

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN

=====***=====



BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG GIAO ĐÒ ĂN FOODLY TRÊN
NỀN TẢNG MOBILE**

Cán bộ hướng dẫn : **ThS. Nguyễn Xuân Hoàng**

Sinh viên thực hiện : **Phạm Văn Cường**

Mã số sinh viên : **2020600079**

Lớp : **CNTT01-K15**

Hà Nội – Năm 2024

MỤC LỤC

MỤC LỤC.....	i
LỜI MỞ ĐẦU	1
CHƯƠNG 1. TỔNG QUAN VỀ CÔNG NGHỆ VÀ CƠ SỞ LÝ THUYẾT	3
1.1. NodeJS	3
1.1.1. Giới thiệu về NodeJS.....	3
1.1.2. Lịch sử phát triển của NodeJS	4
1.1.3. Ưu điểm của NodeJS	5
1.1.4. Nhược điểm	5
1.1.5. Các ứng dụng và ví dụ thực tế.....	6
1.1.6. Sử dụng NodeJS trong dự án Foodly	6
1.2. MongoDB	7
1.2.1. Giới thiệu về MongoDB	7
1.2.2. Ưu điểm MongoDB	8
1.2.3. Nhược điểm MongoDB	8
1.2.4. Sử dụng MongoDB trong Foodly.....	9
1.3. Google Maps API.....	10
1.3.1. Giới thiệu về Google Map API	10
1.3.2. Lịch sử phát triển Google Map API	11
1.3.3. Ưu điểm của Google Maps API	11
1.4. React Native	12
1.4.1. Giới thiệu React Native	12
1.4.2. Lịch sử phát triển.....	13
1.4.3. Ưu điểm của React Native.....	13
1.4.4. Nhược điểm của React Native.....	14
1.4.5. Sử dụng React Native trong Foodly	14
1.5. Fire base	15
1.5.1. Firebase Authentication.....	16

1.6. Expo	17
1.6.1. Lợi ích của Expo	17
1.6.2. Nhược điểm	18
1.7. Bài toán đường đi ngắn nhất với Google Map.....	19
1.7.1. Áp dụng giải thuật Dijkstra	19
1.7.2. Áp dụng Google Maps API vào Foodly	21
CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG VÀ THIẾT KẾ CƠ SỞ DỮ LIỆU	25
2.1. Xác định các yêu cầu chính của hệ thống	25
2.1.1. Phân tích các yêu cầu	25
2.1.2. Các yêu cầu phi chức năng	25
2.2. Xây dựng biểu đồ use case.....	26
2.2.1. Các tác nhân	26
2.2.2. Các use case của hệ thống	26
2.2.3. Biểu đồ use case	27
2.2.4. Mô tả chi tiết các use case	28
2.2.5. Phân tích các use case	44
2.3. Xây dựng cơ sở dữ liệu	57
2.3.1. Biểu đồ Entity Relationship Diagram	57
2.3.2. Cấu trúc thành phần trong cơ sở dữ liệu	58
CHƯƠNG 3. KẾT QUẢ ĐẠT ĐƯỢC	62
3.1. Giao diện chương trình	62
3.1.1. Giao diện đăng nhập	62
3.1.2. Giao diện đăng ký	63
3.1.3. Giao diện trang chủ	64
3.1.4. Màn hình xem sản phẩm theo danh mục	65
3.1.5. Màn hình xem chi tiết sản phẩm	66
3.1.6. Màn hình tìm kiếm sản phẩm	67

3.1.7.	Màn hình giỏ hàng.....	68
3.1.8.	Màn hình thông tin nhà hàng.....	69
3.1.9.	Màn hình đánh giá	70
3.1.10.	Màn hình dẫn đường bởi Google Map	71
3.1.11.	Màn hình đơn hàng.....	72
3.1.12.	Giao diện profile	73
3.1.13.	Màn hình dashboard	74
3.1.14.	Màn hình profile	75
3.1.15.	Màn hình cài đặt	76
3.1.16.	Màn hình Authentication Admin Login	76
3.1.17.	Màn hình xóa sản phẩm.....	77
3.1.18.	Màn hình thêm mới sản phẩm	77
3.1.19.	Màn hình danh sách các danh mục.....	78
3.1.20.	Màn hình update danh mục	78
3.1.21.	Màn hình xóa danh mục	79
	TÀI LIỆU THAM KHẢO.....	82

LỜI MỞ ĐẦU

1. Lý do chọn đề tài

Hiện nay công nghệ thông tin không ngừng phát triển mạnh mẽ và hiện đại, các nền tảng số ngày càng tiếp cận tới người dùng dễ dàng hơn, trong đó có các hệ thống đặt đồ và giao đồ dần được phổ biến trên thị trường hiện nay với nhiều nền tảng công nghệ.

Bên cạnh đó, việc đặt hàng và giao hàng qua mạng giúp cửa hàng có thể tiếp cận được tệp khách hàng rộng lớn và đa dạng hơn. Hơn nữa, mỗi khách hàng chúng ta hiện nay ngày càng ưu chuộng việc tiện lợi, nhanh chóng và mất ít công sức hơn. Đối với thị trường Việt Nam, việc có các ứng dụng đặt và giao đồ ăn đang phổ biến hơn vì sức tiêu dùng về lĩnh vực thực phẩm cực kì cao đặc biệt là các sản phẩm thức ăn đã được hoàn thiện, chế biến sẵn và đóng gói chỉ đợi chuyển tới tay người tiêu dùng có nhu cầu.

Với mong muốn cung cấp và nâng cao kiến thức về lập trình, áp dụng kiến thức để xây dựng một ứng dụng giúp người dùng có thể dễ dàng sử dụng đặt đồ ăn, tham khảo, lựa chọn các thực phẩm mình muốn với mức giá chuẩn xác, sử dụng hệ thống với công nghệ mới nhanh và mượt mà hơn em đã chọn đề tài “Xây dựng ứng dụng giao đồ ăn Foodly trên nền tảng Mobile”.

Để hoàn thành được đồ án tốt nghiệp này, em xin được gửi lời cảm ơn chân thành đến các thầy cô trong khoa Công nghệ thông tin Trường Đại học Công Nghiệp Hà Nội, thầy giáo hướng dẫn đề tài –**Thạc sĩ Nguyễn Xuân Hoàng**– đã tận tụy hết lòng giúp đỡ, hướng dẫn, chỉ dẫn tận tình để giúp em hoàn thành đồ án. Bạn bè, các anh chị em Câu lạc bộ Tin Học HIT, giúp đỡ em trong suốt quá trình làm đồ án.

Hà Nội, Ngày 18 tháng 04 năm 2024

2. Mục tiêu của đề tài

Xây dựng được ứng dụng hoàn thiện gồm các module như quảng cáo nhà hàng , đồ ăn và tích hợp được bài toán đường đi ngắn nhất vào Google Map Api . Rèn luyện các kiến thức và kỹ năng phát triển hệ thống phần mềm, nghiên cứu thuật toán và thái độ làm việc chuyên nghiệp.

3. Nội dung nghiên cứu

- Khảo sát và phân tích yêu cầu quy trình đặt hàng, giao hàng.
- Sử dụng API Google Map để cung cấp bản đồ.
- Bài toán đường đi ngắn nhất với giải thuật Dijkstra
- Tọa độ GPS với kinh độ và vĩ độ
- Phân tích thiết kế hệ thống.
- Thiết kế cơ sở dữ liệu.
- Lập trình backend API.
- Lập trình giao diện và kết nối tới API.
- Có kế hoạch và đánh giá kiểm thử hệ thống.

4. Bố cục đề tài

Ngoài các phần Mở đầu, Kết luận và Tài liệu tham khảo, báo cáo đồ án được bô cục thành ba chương chính sau.

- Chương 1: Trình bày tổng quan về công nghệ và cơ sở lý thuyết.
- Chương 2: Phân tích thiết kế hệ thống và thiết kế cơ sở dữ liệu.
- Chương 3: Cài đặt phần mềm và kết quả đạt được.

CHƯƠNG 1. TỔNG QUAN VỀ CÔNG NGHỆ VÀ CƠ SỞ LÝ THUYẾT

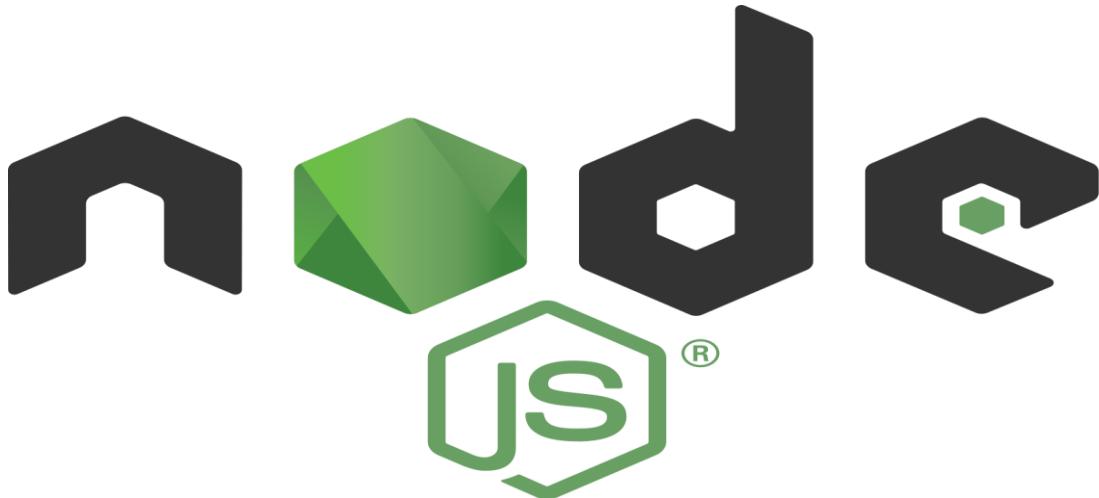
1.1. NodeJS

1.1.1. Giới thiệu về NodeJS

Node.js là một nền tảng phát triển ứng dụng phía máy chủ được xây dựng trên JavaScript runtime của Chrome V8 Engine. Nó cho phép các nhà phát triển sử dụng JavaScript để viết mã ở phía máy chủ thay vì chỉ ở phía trình duyệt web.

Một số tính năng quan trọng của NodeJS:

- **JavaScript Everywhere:** Node.js cho phép viết mã JavaScript không chỉ trên phía máy chủ mà còn trên phía máy khách (trình duyệt web). Điều này giúp giảm thiểu sự phân tách giữa các phần của ứng dụng web và tạo điều kiện thuận lợi cho việc chia sẻ mã giữa phía máy chủ và phía máy khách.
- **Non-blocking và Asynchronous I/O:** Node.js được thiết kế để xử lý hàng triệu kết nối đồng thời với cơ chế không chặn (non-blocking) và I/O bất đồng bộ (asynchronous I/O). Điều này giúp ứng dụng có thể xử lý các yêu cầu I/O mà không phải chờ đợi kết quả trả về, tăng hiệu suất và khả năng mở rộng.
- **Module Ecosystem:** Node.js có một cộng đồng mạnh mẽ và phong phú với hàng ngàn các module và thư viện có sẵn thông qua npm (Node Package Manager). Nhờ vào cộng đồng này, nhà phát triển có thể dễ dàng sử dụng và tái sử dụng mã nguồn mở để giảm thiểu thời gian phát triển và nâng cao hiệu suất của ứng dụng.
- **Scalability:** Tính năng này được sử dụng cho toàn bộ chức năng log trong phạm vi nội bộ và nó được quản lý mặc định.



Hình 1.1 Logo của NodeJS

1.1.2. Lịch sử phát triển của NodeJS

- Node.js được tạo ra bởi Ryan Dahl vào năm 2009. Ông muốn xây dựng một nền tảng cho phép viết mã JavaScript không chỉ trên trình duyệt web mà còn trên máy chủ.
- Node.js đã phát hành phiên bản đầu tiên vào tháng 5 năm 2009. Nó đã nhanh chóng thu hút sự chú ý của cộng đồng lập trình và trở thành một công cụ phát triển phổ biến.
- Năm 2011 Node.js được chuyển từ dự án cá nhân sang dự án mã nguồn mở, giúp thu hút sự tham gia của cộng đồng lập trình viên và tạo điều kiện cho sự phát triển nhanh chóng.
- Cùng năm 2011 npm (Node Package Manager) được giới thiệu, cung cấp cho các nhà phát triển một cách tiện lợi để quản lý các thư viện và phụ thuộc của dự án Node.js.
- Node.js trở nên ngày càng phổ biến và được sử dụng rộng rãi trong các ứng dụng web thương mại và doanh nghiệp. Sự ra đời của các framework như Express.js cũng giúp Node.js trở nên dễ dàng và linh hoạt hơn cho các nhà phát triển.

1.1.3. Ưu điểm của NodeJS

- **Hiệu Suất Cao:** Node.js được xây dựng dựa trên kiến trúc không chặn (non-blocking) và I/O bất đồng bộ (asynchronous I/O), giúp tối ưu hóa hiệu suất cho các ứng dụng yêu cầu xử lý I/O lớn và song song.
- **Thích ứng Với Thực Tế:** Node.js phù hợp với các ứng dụng yêu cầu xử lý thời gian thực và có thể mở rộng dễ dàng để đáp ứng với số lượng lớn người dùng. Giữ đầy đủ các tính năng của Spring Framework.
- **Module Ecosystem Phong Phú:** Node.js có một cộng đồng lớn và sôi nổi, cung cấp hàng ngàn các module và thư viện thông qua npm, giúp nhà phát triển dễ dàng sử dụng và tái sử dụng mã nguồn mở.
- **Linh Hoạt và Dễ Mở Rộng:** Node.js hỗ trợ việc xây dựng các ứng dụng mạng linh hoạt và có thể mở rộng dễ dàng bằng cách sử dụng các cơ chế như clustering hoặc load balancing.

1.1.4. Nhược điểm

- **Đơn luồng:** Mặc dù mô hình đơn luồng của Node.js giúp quản lý nhiều kết nối một cách hiệu quả, nó lại là một bất lợi khi xử lý các tác vụ CPU-bound. Những tác vụ này có thể làm chậm hoặc chặn luồng chính, ảnh hưởng đến hiệu suất tổng thể.
- **Hạn chế về hiệu năng trong các tác vụ CPU-bound:** Node.js hoạt động tốt với các tác vụ I/O-bound nhưng không hiệu quả với các tác vụ CPU-bound như mã hóa, giải mã, hoặc xử lý hình ảnh/video. Những tác vụ này có thể làm chậm hệ thống vì chúng chặn luồng đơn của Node.js.
- **Bảo trì mã phức tạp:** JavaScript, ngôn ngữ chính của Node.js, là một ngôn ngữ động và không có kiểu dữ liệu rõ ràng, điều này có thể làm cho mã nguồn trở nên khó bảo trì và dễ gặp lỗi, đặc biệt là trong các dự án lớn.
- **Callbacks và Promises:** Việc sử dụng callbacks để xử lý không đồng bộ có thể dẫn đến "callback hell" – một tình trạng mã trở nên phức tạp và khó đọc. Mặc dù Promises và async/await đã cải thiện vấn đề này, nhưng không phải mọi dự án đều sử dụng chúng hiệu quả.

1.1.5. Các ứng dụng và ví dụ thực tế

Node.js được sử dụng rộng rãi trong nhiều loại ứng dụng khác nhau nhờ vào khả năng xử lý không đồng bộ và hiệu suất cao.

- **Ứng dụng Web thời gian thực (Real-time Web Applications):** Một ví dụ điển hình là ứng dụng chat như WhatsApp Web hay Facebook Messenger. Node.js cho phép xử lý nhiều kết nối cùng lúc và cập nhật trạng thái chat theo thời gian thực.
- **Ứng dụng Streaming:** Netflix sử dụng Node.js cho nhiều phần của hệ thống của họ. Khả năng xử lý đồng thời của Node.js giúp tối ưu hóa việc truyền tải dữ liệu video đến người dùng.
- **Ứng dụng Microservices:** PayPal đã chuyển từ Java sang Node.js cho các ứng dụng của mình để tăng tốc độ phát triển và xử lý lượng lớn giao dịch.
- **Internet of Things (IoT):** Smart Home Systems sử dụng Node.js có thể được sử dụng để điều khiển các thiết bị IoT, thu thập và xử lý dữ liệu từ các cảm biến trong thời gian thực.
- **E-commerce Platforms:** Nhiều nền tảng thương mại điện tử như eBay đã sử dụng Node.js để cải thiện hiệu suất của trang web và ứng dụng của họ, đặc biệt là trong xử lý thanh toán và giỏ hàng.

1.1.6. Sử dụng NodeJS trong dự án Foodly

- Node.js được xây dựng dựa trên kiến trúc không chặn (non-blocking) và I/O bất đồng bộ (asynchronous I/O), giúp tối ưu hóa hiệu suất cho các ứng dụng yêu cầu xử lý I/O lớn và song song đặc biệt là trong xử lý thanh toán và giỏ hàng.
- Kho thư viện và module phong phú với hơn một triệu gói, giúp giảm bớt công sức phát triển và tái sử dụng mã nguồn.
- Sử dụng Node.js cho phần server của ứng dụng di động, giúp tăng tốc độ xử lý và cải thiện hiệu suất tổng thể.
- Node.js có thời gian khởi động nhanh, giúp dễ dàng triển khai và khởi động lại các ứng dụng.

- So với các công nghệ khác thì thống nhất ngôn ngữ lập trình giữa frontend và backend trong Foodly đều là nền tảng từ Javascript, thích hợp cho các ứng dụng cần xử lý nhiều kết nối đồng thời.
- Tốt cho các ứng dụng yêu cầu xử lý nhiều yêu cầu cùng lúc.
- Phù hợp cho các ứng dụng yêu cầu xử lý nhiều kết nối cùng lúc mà không bị chặn.

1.2. MongoDB

1.2.1. Giới thiệu về MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) mã nguồn mở, được thiết kế để lưu trữ và xử lý dữ liệu có cấu trúc linh hoạt, phân tán và có khả năng mở rộng.

MongoDB sử dụng mô hình cấu trúc dữ liệu linh hoạt gọi là BSON (Binary JSON), cho phép lưu trữ dữ liệu dưới dạng tài liệu (document) không có cấu trúc cố định. Điều này giúp dễ dàng thay đổi cấu trúc của dữ liệu mà không cần phải thay đổi schema.

MongoDB được thiết kế để mở rộng dễ dàng và linh hoạt, cho phép mở rộng theo cả chiều ngang (horizontal scaling) và chiều dọc (vertical scaling). Điều này giúp MongoDB phù hợp với các ứng dụng có yêu cầu về lưu trữ dữ liệu lớn và tốc độ truy vấn cao bao gồm các truy vấn phức tạp, truy vấn toàn văn (text search), và các truy vấn địa lý (geospatial queries), giúp cho việc tìm kiếm và truy xuất dữ liệu trở nên thuận tiện và hiệu quả.



Hình 1.2. Mongo DB

1.2.2. Ưu điểm MongoDB

- **Cấu trúc dữ liệu linh hoạt:** MongoDB sử dụng cấu trúc dữ liệu JSON dưới dạng BSON (Binary JSON), cho phép lưu trữ dữ liệu dưới dạng tài liệu linh hoạt và có thể thay đổi dễ dàng. Điều này rất hữu ích cho các ứng dụng có nhu cầu lưu trữ dữ liệu không cố định hoặc thay đổi thường xuyên.
- **Khả năng mở rộng dễ dàng:** MongoDB thiết kế để mở rộng mà không cần phải thay đổi cấu trúc dữ liệu hoặc ứng dụng. Bằng cách sử dụng các cụm (clusters) và các phân đoạn (shards), MongoDB có thể mở rộng từ các dự án nhỏ đến các ứng dụng với hàng triệu hoặc thậm chí hàng tỷ tài liệu.
- **Tích hợp tốt với ngôn ngữ lập trình:** MongoDB cung cấp các driver cho nhiều ngôn ngữ lập trình phổ biến như Python, JavaScript, Java, Ruby, và nhiều ngôn ngữ khác. Điều này làm cho việc phát triển ứng dụng với MongoDB trở nên dễ dàng và linh hoạt.
- **Tính toàn vẹn và độ tin cậy:** MongoDB hỗ trợ các tính năng như replica sets và failover, giúp đảm bảo tính toàn vẹn và độ tin cậy của dữ liệu trong trường hợp có sự cố xảy ra.
- **Hiệu suất cao:** MongoDB có thể được cấu hình để hoạt động với hiệu suất cao, đặc biệt là khi sử dụng các công nghệ như WiredTiger storage engine và các index phức tạp.
- **Hỗ trợ cho các tính năng nâng cao:** MongoDB hỗ trợ nhiều tính năng nâng cao như MapReduce, aggregation pipeline, và các công cụ tìm kiếm text, giúp phát triển các ứng dụng phức tạp một cách dễ dàng.

1.2.3. Nhược điểm MongoDB

- Đối với các ứng dụng yêu cầu giao dịch phức tạp và nhất quán cao, MongoDB có thể không phải là lựa chọn tốt nhất.
- MongoDB thường sử dụng nhiều bộ nhớ để đảm bảo hiệu suất cao. Điều này có thể dẫn đến việc tiêu tốn nhiều tài nguyên hệ thống hơn, đặc biệt khi xử lý các khối lượng dữ liệu lớn.
- Mặc dù việc không yêu cầu schema cứng nhắc mang lại sự linh hoạt, nhưng điều này cũng có thể dẫn đến các vấn đề về tính nhất quán và khó khăn trong

việc duy trì và kiểm tra dữ liệu. Nếu không được quản lý cẩn thận, dữ liệu có thể trở nên không đồng nhất và khó xử lý.

- MongoDB không hỗ trợ JOINs phức tạp như trong các cơ sở dữ liệu quan hệ. Mặc dù Aggregation Framework và các công cụ khác có thể giúp giải quyết một số trường hợp, nhưng việc thao tác dữ liệu liên kết phức tạp có thể trở nên khó khăn và không hiệu quả.
- Mặc dù sharding giúp MongoDB mở rộng theo chiều ngang, nhưng việc thiết lập và quản lý sharding có thể phức tạp. Các vấn đề về phân mảnh dữ liệu và tái cân bằng (rebalancing) có thể gây ra gián đoạn và yêu cầu quản trị cẩn thận.
- Mặc dù MongoDB cung cấp các tính năng bảo mật như xác thực và mã hóa, nhưng việc cấu hình bảo mật đúng cách có thể phức tạp và dễ dẫn đến lỗi nếu không được thực hiện cẩn thận. Ngoài ra, các phiên bản cũ của MongoDB từng bị chỉ trích vì các lỗ hổng bảo mật, mặc dù điều này đã được cải thiện trong các phiên bản gần đây.

1.2.4. Sử dụng MongoDB trong Foodly

- MongoDB lưu trữ dữ liệu dưới dạng các tài liệu JSON (hoặc BSON) giúp dữ liệu có cấu trúc linh hoạt hơn so với các bảng trong cơ sở dữ liệu quan hệ.
- Không cần phải định nghĩa trước cấu trúc bảng, cho phép dễ dàng thay đổi và mở rộng cấu trúc dữ liệu mà không gây gián đoạn.
- MongoDB có thể xử lý lượng lớn dữ liệu và các truy vấn phức tạp nhanh chóng nhờ vào việc lưu trữ dữ liệu dưới dạng tài liệu.
- Hỗ trợ sharding (phân mảnh dữ liệu), giúp dễ dàng mở rộng cơ sở dữ liệu theo chiều ngang bằng cách thêm nhiều máy chủ
- MongoDB có khả năng xử lý các hoạt động đọc và ghi với tốc độ cao, phù hợp cho các ứng dụng cần phản hồi ngay lập tức như các hệ thống giám sát, phân tích dữ liệu thời gian thực.
- MongoDB hỗ trợ replication, cho phép sao lưu dữ liệu và đảm bảo tính sẵn sàng của hệ thống ngay cả khi có sự cố xảy ra.
- Hỗ trợ nhiều loại chỉ mục (indexes) giúp tăng hiệu suất truy vấn.

- Hỗ trợ tốt các truy vấn không gian, rất hữu ích cho các ứng dụng liên quan đến định vị và bản đồ.
- MongoDB sử dụng JSON-like documents, dễ dàng tương thích với JavaScript và các ngôn ngữ lập trình phổ biến khác.
- Kết hợp tốt với Node.js, một trong những công nghệ phổ biến nhất hiện nay cho việc xây dựng các ứng dụng web và dịch vụ backend.

1.3. Google Maps API

1.3.1. Giới thiệu về Google Map API

Google Maps API là một bộ công cụ mạnh mẽ do Google cung cấp để phát triển các ứng dụng liên quan đến bản đồ, địa điểm và dịch vụ địa lý.

- **Bản đồ tương tác:** Google Maps API cho phép bạn nhúng bản đồ tương tác vào các ứng dụng web hoặc di động của mình. Bạn có thể hiển thị bản đồ, điểm đánh dấu, đường đi, và nhiều nội dung khác trực tiếp trên bản đồ.
- **Dữ liệu địa lý:** API cung cấp quyền truy cập vào dữ liệu địa lý của Google, bao gồm các địa điểm, địa chỉ, điểm quan trọng và dịch vụ như tìm địa điểm, xác định vị trí, và định vị địa lý.
- **Hỗ trợ đa nền tảng:** Google Maps API hỗ trợ phát triển ứng dụng trên nhiều nền tảng, bao gồm web, Android và iOS.



Hình 1.3 Google Map

1.3.2. Lịch sử phát triển Google Map API

Google Maps API đã trải qua một hành trình phát triển đáng kinh ngạc từ khi ra mắt vào những năm đầu của thế kỷ 21.

Google Maps API được giới thiệu lần đầu tiên vào tháng 6 năm 2005. Ban đầu, API này chỉ hỗ trợ một số tính năng cơ bản như hiển thị bản đồ và điểm đánh dấu.

Mở rộng tính năng (2006-2010): Trong giai đoạn này, Google liên tục cập nhật và mở rộng tính năng của Google Maps API. API được cải tiến để hỗ trợ thêm các tính năng như tìm địa điểm, hiển thị thông tin về vị trí, và tích hợp các dịch vụ như Street View.

Google Maps API for Business (2011): Google giới thiệu phiên bản dành cho doanh nghiệp của Google Maps API, cung cấp các tính năng và hỗ trợ cao cấp hơn cho các tổ chức và doanh nghiệp muốn tích hợp bản đồ vào sản phẩm của họ.

Tích hợp tính năng định vị GPS (2012): Google Maps API bổ sung tính năng định vị GPS, cho phép ứng dụng sử dụng thông tin vị trí của người dùng để cung cấp các dịch vụ địa lý dựa trên vị trí.

Google Maps Platform (2018): Google tiếp tục cải tiến và thay đổi cách họ cung cấp dịch vụ địa lý, đưa ra Google Maps Platform là một bộ sản phẩm mới bao gồm nhiều dịch vụ như Maps, Routes, Places và nhiều hơn nữa, thay thế cho các phiên bản API cũ.

1.3.3. Ưu điểm của Google Maps API

- **Dữ liệu địa lý phong phú:** Google Maps API cung cấp quyền truy cập vào một cơ sở dữ liệu địa lý lớn với hàng triệu địa điểm, địa chỉ và dịch vụ khác nhau trên khắp thế giới. Điều này giúp cho việc tạo ra các ứng dụng địa lý phong phú và đa dạng.
- **Tích hợp dễ dàng:** Google Maps API được thiết kế để tích hợp dễ dàng vào các ứng dụng web, di động và desktop. Có sẵn các thư viện và SDK cho nhiều ngôn ngữ lập trình, giúp nhà phát triển dễ dàng tích hợp và sử dụng API trong các dự án của họ.

- **Bản đồ tương tác:** API cung cấp các tính năng tương tác như phóng to, thu nhỏ, di chuyển bản đồ, và thậm chí xoay để cung cấp trải nghiệm người dùng tốt nhất. Người dùng có thể tương tác với bản đồ, điều này tạo ra một trải nghiệm người dùng tốt hơn.
- **Tính đa nền tảng:** Google Maps API hỗ trợ phát triển ứng dụng trên nhiều nền tảng, bao gồm web, Android và iOS, giúp các nhà phát triển xây dựng ứng dụng một cách linh hoạt và toàn diện.

1.4. React Native

1.4.1. Giới thiệu React Native

React Native là một framework phát triển ứng dụng di động mã nguồn mở được phát triển bởi Facebook. Nó cho phép nhà phát triển sử dụng JavaScript và React để xây dựng các ứng dụng di động đa nền tảng, bao gồm cả iOS và Android, từ một mã nguồn duy nhất.

Trong Native, giao diện được chia thành các phần nhỏ gọi là component và có thể dễ dàng tái sử dụng.

React Native giúp giao diện của ứng dụng Mobile có thể phản ứng nhanh nhẹn và mượt mà hơn.



Hình 1.1. React Native

1.4.2. Lịch sử phát triển

React Native được giới thiệu lần đầu tiên bởi Facebook vào tháng 3 năm 2015 tại F8, sự kiện hàng năm của họ. Facebook chia sẻ rằng họ đã sử dụng React Native để xây dựng một số ứng dụng di động của mình như Facebook và Instagram, và họ muốn chia sẻ công nghệ này với cộng đồng nhà phát triển.

React Native nhanh chóng thu hút sự chú ý của cộng đồng phát triển. Nhiều công ty và nhà phát triển bắt đầu sử dụng nó để phát triển ứng dụng di động đa nền tảng với hiệu suất cao và chi phí thấp.

1.4.3. Ưu điểm của React Native

- **Phát triển đa nền tảng:** React Native cho phép phát triển ứng dụng di động đa nền tảng từ một mã nguồn duy nhất. Điều này giúp giảm chi phí và thời gian phát triển, vì bạn chỉ cần viết mã một lần và có thể triển khai cho cả iOS và Android.
- **JavaScript và React:** Với React Native, nhà phát triển có thể sử dụng JavaScript, một ngôn ngữ lập trình phổ biến và dễ học, cùng với React, một

thư viện phổ biến cho việc xây dựng giao diện người dùng, để phát triển các ứng dụng di động một cách hiệu quả và linh hoạt.

- **Hiệu suất gần native:** React Native sử dụng các thành phần UI native và cơ chế giao tiếp thông qua bridge, giúp tạo ra ứng dụng với hiệu suất gần như native. Điều này đảm bảo rằng ứng dụng của bạn có thể chạy mượt mà và đáp ứng người dùng mong đợi.
- **Tính tùy chỉnh và linh hoạt:** React Native cho phép tùy chỉnh giao diện người dùng của ứng dụng bằng cách sử dụng các thành phần UI native, thư viện bên thứ ba hoặc thậm chí tạo ra các thành phần tùy chỉnh của riêng bạn.
- **Hot Reloading và Live Reload:** React Native hỗ trợ hot reloading và live reloading, giúp nhà phát triển xem các thay đổi ngay lập tức khi thực hiện mà không cần phải tải lại ứng dụng hoặc mô phỏng lại các trạng thái.

1.4.4. Nhược điểm của React Native

- **Độ trễ giao diện người dùng (UI):** Mặc dù React Native cung cấp một cơ chế khá hiệu quả để tạo giao diện người dùng (UI), nhưng có thể gặp phải độ trễ trong việc hiển thị UI so với ứng dụng native, đặc biệt là trong các ứng dụng có nhiều chức năng phức tạp hoặc animation.
- **Sự không ổn định trên các thiết bị cũ:** Một số lỗi và vấn đề liên quan đến hiệu suất có thể xuất hiện trên các thiết bị cũ hoặc không được hỗ trợ tốt bởi thư viện React Native.
- **Sự tương thích giữa các phiên bản React Native:** Việc nâng cấp hoặc thêm các tính năng mới vào ứng dụng React Native có thể gặp khó khăn do sự không tương thích giữa các phiên bản React Native và các thư viện và plugins bên thứ ba.
- **Tiêu tốn bộ nhớ:** Ứng dụng React Native có thể tiêu tốn nhiều bộ nhớ hơn so với ứng dụng Native tương tự, đặc biệt khi sử dụng nhiều thư viện bên thứ ba hoặc khi xử lý dữ liệu lớn.

1.4.5. Sử dụng React Native trong Foodly

- **Tái sử dụng thành phần (Component Reusability):** React Native cho phép tái sử dụng các thành phần đã phát triển, giúp tăng tốc độ phát triển và dễ dàng duy trì mã nguồn.

- **Chia sẻ logic code:** Không chỉ UI, mà cả logic ứng dụng cũng có thể được chia sẻ giữa các nền tảng.
- **Bridge to Native:** React Native sử dụng bridge để giao tiếp giữa mã JavaScript và các thành phần native, mang lại hiệu suất cao gần giống như các ứng dụng native, khác biệt so với các framework khác có hiệu suất thấp hơn.
- **Hot Reloading:** Tính năng hot reloading cho phép các nhà phát triển thấy ngay lập tức kết quả của các thay đổi mã mà không cần phải build lại ứng dụng, tăng tốc độ phát triển và giảm thời gian phát triển.
- **Developer Experience:** React Native cung cấp một trải nghiệm phát triển viên tốt nhờ vào các công cụ hỗ trợ phát triển mạnh mẽ như React Developer Tools, Expo, và nhiều IDE plugin.
- **Component-Based:** Mô hình phát triển dựa trên component giúp dễ dàng xây dựng các giao diện người dùng hiện đại, tái sử dụng các thành phần và quản lý trạng thái hiệu quả.
- **Tiết kiệm tài nguyên:** Phát triển ứng dụng với React Native thường ít tốn kém hơn so với việc phát triển riêng biệt cho iOS và Android, nhờ vào việc tái sử dụng mã nguồn và giảm số lượng phát triển viên cần thiết.
- **Native Modules:** Khi cần hiệu suất tối ưu hoặc các tính năng không được hỗ trợ trực tiếp bởi React Native, các nhà phát triển có thể viết các module native bằng Java, Swift, hoặc Objective-C và tích hợp chúng dễ dàng với mã JavaScript.

1.5. Fire base

Firestore là cơ sở dữ liệu NoSQL thẻ hệ tiếp theo của Firebase, hỗ trợ truy vấn linh hoạt hơn và khả năng mở rộng tốt hơn so với Realtime Database.

Quản lý dữ liệu phức tạp với các truy vấn nâng cao, tích hợp với các dịch vụ Google Cloud khác.



Hình 1.5.1 Fire Base

1.5.1. Firebase Authentication

- Firebase Authentication cung cấp các phương thức xác thực người dùng như email/password, Google, Facebook, Twitter, v.v.
- Đăng nhập/đăng ký người dùng, quản lý phiên người dùng.

```
const jwt = require('jsonwebtoken')
const admin = require('firebase-admin')

module.exports = {
  createUser: async (req, res) => {
    const user = req.body;

    try {
      await admin.auth().getUserByEmail(user.email);

      res.status(400).json({ message: "Email already registered" })
    } catch (error) {
      if (error.code === 'auth/user-not-found') {
        try {
          const userResponse = await admin.auth().createUser({
            email: user.email,
            password: user.password,
            emailVerified: false,
            disabled: false
          })

          const newUser = new User({
            username: user.username,
            email: user.email,
            password: CryptoJS.AES.encrypt(user.password, process.env.SECRET).toString(),
            uid: userResponse.uid,
            userType: 'Client'
          })

          await newUser.save()

          res.status(201).json({ status: true })
        } catch (error) {
          res.status(500).json({ status: false, error: "Error creating user" })
        }
      }
    }
  }
}
```

Identifier	Providers	Created	Signed In	User UID
admin@gmail.com	✉️	May 5, 2024		RKV2oFXjsMs4Kx7wWrLix8L...
vancuong442002@gmail.com	✉️	May 4, 2024		nHalVBM6BuWkj4BaKPm85z...
cuongcter442002@gmail.com	✉️	May 3, 2024		3xcTMxBg9APALvMac7Rl1qc...

Hình 1.5.2 Sử dụng Fire Base Authentication

1.6. Expo

Expo cung cấp một cách nhanh chóng và dễ dàng để phát triển ứng dụng di động với React Native, giúp các nhà phát triển tập trung vào việc xây dựng chức năng và trải nghiệm người dùng mà không cần lo lắng về các chi tiết kỹ thuật phức tạp.

Hình 1.6.1 Expo

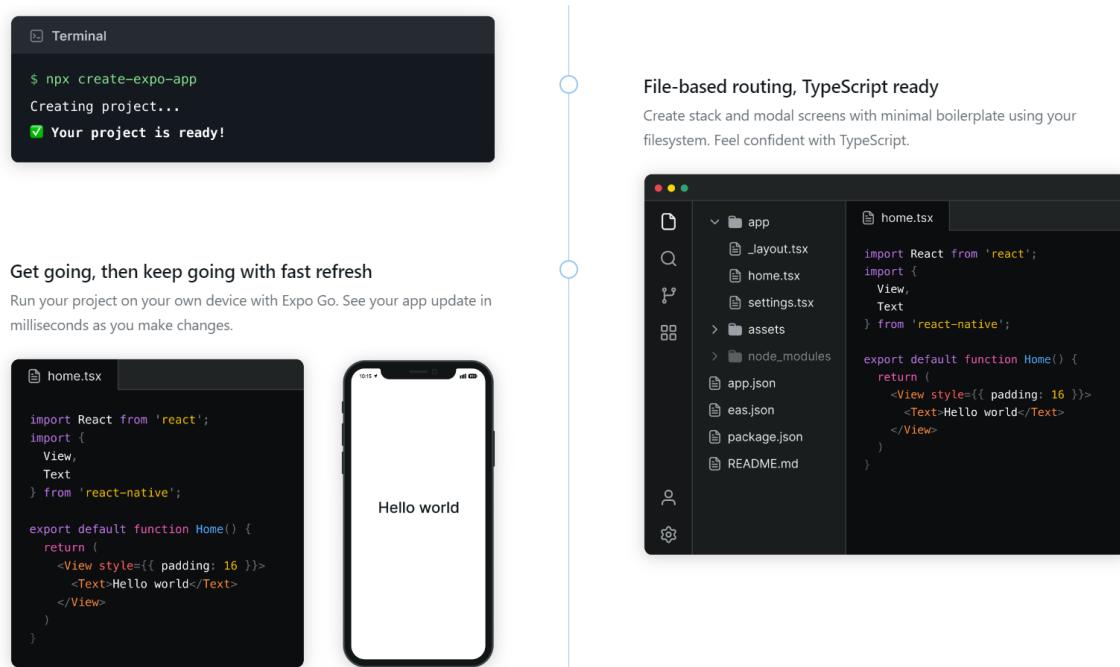
1.6.1. Lợi ích của Expo

- Thiết lập dễ dàng:** Bạn có thể nhanh chóng khởi tạo một dự án mới và bắt đầu phát triển mà không cần cấu hình phức tạp.
- Truy cập nhanh vào các tính năng di động:** Expo cung cấp sẵn các API để truy cập vào các tính năng của thiết bị như camera, địa điểm, thông báo đẩy, v.v.

- **Phát triển đa nền tảng:** Với Expo, bạn có thể phát triển ứng dụng cho cả iOS và Android từ cùng một mã nguồn.
- **Xem trước ứng dụng ngay lập tức:** Với Expo Go, bạn có thể xem trước ứng dụng của mình trên thiết bị di động mà không cần phải xây dựng và cài đặt.

1.6.2. Nhược điểm

- **Giới hạn của Managed Workflow:** Mặc dù Expo cung cấp rất nhiều tính năng, nhưng một số tính năng nâng cao hoặc tùy chỉnh sâu có thể không được hỗ trợ trong Managed Workflow.
- **Dung lượng ứng dụng lớn hơn:** Ứng dụng xây dựng bằng Expo thường có kích thước lớn hơn vì bao gồm nhiều thư viện mặc định của Expo.
- **Độ trễ cập nhật:** Một số tính năng mới của React Native hoặc các thư viện bên ngoài có thể không ngay lập tức được hỗ trợ trong Expo.





Hình 1.6.1 Sử dụng Expo trong dự án

1.7. Bài toán đường đi ngắn nhất với Google Map

1.7.1. Áp dụng giải thuật Dijkstra

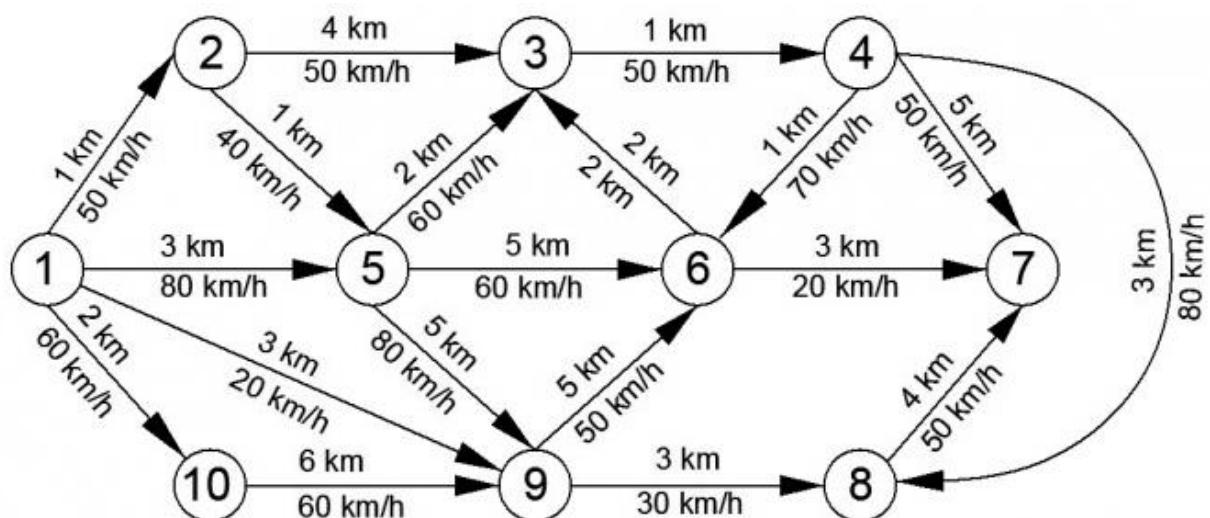
- Thuật toán Dijkstra có thể giải quyết bài toán tìm đường đi ngắn nhất trên đồ thị vô hướng lẫn có hướng miễn là trọng số không âm.

- **Khởi tạo:** Gán một giá trị khoảng cách ban đầu cho tất cả các đỉnh trong đồ thị. Đỉnh bắt đầu có giá trị khoảng cách là 0, còn lại là vô cùng. Gán một tập hợp trống để lưu trữ các đỉnh đã được xác định khoảng cách ngắn nhất.
- **Chọn đỉnh xuất phát:** Chọn một đỉnh xuất phát và đặt khoảng cách từ đỉnh này đến chính nó là 0.
- **Lặp lại cho đến khi tất cả các đỉnh được xác định:** Lặp lại các bước sau
 - Chọn đỉnh x với khoảng cách nhỏ nhất trong tập hợp chưa được xác định.
 - Đánh dấu đỉnh x này là đã xác định.
 - Cập nhật khoảng cách của tất cả các đỉnh kề với đỉnh x. Nếu khoảng cách mới ngắn hơn khoảng cách hiện tại, cập nhật khoảng cách.

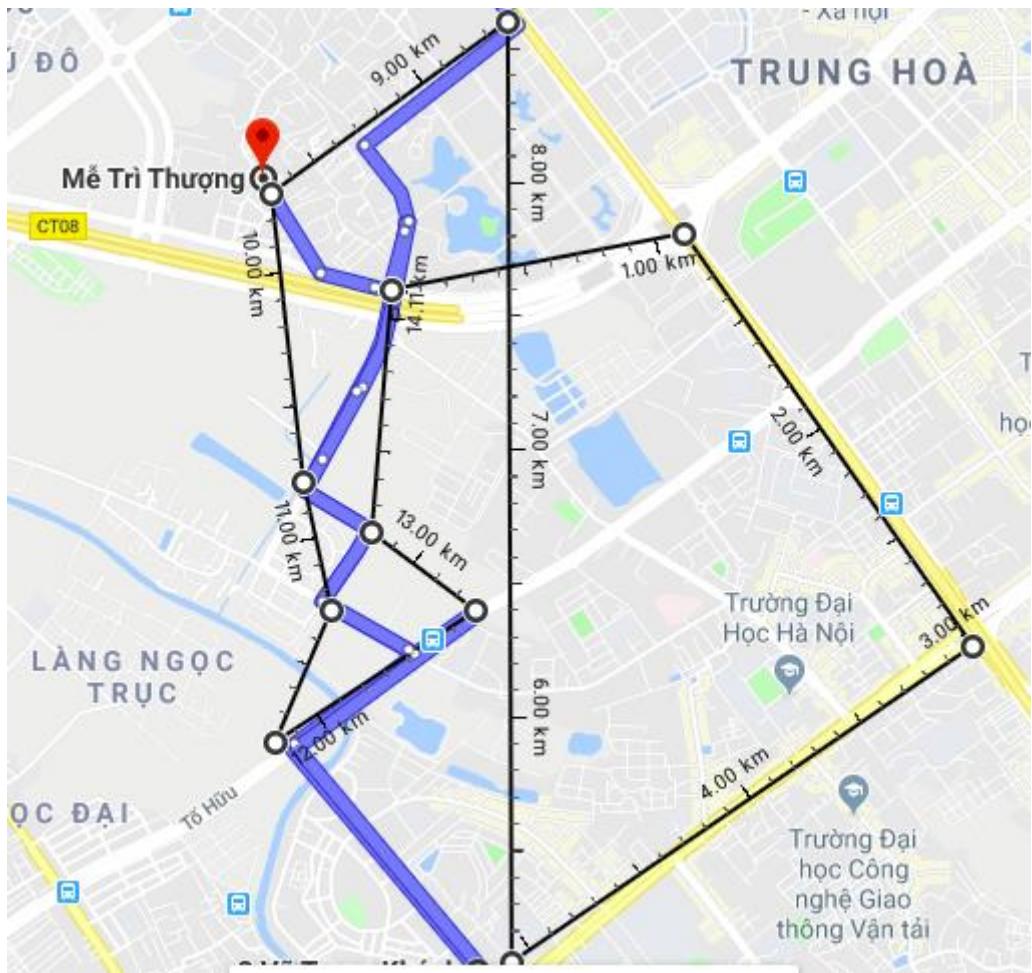
- **Kết thúc:** Khi tất cả các đỉnh đã được xác định khoảng cách ngắn nhất từ đỉnh xuất phát, thuật toán kết thúc.

Thuật toán Dijkstra đảm bảo rằng mỗi đỉnh chỉ được xác định khoảng cách ngắn nhất một lần, và sau khi một đỉnh được xác định, khoảng cách ngắn nhất từ đỉnh xuất phát đến đỉnh đó là khoảng cách ngắn nhất thực sự.

Một điều quan trọng cần lưu ý là thuật toán Dijkstra chỉ hoạt động với đồ thị có trọng số không âm. Điều này đảm bảo rằng khi một đỉnh được xác định khoảng cách ngắn nhất, thì khoảng cách đó không thể được cải thiện.



Hình 1.7.1 Dijkstra



Hình 1.7.2 Dijkstra với Google Map

1.7.2. Áp dụng Google Maps API vào Foodly

Tạo Project và lấy API Key:

- Truy cập Google Cloud Console.
- Tạo một project mới và kích hoạt Google Maps JavaScript API.
- Tạo một API key để sử dụng trong ứng dụng của bạn.

The screenshot shows the Google Cloud Console interface for managing credentials. The left sidebar has a 'Navigation menu' with 'Services' selected. Under 'Services', 'Enabled APIs & services' is expanded. The main content area is titled 'Credentials' and includes sections for 'API Keys', 'OAuth 2.0 Client IDs', and 'Service Accounts'. In the 'API Keys' section, there are two entries: 'Clé API 2' and 'Maps API Key', both created on Mar 15, 2024, with 'None' restrictions. A button 'CONFIGURE CONSENT SCREEN' is visible.

Hình 1.7.3. Google Cloud Console.

This screenshot shows a modal dialog for a 'Maps API Key'. The title is 'Maps API Key'. It contains a text input field labeled 'Your API key' with the value 'AIzaSyBcLEWrr020gkXN6uz_WYzMz21Fg0oXU8'. Below the input field is a warning message: '⚠ This key is unrestricted. To prevent unauthorized use, we recommend restricting where and for which APIs it can be used. [Edit API key](#) to add restrictions. [Learn more](#)'.

Hình 1.7.4. Map API Key

```
const apiKey = "AIzaSyC9RowylChjTPyyYtzol-9Djgz09WzfH9g"; // Replace with your API Key

const calculateDistanceAndTime = async (startLat, startLng, destinationLat, destinationLng, mode = 'bicycling') => {
    const baseUrl = "https://maps.googleapis.com/maps/api/distancematrix/json?";
    const ratePerKm = 1;

    const requestUrl = `${baseUrl}origins=${startLat},${startLng}&destinations=${destinationLat},${destinationLng}&mode=${mode}&key=${apiKey}`;

    try {
        const response = await fetch(requestUrl);
        const data = await response.json();
    }
}
```

Hình 1.7.5. Tích hợp vào mã nguồn

Nhúng API vào ứng dụng Foodly sử dụng để tìm kiếm địa điểm, tự động hoàn thành địa chỉ, và thông tin địa điểm chi tiết.

```

39  const fetchDirections = async (
40    startLat,
41    startLng,
42    destinationLat,
43    destinationLng
44  ) => {
45    try {
46      const url = `https://maps.googleapis.com/maps/api/directions/json?origin=${startLat},${startLng}&destination=${destinationLat},${destinationLng}&key=${apiKey}`;
47      const response = await fetch(url);
48      console.log(response);
49      const data = await response.json().then((data) => {
50        setDirections(data);
51        const encodedPolyline = data.routes[0].overview_polyline.points;
52        const coordinates = decode(encodedPolyline);
53        setCoordinates(coordinates);
54      });
55    } catch (error) {
56      console.error(error);
57    }
58  };
59
60  const decode = (encoded) => {
61    const points = [];
62    let index = 0,
63      len = encoded.length;
64    let lat = 0,
65    lng = 0;
66
67    while (index < len) {
68      let shift = 0,
69        result = 0;
70      let byte;
71      do {
72        byte = encoded.charCodeAt(index++) - 63;
73        result |= (byte & 0x1f) << shift;
74        shift += 5;

```



```

60  const decode = (encoded) => {
75    } while (byte >= 0x20);
76    const deltaLat = result & 1 ? ~(result >> 1) : result >> 1;
77    lat += deltaLat;
78
79    shift = 0;
80    result = 0;
81    do {
82      byte = encoded.charCodeAt(index++) - 63;
83      result |= (byte & 0x1f) << shift;
84      shift += 5;
85    } while (byte >= 0x20);
86    const deltaLng = result & 1 ? ~(result >> 1) : result >> 1;
87    lng += deltaLng;
88
89    points.push({ latitude: lat / 1e5, longitude: lng / 1e5 });
90  }
91
92  return points;
93};
94
95  return (
96    <View style={styles.mapContainer}>
97      <MapView
98        style={styles.map}
99        provider={PROVIDER_GOOGLE}
100       showsUserLocation={true}
101       region={mapRegion}
102     >
103       <Marker title="My Location" coordinate={mapRegion} />
104
105       {placeList.map(
106         (item, index) => index <= 1 && <PlaceMarker coordinates={item} />
107       )}
108
109       <Polyline coordinates={coordinates} strokeColor={COLORS.primary} strokeWidth={5}/>
110     </MapView>
111   </View>

```

```

const Directions = () => {
  const { restaurantObj, setRestaurantObj } = useContext(RestaurantContext);
  const coords = restaurantObj.coords;

  const onDirectionClick = () => {
    const url = Platform.select({
      ios: "maps:" + coords.latitude + "," + coords.longitude,
      android: "geo:" + coords.latitude + "," + coords.longitude + "?z=16",
    })
    Linking.openURL(url);
  }

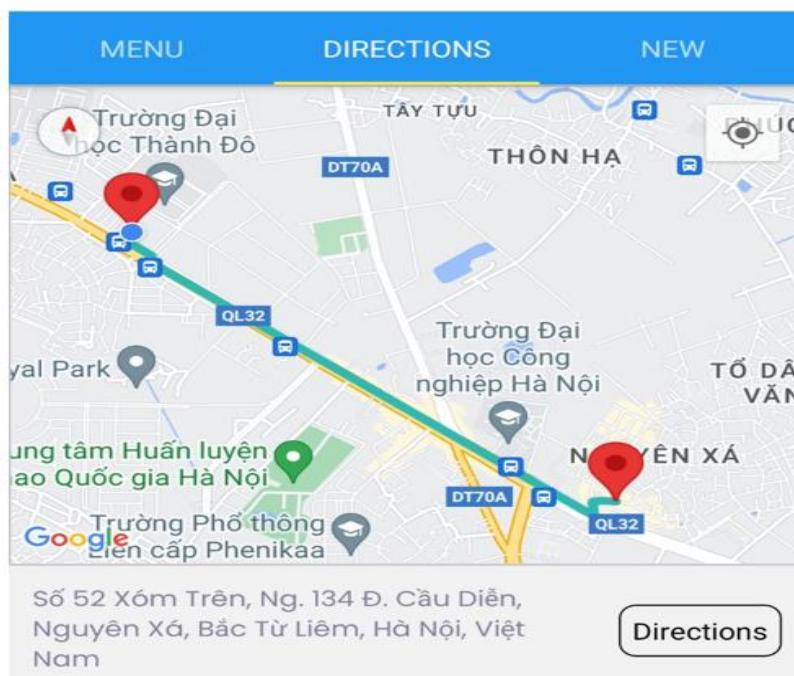
  return (
    <View>
      <GoogleMapView placeList={[coords]} />

      <View
        style={{{
          flexDirection: "row",
          justifyContent: "space-between",
          alignItems: "center",
          margin: 12,
        }}>
        <Text style={[styles.small, { width: SIZES.width / 1.6 }]}>
          {coords.address}
        </Text>

        <TouchableOpacity style={styles.ratingBtn} onPress={()=> onDirectionClick()}>
          <Text>Directions</Text>
        </TouchableOpacity>
      </View>
    </View>
  );
};

export default Directions;

```



Hình 1.5.5 Nhúng Google Map vào ứng dụng

CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HỆ THỐNG VÀ THIẾT KẾ CƠ SỞ DỮ LIỆU

2.1. Xác định các yêu cầu chính của hệ thống

2.1.1. Phân tích các yêu cầu

Những quyền lợi đối với khách hàng

- Xem trang tĩnh: Khách hàng có quyền xem nội dung trang tĩnh trên ứng dụng trên điện thoại.
- Đăng nhập: Khách hàng có thể đăng nhập hệ thống.
- Đăng ký: Khách hàng đăng ký vào hệ thống.
- Xem sản phẩm: Khách hàng có quyền xem danh sách sản phẩm và chi tiết sản phẩm.
- Thêm vào giỏ hàng: Khách hàng có thể thêm bất kỳ sản phẩm nào vào giỏ hàng.
- Đặt hàng: Khách hàng có quyền đặt hàng trên ứng dụng
- Tra cứu đơn hàng: Khách hàng xem được những đơn hàng mà mình đã đặt trước đó.
- Quản lý giỏ hàng: Khách hàng có quyền xem, thêm, sửa, xoá các sản phẩm trong giỏ hàng.

Những quyền lợi đối với quản trị viên

- Đăng nhập: Quản trị viên cần đăng nhập vào trang quản lý để thực hiện các chức năng quản trị hệ thống.
- Bảo trì danh mục: xem, thêm, sửa, xoá danh mục.
- Bảo trì sản phẩm: xem, thêm, sửa, xoá sản phẩm.
- Quản lý người dùng: xem, cập nhật quyền, cập nhật trạng thái tài khoản.
- Quản lý đơn hàng: xem, cập nhật trạng thái đơn hàng.

2.1.2. Các yêu cầu phi chức năng

- Hiệu năng cao, trải nghiệm mượt mà, ổn định.

- Giao diện thân thiện.
- Tích hợp Google Map
- Đáp ứng được lượng lớn truy cập đồng thời.
- Dễ dàng bảo trì và mở rộng.

2.2. Xây dựng biểu đồ use case

2.2.1. Các tác nhân

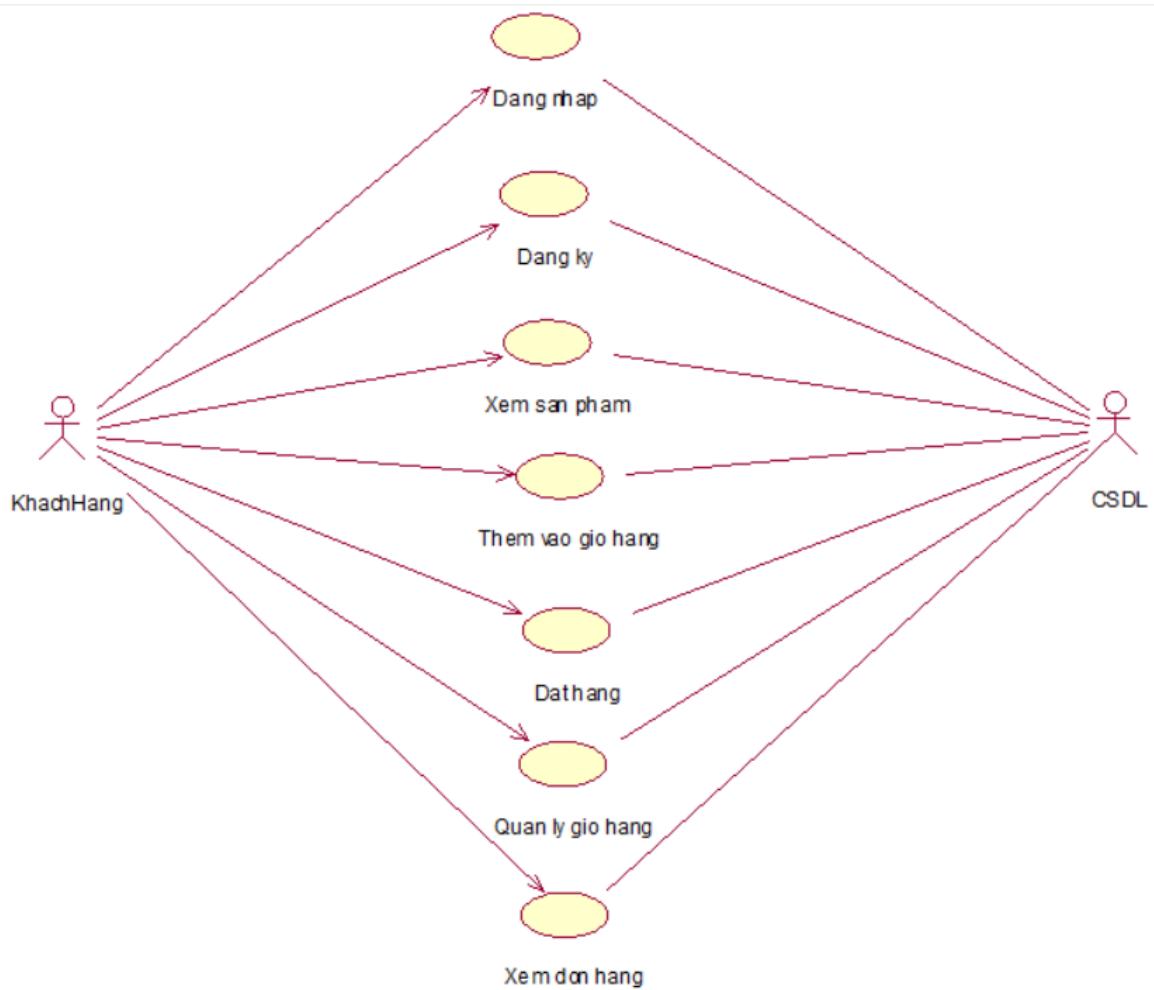
- Người quản trị: Quản lý nội dung app mobile, xử lý các yêu cầu từ khách hàng yêu cầu.
- Khách hàng: Tìm kiếm, xem thông tin sản phẩm, đặt hàng.

2.2.2. Các use case của hệ thống

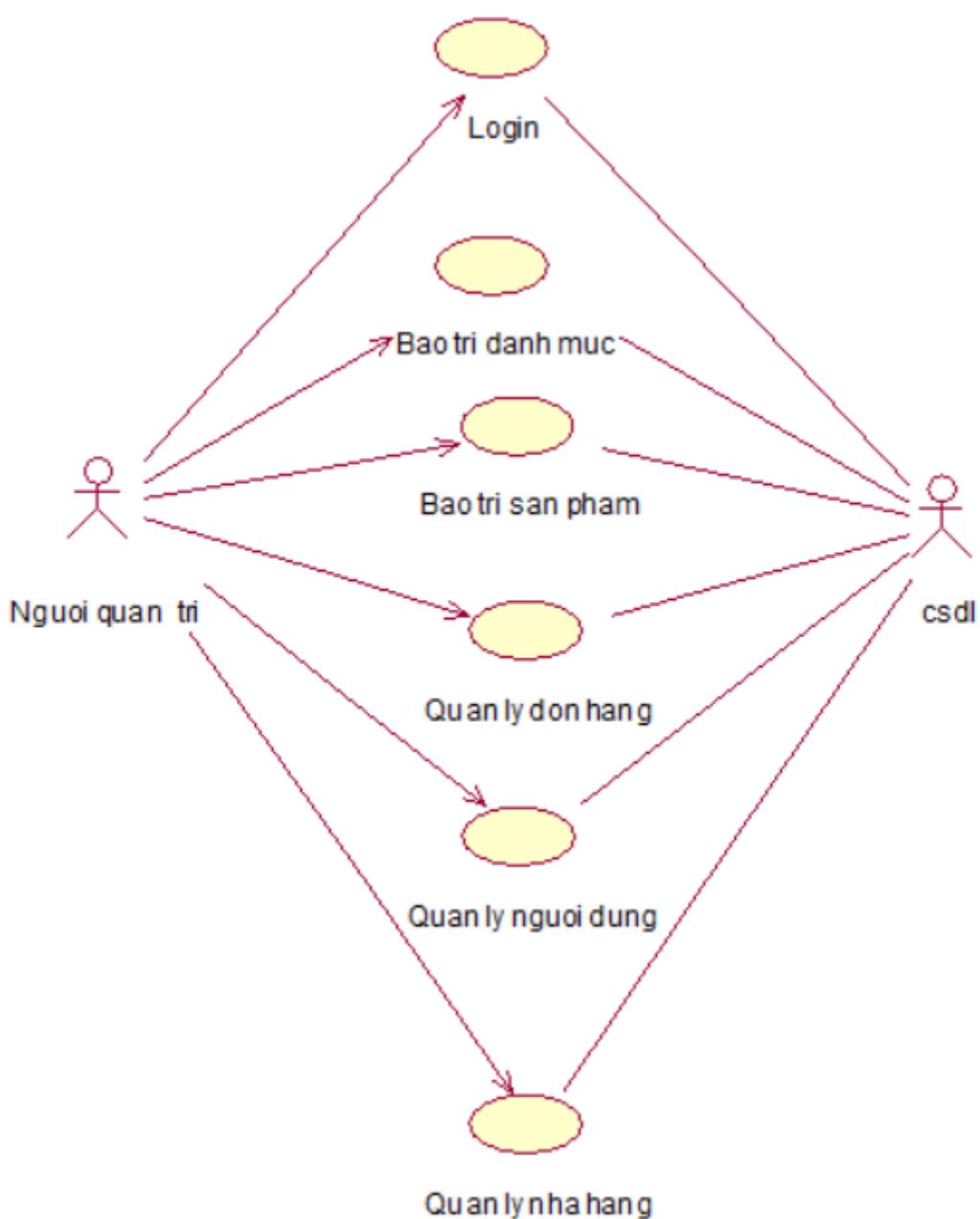
Bảng 2.1. Các use case hệ thống

Tác nhân	Các case sử dụng
Khách hàng	Đăng nhập Đăng ký Xem sản phẩm Thêm vào giỏ hàng Đặt hàng Quản lý giỏ hàng Tra cứu đơn hàng
Người quản trị	Bảo trì danh mục Bảo trì sản phẩm Quản lý đơn hàng Quản lý người dùng Quản lý nhà hàng

2.2.3. Biểu đồ use case



Hình 2.2. Các use case chính



Hình 2.3. Các use case thứ cấp

2.2.4. Mô tả chi tiết các use case

2.2.4.1. Use case Đăng nhập

- **Mô tả văn tắt:** Use case này cho phép khách hàng đăng nhập vào hệ thống.

- **Luồng sự kiện:**

- **Luồng cơ bản:**

1. Use case này bắt đầu khi khách hàng nhấn vào nút đăng nhập. Hệ thống sẽ yêu cầu khách hàng nhập email và mật khẩu.
2. Khách hàng nhập email và mật khẩu sau đó nhấn vào nút “Login”. Hệ thống sẽ kiểm tra email, mật khẩu của khách hàng trong bảng USERS, nếu có hệ thống sẽ thông báo đăng nhập thành công và chuyển hướng đến trang chủ.

Use case kết thúc.

- **Luồng rẽ nhánh**

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 2 trong luồng cơ bản, khi người dùng nhập thiếu thông tin hoặc thông tin không hợp lệ hoặc không tìm thấy bản ghi trong bảng USERS. Hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại. Người dùng có thể nhập lại để tiếp tục hoặc kết thúc thao tác.

Các yêu cầu đặc biệt: Khách hàng cần nhập đúng thông tin để yêu cầu đăng nhập.

- **Các yêu cầu đặc biệt:**

Không có.

- **Tiền điều kiện:**

Không có.

- **Hậu điều kiện:**

Không có.

- **Điểm mở rộng:**

Không có.

2.2.4.2. Use case Đăng ký

- **Mô tả văn tắt:** Use case này cho phép người dùng đăng ký tài khoản trên hệ thống.
- **Luồng sự kiện:**

- **Luồng cơ bản:**

1. Use case này bắt đầu khi người dùng kích chuột vào icon user ở ứng dụng và chọn nút “Register” ở trang Login. Hệ thống điều hướng sang màn hình chứa form “Sign Up”. Hệ thống yêu cầu nhập thông tin đăng ký: name, email, password.
2. Khách hàng nhập username, email và password sau đó nhấn vào nút “Sign Up”. Hệ thống sẽ kiểm tra email, password của khách hàng trong bảng USERS. Sau khi người dùng xác thực tài khoản thành công ứng dụng sẽ điều hướng vào Profile của ứng dụng cá nhân.

Use case kết thúc.

- **Luồng rẽ nhánh**

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 2 trong luồng cơ bản, khi người dùng nhập thiếu thông tin, thông tin không hợp lệ hoặc email đã được đăng ký trong bảng USERS. Hệ thống sẽ hiển thị thông báo lỗi và yêu cầu nhập lại. Người dùng có thể nhập lại để tiếp tục hoặc kết thúc thao tác.

Các yêu cầu đặc biệt: Khách hàng cần nhập đúng thông tin để yêu cầu đăng nhập.

- **Các yêu cầu đặc biệt:**

Không có.

- **Tiền điều kiện:**

Không có.

- **Hậu điều kiện:**

Không có.

- **Điểm mở rộng:**

Không có.

2.2.4.3. Use case Thêm vào giỏ hàng

- **Mô tả ngắn:** Use case này cho phép khách hàng thêm bất kỳ sản phẩm nào vào giỏ hàng

- **Luồng sự kiện**

- **Luồng cơ bản:**

1. Người dùng nhấn vào icon dấu “+” trong chi tiết sản phẩm sau khi chọn số lượng cần đặt.
2. Hệ thống sẽ lấy ra thông tin các sản phẩm người dùng đã thêm vào giỏ hàng của mình từ bảng “FOODS”, nếu sản phẩm muốn thêm đã tồn tại trong giỏ hàng hệ thống sẽ cộng số lượng sản phẩm muốn thêm vào sản phẩm trong giỏ hàng, nếu sản phẩm muốn thêm chưa tồn tại hệ thống sẽ tạo mới một bản ghi và lưu vào bảng “CARTS”

- **Luồng rẽ nhánh**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi.
2. Tại bước 1 trong luồng cơ bản, nếu người dùng chưa chọn số lượng sản phẩm muốn thêm vào giỏ hàng. Hệ thống sẽ hiển thị thông báo lỗi và yêu cầu đăng nhập. Người dùng có thể nhập lại để tiếp tục hoặc kết thúc thao tác.

Use case kết thúc.

- **Các yêu cầu đặc biệt**

Không có

- **Tiền điều kiện**

Khách hàng cần đăng nhập trước khi thực hiện

- **Hậu điều kiện**

Không có

- **Điểm mở rộng:**

Không có

2.2.4.4. Use case *Đặt hàng*

- **Mô tả ngắn tắt:** Use case này cho phép khách hàng đặt những sản phẩm trên ứng dụng.

- **Luồng sự kiện:**

- **Luồng cơ bản:**

1. Tại màn hình chi tiết sản phẩm, người dùng bấm vào nút “Order”, hệ thống sẽ chuyển sang màn hình order.
2. Người dùng nhấn nút “Create Order” hệ thống sẽ thực hiện tạo thêm bản ghi mới vào bảng ORDERS và sau đó hiển thị “Đặt hàng thành công”.

Use case kết thúc.

- **Luồng rẽ nhánh**

1. Tại bất kỳ thời điểm nào trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống hiển thị thông báo lỗi.

- **Các yêu cầu đặc biệt**

Không có

- **Tiền điều kiện**

Khách hàng cần đăng nhập trước khi đặt hàng

- **Hậu điều kiện**

Khi use case kết thúc thành công, các thông tin sẽ được lưu trữ vào bảng ORDERS

- **Điểm mở rộng:**

Không có

2.2.4.5. Use case Xem đơn hàng

- **Mô tả ngắn tắt:** Use case này cho phép người dùng xem được những đơn hàng mà mình đã đặt trước đó.
 - **Luồng sự kiện:**
 - **Luồng cơ bản:**
 1. Use case này bắt đầu khi người dùng nhấp vào nút “Orders” trên thanh công cụ của người dùng. Hệ thống sẽ thực hiện lấy thông tin về các đơn hàng của người dùng đang đăng nhập từ bảng ORDERS sau đó hiển thị lên màn hình.
 2. Người dùng nhấp vào đơn hàng, hệ thống sẽ hiển thị thông tin chi tiết của đơn hàng đó bao gồm thông tin nhà hàng, danh sách sản phẩm trong đơn hàng từ bảng ORDERS
 - **Luồng rẽ nhánh**
 1. Tại thời điểm bất kỳ trong quá trình thực hiện use case, nếu không kết nối được tới cơ sở dữ liệu thì sẽ hiển thị thông báo lỗi.
 2. Tại bước 1 trong luồng rẽ nhánh, nếu người dùng chưa đăng nhập hệ thống sẽ chuyển hướng sang trang đăng nhập.
- Use case kết thúc.

- **Các yêu cầu đặc biệt**

Không có

- **Tiền điều kiện**

Người dùng cần đăng nhập hệ thống

- **Hậu điều kiện**

Không có

- **Điểm mở rộng:**

Không có

2.2.4.6. Use case Quản lý giỏ hàng

- **Mô tả ngắn tắt:** Use case này cho phép khách hàng xem, thêm, sửa, xoá sản phẩm ra khỏi giỏ hàng.
- **Luồng sự kiện**
 - **Luồng cơ bản:**
 1. Use case này bắt đầu khi người dùng kích vào icon “Cart” trên thanh công cụ. Hệ thống sẽ lấy thông tin chi tiết giỏ hàng của người dùng đó gồm: mã giỏ hàng từ bảng “CART” và danh sách các sản phẩm thuộc giỏ hàng này từ bảng “FOODS”, hiển thị danh sách các sản phẩm lên màn hình.
 2. Xoá sản phẩm trong giỏ hàng: Người dùng nhấn vào biểu tượng xoá trên màn hình, hệ thống sẽ thực hiện loại bỏ sản phẩm đó ra khỏi giỏ hàng.
 - **Luồng rẽ nhánh**
 1. Tại thời điểm bất kỳ trong quá trình thực hiện use case, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị thông báo lỗi.
 2. Tại bước 1 trong luồng cơ bản, nếu chưa có sản phẩm nào trong giỏ hàng sẽ hiển thị “Giỏ hàng trống”.

Use case kết thúc.
- **Các yêu cầu đặc biệt**
Không có
- **Tiền điều kiện**
Khách hàng cần đăng nhập vào hệ thống
- **Hậu điều kiện**
Không có
- **Điểm mở rộng:**
Không có

2.2.4.7. Use case Bảo trì danh mục

- **Mô tả văn tắt:** Use case này cho phép quản trị viên xem, thêm, sửa, xoá danh mục.
- **Luồng sự kiện**
 - **Luồng cơ bản:**
 1. Use case này bắt đầu khi người dùng nhấn vào nút “Category” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng CATEGORIES gồm các thông tin: id, tên, ảnh và hiển thị lên màn hình.
 2. Thêm danh mục mới:
 - a. Người quản trị nhấn vào nút “Thêm” trong màn hình danh sách danh mục. Hệ thống sẽ hiển thị modal yêu cầu nhập thông tin cho danh mục gồm tên danh mục và mô tả cho danh mục.
 - b. Người quản trị nhập thông tin cho danh mục mới và nhấn vào nút “Xác nhận”. Hệ thống sẽ thêm một bản ghi mới vào bảng CATEGORIES sau đó lấy thông tin về danh sách danh mục mới gồm id, title, image và hiển thị lên màn hình.
 3. Sửa danh mục:
 - a. Người quản trị nhấn vào biểu tượng chỉnh sửa trên một dòng danh mục. Hệ thống sẽ lấy title, image của danh mục được chọn từ bảng CATEGORIES.
 - b. Người quản trị sẽ sửa tên, mô tả của danh mục sau đó nhấn vào nút “Cập nhật”. Hệ thống cập nhật thông tin danh mục vào bảng CATEGORIES sau đó lấy thông tin danh sách danh mục gồm id, title, image và hiển thị lên màn hình.
 4. Xoá danh mục:
 - a. Người quản trị nhấn vào biểu tượng xoá trên một dòng danh mục. Hệ thống sẽ hiển thị lên popup để xác nhận xoá.
 - b. Người quản trị nhấn vào nút “Thêm”, hệ thống sẽ thực hiện xoá danh mục đã chọn khỏi bảng CATEGORIES sau đó lấy thông

tin danh sách danh mục gồm id, title, image và hiển thị lên màn hình.

Use case kết thúc.

- **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 1 trong luồng cơ bản, nếu không có bản ghi nào trong bảng CATEGORIES thì hệ thống sẽ hiển thị “Không có danh mục sản phẩm” và use case kết thúc.
3. Tại bước 2b và 3b trong luồng cơ bản, khi quản trị viên nhập thông tin danh mục không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi yêu cầu người quản trị nhập lại, quản trị viên có thể tiếp tục nhập lại hoặc nhấn đóng modal để kết thúc.
4. Tại bước 4b trong luồng cơ bản, nếu người quản trị nhấn vào nút không thì hệ thống sẽ bỏ qua thao tác xoá và hiển thị danh sách các danh mục trong bảng CATEGORIES lên màn hình và use case kết thúc.

- **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

- **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

- **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng CATEGORIES

- **Điểm mở rộng**

Không có

2.2.4.8. Use case Bảo trì nhà hàng

- **Mô tả ngắn tắt:** Use case này cho phép quản trị viên xem, thêm, sửa, xoá nhà hàng.
- **Luồng sự kiện**
 - **Luồng cơ bản:**
 5. Use case này bắt đầu khi người dùng nhấn vào nút “Restaurants” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng RESTAUTANTS gồm các thông tin: title, ảnh, address, rating và hiển thị lên màn hình.
 6. Thêm danh mục mới:
 - a. Người quản trị nhấn vào nút “Thêm” trong màn hình danh sách danh mục. Hệ thống sẽ hiển thị modal yêu cầu nhập thông tin cho danh mục gồm title, ảnh, address.
 - b. Người quản trị nhập thông tin cho danh mục mới và nhấn vào nút “Xác nhận”. Hệ thống sẽ thêm một bản ghi mới vào bảng RESTAUTANTS sau đó lấy thông tin về danh sách danh mục mới gồm title, ảnh, address và hiển thị lên màn hình.
 7. Sửa danh mục:
 - a. Người quản trị nhấn vào biểu tượng chỉnh sửa trên một dòng danh mục. Hệ thống sẽ lấy title, ảnh, address của nhà hàng được chọn từ bảng RESTAURANT.
 - b. Người quản trị sẽ sửa tên, mô tả của danh mục sau đó nhấn vào nút “Cập nhật”. Hệ thống cập nhật thông tin danh mục vào bảng RESTAURANT sau đó lấy thông tin danh sách danh mục gồm title, ảnh, address và hiển thị lên màn hình.
 8. Xoá danh mục:
 - a. Người quản trị nhấn vào biểu tượng xoá trên một dòng danh mục. Hệ thống sẽ hiển thị lên popup để xác nhận xoá.
 - b. Người quản trị nhấn vào nút “Thêm”, hệ thống sẽ thực hiện xoá danh mục đã chọn khỏi bảng RESTAUTANTS sau đó lấy

thông tin danh sách danh mục gồm title, ảnh, address và hiển thị lên màn hình.

Use case kết thúc.

- **Luồng rẽ nhánh:**

5. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
6. Tại bước 1 trong luồng cơ bản, nếu không có bản ghi nào trong bảng RESTAUTANTS thì hệ thống sẽ hiển thị “Không có nhà hàng” và use case kết thúc.
7. Tại bước 2b và 3b trong luồng cơ bản, khi quản trị viên nhập thông tin danh mục không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi yêu cầu người quản trị nhập lại, quản trị viên có thể tiếp tục nhập lại hoặc nhấn đóng modal để kết thúc.
8. Tại bước 4b trong luồng cơ bản, nếu người quản trị nhấn vào nút không thì hệ thống sẽ bỏ qua thao tác xoá và hiển thị danh sách các danh mục trong bảng RESTAUTANTS lên màn hình và use case kết thúc.

- **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

- **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

- **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng RESTAUTANTS

- **Điểm mở rộng**

Không có

2.2.4.9. Use case Bảo trì sản phẩm

- **Mô tả vắn tắt:** Use case này cho phép quản trị viên xem, thêm, sửa, xoá sản phẩm.
- **Luồng sự kiện**
 - **Luồng cơ bản:**
 1. Use case này bắt đầu khi người dùng nhấp vào nút “Foods” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng FOODS, CATEGORIES gồm các thông tin: id, title, stock, price, brand, category, thumbnail và hiển thị lên màn hình.
 2. Thêm sản phẩm mới:
 - a. Người quản trị nhấp vào nút “Thêm” trong màn hình danh sách sản phẩm. Hệ thống sẽ hiển thị modal yêu cầu nhập thông tin cho sản phẩm gồm title, description, price, image, category.
 - b. Người quản trị nhập thông tin cho sản phẩm mới và nhấp vào nút “Thêm”. Hệ thống sẽ thêm một bản ghi mới vào bảng FOODS sau đó lấy thông tin về danh sách sản phẩm mới gồm title, description, price, image, category, lên màn hình.
 3. Sửa sản phẩm:
 - a. Người quản trị nhấp vào icon “Sửa” trên một dòng sản phẩm. Hệ thống sẽ title, description, price, image, category từ bảng FOODS.
 - b. Người quản trị sẽ sửa các thông tin của sản phẩm sau đó nhấp vào nút “Thêm”. Hệ thống cập nhật thông tin sản phẩm vào bảng FOODS sau đó lấy thông tin danh sách sản phẩm gồm title, description, price, image, category và hiển thị lên màn hình.
 4. Xoá sản phẩm:
 - a. Người quản trị nhấp vào biểu tượng xoá trên một dòng sản phẩm. Hệ thống sẽ hiển thị lên popup để xác nhận xoá.

- b. Người quản trị nhấn vào nút “Xác nhận”, hệ thống sẽ thực hiện xoá thương hiệu đã chọn khỏi bảng FOODS sau đó lấy thông tin danh sách sản phẩm gồm gồm title, description, price, image, category và hiển thị lên màn hình.

Use case kết thúc.

- **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
2. Tại bước 1 trong luồng cơ bản, nếu không có bản ghi nào trong bảng FOODS thì hệ thống sẽ hiển thị “Không có sản phẩm” và use case kết thúc.
3. Tại bước 2b và 3b trong luồng cơ bản, khi quản trị viên nhập thông tin sản phẩm không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi yêu cầu người quản trị nhập lại, quản trị viên có thể tiếp tục nhập lại hoặc nhấn đóng modal để kết thúc.
4. Tại bước 4b trong luồng cơ bản, nếu người quản trị nhấn vào nút không thì hệ thống sẽ bỏ qua thao tác xoá và hiển thị danh sách các sản phẩm trong bảng FOODS lên màn hình và use case kết thúc.

- **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

- **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

- **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng PRODUCTS.

- **Điểm mở rộng**

Không có

2.2.4.10. Use case Quản lý đơn hàng

- **Mô tả ngắn tắt:** Use case này cho phép quản trị viên xem, thay đổi trạng thái đơn hàng.
- **Luồng sự kiện**
 - **Luồng cơ bản:**
 1. Use case này bắt đầu khi người dùng nhấn vào nút “Quản lý đơn hàng” trên menu quản trị. Hệ thống sẽ lấy trong bảng ORDERS gồm các thông tin: orderTotal, deliveryFee, grandTotal, paymentStatus, orderStatus, tổng số lượng sản phẩm, ngày tạo và hiển thị lên màn hình.
 2. Cập nhật trạng thái đơn hàng:
 - a. Người quản trị nhấn vào biểu tượng cập nhật trên một dòng đơn đặt hàng. Hệ thống sẽ thay đổi trạng thái của đơn hàng theo lựa chọn của người quản trị. Các trạng thái là: Placed, Preparing, Out for Delivery, Delivery.
 3. Huỷ đơn hàng:
 - a. Người quản trị nhấn vào biểu tượng huỷ trên một dòng đơn hàng. Hệ thống sẽ thay đổi trạng thái của đơn hàng sang đã huỷ và hiển thị lên màn hình.

Use case kết thúc.
 - **Luồng rẽ nhánh:**
 1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.
- **Các yêu cầu đặc biệt**
Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.
- **Tiền điều kiện**
Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

- **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng ORDERS.

- **Điểm mở rộng**

Không có

2.2.4.11. Use case Quản lý người dùng

- **Mô tả ngắn tắt:** Use case này cho phép quản trị viên xem, cập nhật người dùng.
- **Luồng sự kiện**
 - **Luồng cơ bản:**
 1. Use case này bắt đầu khi người dùng nhấn vào nút “Quản lý người dùng” trên thanh menu quản trị. Hệ thống sẽ lấy trong bảng USERS gồm các thông tin: username, email, userType, profile và hiển thị lên màn hình.
 2. Thêm người dùng mới:
 - a. Người quản trị nhấn vào nút “Tạo mới” trong màn hình danh sách người dùng. Hệ thống sẽ hiển thị modal yêu cầu nhập thông tin cho người dùng gồm username, email, password của người dùng.
 - b. Người quản trị nhập thông tin cho người dùng mới và nhấn vào nút “Tạo mới”. Hệ thống sẽ thêm một bản ghi mới vào bảng USERS sau đó lấy thông tin về danh sách người dùng mới gồm username, email, profile và hiển thị lên màn hình.
 3. Sửa thông tin người dùng:
 - a. Người quản trị nhấn vào nút “Sửa” trên một dòng người dùng. Hệ thống hiển thị các thông tin người dùng gồm username, email và hiển thị lên màn hình.
 - b. Người quản trị sẽ sửa các thông tin của người dùng sau đó nhấn vào nút “Cập nhật”. Hệ thống cập nhật thông tin người dùng vào bảng USERS sau đó lấy thông tin danh sách người dùng gồm họ tên, email, ngày sinh, số điện thoại, địa chỉ, giới tính và hiển thị lên màn hình.
 4. Xoá người dùng:

a. Người quản trị nhấn vào biểu tượng xoá trên một dòng người dùng. Hệ thống sẽ hiển thị lên popup để xác nhận xoá.

b. Người quản trị nhấn vào nút “Đồng ý”, hệ thống sẽ thực hiện xoá thương hiệu đã chọn khỏi bảng USERS

Use case kết thúc.

- **Luồng rẽ nhánh:**

1. Tại thời điểm bất kỳ trong quá trình thực hiện use case nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

2. Tại bước 1 trong luồng cơ bản, nếu không có bản ghi nào trong bảng USERS thì hệ thống sẽ hiển thị “Không có người dùng” và use case kết thúc.

3. Tại bước 2b và 3b trong luồng cơ bản, khi quản trị viên nhập thông tin người dùng không hợp lệ, hệ thống sẽ hiển thị thông báo lỗi yêu cầu người quản trị nhập lại, quản trị viên có thể tiếp tục nhập lại hoặc nhấn đóng modal để kết thúc.

4. Tại bước 4b trong luồng cơ bản, nếu người quản trị nhấn vào nút không thì hệ thống sẽ bỏ qua thao tác xoá và hiển thị danh sách các sản phẩm trong bảng USERS lên màn hình và use case kết thúc.

- **Các yêu cầu đặc biệt**

Cần kiểm soát quyền thực hiện use case này để đảm bảo tính an toàn và bảo mật.

- **Tiền điều kiện**

Người quản trị cần đăng nhập với quyền quản trị trước khi thực hiện use case.

- **Hậu điều kiện**

Sau khi use case kết thúc thành công thì lưu các thông tin vào bảng USERS

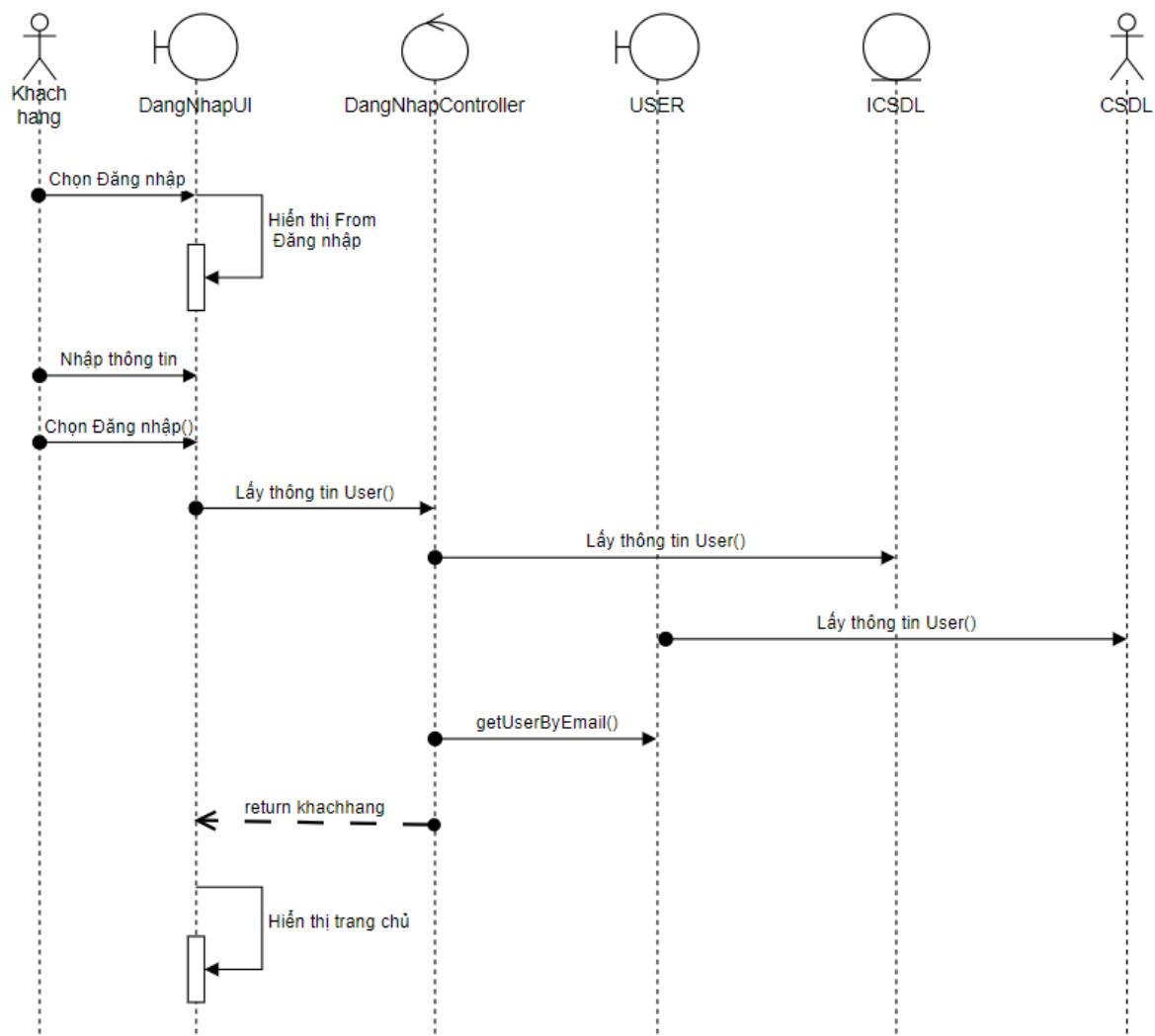
- **Điểm mở rộng**

Không có

2.2.5. Phân tích các use case

2.2.5.1. Use case Đăng nhập

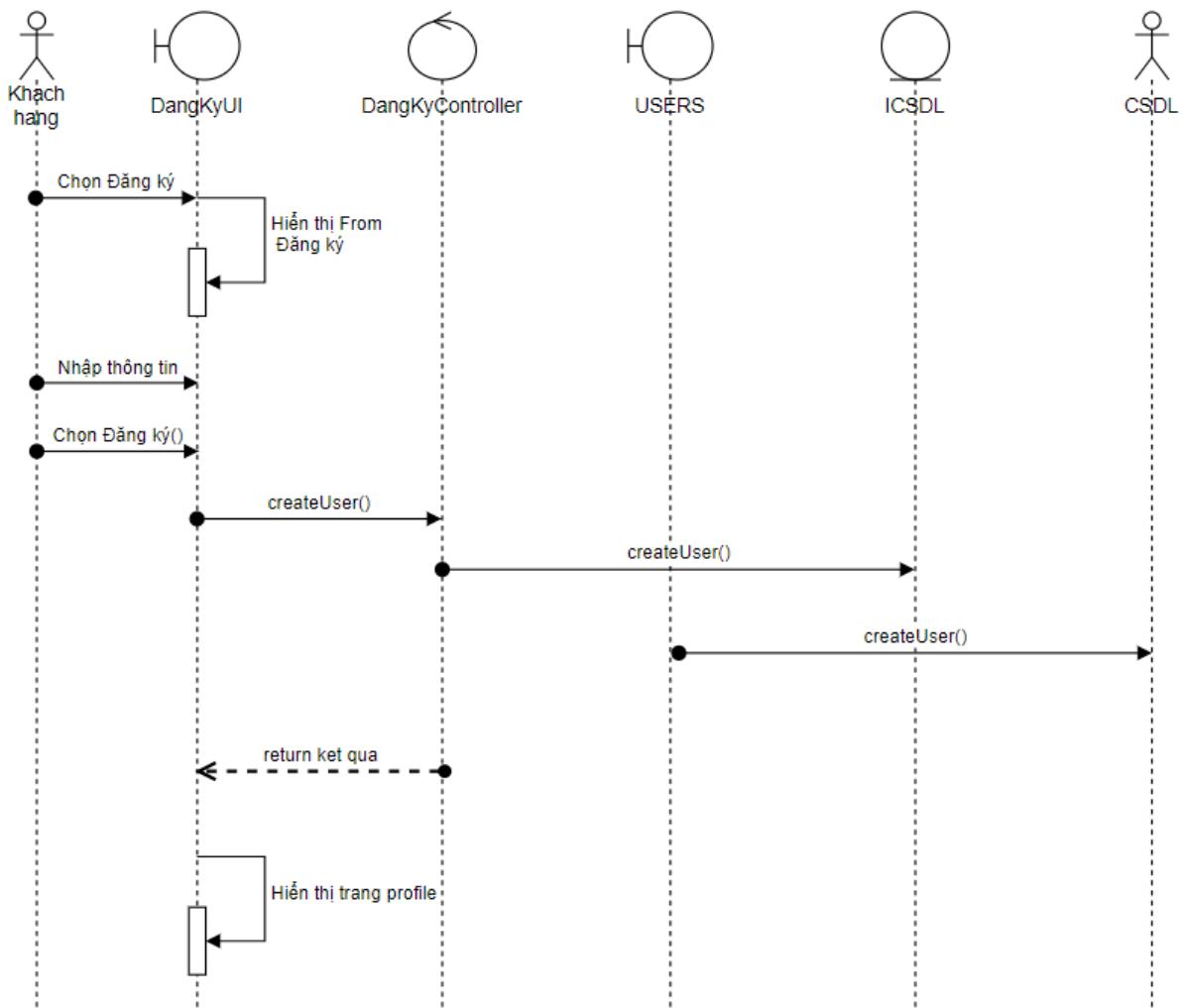
- Biểu đồ trình tự



Hình 2.1. Biểu đồ trình tự use case Đăng nhập

2.2.5.2. Use case Đăng ký

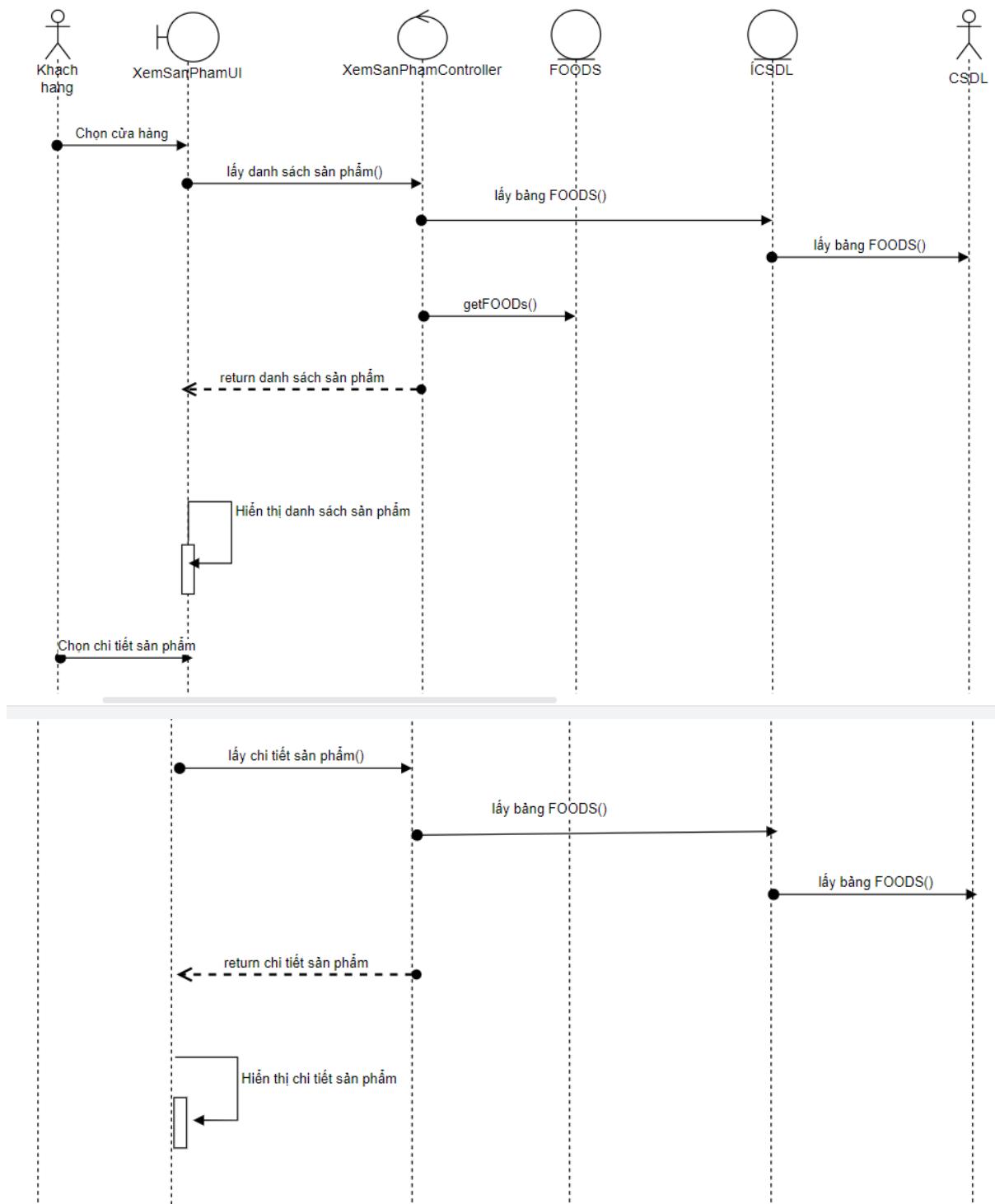
- Biểu đồ trình tự



Hình 2.5. Biểu đồ trình tự use case Đăng kí tài khoản

2.2.5.3. Use case Xem sản phẩm

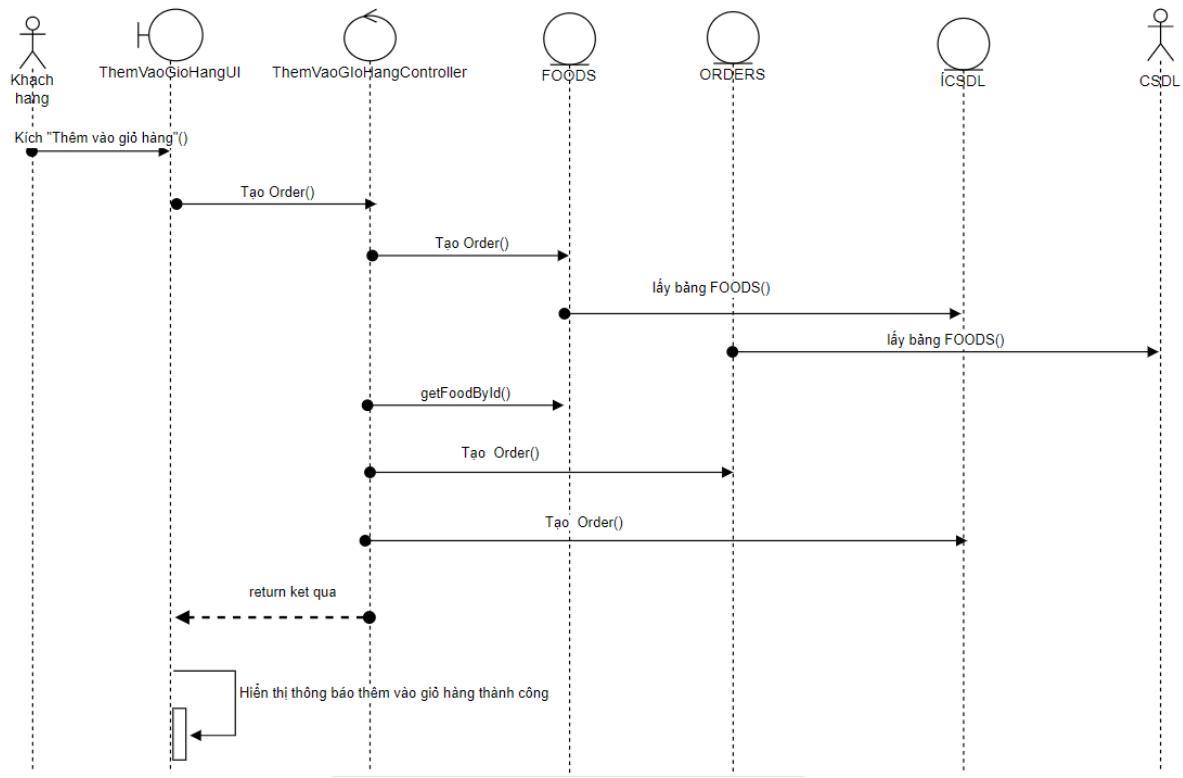
- Biểu đồ trình tự



Hình 2.6. Biểu đồ trình tự use case Xem sản phẩm

2.2.5.4. Use case *Thêm vào giỏ hàng*

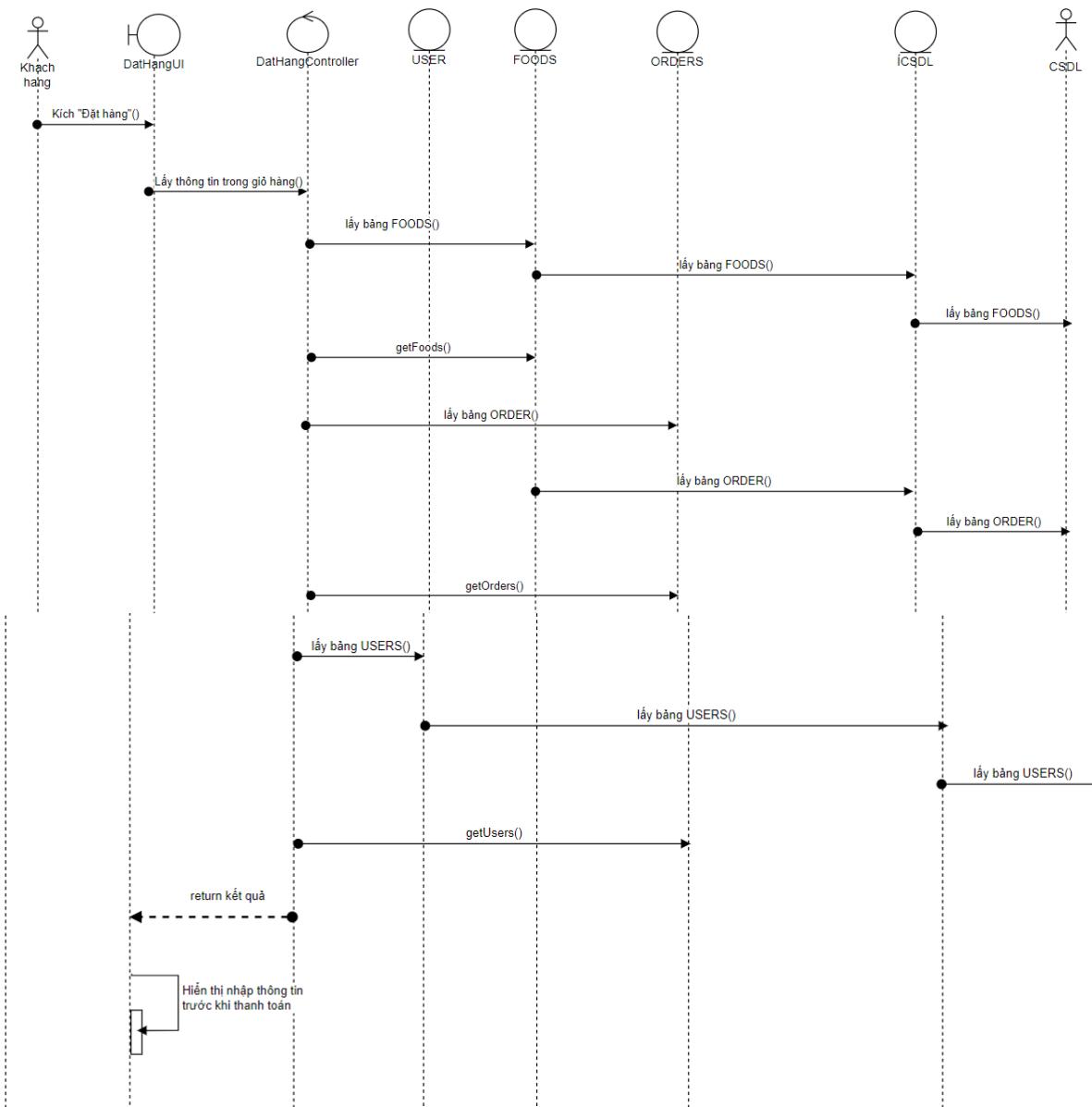
- Biểu đồ trình tự



Hình 2.7. Biểu đồ trình tự use case *Thêm vào giỏ hàng*

2.2.5.5. Use case Đặt hàng

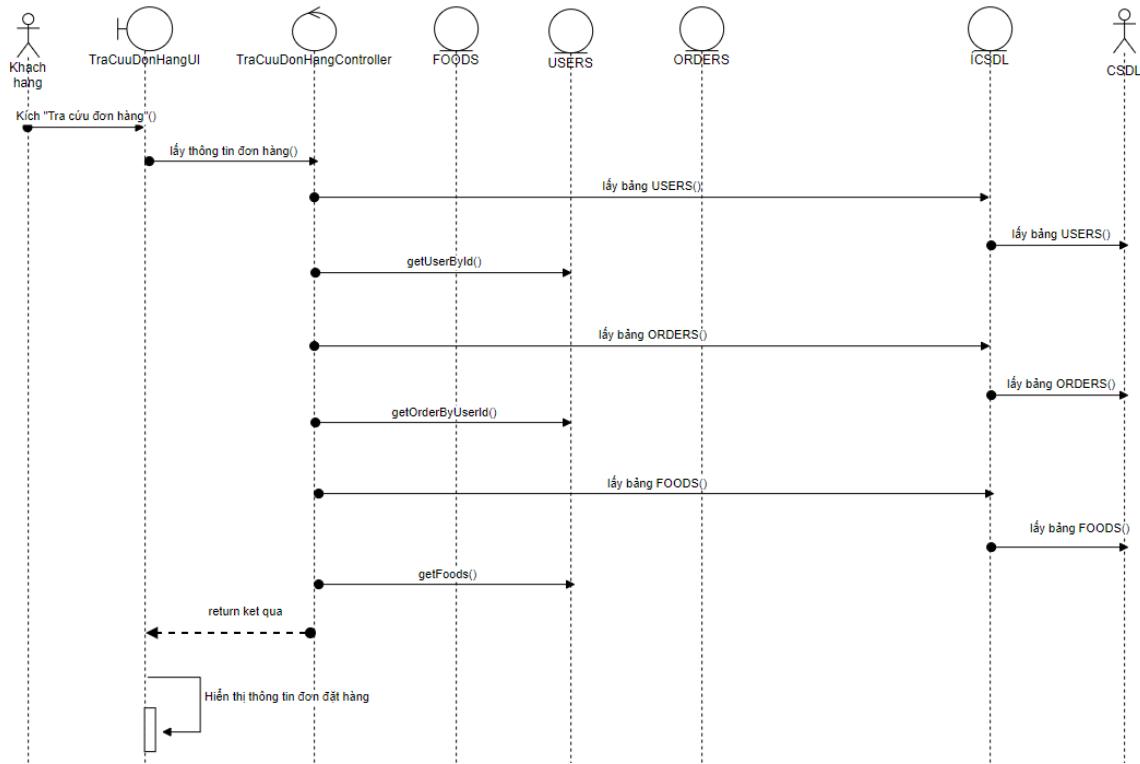
- Biểu đồ trình tự



Hình 2.8. Biểu đồ trình tự use case Đặt hàng

2.2.5.6. Use case Tra cứu đơn hàng

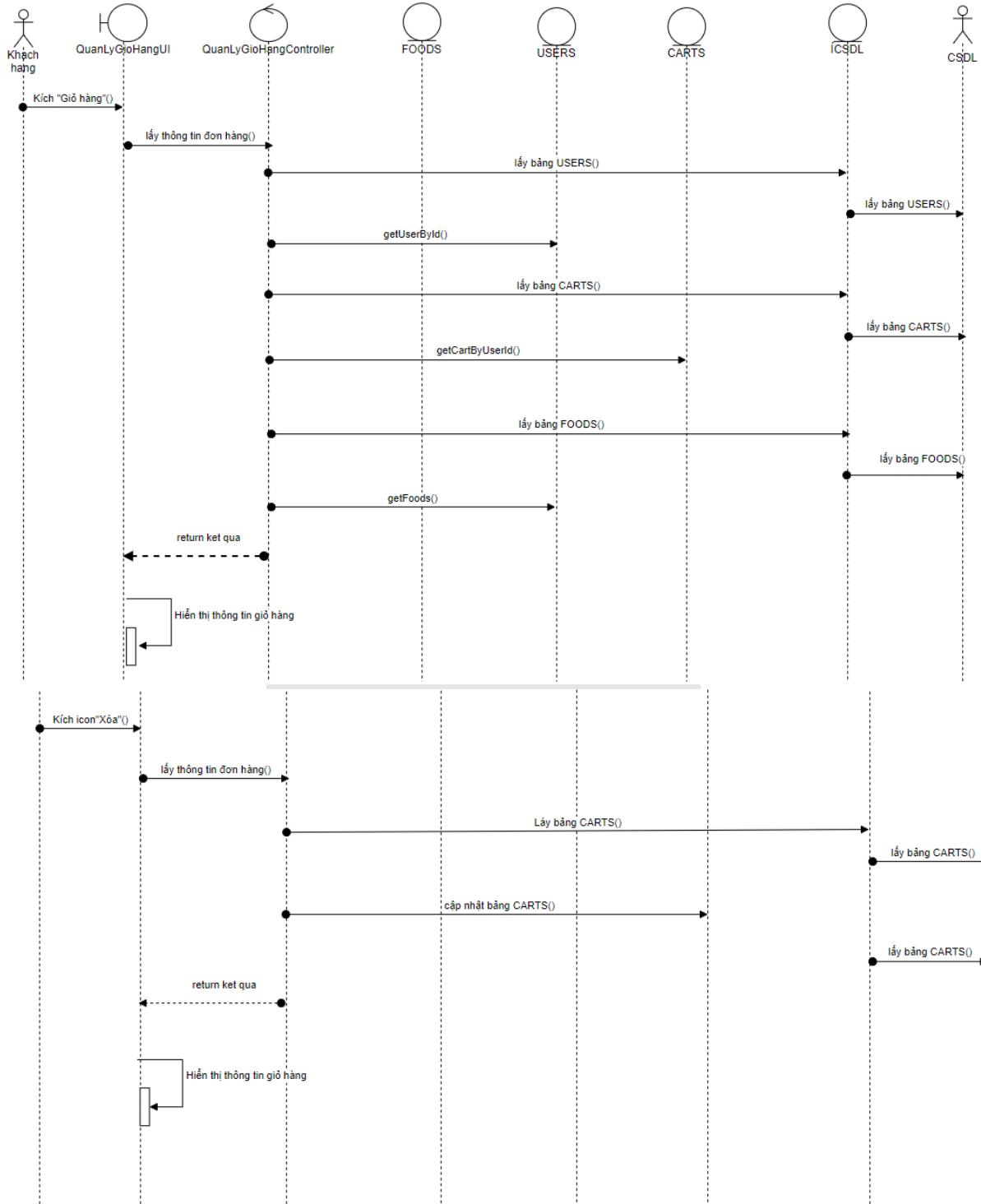
- Biểu đồ trình tự



Hình 2.9. Biểu đồ trình tự use case Tra cứu đơn hàng

2.2.5.7. Use case Quản lý giỏ hàng

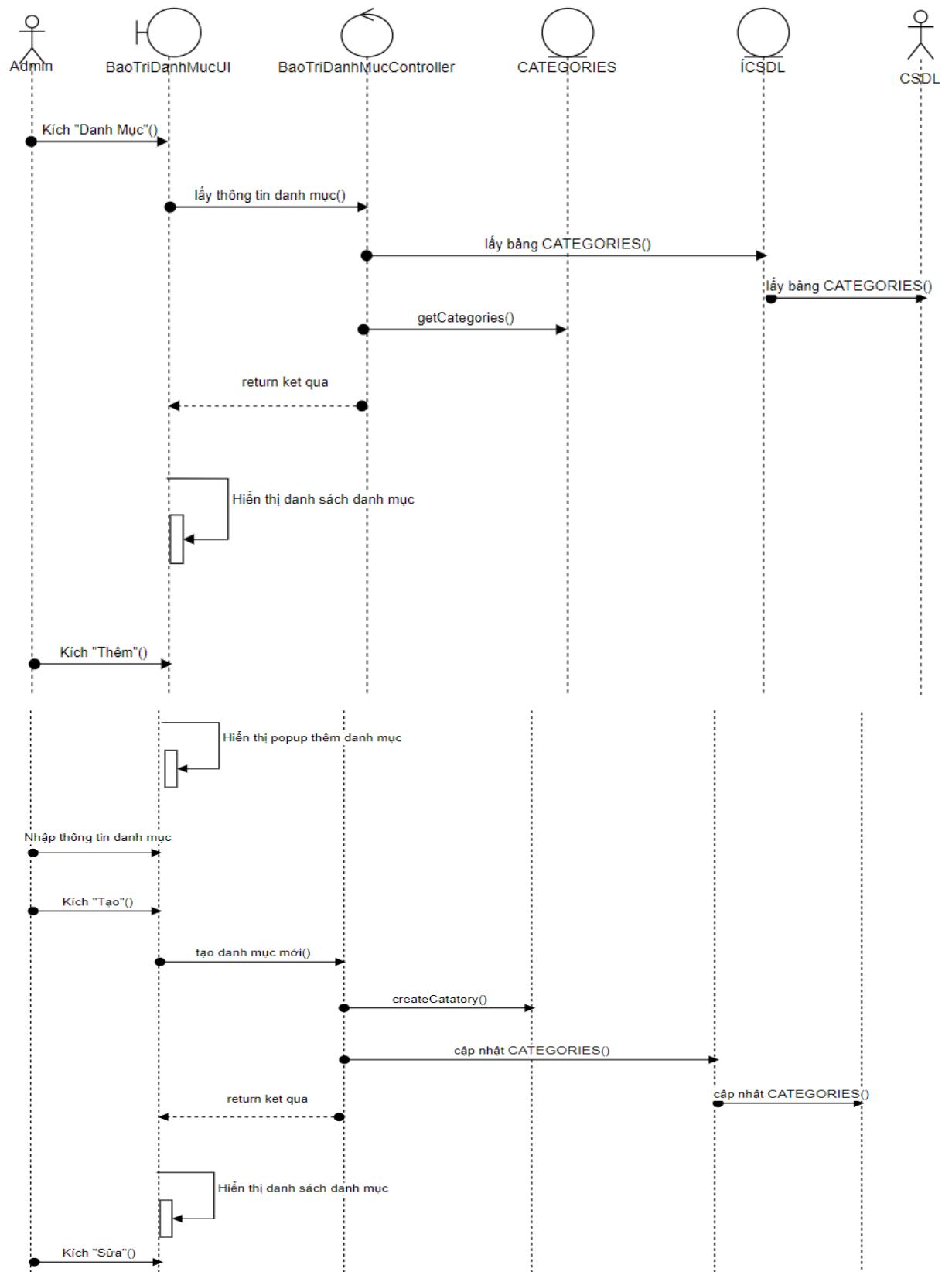
- Biểu đồ trình tự

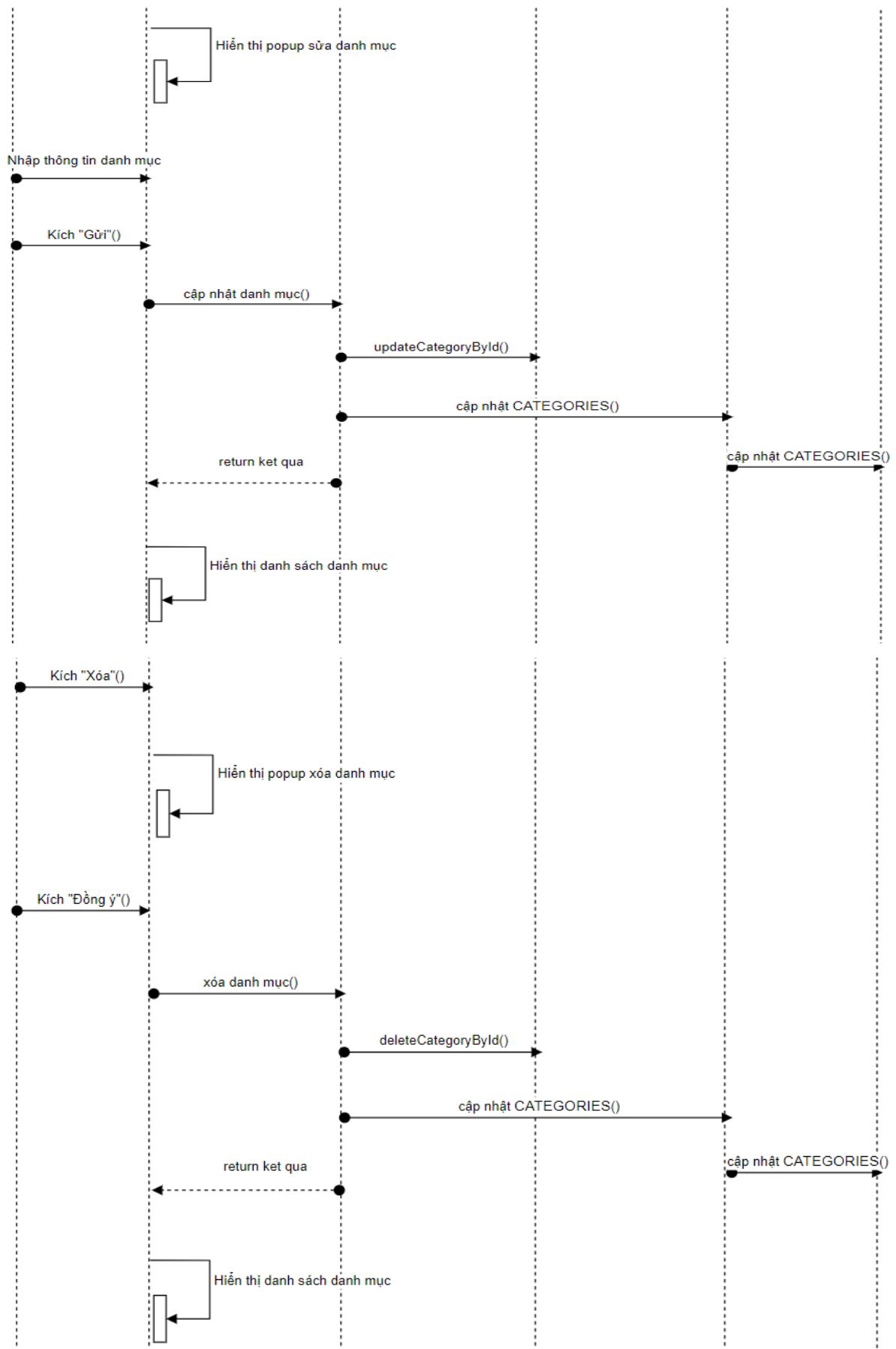


Hình 2.10. Biểu đồ trình tự use case Quản lý giỏ hàng

2.2.5.8. Use case Bảo trì danh mục

- Biểu đồ trình tự

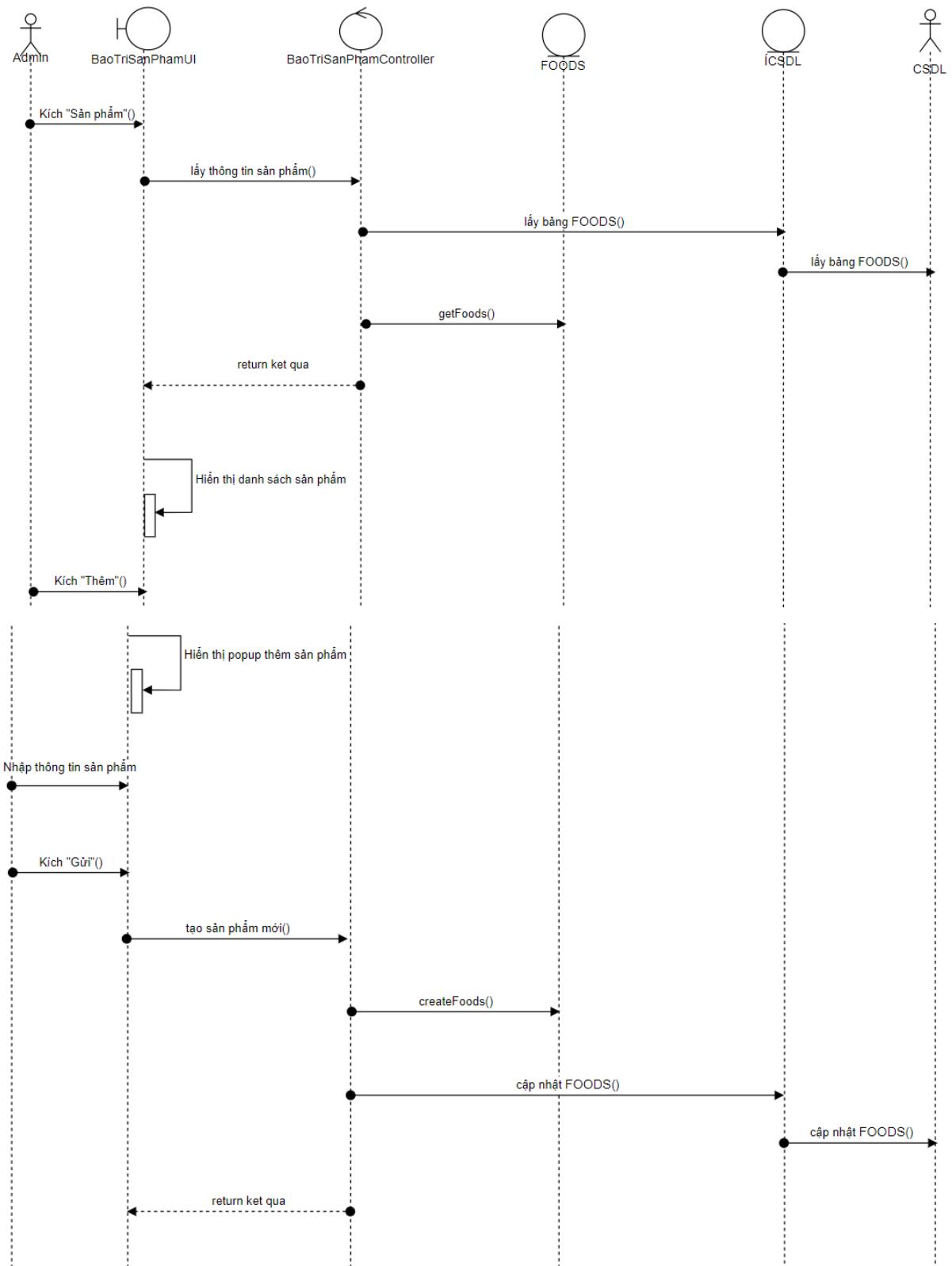


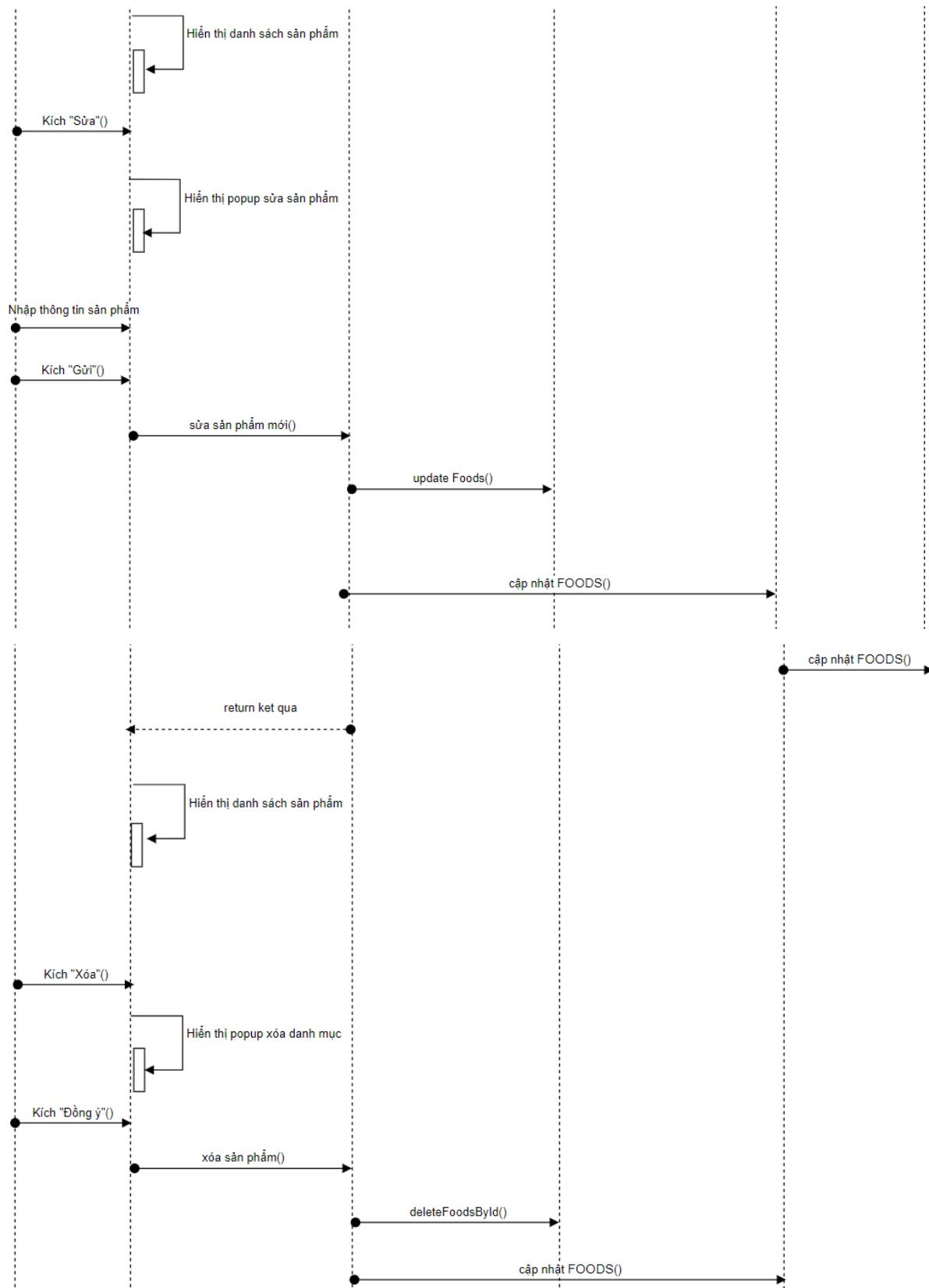


Hình 2.11. Biểu đồ trình tự use case Bảo trì danh mục

2.2.5.9. Use case *Bảo trì sản phẩm*

- Biểu đồ trình tự



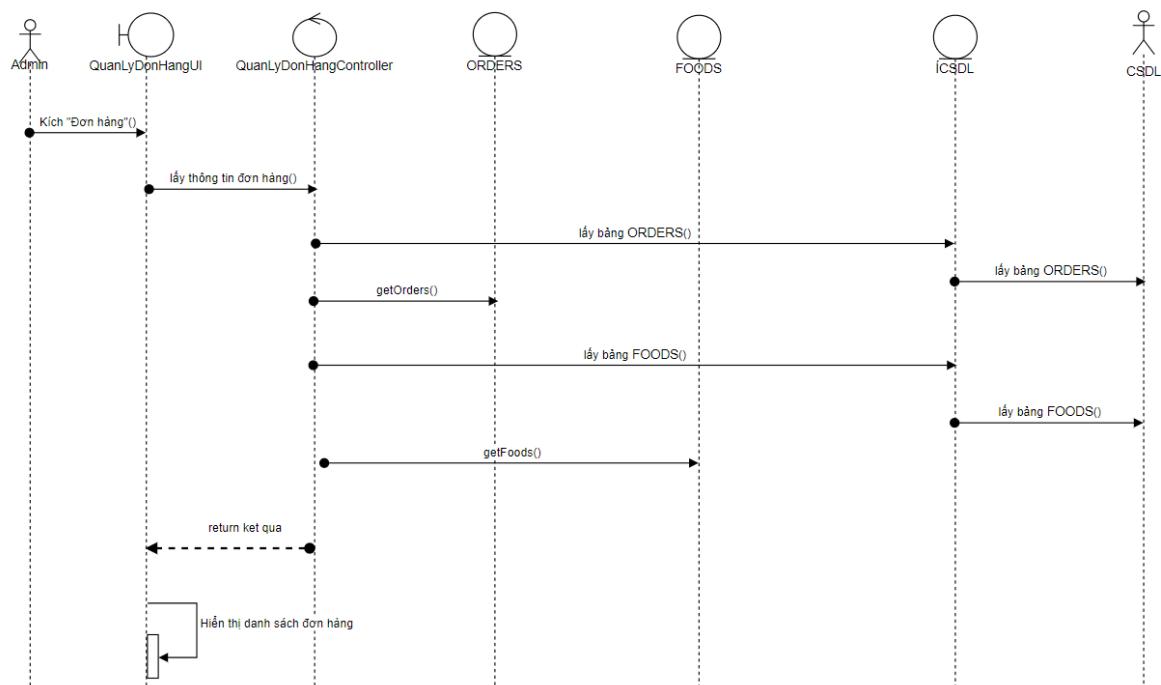




Hình 2.12. Biểu đồ trình tự use case Bảo trì sản phẩm

2.2.5.10. Use case Quản lý đơn hàng

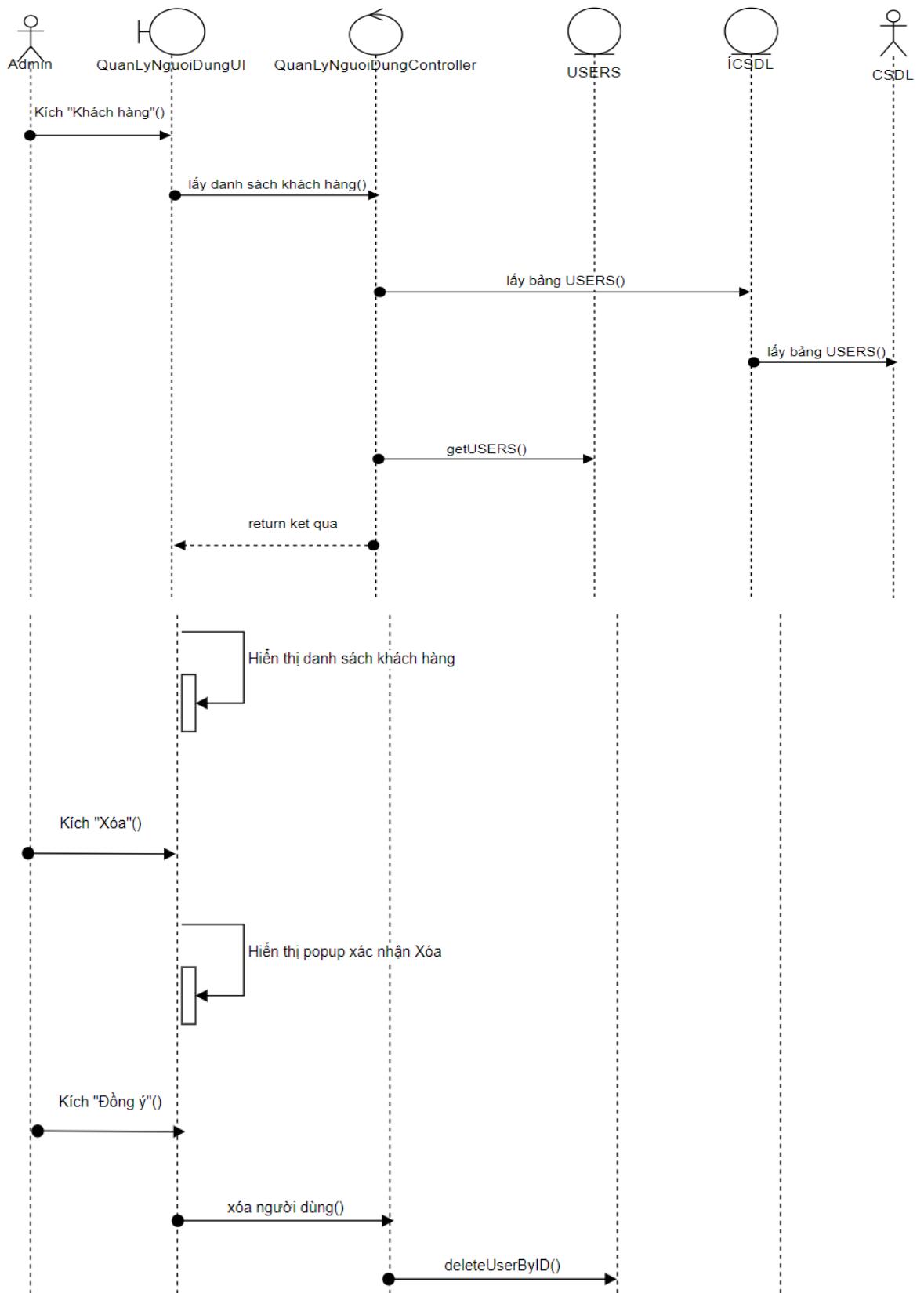
- Biểu đồ trình tự

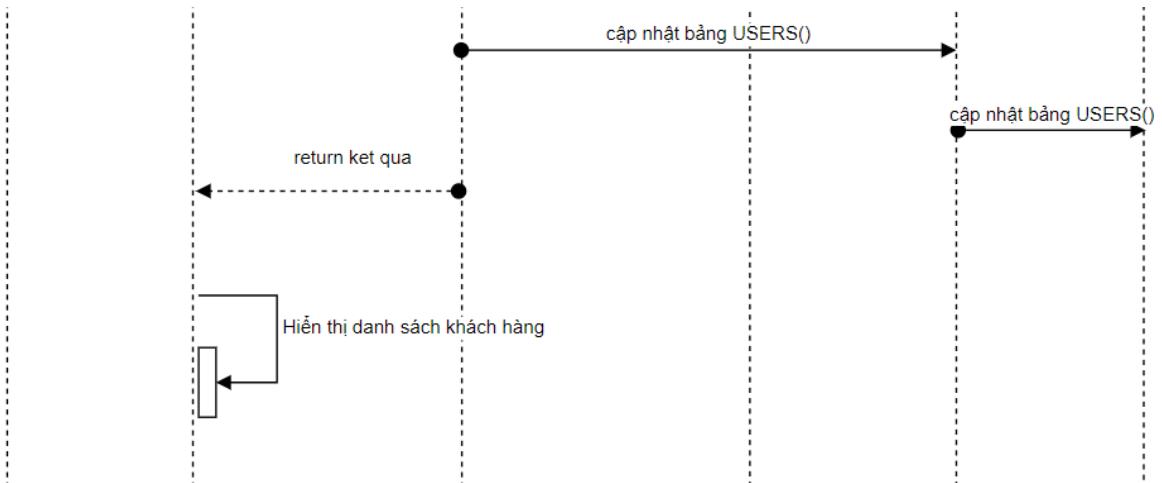


Hình 2.13. Biểu đồ trình tự use case Quản lý đơn hàng

2.2.5.11. Use case Quản lý người dùng

- Biểu đồ trình tự

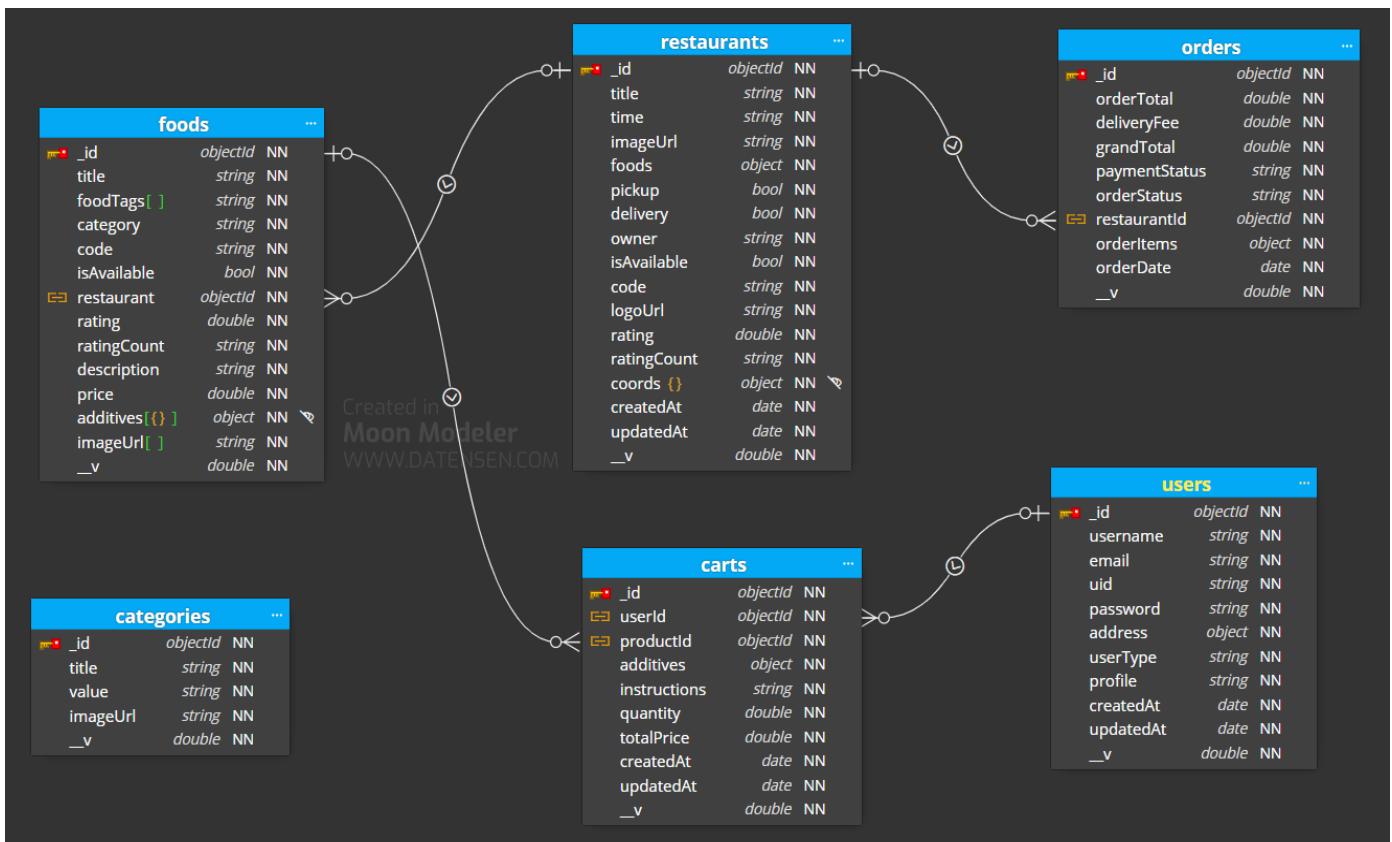




Hình 2.14. Biểu đồ trình tự use case Quản lý người dùng

2.3. Xây dựng cơ sở dữ liệu

2.3.1. Biểu đồ Entity Relationship Diagram



Hình 2.15. Thiết kế cơ sở dữ liệu

2.3.2. Cấu trúc thành phần trong cơ sở dữ liệu

USERS SCHEMA

```
const UserSchema = new mongoose.Schema({
    username: {type: String, required: true},
    email: {type: String, required: true, unique: true},
    uid: {type: String, required: true, unique: true},
    password: {type: String, required: true},
    address: {type: Array, required: false},
    phone: {type: String, required: false},
    userType: {type: String, required: true, default: "Client", enum: ['Admin', 'Driver', 'Client', 'Vendor']},
    profile: {
        type: String,
        required: true,
        default: 'https://d326fntlu7tb1e.cloudfront.net/uploads/bdec9d7d-0544-4fc4-823d-3b898f6dbbbf-vinci_03.jpeg'
    }
}, {timestamps: true});

module.exports = mongoose.model('User', UserSchema)
```

Hình 2.16. Users schema

RESTAURANT SCHEMA

```
const restaurantSchema = new mongoose.Schema({
    title: {type: String, required: true},
    time: {type: String, required: true},
    imageUrl: {type: String, required: true},
    foods: {type: Array},
    pickup: {type: Boolean, required: false, default: true},
    delivery: {type: Boolean, required: false, default: true},
    owner: {type: String, required: true},
    isAvailable: {type: Boolean, default: true},
    code: {type: String, required: true},
    logoUrl: {
        type: String,
        required: true,
        default: 'https://d326fntlu7tb1e.cloudfront.net/uploads/bdec9d7d-0544-4fc4-823d-3b898f6dbbbf-vinci_03.jpeg'
    },
    rating: {type: Number, min: 1, max: 5},
    ratingCount: {type: String},
    coords: {
        id: {type: String, required: true},
        latitude: {type: Number, required: true},
        longitude: {type: Number, required: true},
        latitudeDelta: {type: Number, required: true, default: 0.0122},
        longitudeDelta: {type: Number, required: true, default: 0.0221},
        address: {type: String, required: true},
        title: {type: String, required: true}
    }
}, {timestamps: true});
```

Hình 2.17. Restaurants schema

CATEGORIES SCHEMA

```
const categorySchema = new mongoose.Schema({
    title: {type: String, required: true},
    value: {type: String, required: true},
    imageUrl: {type: String, required: true}
}, {timestamps: false});
```

Hình 2.18. Categories schema

CARTS

```
const cartSchema = new mongoose.Schema({
    userId: {type: mongoose.Schema.Types.ObjectId, ref: 'User'},
    productId: {type: mongoose.Schema.Types.ObjectId, ref: 'Food'},
    additives: {type: []},
    instructions: {type: String, default: ''},
    quantity: {type: Number, default: 1},
    totalPrice: {type: Number, required: true}
}, {timestamps: true});
```

Hình 2.19. Carts schema

ODERS SCHEMA

```
const orderSchema = new mongoose.Schema({
    userId: {type: mongoose.Schema.Types.ObjectId, ref: "User"},
    orderItems: [orderItemSchema],
    orderTotal: {type: Number, required: true},
    deliveryFee: {type: Number, required: true},
    grandTotal: {type: Number, required: true},
    deliveryAddress:{type: mongoose.Schema.Types.ObjectId, ref: "Address"},
    paymentMethod: {type: String},
    paymentStatus: {type: String, default: "Pending", enum: ["Pending", "Completed", "Failed"]},
    orderStatus: {type: String, default: "Placed", enum: ["Placed", "Preparing", "Out for Delivery", "Delivery"]},
    orderDate: {type: Date, default: Date.now},
    restaurantId: {type: mongoose.Schema.Types.ObjectId, ref: "Restaurant", required: true},
    driverId: {type: mongoose.Schema.Types.ObjectId, ref: "Driver"},
    rating: {type: Number, min: 1, max: 5},
    feedBack: {type: String},
    promoCode: {type: String},
    discountAmount: {type: Number},
    notes: {type: String}
}, {timestamps: false});
```

Hình 2.20. Orders schema

DRIVER SCHEMA

```
const driverSchema = new mongoose.Schema({
  driver: {type: mongoose.Schema.Types.ObjectId, ref: 'User'},
  vehicleType: {type: String, required: true, enum: ['Bike', 'Scooter', 'Car']},
  vehicleNumber: {type: String, required: true},
  currentLocation: {
    latitude: {type: Number, required: true},
    longitude: {type: Number, required: true},
    latitudeDelta: {type: Number, required: true, default: 0.0122},
    longitudeDelta: {type: Number, required: true, default: 0.0221},
  },
  isAvailable: {type: Boolean, required: true},
  rating: {type: Number, required: true},
  totalDeliveries: {type: Number, default: 0},
  profileImage: {type: String, defalut: "https://d326fntlu7tb1e.cloudfront.net/uploads/bdec9d7d-0544-4fc4-823d-3b898f6dbbbf-vinci_03.jpeg"}
}, {timestamps: true})
```

Hình 2.21. Driver schema

ADDRESS SCHEMA

```
const addressSchema = new mongoose.Schema({
  userId: {type: String, required: true},
  addressLine1: {type: String, required: true},
  city: {type: String, required: true},
  state: {type: String, required: true},
  district: {type: String, required: true},
  postalCode: {type: String, required: true},
  country: {type: String, required: true},
  deliveryInstructions: {type: String},
  default: {type: Boolean, default: true},
}, {timestamps: false});
```

Hình 2.22. Address schema

FOOD SCHEMA

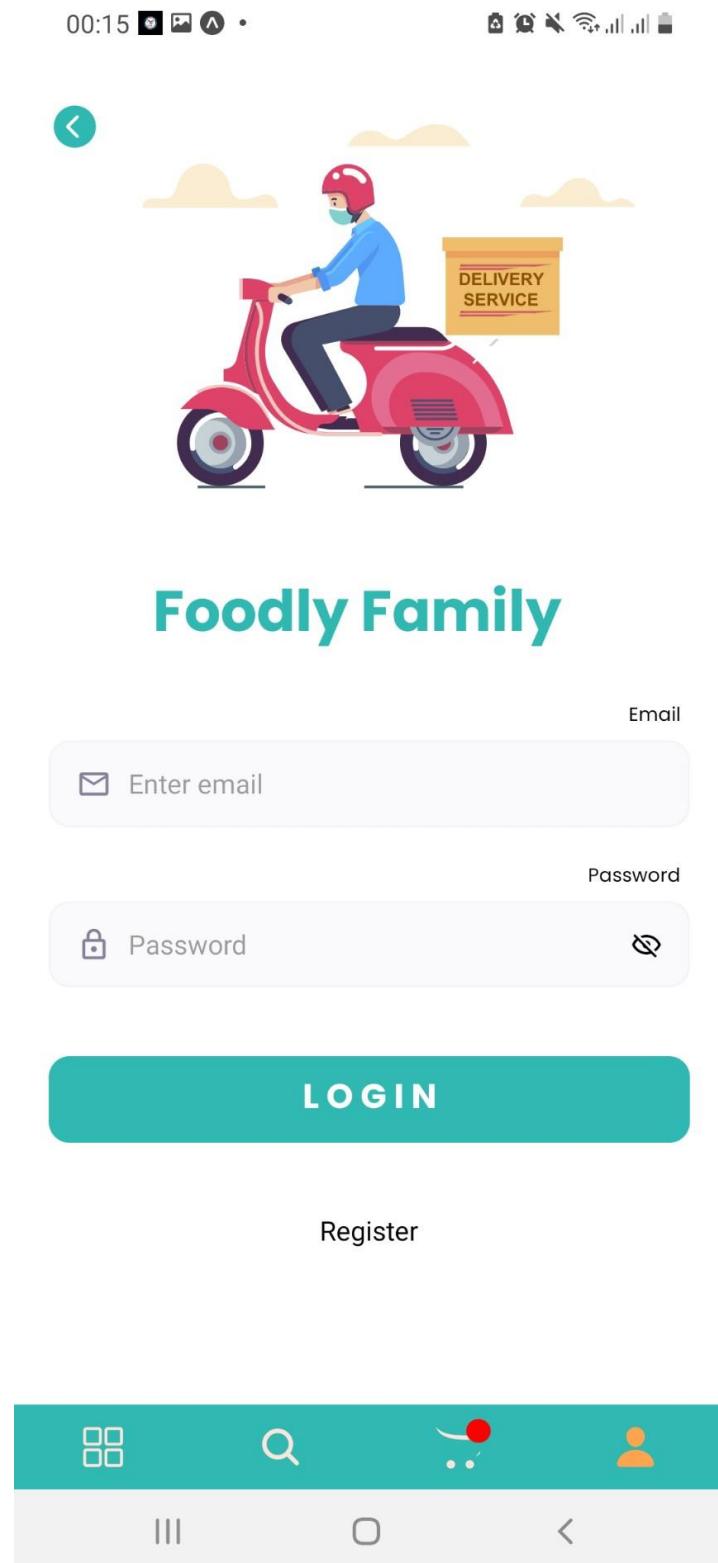
```
const foodSchema = new mongoose.Schema({
    title: {type: String, required: true},
    foodTags: {type: Array, required: true},
    category: {type: String, required: true},
    code: {type: String, required: true},
    isAvailable: {type: Boolean, required: true, default: true},
    restaurant: {type: mongoose.Schema.Types.ObjectId, ref: 'Restaurant'},
    rating: {
        type: Number,
        min: 1,
        max: 5,
        default: 5
    },
    ratingCount: {type: String},
    description: {type: String, required: true},
    price: {type: Number, required: true},
    additives: {type: Array, required: true},
    imageUrl: {type: Array, required: true}
});
```

Hình 2.23. Foods schema

CHƯƠNG 3. KẾT QUẢ ĐẠT ĐƯỢC

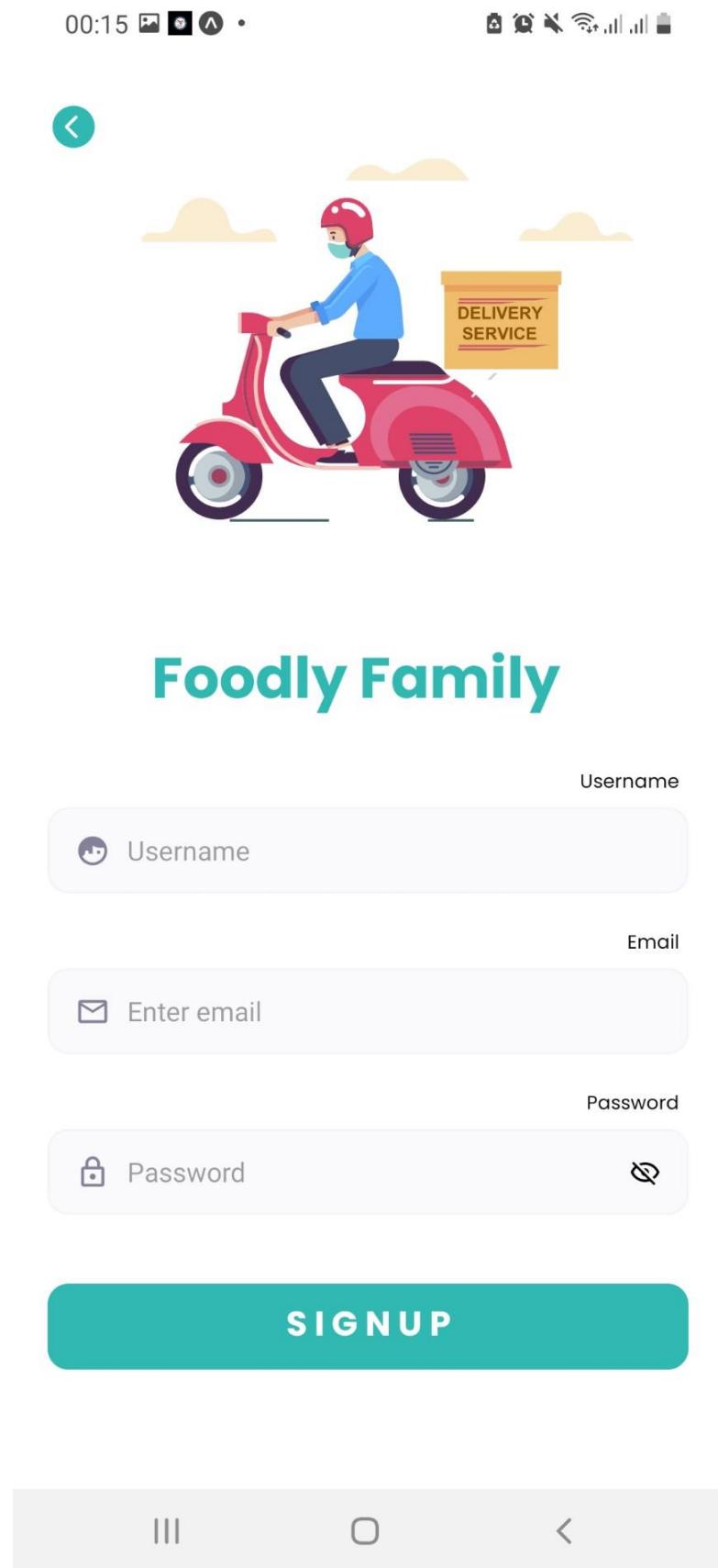
3.1. Giao diện chương trình

3.1.1. Giao diện đăng nhập



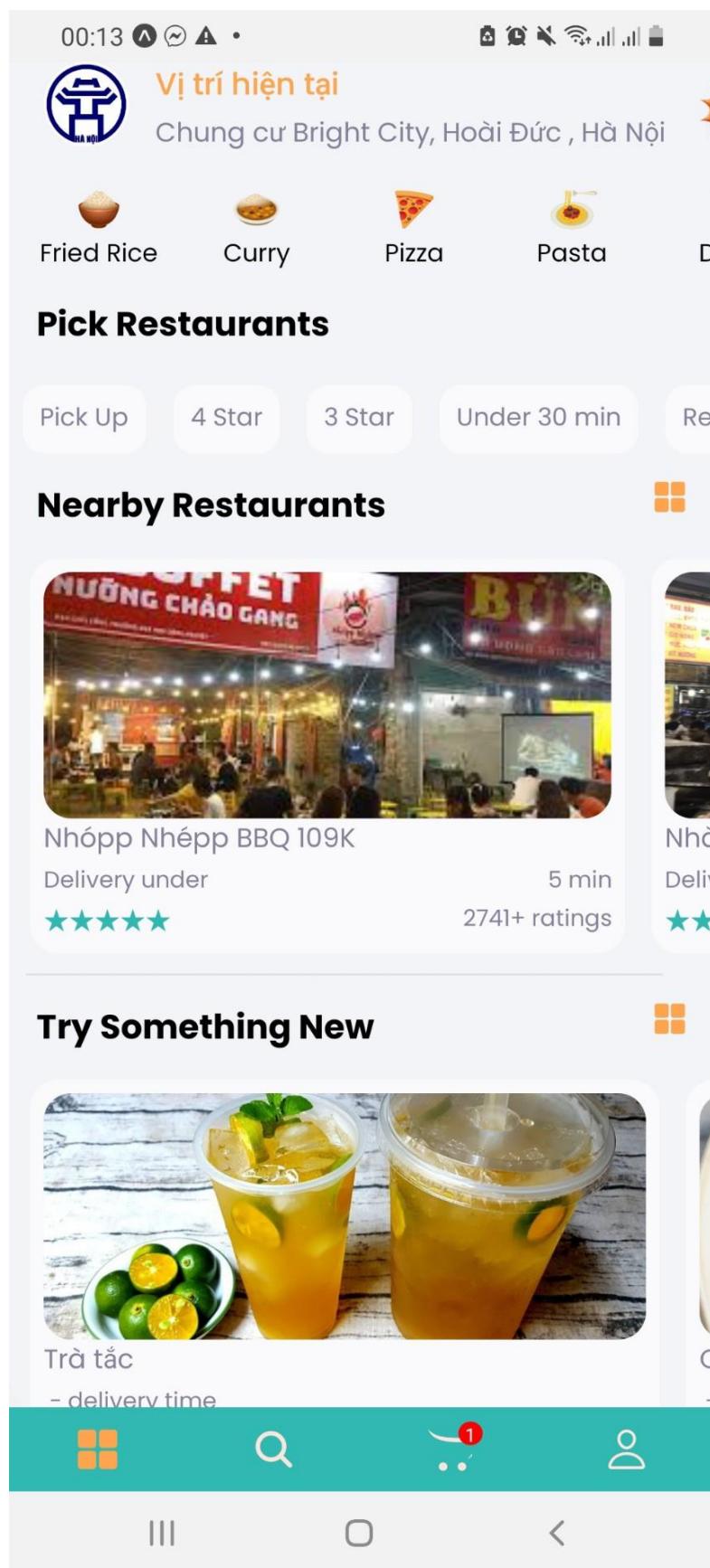
Hình 3.1. Màn hình đăng nhập

3.1.2. Giao diện đăng ký



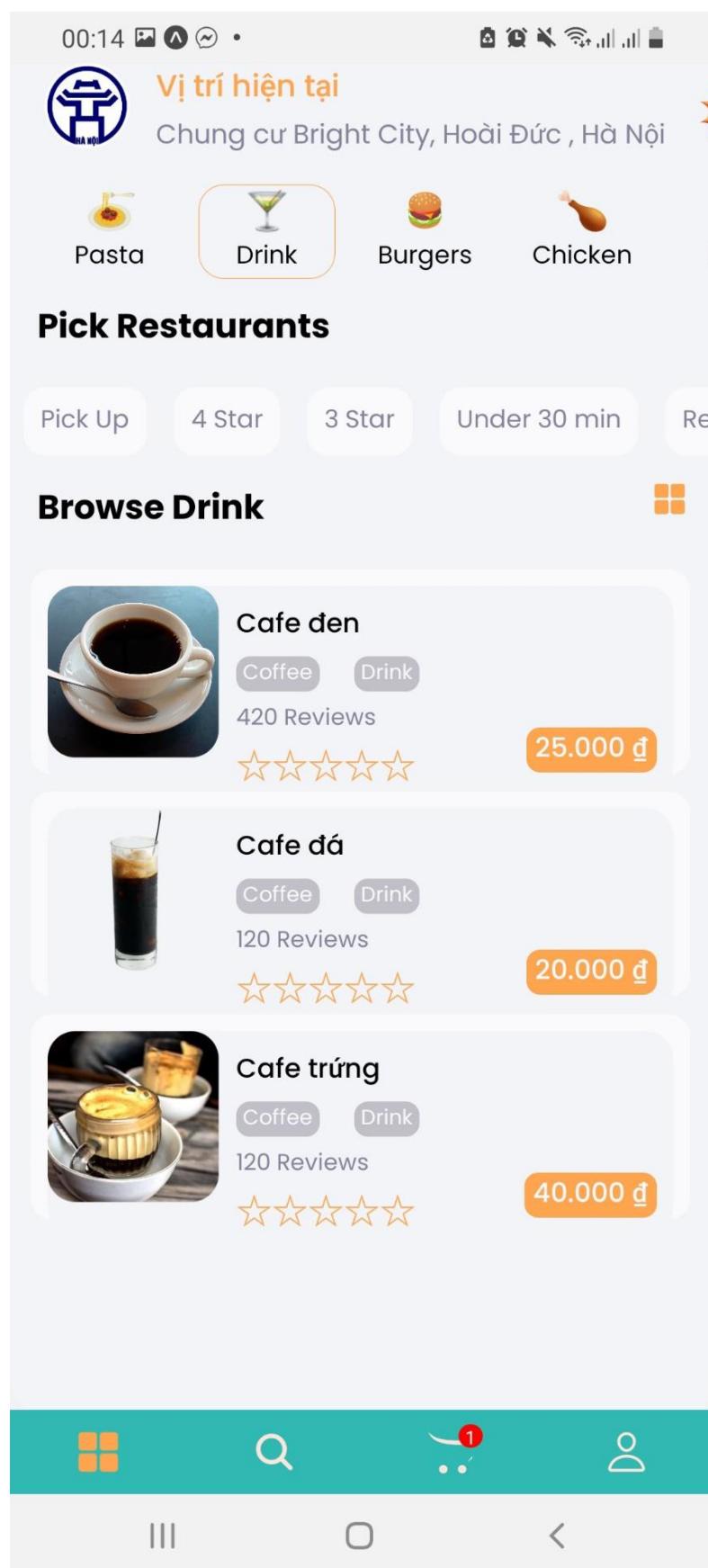
Hình 3.2. Màn hình đăng ký

3.1.3. Giao diện trang chủ



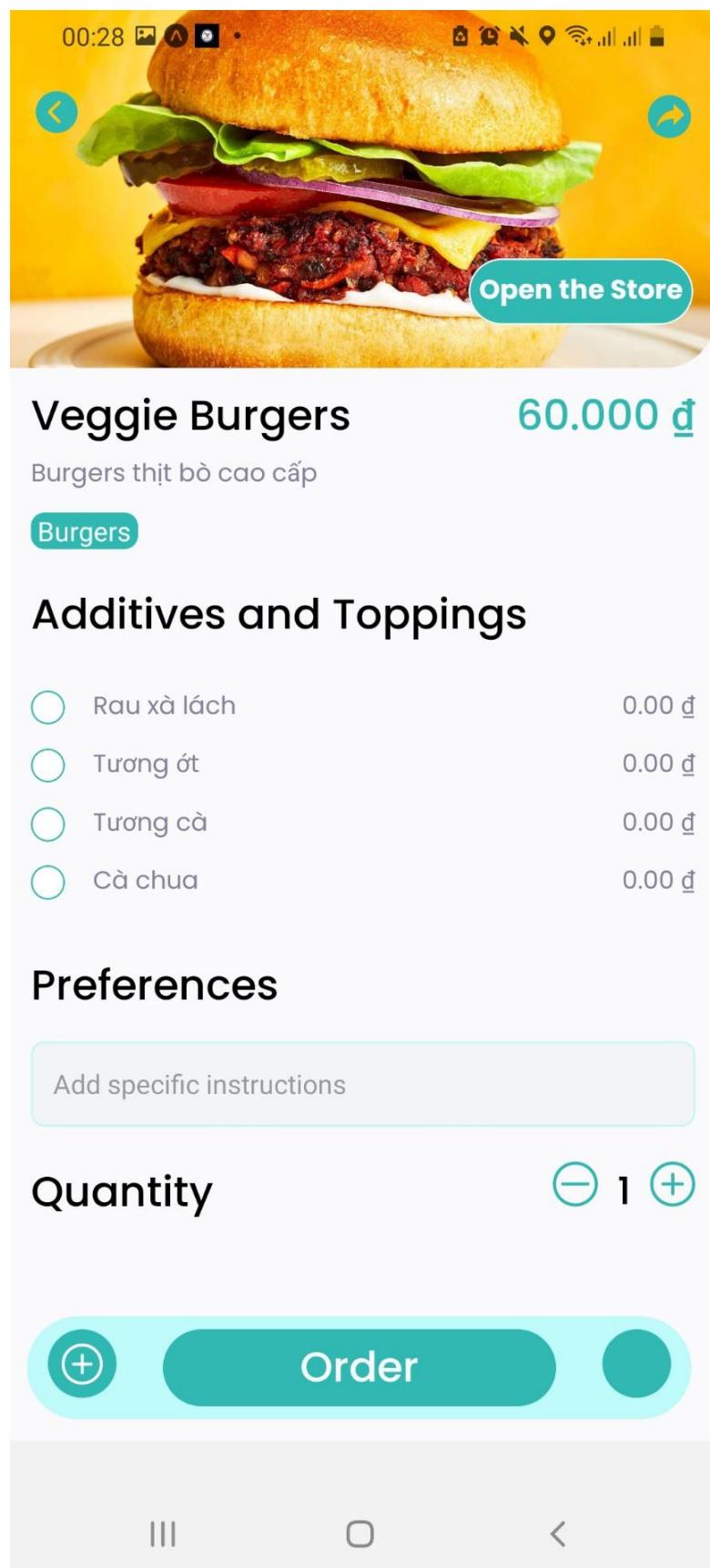
Hình 3.3. Màn hình trang chủ

3.1.4. Màn hình xem sản phẩm theo danh mục



Hình 3.4. Màn hình xem sản phẩm theo danh mục

3.1.5. Màn hình xem chi tiết sản phẩm



Hình 3.5. Màn hình xem chi tiết sản phẩm

3.1.6. Màn hình tìm kiếm sản phẩm

00:14

Cơm

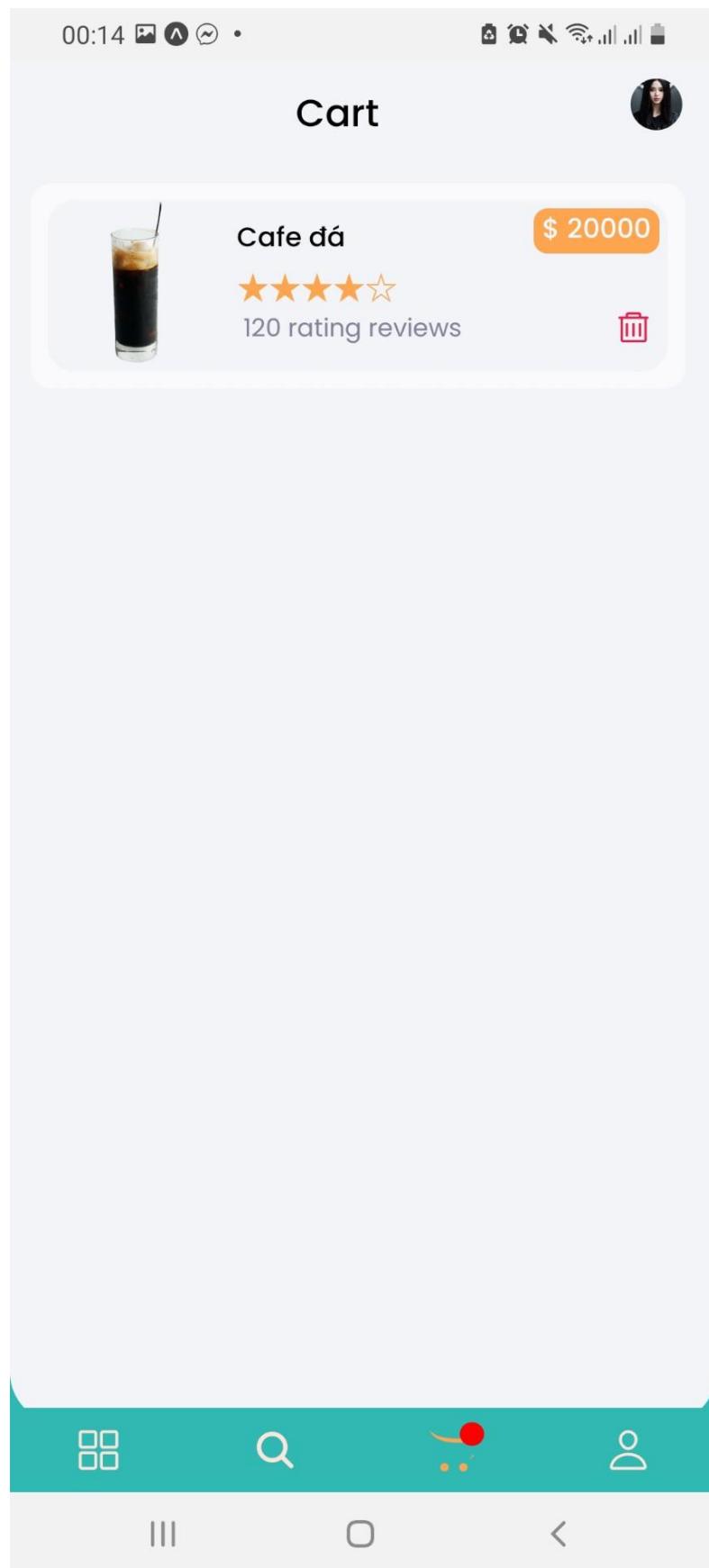
Cơm chiên trứng
Cơm chiên nóng hổi
40.000 đ
Rating: 4.4 (58 reviews)

Cơm rang gà
Cơm chiên gà nóng hổi
60.000 đ
Rating: 4.9 (144 reviews)

Cơm rang cà ry
Cơm cà ry Ấn Độ
55.000 đ
Rating: 4.5 (174 reviews)

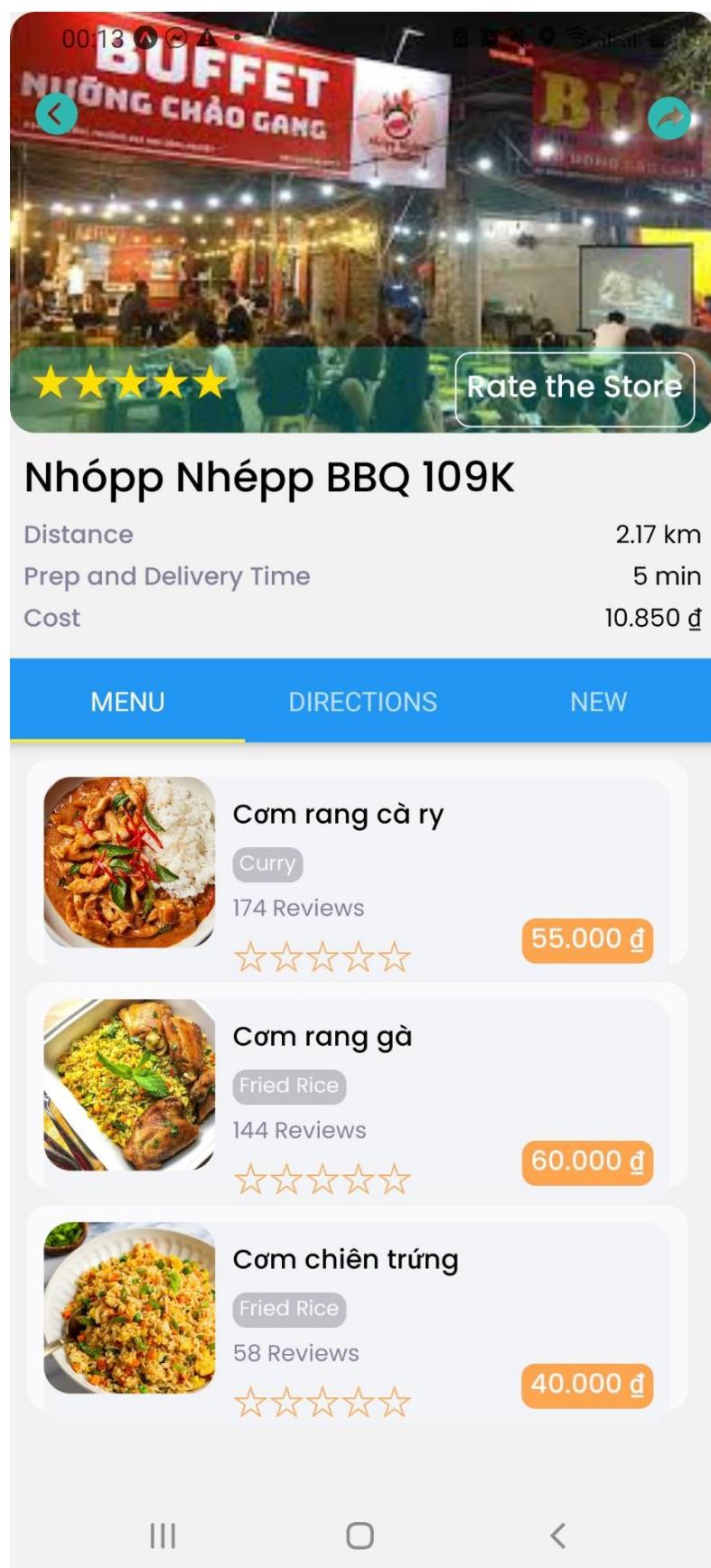
3.6. Màn hình tìm kiếm sản phẩm

3.1.7. Màn hình giỏ hàng



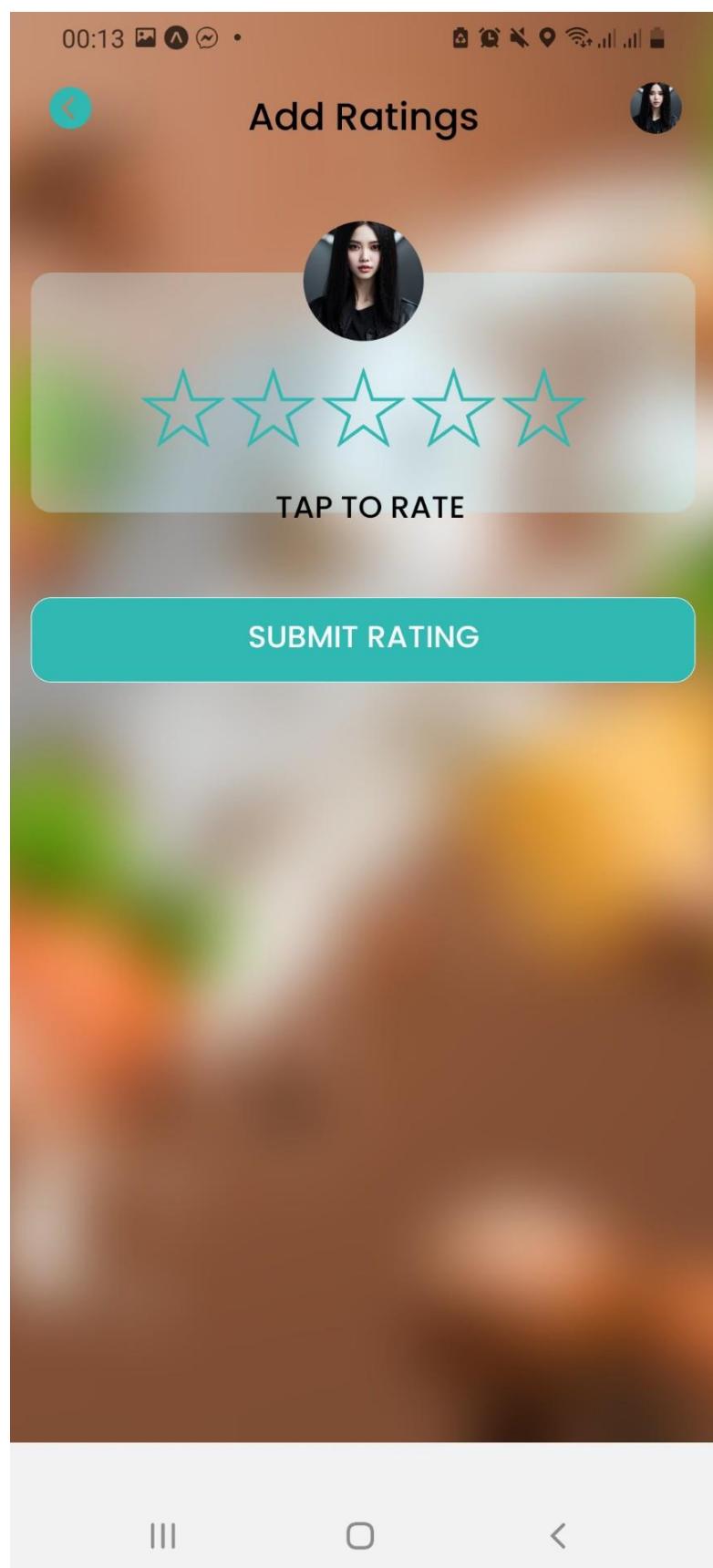
3.7. Màn hình giỏ hàng

3.1.8. Màn hình thông tin nhà hàng



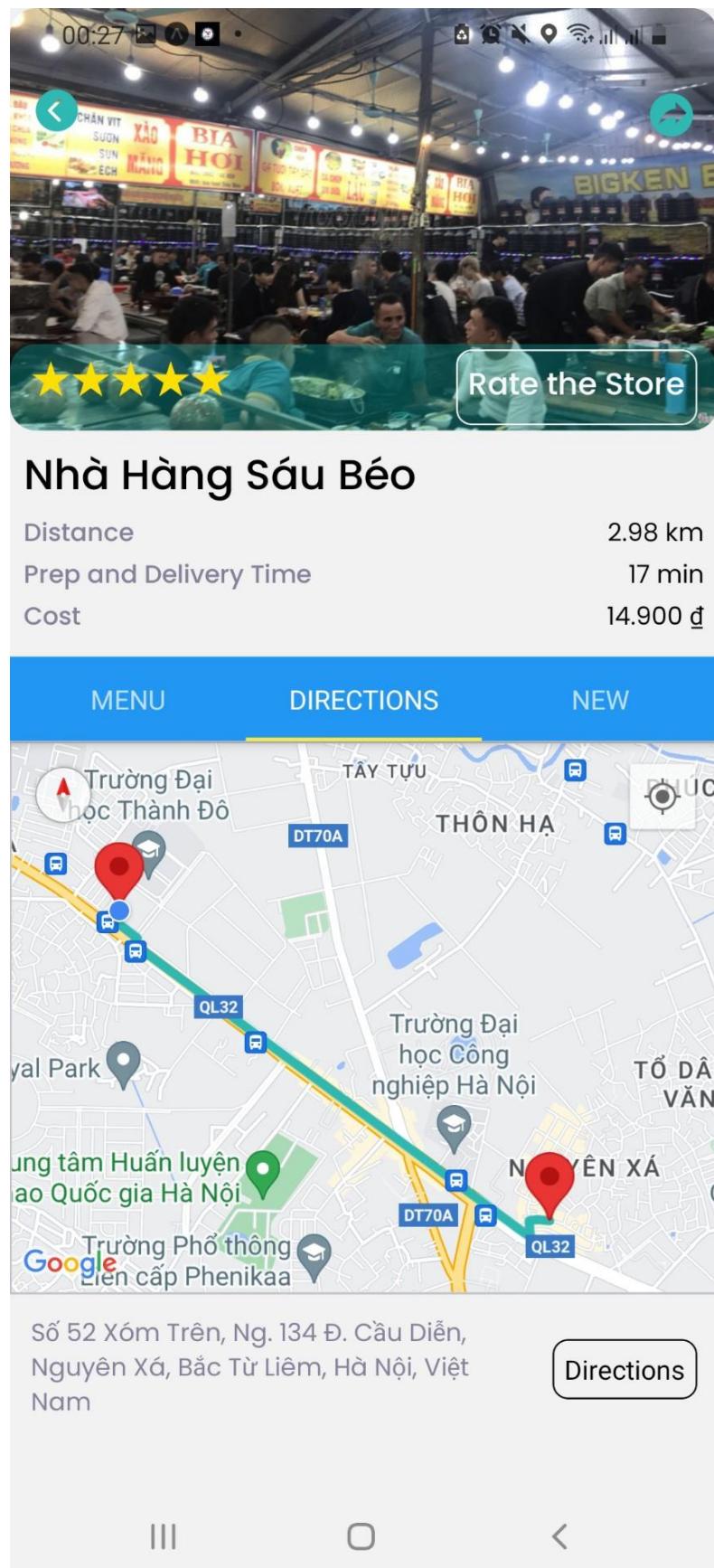
3.8. Màn hình thông tin nhà hàng

3.1.9. Màn hình đánh giá



3.9 Màn hình đánh giá

3.1.10. Màn hình dẫn đường bởi Google Map



3.10. Màn hình dẫn đường bởi Google Map

3.1.11. Màn hình đơn hàng

00:14

My order

Order Date: 9/5/2024
Order Total: 29.002 ₫
Order Status: Placed

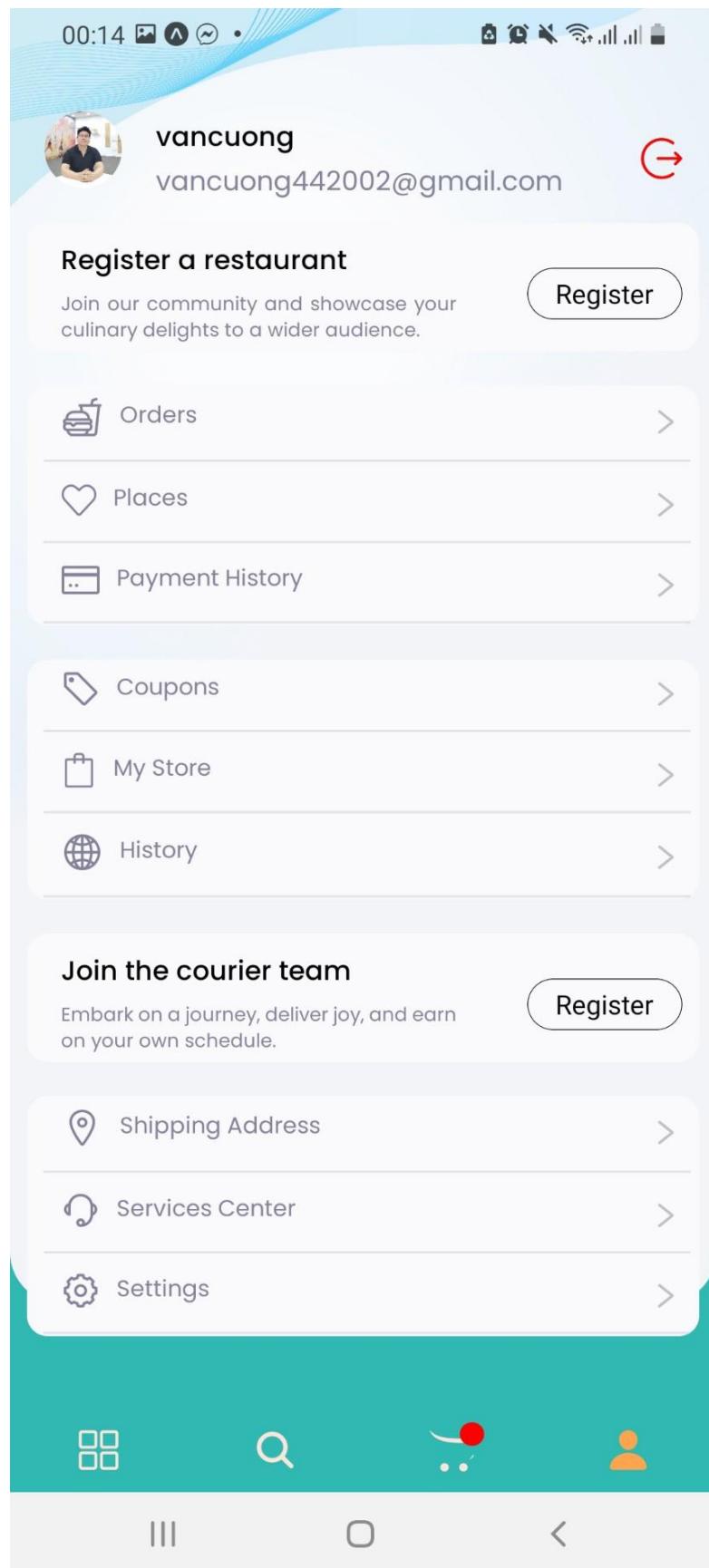
Order Date: 9/5/2024
Order Total: 44.000 ₫
Order Status: Placed

Order Date: 10/5/2024
Order Total: 42.000 ₫
Order Status: Placed

||| □ <

3.11. Màn hình đơn hàng

3.1.12. Giao diện profile



Hình 3.12. Màn hình profile người dùng

3.1.13. Màn hình dashboard

Top Dashboard View:

The top dashboard view features four summary cards and a dual-line chart. The cards show:

- Total views: 125 (0.43% ↑)
- Total Restaurants: 4 (4.35% ↑)
- Total Product: 12 (2.59% ↑)
- Total Account: 3 (0.95% ↓)

The chart compares Total Revenue (blue line) and Total Sales (light blue line) from September 2024 to August 2024. The Y-axis ranges from 0 to 100.

Month	Total Revenue	Total Sales
Sep	~25	~30
Oct	~10	~20
Nov	~20	~30
Dec	~25	~30
Jan	~10	~40
Feb	~20	~35
Mar	~35	~65
Apr	~20	~55
May	~45	~60
Jun	~20	~45
Jul	~30	~40
Aug	~45	~55

Bottom Dashboard View:

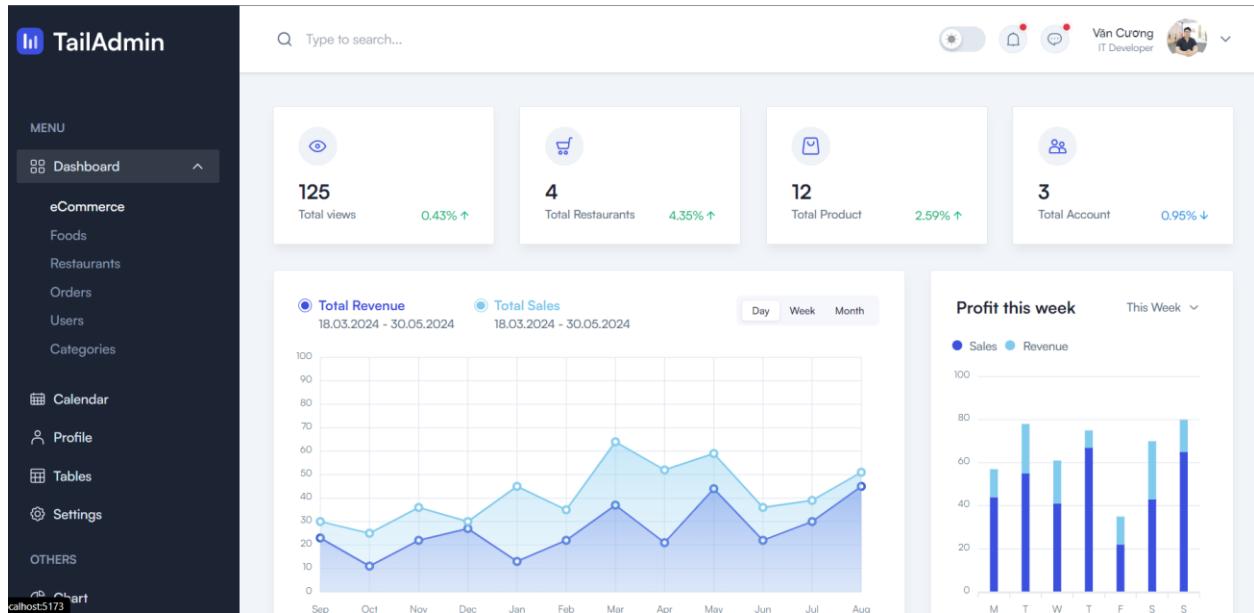
The bottom dashboard view includes the following components:

- Visitors Analytics:** A donut chart showing device usage distribution. Legend: Desktop (Blue), Mobile (Light Blue), Tablet (Dark Blue), Unknown (Teal). Data: Desktop 65%, Mobile 45%, Tablet 34%, Unknown 12%.
- Region Labels:** A map of Vietnam with a callout to Hanoi.
- Top Channels:** A table showing traffic sources and performance metrics. Legend: SOURCE (Google icon), VISITORS (3.5K), REVENUES (\$5,768), SALES (590), CONVERSION (4.8%).
- Chats:** A sidebar showing two active chat conversations.

SOURCE	VISITORS	REVENUES	SALES	CONVERSION
Google	3.5K	\$5,768	590	4.8%

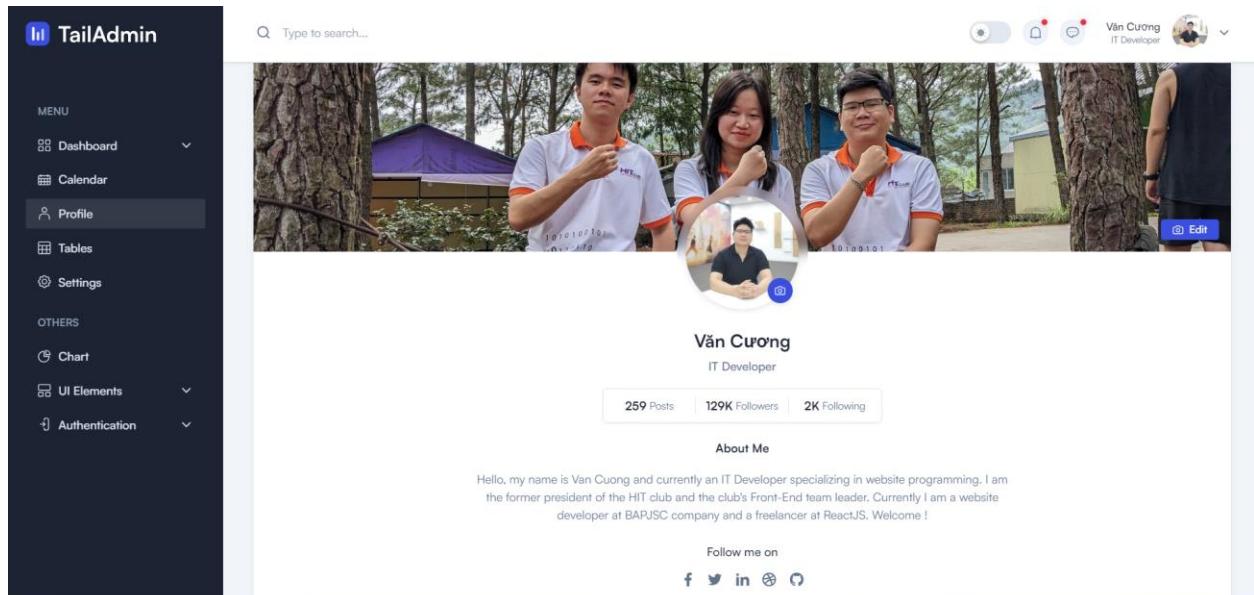
Chats (Visible):

- David Heilo: "How are you?", 12 min ago
- Henry Fisher: "Waiting for you!", 10 min ago



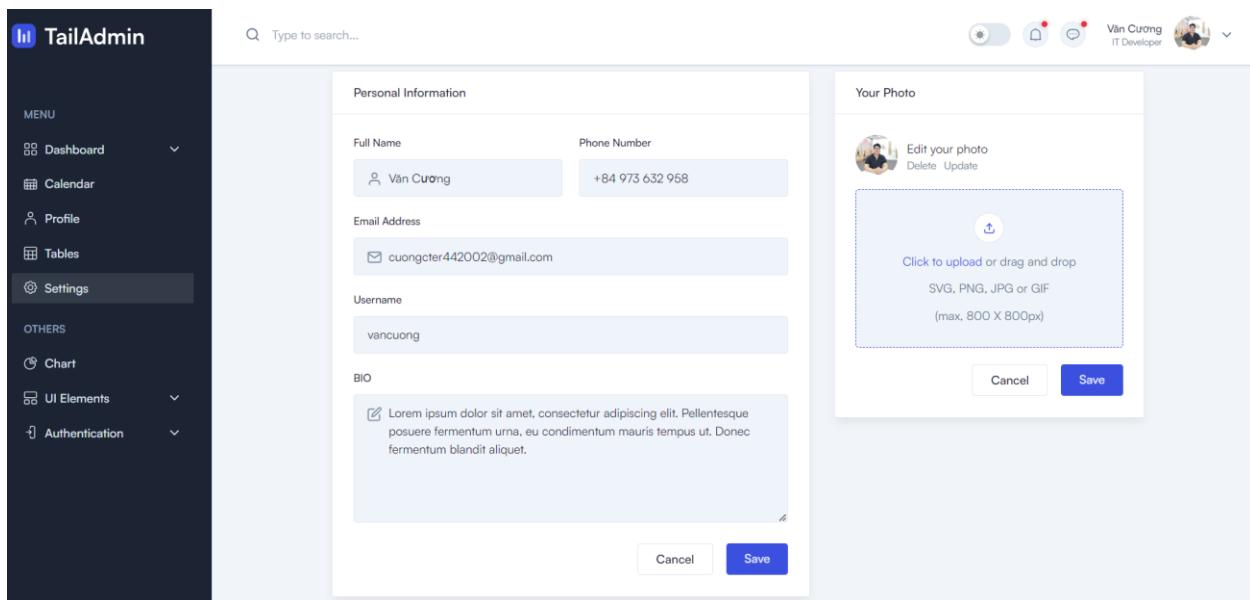
3.13. Màn hình dashboard

3.1.14. Màn hình profile



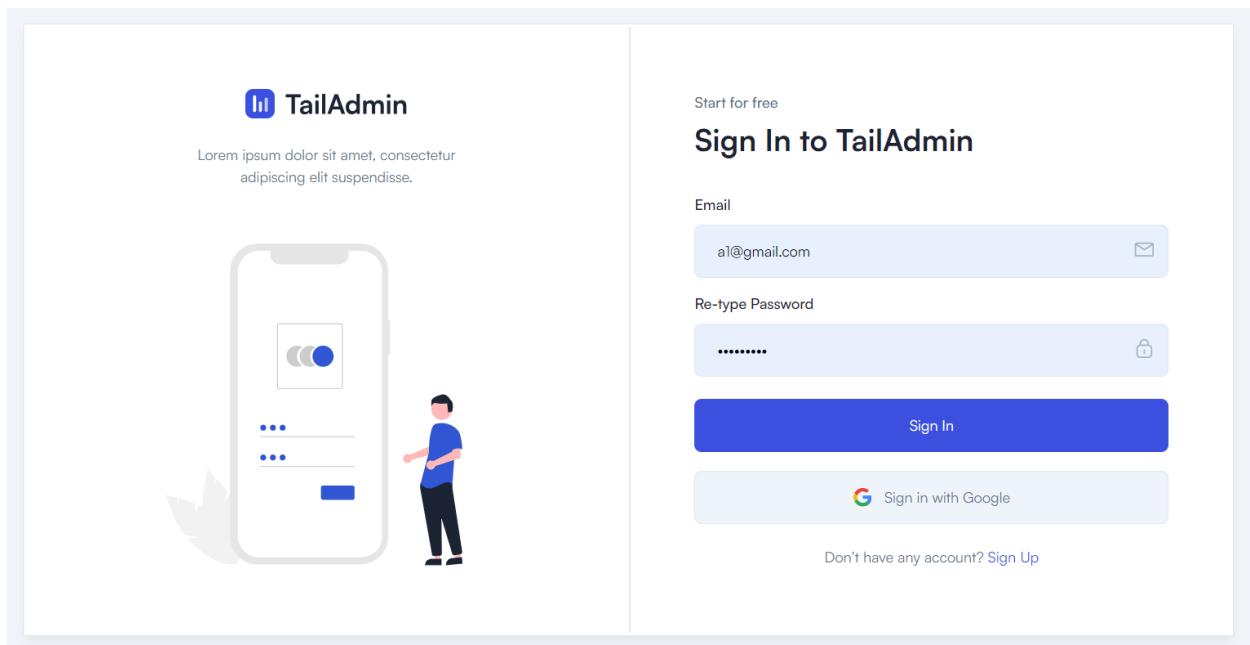
3.14. Màn hình profile

3.1.15. Màn hình cài đặt



3.15. Màn hình setting

3.1.16. Màn hình Authentication Admin Login



3.16. Màn hình Authentication Admin Login

3.1.17. Màn hình Authentication Admin SignUp

Start for free

Sign Up to TailAdmin

Name

Email

Password

Re-type Password

Create account

Sign up with Google

Already have an account? [Sign in](#)

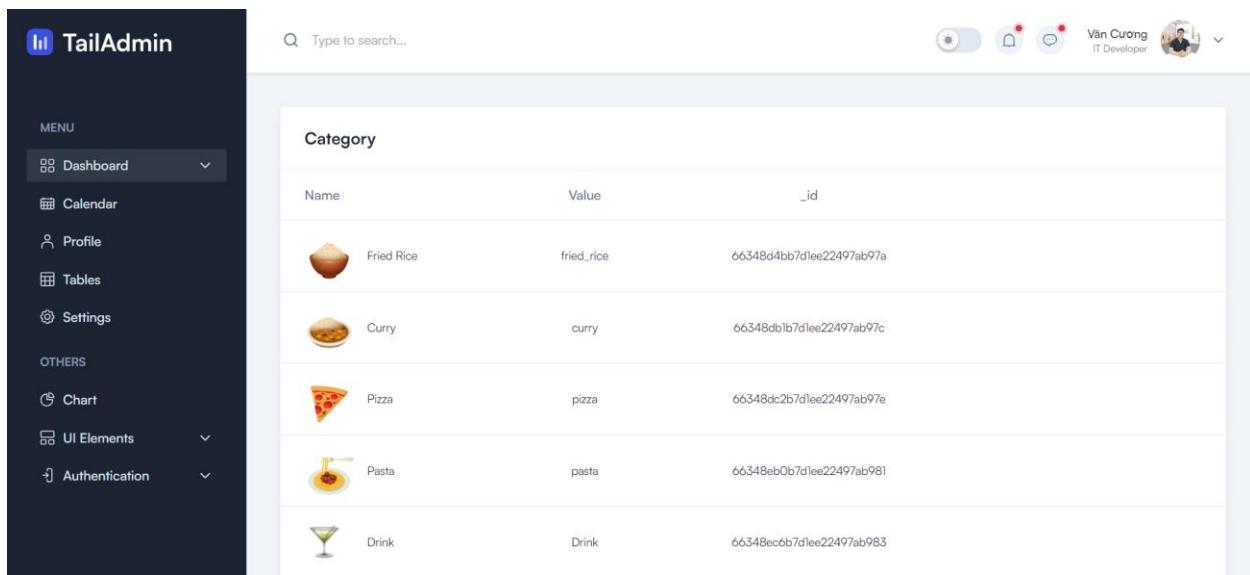
3.17. Màn hình Authentication Admin SignUp

3.1.18. Màn hình sản phẩm

FOODS	FOOD TAGS	PRICE	RATING	COUNT
Cafe đen	Coffee	25.000 ₫	4.9	420
Cafe đá	Coffee	20.000 ₫	4.9	120
Cafe trứng	Coffee	40.000 ₫	4.9	120
Dùi gà chiên xù	Chicken	29.000 ₫	4.6	213
Chân gà rút xương	Chicken	40.000 ₫	4.6	213

3.18. Màn hình thêm mới sản phẩm

3.1.19. Màn hình quản lý danh mục

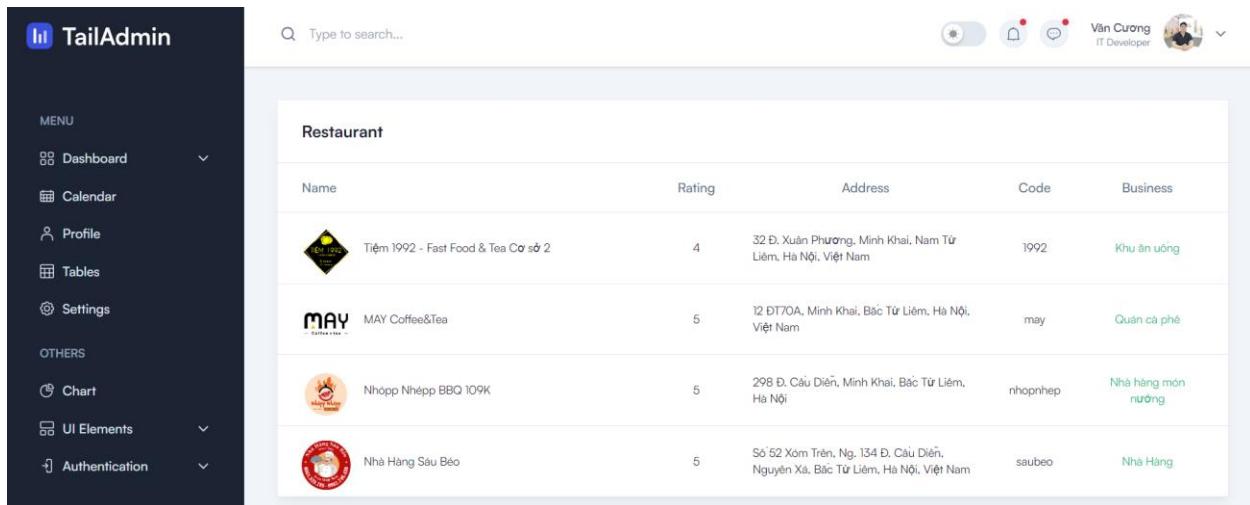


The screenshot shows a dark-themed dashboard titled "TailAdmin". On the left, a sidebar lists various menu items: Dashboard, Calendar, Profile, Tables, Settings, UI Elements, and Authentication. The "Tables" item is currently selected. The main content area has a search bar at the top. Below it is a table titled "Category" with columns: Name, Value, and _id. The table contains five rows of data:

Name	Value	_id
Fried Rice	fried_rice	66348d4bb7d1ee22497ab97a
Curry	curry	66348db1b7d1ee22497ab97c
Pizza	pizza	66348dc2b7d1ee22497ab97e
Pasta	pasta	66348eb0b7d1ee22497ab981
Drink	Drink	66348ec6b7d1ee22497ab983

3.19. Màn hình quản lý danh mục

3.1.20. Màn hình quản lý nhà hàng



The screenshot shows a dark-themed dashboard titled "TailAdmin". The sidebar includes the same menu items as the previous screenshot. The "Tables" item is selected. The main content area displays a table titled "Restaurant" with columns: Name, Rating, Address, Code, and Business. The table lists four restaurants:

Name	Rating	Address	Code	Business
Tiệm 1992 - Fast Food & Tea Cơ sở 2	4	32 Đ. Xuân Phương, Minh Khai, Nam Từ Liêm, Hà Nội, Việt Nam	1992	Khu ăn uống
MAY Coffee&Tea	5	12 ĐT70A, Minh Khai, Bắc Từ Liêm, Hà Nội, Việt Nam	may	Quán cà phê
Nhôpp Nhêpp BBQ 109K	5	298 D. Cầu Diễn, Minh Khai, Bắc Từ Liêm, Hà Nội	nhopnhep	Nhà hàng món nướng
Nhà Hàng Sáu Béo	5	Số 52 Xóm Trên, Ng. 134 D. Cầu Diễn, Nguyễn Xã, Bắc Từ Liêm, Hà Nội, Việt Nam	saubeo	Nhà Hàng

3.20. Màn hình quản lý nhà hàng

3.1.21. Màn hình quản lý đơn hàng

User	Giá trị đơn	Phi giao hàng	Tổng tiền	Địa điểm đến	Trạng thái đơn	Thanh toán
vancuong442002@gmail.com	29000	2	29002	298 Đ.Cầu Diễn	Placed	Pending
vancuong442002@gmail.com	40000	4000	44000	298 Đ.Cầu Diễn	Placed	Pending
vancuong442002@gmail.com	38000	4000	42000	298 Đ.Cầu Diễn	Placed	Pending

3.21. Màn hình quản lý đơn hàng

3.1.22. Màn hình quản lý người dùng

Username	Email	userType
CuongCter	cuongcter442002@gmail.com	Client
vancuong	vancuong442002@gmail.com	Client
admin	admin@gmail.com	Admin

3.22. Màn hình quản lý người dùng

3.2. Kiểm thử hệ thống

3.2.1. Kiểm thử chức năng

Bảng 3.23. Bảng kiểm thử hệ thống

STT	Chức năng	Trường hợp kiểm thử	Đầu vào	Đầu ra thực tế	Kết quả
1	Xác thực người dùng	Đăng nhập tài	Nhập tài khoản	Chuyển hướng sang trang dashboard	Pass
			Nhập dữ liệu	Hiển thị thông báo lỗi	Pass

		khoản	không hợp lệ		
2	Quy trình quản lý sản phẩm	Đăng xuất tài khoản	Click nút “Logout”	Chuyển hướng trang đăng nhập	Pass
			Nhập dữ liệu không hợp lệ	Hiển thị thông báo lỗi	Pass
		Thêm sản phẩm	Nhập dữ liệu hợp lệ	Thêm mới tài liệu và hiển thị danh sách tài liệu	Pass
			Nhập dữ liệu không hợp lệ	Hiển thị thông báo lỗi	Pass
		Update thông tin sản phẩm	Click vào nút “Edit” tại màn hình danh sách sản phẩm	Hiển thị thông tin sản phẩm sau khi cập nhật tại màn hình danh sách người dùng.	Pass
		Xoá sản phẩm	Click nút Xóa trên màn hình	+ Hệ thống hiển thị dialog xác nhận xoá + Hệ thống hiển thị thông báo xoá thành công	Pass
3	Quản lý người dùng	Thêm mới người dùng	Nhập dữ liệu hợp lệ	Hiển thị người dùng mới tại màn hình danh sách người dùng	Pass
			Nhập dữ liệu không hợp lệ	Hiển thị thông báo lỗi	Pass
		Update thông tin người dùng	Nhập dữ liệu hợp lệ.	Hiển thị thông tin người dùng sau khi cập nhật tại màn hình danh sách người dùng.	Pass
		Xoá người dùng	Click “Delete” tại màn hình danh sách người dùng + click nút “OK” để xác nhận xoá	+ Hệ thống hiển thị dialog xác nhận xoá + Hệ thống hiển thị thông báo xoá thành công	Pass

3.2.2. Kiểm thử tích hợp

STT	Tích hợp	Trường hợp kiểm thử	Đầu vào	Đầu ra thực tế	Kết quả
1	Google Maps API	Hiển thị Google Map	API Key	Hiển thị bản đồ thực tế	Pass
			Dữ liệu không hợp lệ	Hiển thị thông báo lỗi	Pass
		Hiển thị hệ thống dẫn đường	Kinh độ vĩ độ GPS	Đường line dẫn đường và tính toán đường đi ngắn nhất	Pass
			Dữ liệu không hợp lệ	Hiển thị thông báo lỗi	Pass
2	Fire base	Đăng ký tài khoản	Dữ liệu hợp lệ	Dữ liệu đưa lên cơ sở dữ liệu của Firebase	Pass
			Dữ liệu email đã tồn tại	Hiển thị thông báo lỗi	Pass
		Đăng nhập tài khoản	Dữ liệu hợp lệ	Kiểm tra với CSDL Firebase rồi điều hướng	Pass
			Dữ liệu không hợp lệ	Hiển thị thông báo lỗi	Pass

TÀI LIỆU THAM KHẢO

- [1]. Trường Đại học Công nghiệp Hà Nội. (1/2023). Bài giảng điện tử, Nhập môn công nghệ phần mềm, Tổng quan về công nghệ phần mềm.
- [2]. Trường Đại học Công nghiệp Hà Nội. (1/2023). Bài giảng điện tử, Nhập môn công nghệ phần mềm, Thiết kế phần mềm.
- [3]. React Native Development Environment. (Truy cập ngày: 18/03/2024).
<https://reactnative.dev/>
- [4]. NodeJS Document. (Truy cập ngày: 18/03/2024). <https://nodejs.org/en>
- [5]. Express Document. (Truy cập ngày: 20/03/2024). <https://expressjs.com/>
- [6]. Google Maps API. (Truy cập ngày: 20/03/2024). Tài liệu của Google
<https://developers.google.com/maps/documentation?hl=vi>
- [7]. Firebase. (Truy cập ngày: 28/03/2024). Tài liệu nghiên cứu của Google
<https://firebase.google.com/>
- [8]. Nguyễn Thị Hoa - Tài liệu tham khảo và chi tiết về Google Maps API. (Truy cập ngày: 20/03/2024). <https://viblo.asia/p/tim-hieu-ve-google-map-api-ZWApGxJ3R06y> Tài liệu của Google Cloud – API Google.
- [9].Google Doc - Tài liệu về Google Firebase (Truy cập ngày: 28/03/2024).
<https://firebase.google.com/docs?hl=vi>
- [10].Expo - Tài liệu tham khảo về Expo (Truy cập ngày: 28/03/2024).
<https://docs.expo.dev/>
- [12].GPS - Tài liệu GPS tọa độ Google Map(Truy cập ngày: 29/03/2024).
<https://support.google.com/maps/answer/18539?hl=en&co=GENIE.Platform%3DAndroid>

