đăng xuất, kiểm tra token hợp lệ.

**4.4.7. FoodService**

Lấy danh sách, chi tiết, tìm kiếm món ăn, hỗ trợ pagination.

**4.4.8. CartService**

Quản lý giỏ hàng: thêm, cập nhật, xóa, tính tổng tiền.

**4.4.9. OrderService**

Tạo, xem, hủy và theo dõi đơn hàng.

**4.4.10. Data Models**

Các model ánh xạ dữ liệu JSON: User, Food, Category, Order, OrderItem, CartItem. Mỗi model có phương thức fromJson() và toJson() để chuyển đổi dữ liệu dễ dàng.

**4.4.11. State Management với Provider**

* AuthProvider: Quản lý trạng thái đăng nhập và token.
* CartProvider: Quản lý giỏ hàng và tổng tiền (tự động cập nhật khi thay đổi).
* ThemeProvider: Quản lý theme, font size và ngôn ngữ hiển thị.

**4.4.12. Xử lý lỗi và bảo mật**

Trong Flutter, thêm xử lý lỗi toàn cục (global error handler) để hiển thị thông báo thân thiện. Sử dụng flutter\_secure\_storage để lưu token an toàn, tránh rủi ro lộ thông tin.

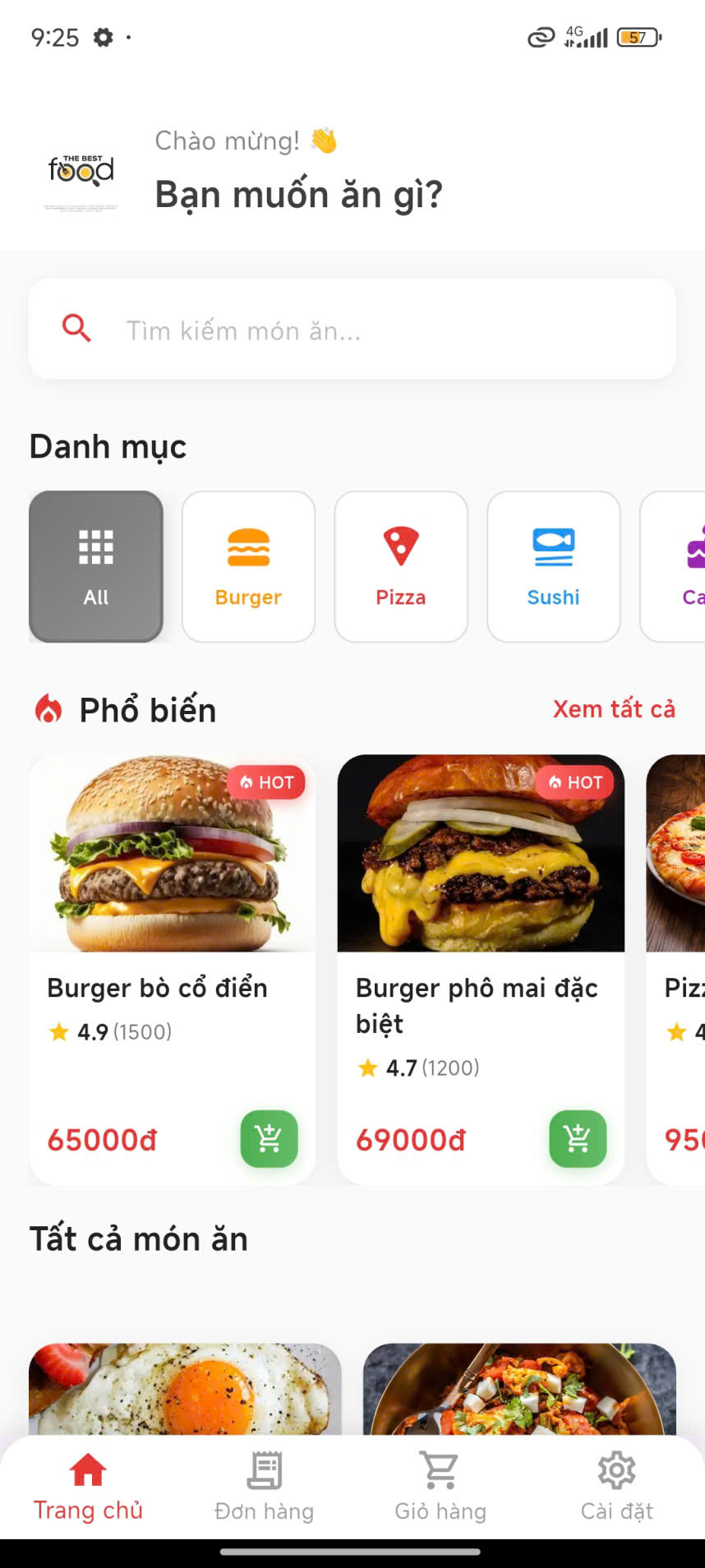
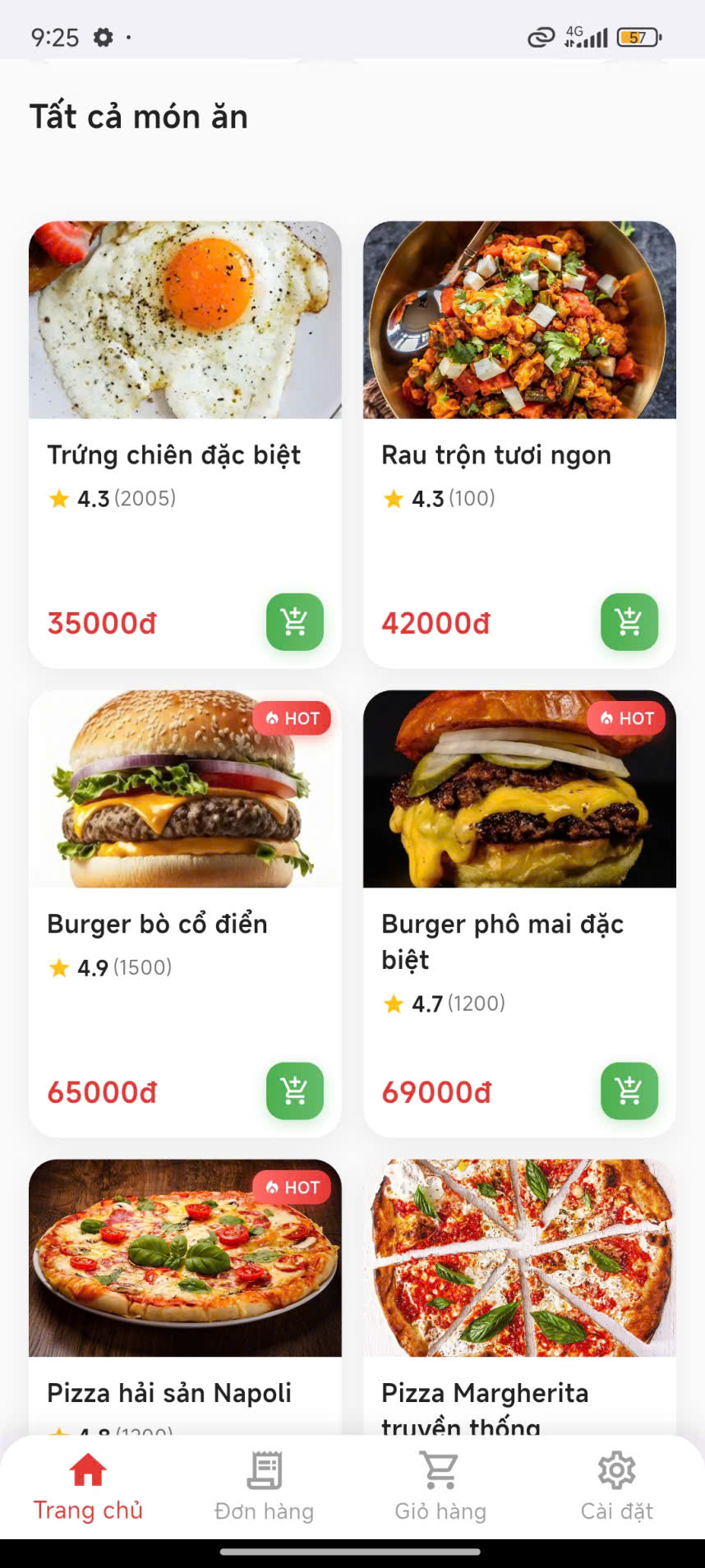
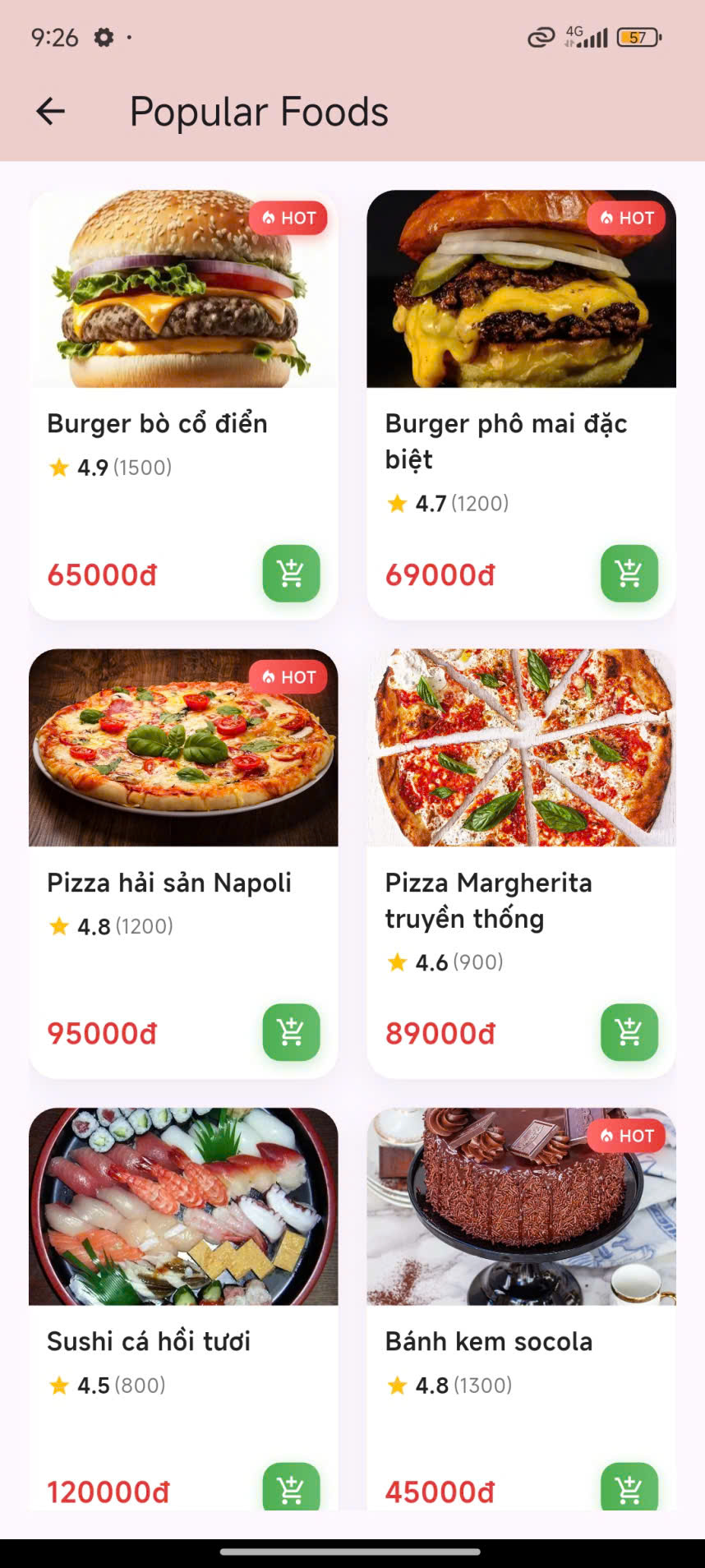
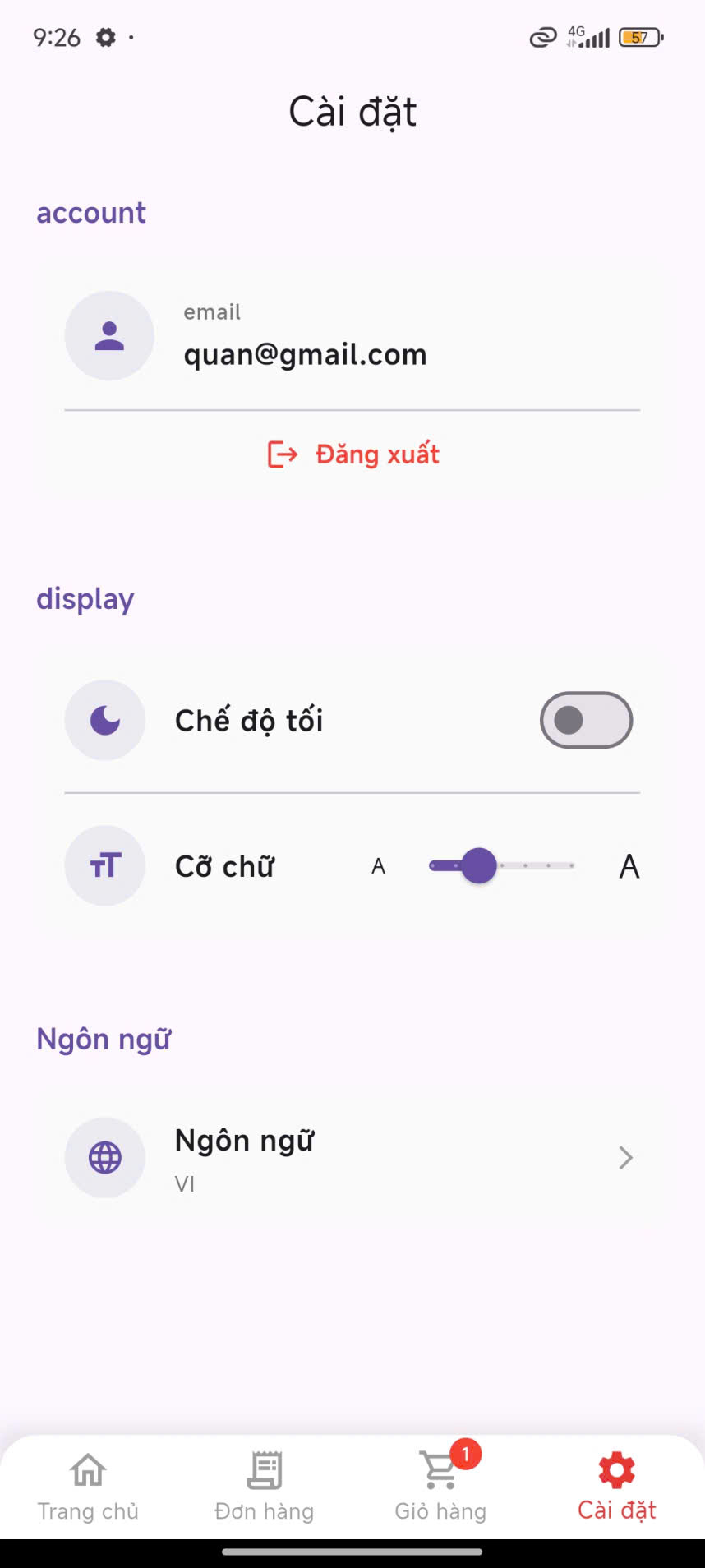
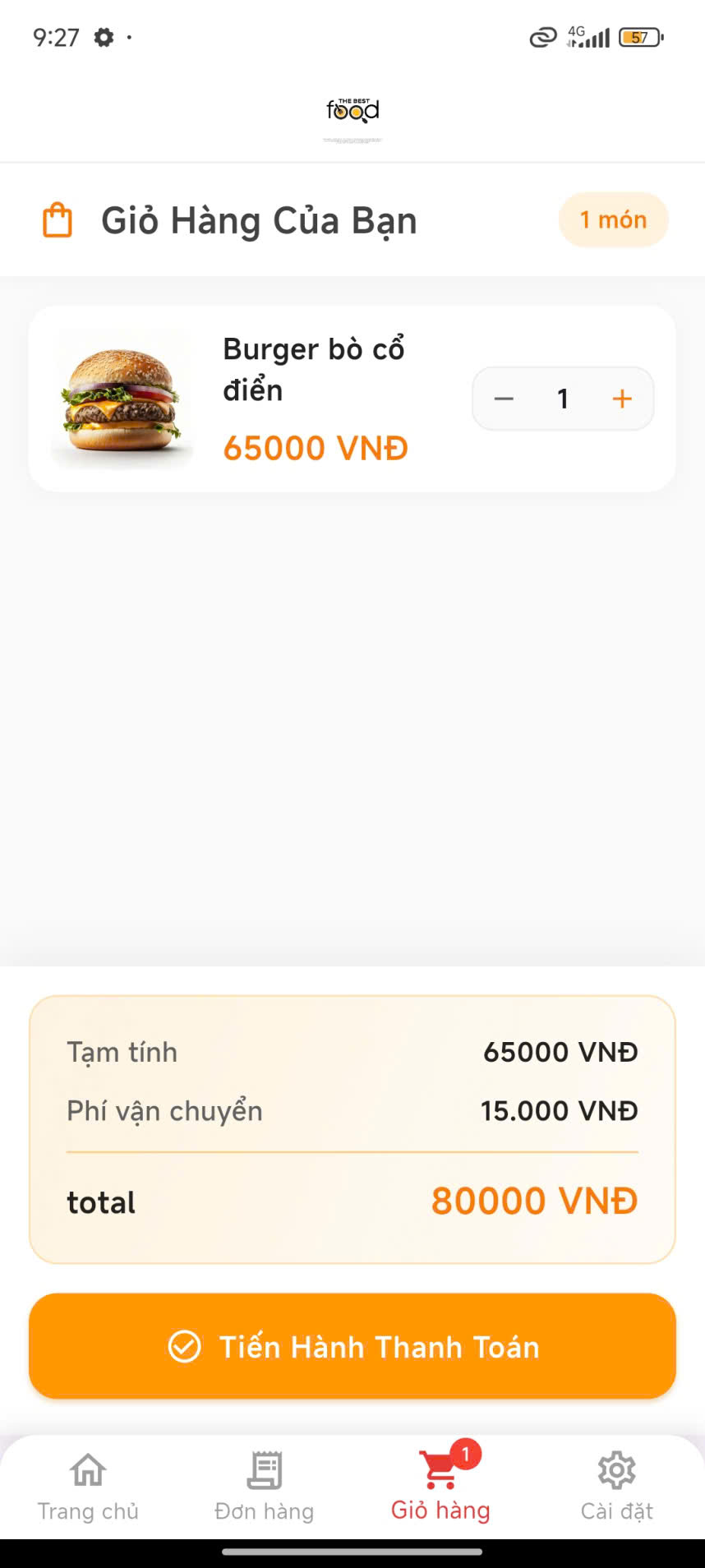
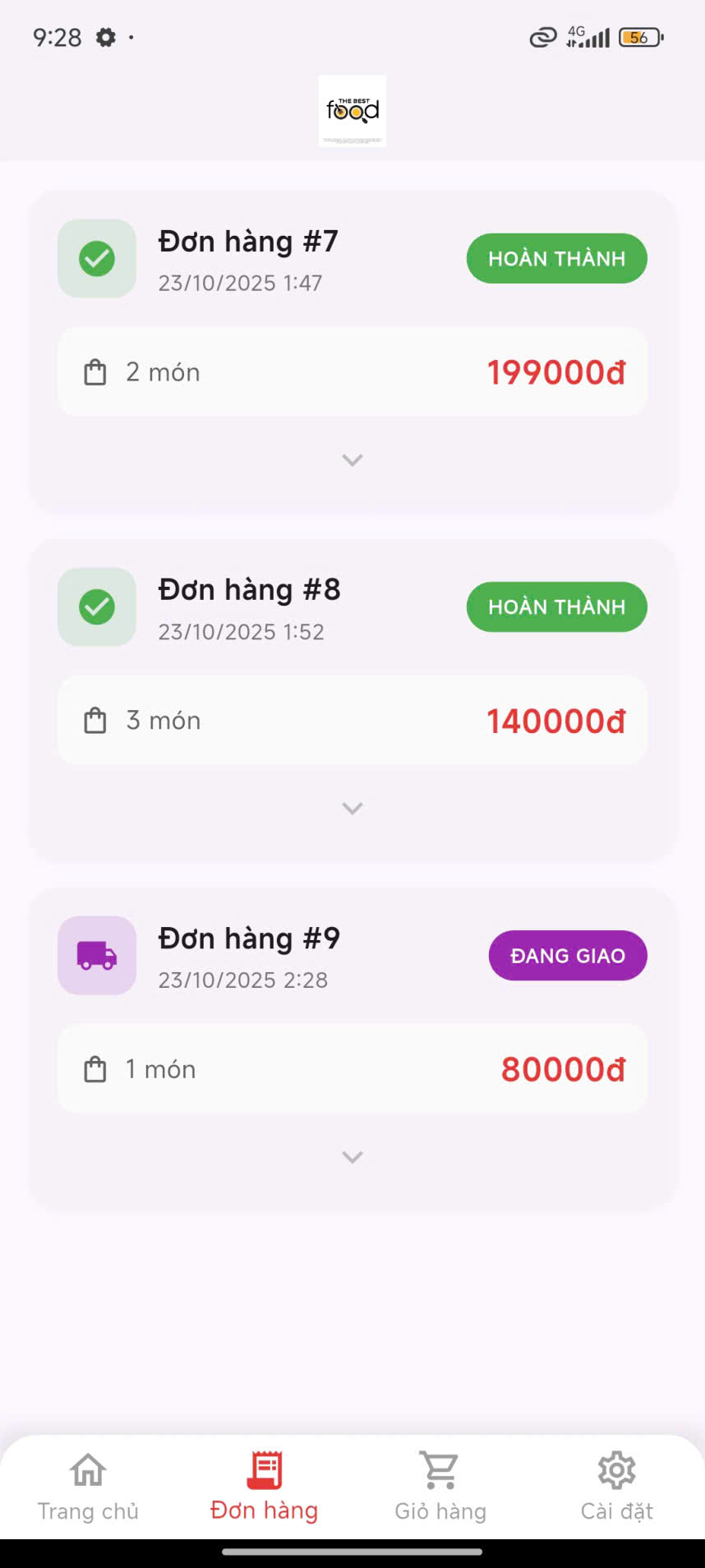
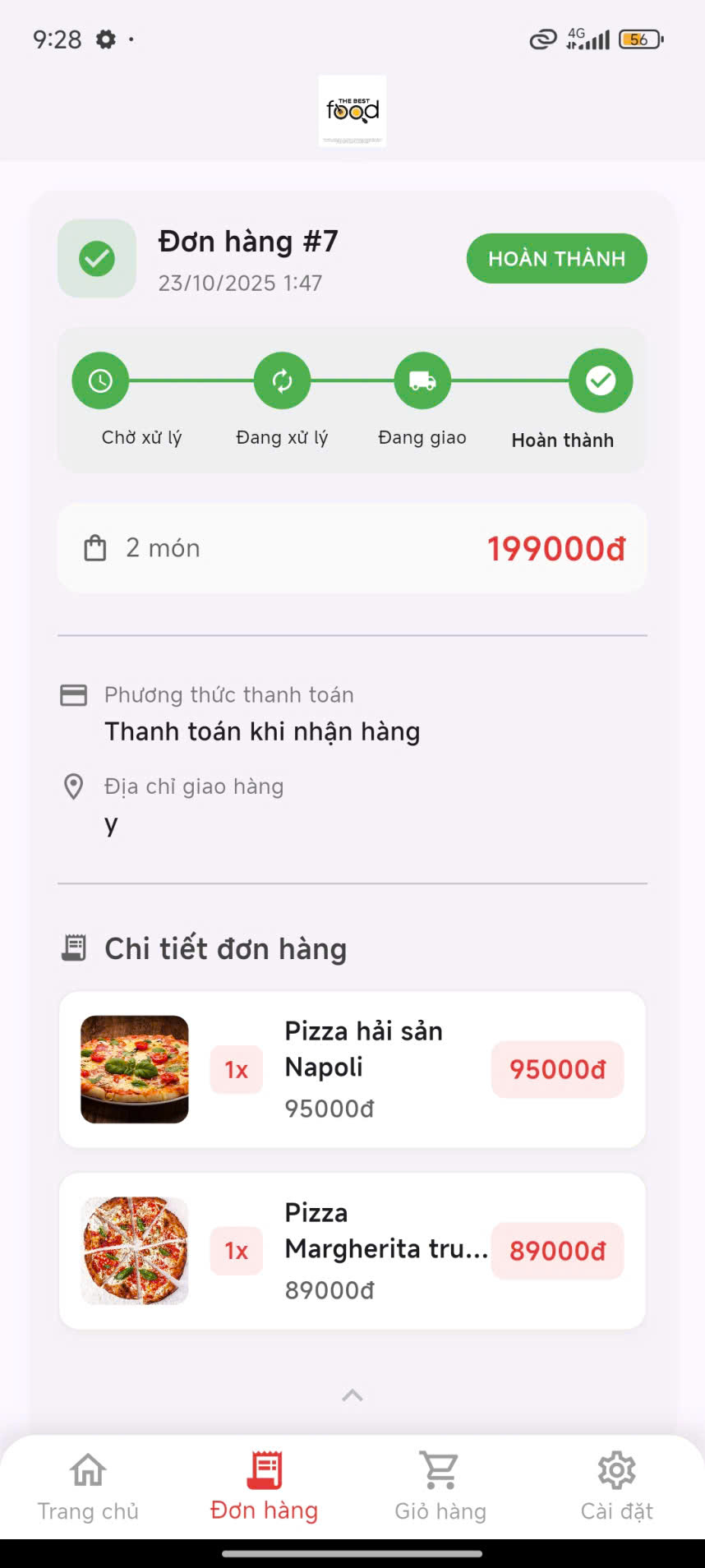
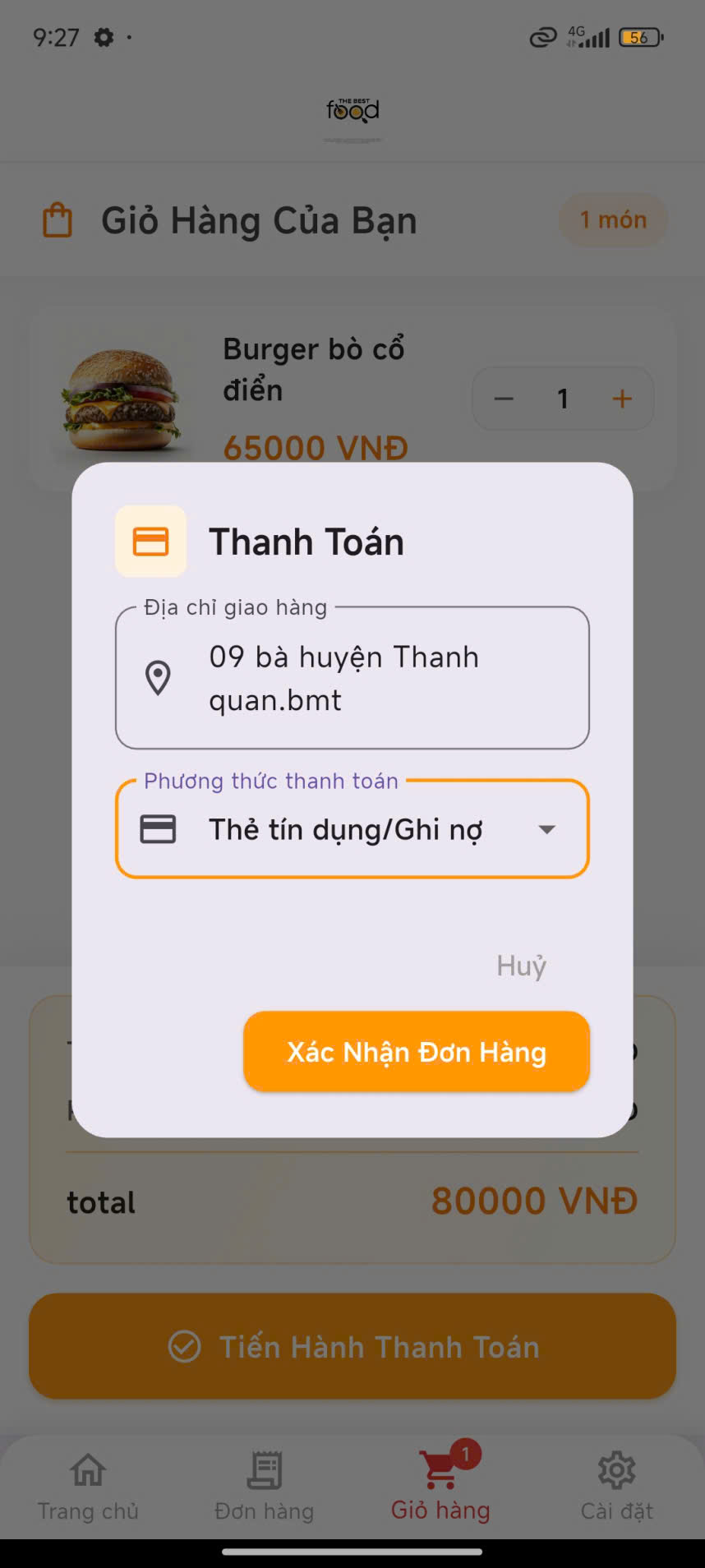
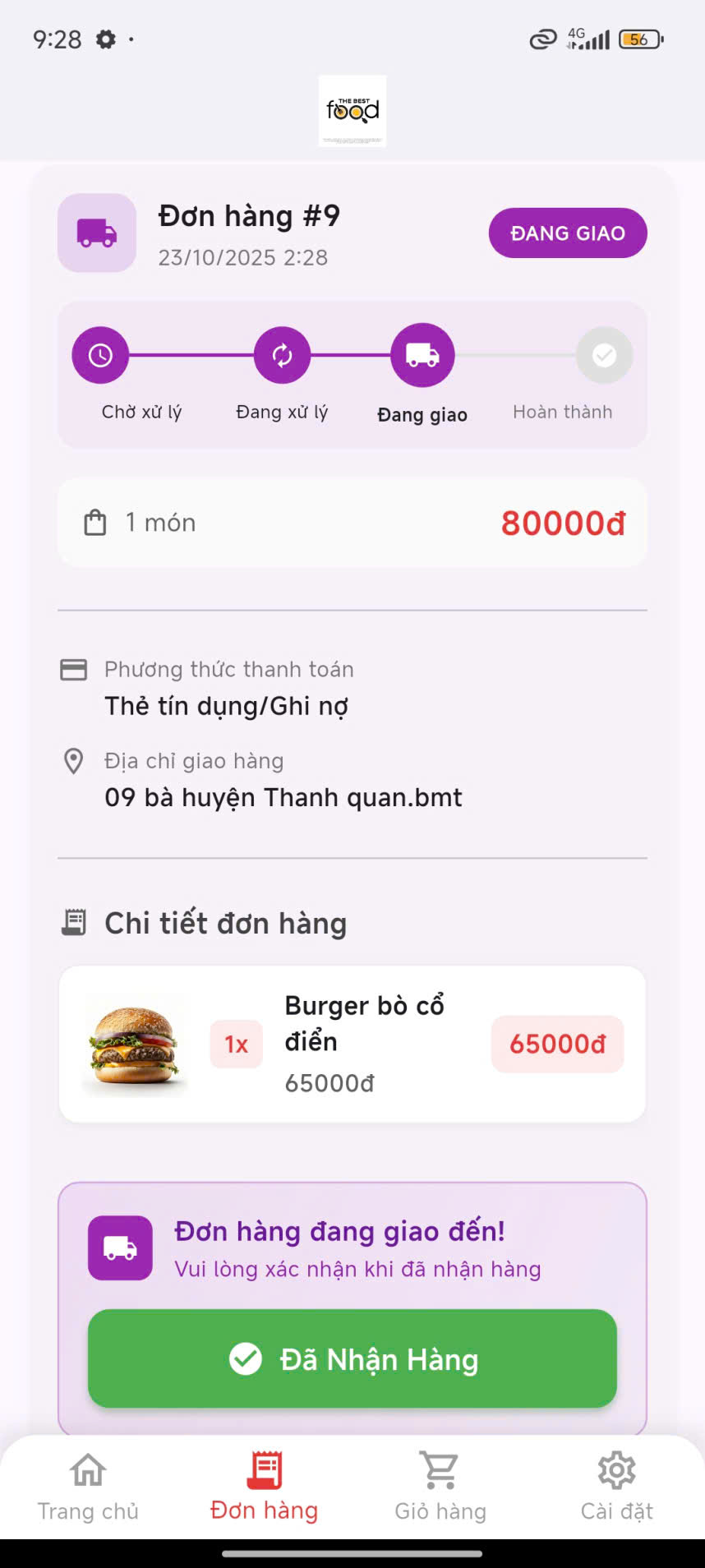
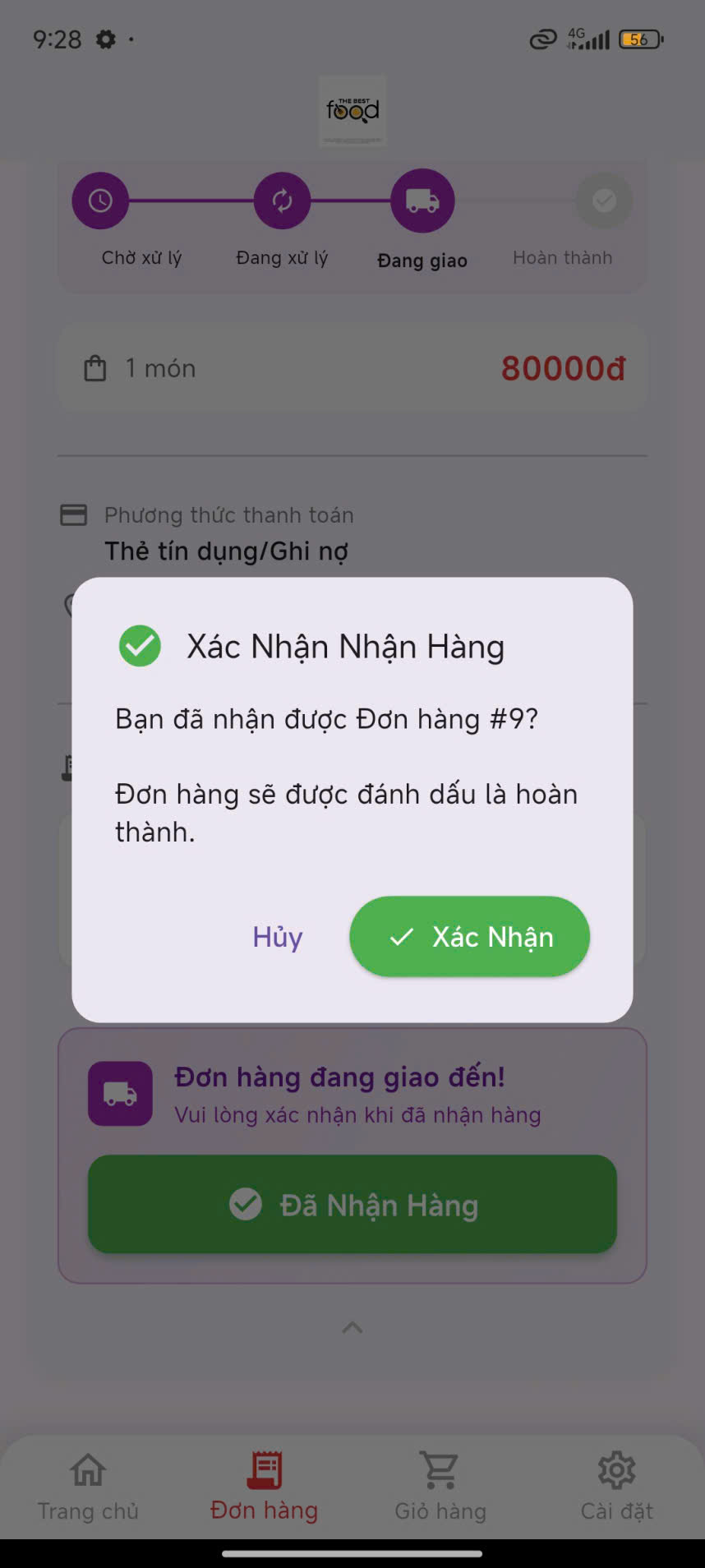
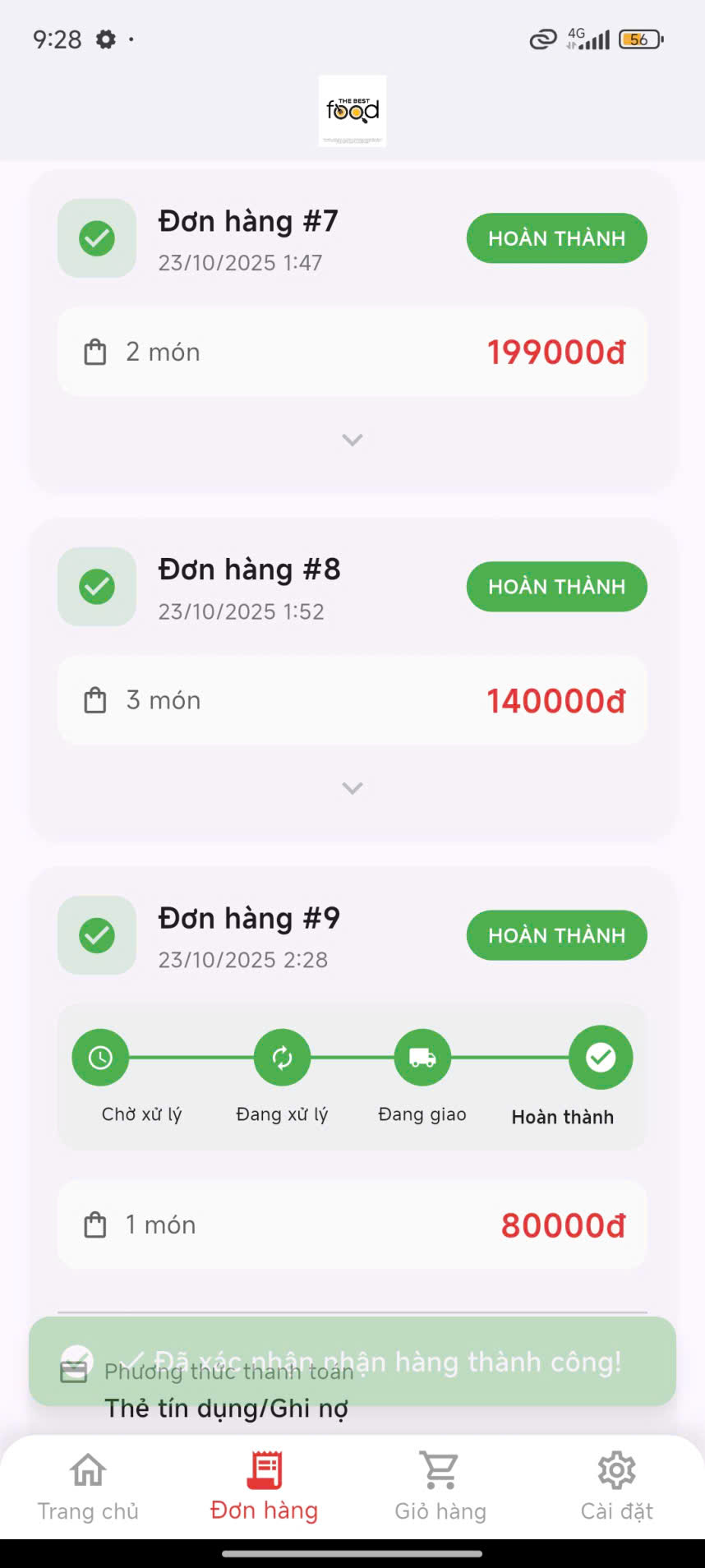
**4.4.13. Tích hợp và Kiểm thử**

Thực hiện kiểm thử toàn diện:

* API Connection: Sử dụng Postman để test endpoints, đảm bảo kết nối server ổn định.
* Authentication Flow: Kiểm thử đăng ký, đăng nhập, đăng xuất.
* Cart Flow: Kiểm thử thêm, cập nhật, xóa và thanh toán.
* Order Flow: Tạo, xem, hủy đơn hàng.
* UI Testing: Sử dụng Flutter's integration\_test package để đảm bảo giao diện tương thích đa màn hình.

# **5. KẾT QUẢ**

**Giao diện ứng dụng**



# **6. ĐÁNH GIÁ VÀ HƯỚNG PHÁT TRIỂN**

## 6.1. Ưu điểm của hệ thống

- Giao diện thân thiện, dễ sử dụng, phù hợp với nhiều đối tượng người dùng.

- Các chức năng đặt món, giỏ hàng, quản lý đơn hàng hoạt động ổn định.

- Ứng dụng được xây dựng bằng Android Studio, dễ triển khai và mở rộng.

- Dữ liệu lưu trữ rõ ràng, có thể quản lý được cả khách hàng và cửa hàng.

- Phản hồi nhanh, thao tác mượt mà trên các thiết bị Android

## 6.2. Hạn chế của hệ thống

- Mới chỉ hoạt động trên nền tảng Android, chưa có phiên bản iOS.

- Chưa có chức năng định vị hoặc gợi ý quán ăn gần người dùng.

- Cần có hệ thống quản trị web riêng để chủ cửa hàng quản lý tốt hơn.

## 6.3. Hướng phát triển

- Phát triển phiên bản iOS để mở rộng đối tượng người dùng.

- Tích hợp thanh toán online (MoMo, ZaloPay, VNPay,...) giúp quá trình đặt hàng tiện lợi hơn.

- Thêm bản đồ định vị cửa hàng và gợi ý món ăn theo vị trí người dùng.

- Nâng cấp giao diện (UI/UX) hiện đại, sinh động và trực quan hơn.

- Xây dựng hệ thống quản trị giúp cửa hàng theo dõi doanh thu và phản hồi khách hàng.

- Ứng dụng có thể mở rộng thành nền tảng đặt đồ ăn toàn diện, liên kết nhiều quán khác nhau trong khu vực.

# **TÀI LIỆU THAM KHẢO**