TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN

Công nghệ phần mềm

Ứng dụng Đặt hàng & Nhà hàng

GVHD: Từ Lãng Phiêu
SV: Trần Duy Hoành - 3122410132
Trần Ngô Nhật Nam - 3122410253
Nguyễn Ngọc Thùy Trân - 3119480093
Nguyễn Huỳnh Phương Uyên - 3122410461

TP. HỒ CHÍ MINH, THÁNG 4/2025

Mục lục

1 Khai thác yêu cầu

1.1

1.1.1 Xác định bối cảnh của dự án

Trong thời đại công nghệ số hiện nay, lúc mà ai ra đường cũng có thể dễ dàng sở hữu tối thiểu một chiếc smart phone. Đó là lúc các nhà hàng đang có xu hướng ứng dụng hệ thống phần mềm để quản lý hoạt động hiệu quả hơn, giảm tải công việc cho nhân viên, đồng thời nâng cao trải nghiệm cho khách hàng. Dự án này hướng đến việc xây dựng một hệ thống quản lý nhà hàng với các chức năng chính như: đặt bàn, gọi món, theo dõi chế biến, thanh toán, và quản lý thực đơn. Hệ thống được thiết kế để phục vụ cả khách hàng lẫn nhân viên trong việc xử lý nhanh chóng, chính xác các hoạt động thường ngày tại nhà hàng.

1.1.2 Stock holder

- Khách hàng : đối tượng mà nhà hàng phục vụ.
- Nhân viên : là nhân viên tư vấn cho khách hàng, xác nhận đặt món từ
- khác hàng (trực tiếp hoặc gián tiếp).
- Đầu bếp : nhận thông tin đặt món từ nhân viên.
- Quản lý : là người điều hành chính nhà hàng.
- Chủ nhà hàng : là người sở hữu nhà hàng. Trả tiền để điều hành nhà hàng.
- Bảo vệ : là người bảo vệ nhà hàng.
- Bộ phận Công nghệ Thông tin : là người chỉnh sửa hệ thống, bảo trì hệ thống,...

1.1.3 Kỳ vọng thực hiện được

- Tự động hóa quy trình đặt bàn và gọi món
- Rút ngắn thời gian phục vụ, giảm sai sót do thao tác thủ công
- Cung cấp công cụ thống kê, báo cáo hỗ trợ quản lý ra quyết định
- Tăng sự hài lòng và trải nghiệm của khách hàng
- Giảm thiểu chi phí vận hành lâu dài

1.1.4 Phạm vi của dự án

- Hệ thống đặt bàn trực tuyến
- Gọi món từ thiết bị.
- Cập nhật trạng thái đơn hàng và món ăn
- Thanh toán online hoặc tại quầy
- Quản lý thực đơn, nguyên liệu, và nhân viên
- Giao diện cho các đối tượng: khách, nhân viên, bếp, quản lý
- Không tích hợp với hệ thống giao hàng từ bên ngoài

1.2

1.2.1 Chức năng

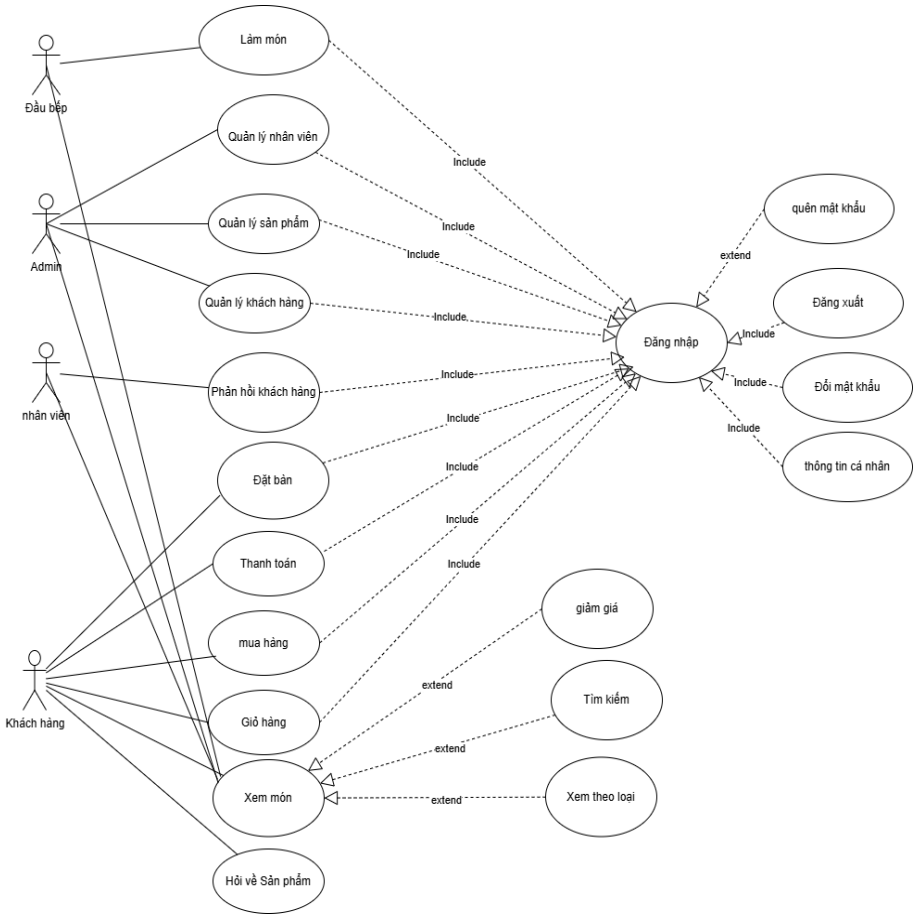
- **Khách hàng**
  - Xem, sửa, xoá các sản phẩm trong giỏ hàng.
  - Đặt hàng tại nhà và hẹn giờ mang đi. Khách hàng có thể:
    - Tự đến lấy.
    - Nhờ người khác lấy hộ.
  - Hủy đơn hàng trong thời gian cho phép.
  - Đánh giá sản phẩm sau khi mua.
- **Admin**
  - Thống kê doanh thu.
  - Thống kê sản phẩm đã bán theo:
    - Ngày.
    - Tháng.
    - Năm.
  - Quản lý thông tin:
    - Thêm, sửa, xoá nhân viên.
    - Thêm, sửa, xoá khách hàng.
    - Thêm, sửa, xoá sản phẩm.
- **Đầu bếp**
  - Khi nhân viên báo cáo làm món, phản hồi thời gian dự kiến hoàn thành món.
  - Kiểm tra món còn/tồn kho và món hết để thông báo cho khách hàng.
- **Nhân viên**
  - Giao tiếp, tư vấn cho khách hàng.
  - Khi khách đặt món, lên danh sách món và gửi cho đầu bếp.
- **Đăng nhập/Đăng ký**
  - Kiểm tra đầu vào hợp lệ.
  - Hỗ trợ liên kết tài khoản qua Gmail hoặc Facebook.
- **Đặt bàn**
  - Khách hàng có thể đặt bàn từ xa, tại nhà.

1.2.2 Yêu cầu phi chức năng

- **Hiệu suất (Performance)**
  - **Tốc độ tải trang:**
    - Khi người dùng truy cập, trang web phải tải tối đa trong vòng 2 giây.
    - Hình ảnh trên trang chủ (Home) phải tải nhanh, không bị trễ.
    - Khi người dùng thêm sản phẩm vào giỏ hàng, thời gian phản hồi không quá 1.5 giây.
  - **Số lượng truy cập đồng thời:**
    - Hệ thống phải đảm bảo hoạt động mượt mà với ít nhất 100 người dùng truy cập cùng lúc.
    - Trong các khung giờ cao điểm (ăn trưa, ăn tối), hệ thống không được xảy ra giật, lag gây khó chịu cho khách hàng.
- **Tính bảo mật (Security)**
  - Không cung cấp hoặc bán thông tin người dùng.
  - Sử dụng cổng thanh toán an toàn và bảo mật.
  - Thông tin khách hàng phải được giữ bí mật tuyệt đối, chỉ người dùng mới có quyền truy cập vào thông tin của họ.
- **Tính sẵn sàng (Availability)**
  - Hệ thống phải hoạt động liên tục 24/7.
  - Phải đảm bảo hoạt động ổn định trong các khung giờ cao điểm.
- **Khả năng bảo trì (Maintainability)**
  - Thời gian bảo trì hệ thống để nâng cấp mã nguồn không được vượt quá 1 giờ.
- **Khả năng mở rộng (Scalability)**
  - Khi số lượng người dùng tăng lên, hệ thống vẫn phải đảm bảo hoạt động mượt mà và ổn định.
- **Khả năng sử dụng (Usability)**
  - Giao diện người dùng phải dễ sử dụng, dễ hiểu và thân thiện với người dùng.

1.3

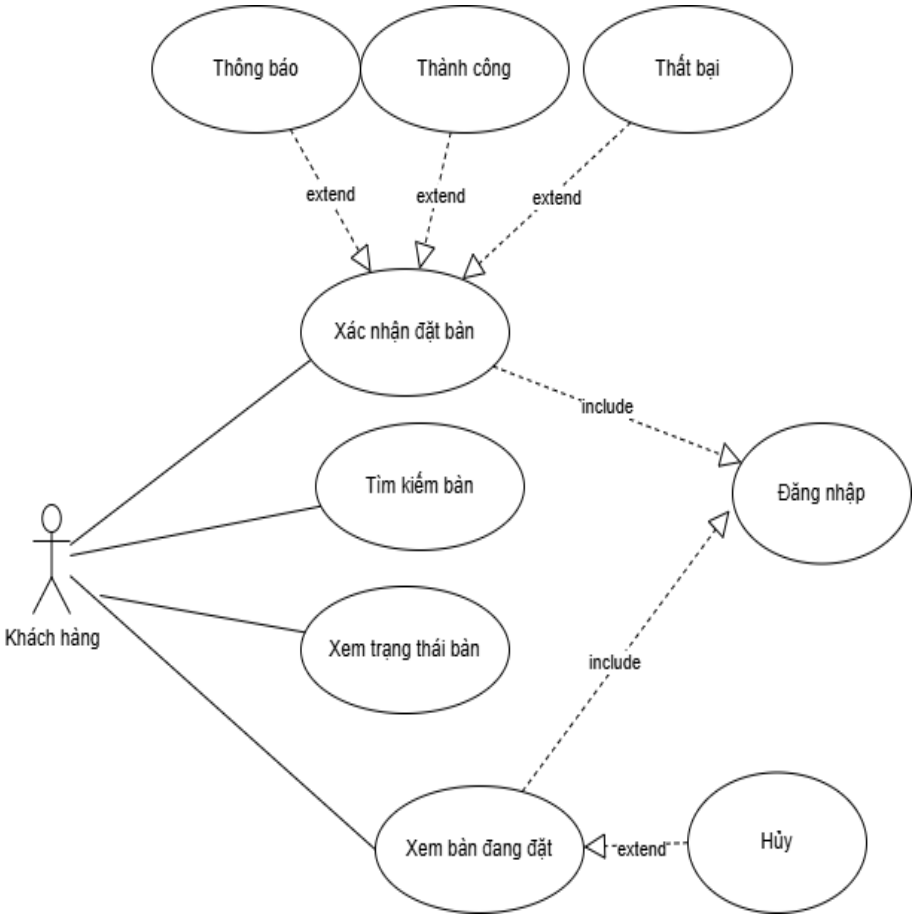
1.3.1 Sơ đồ use-case cho toàn hệ thống



Hình 1: Sơ đồ use case cho toàn hệ thống

1.3.2

1.3.2.a Sơ đồ use-case của chức năng đặt bàn



Hình 2: Sơ đồ use case chức năng đặt bàn

1.3.2.b Bảng mô tả sơ đồ use-case: Đặt bàn

Thuộc tính	**Chi tiết**
Tên use-case	Đặt bàn
Mô tả	Cho phép khách hàng đặt bàn online
Actor chính	Khách hàng
Actor phụ	Không có
Tiền điều kiện	Có tài khoản
Hậu điều kiện	Bàn được đặt thành công, thông tin được lưu, bàn chuyển trạng thái "đã đặt".

1.3.3 Luồng sự kiện chính

Khách hàng	**Hệ thống**
1. Chọn đặt bàn	2. Hiển thị giao diện đặt bàn
3. Tìm bàn	4. Kiểm tra bàn còn trống
	5. Hiển thị thông tin bàn còn trống
6. Chọn bàn	7. Hiển thị form nhập thông tin đặt bàn
	8. Yêu cầu khách hàng nhập thông tin (ngày/giờ dùng bàn, số lượng người, sđt hoặc email khác với tài khoản nếu muốn...)
9. Nhập thông tin	10. Lưu tạm thông tin đặt bàn vào local storage
	11. Kiểm tra thông tin đặt bàn
12. Đặt bàn	13. Kiểm tra bàn ở thời điểm hiện tại
	14. Gửi thông báo đặt bàn thành công, cập nhật thông tin bàn đã được đặt lên CSDL

1.3.4 Luồng sự kiện thay thế

	4.1 Hệ thống thông báo hết bàn, gợi ý khách hàng quay lại vào thời gian khác.
4.2 Khách hàng trở về trang chủ của hệ thống.	
	8.1 Dùng thông tin có sẵn trong local storage.
	12.1 Hệ thống thông báo bàn vừa được khách khác đặt.
12.2 Khách hàng quay về bước 6.	12.3 Hệ thống quay về bước 5.

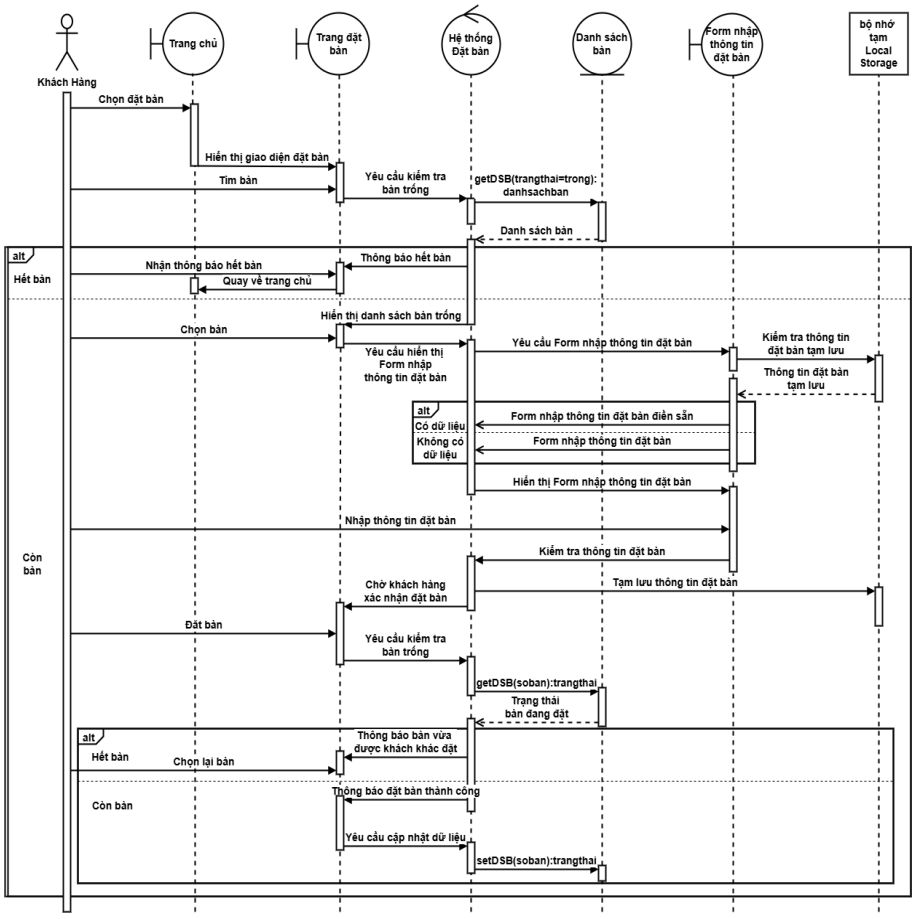
2 Mô hình hóa hệ thống

2.1 Sơ đồ hoạt động cho toàn hệ thống

2.2 Sơ đồ trình tự cho chức năng đặt bàn

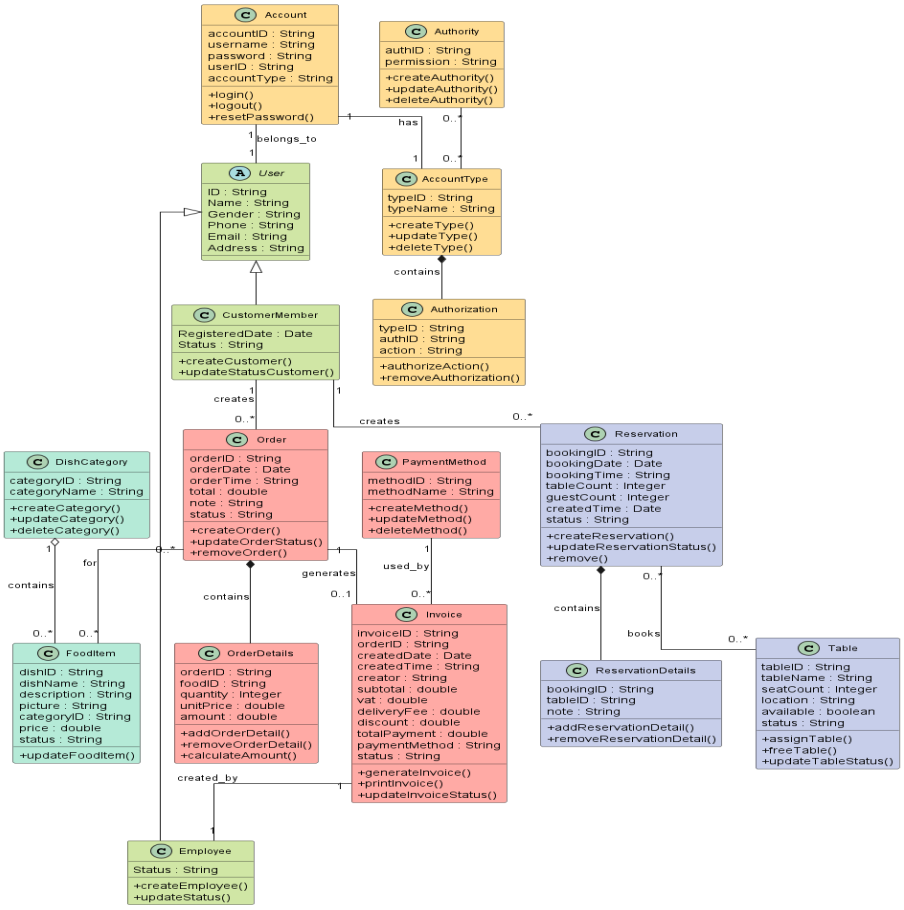
2.2.1 Phân tích các thành phần theo mô hình UML

Thành phần	**Mô tả**
Actor	Khách hàng
Boundary	Giao diện trang chủ, giao diện đặt bàn, form điền thông tin đặt bàn
Control	Control đặt bàn
Entity	Danh sách bàn



Hình 3: Sơ đồ trình tự cho chức năng đặt bàn

2.3 Sơ đồ lớp cho toàn hệ thống



Hình 4: Sơ đồ lớp cho toàn hệ thống

3 Thiết kế kiến trúc

3.1 Kiến trúc hệ thống - Modular MVC + Dependency Injection (Clean Architecture-oriented MVC)

3.1.1 Mô hình MVC là gì?

MVC (Model – View – Controller) là kiến trúc phần mềm phổ biến trong các ứng dụng web truyền thống, giúp phân tách rõ ràng giữa giao diện, logic và dữ liệu.

- **Model (M):** Xử lý logic nghiệp vụ và tương tác với cơ sở dữ liệu.
- **View (V):** Hiển thị dữ liệu cho người dùng, thường dưới dạng HTML.
- **Controller (C):** Nhận yêu cầu từ client, điều phối giữa Model và View.

Trong mô hình này, cả ba thành phần đều nằm trong backend. Backend vừa xử lý logic vừa render giao diện, dẫn đến sự phụ thuộc chặt giữa xử lý và hiển thị.

3.1.2 Khái quát

Module MVC kết hợp Dependency Injection (DI) là kiến trúc hiện đại được NestJS áp dụng nhằm xây dựng backend theo hướng tách biệt, linh hoạt và dễ mở rộng.

Thay vì để backend xử lý cả giao diện như mô hình MVC truyền thống, kiến trúc này xem backend như một **API Provider** – chỉ xử lý logic nghiệp vụ và trả về dữ liệu (thường là JSON) cho frontend (web, mobile, IoT,...) hiển thị.

3.1.3 Thành phần chính

Hệ thống Module MVC + DI trong NestJS vẫn giữ các thành phần tương tự mô hình MVC truyền thống nhưng được điều chỉnh phù hợp với cơ chế cung cấp API:

Thành phần NestJS	Vai trò trong MVC
Controller	Controller – tiếp nhận request, định tuyến
Service	Model – xử lý logic nghiệp vụ
JSON response	View – dữ liệu trả về cho client

Đặc điểm nổi bật:

- Tách biệt frontend – backend: Giao diện do frontend xử lý, backend chỉ cung cấp dữ liệu.
- Tổ chức hệ thống theo module domain như `user`, `auth`, `product`,... thay vì chia theo tầng.
- Hỗ trợ mạnh mẽ DI (Dependency Injection): tự động cung cấp các phụ thuộc (service, repository,...) giúp mã nguồn gọn gàng, dễ mở rộng, bảo trì và kiểm thử.

3.2 Công nghệ sử dụng

3.2.1 React

React là thư viện JavaScript mã nguồn mở dùng để xây dựng giao diện người dùng (UI) theo hướng component.

React giúp phát triển UI linh hoạt, tái sử dụng và dễ mở rộng, đặc biệt phù hợp với ứng dụng đơn trang (SPA).

3.2.2 NestJS – Backend Framework (Node.js + TypeScript)

NestJS là một Framework Node.js hiện đại dựa trên TypeScript, tận dụng Dependency Injection (DI) và tổ chức module hóa mạnh mẽ, hỗ trợ tích hợp với ORM/ODM như TypeORM, Prisma, Mongoose.

Đặc điểm nổi bật khi kết hợp React và NestJS:

- Tách biệt rõ ràng frontend và backend: React tập trung vào UI, còn NestJS xử lý logic, API, database, authentication.
- Modular hóa tốt ở cả hai phía: NestJS tổ chức theo module, React tổ chức bằng `components/`, `pages/`, `context/`.

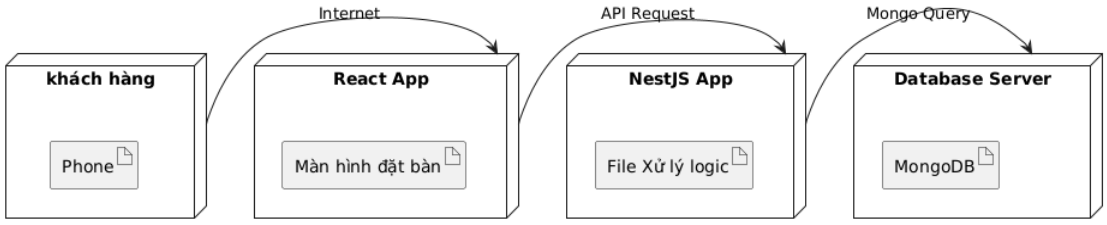
3.2.3 Database – MongoDB & Mongoose

MongoDB là hệ quản trị cơ sở dữ liệu NoSQL, lưu trữ dữ liệu dưới dạng document (tài liệu) theo định dạng JSON. Rất phù hợp với mô hình domain-driven design.

Mongoose là thư viện ORM/ODM cho MongoDB trong môi trường Node.js. Cho phép định nghĩa Schema linh hoạt, kiểm tra dữ liệu, middleware, tối ưu hiệu suất giúp API trả dữ liệu nhanh, chính xác.

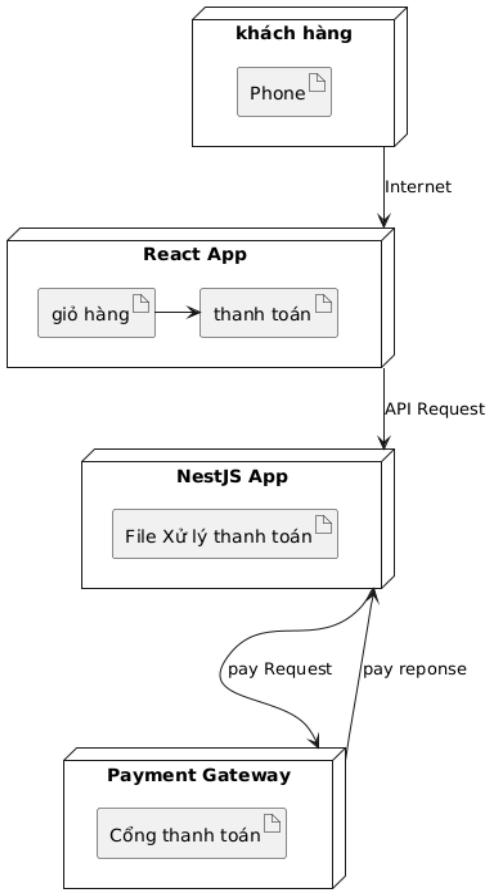
3.3 Sơ đồ triển khai cho các yêu cầu chức năng chính

3.3.1 sơ đồ triển khai cho các yêu cầu chức năng Đặt bàn



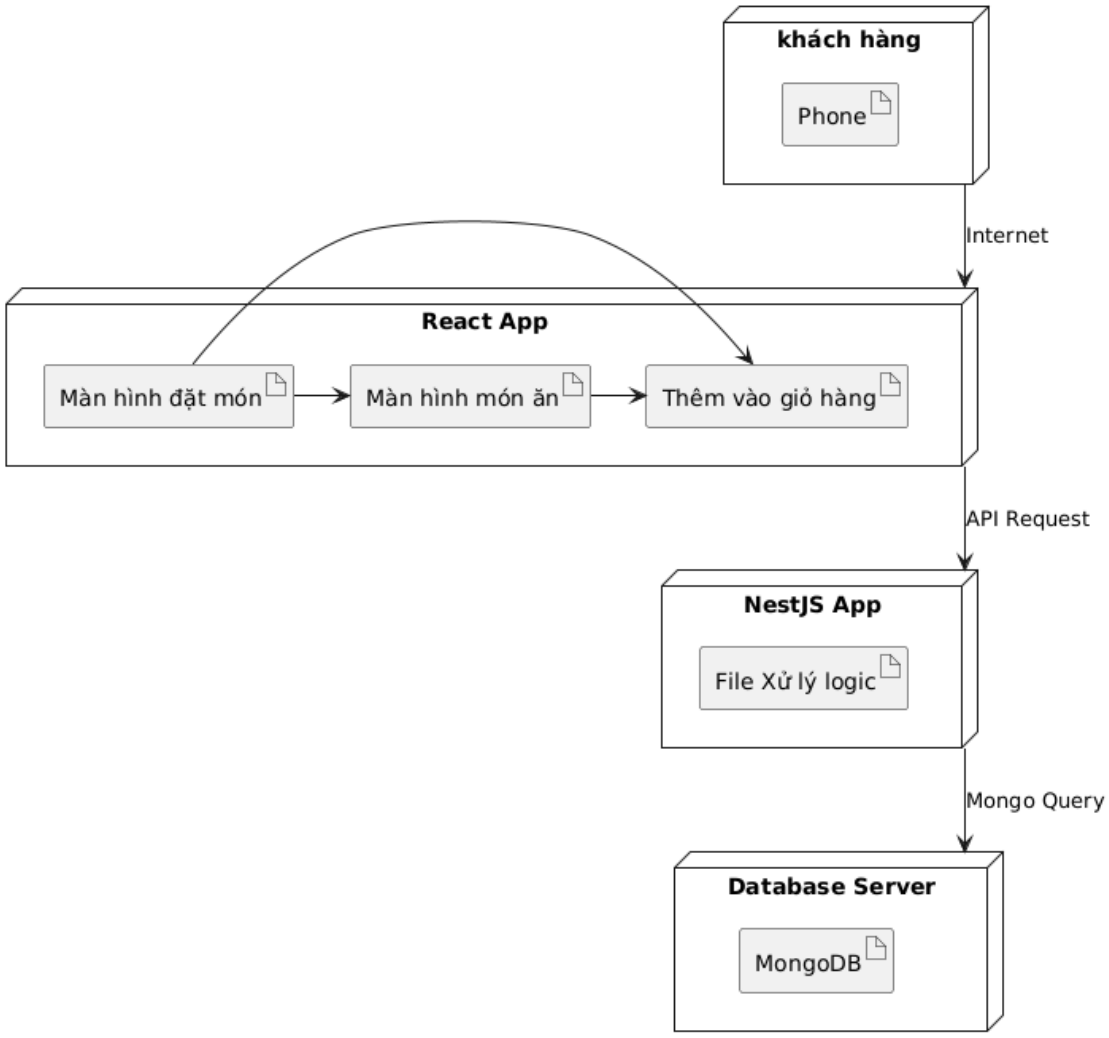
Hình 5: sơ đồ triển khai cho các yêu cầu chức năng Đặt bàn

3.3.2 sơ đồ triển khai cho các yêu cầu chức năng Thanh toán trực tuyến



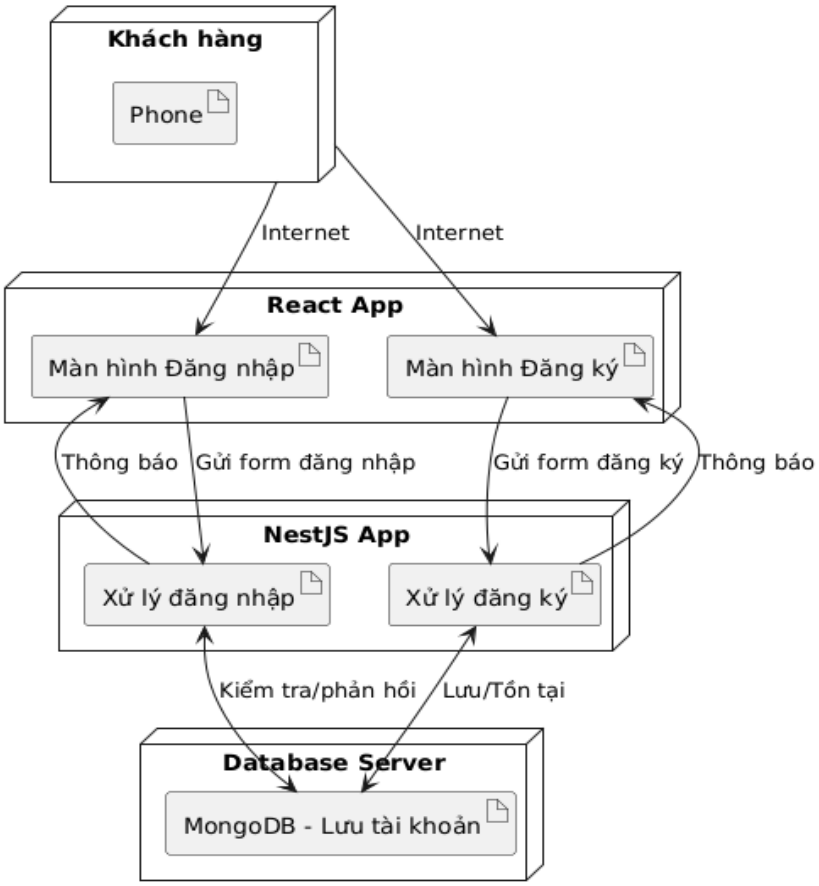
Hình 6: sơ đồ triển khai cho các yêu cầu chức năng Thanh toán trực tuyến

3.3.3 sơ đồ triển khai cho các yêu cầu chức năng Đặt món



Hình 7: sơ đồ triển khai cho các yêu cầu chức năng Đặt món

3.3.4 sơ đồ triển khai cho các yêu cầu chức năng đăng nhập, đăng ký



Hình 8: sơ đồ triển khai cho các yêu cầu chức năng đăng nhập, đăng ký

4 Triển khai - Sprint 1

4.1 Kho lưu trữ github

https://github.com/hoanhviplengend/Assigment_CNPM

4.2 Thêm Tài Liệu và Thư Mục

Cấu trúc thư mục trong kho lưu trữ

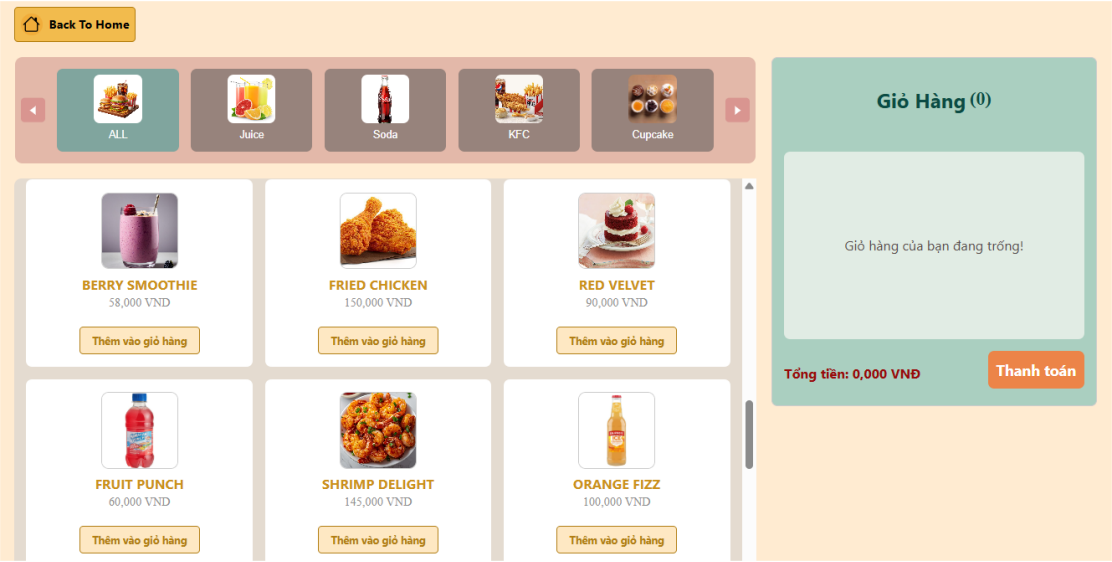
- **requirements/**: Chứa các tài liệu yêu cầu hệ thống.
- **system-modeling/**: Chứa các sơ đồ mô hình hệ thống.
- **architecture-design/**: Chứa các tài liệu và sơ đồ thiết kế kiến trúc.

4.3 Triển Khai MVP cho Màn Hình Menu

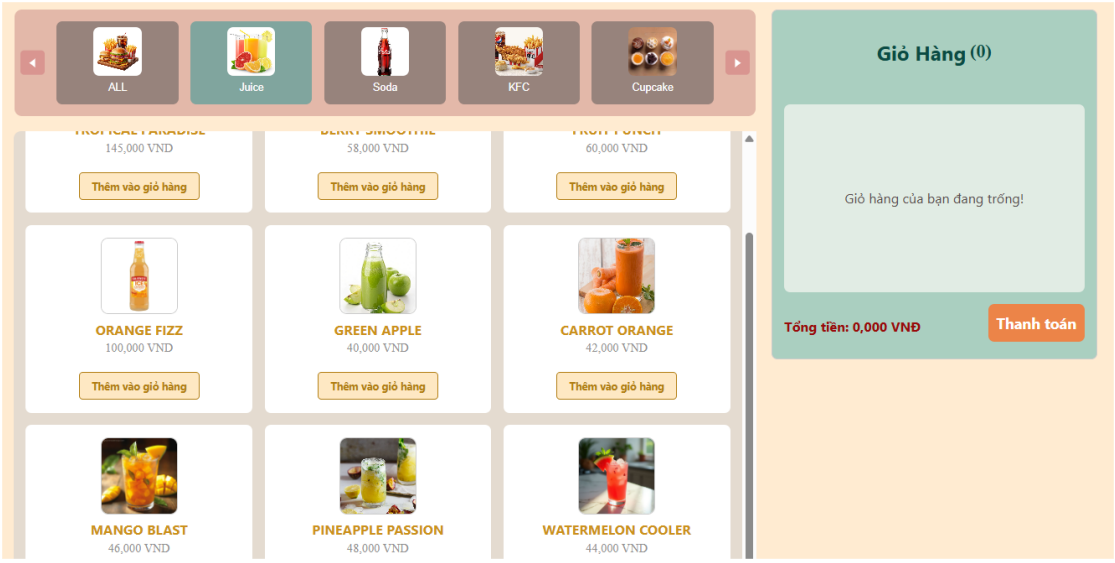
MVP (Minimum Viable Product) (Sản phẩm khả thi tối thiểu). Đây là một phiên bản ban đầu của sản phẩm với những tính năng cốt lõi, đủ để đáp ứng nhu cầu cơ bản của người dùng và có thể đưa ra thị trường hoặc cho khách hàng sử dụng thử nghiệm.

MVP thường được sử dụng trong phát triển phần mềm hoặc khởi nghiệp để nhanh chóng kiểm chứng ý tưởng mà không cần phải phát triển một sản phẩm hoàn thiện

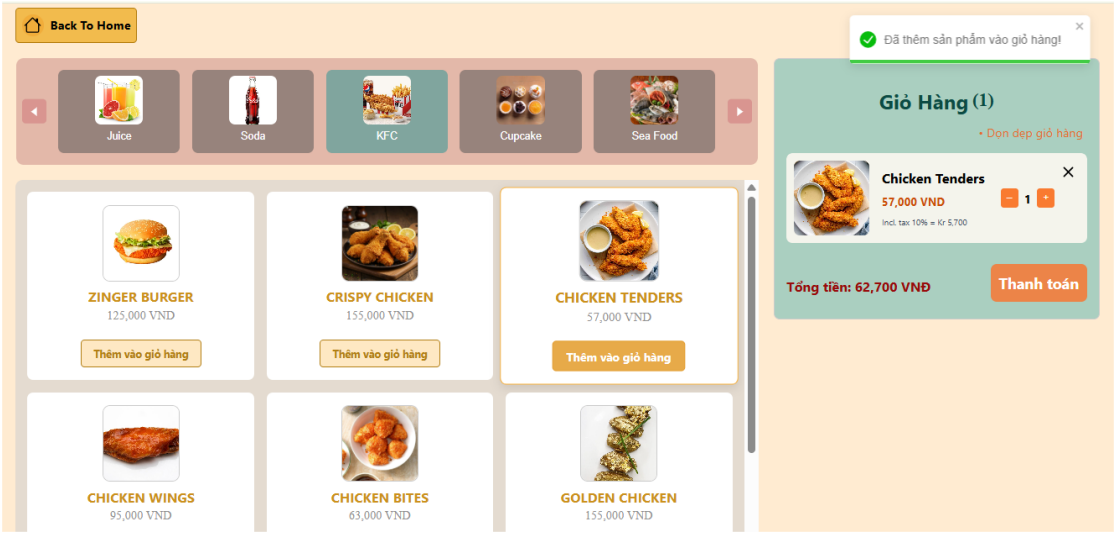
- **Chức năng cơ bản:** Hiển thị các lựa chọn món ăn cho khách hàng.



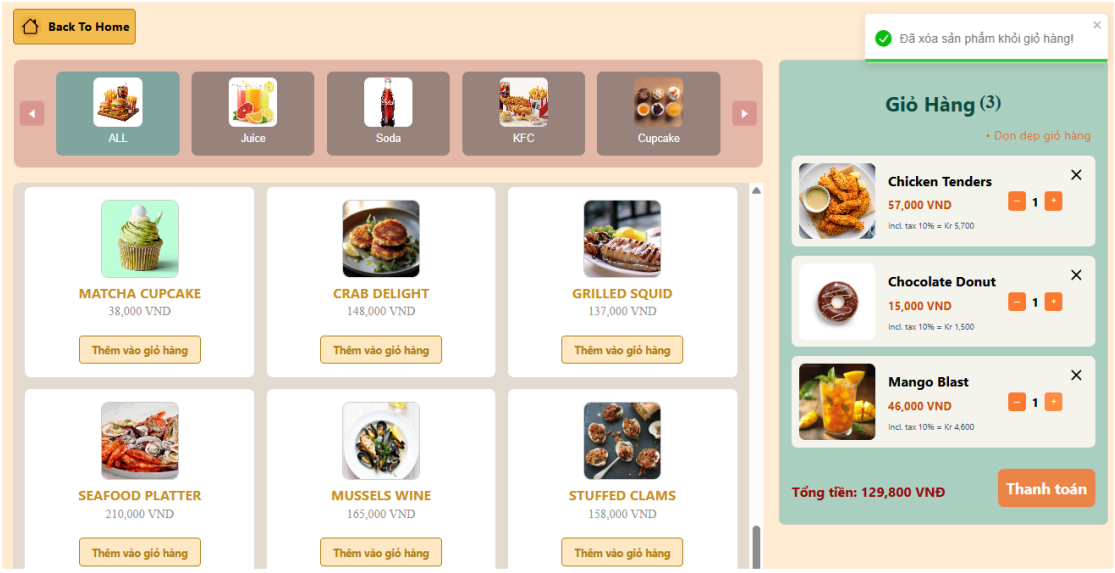
Hình 9: Hiển thị danh sách món ăn



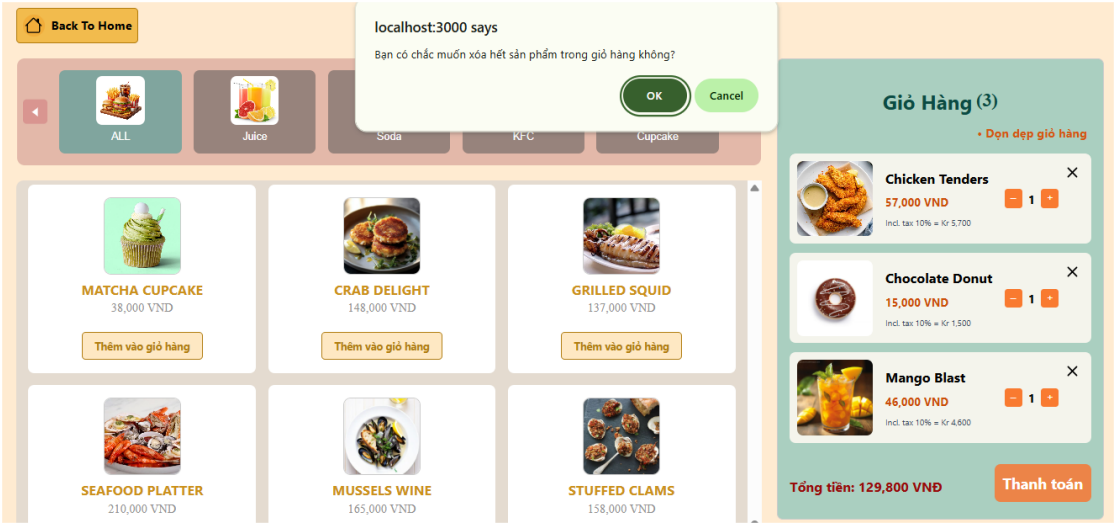
Hình 10: Phân loại món ăn theo danh mục



Hình 11: Thêm sản phẩm vào giỏ hàng



Hình 12: Xóa sản phẩm trong giỏ hàng



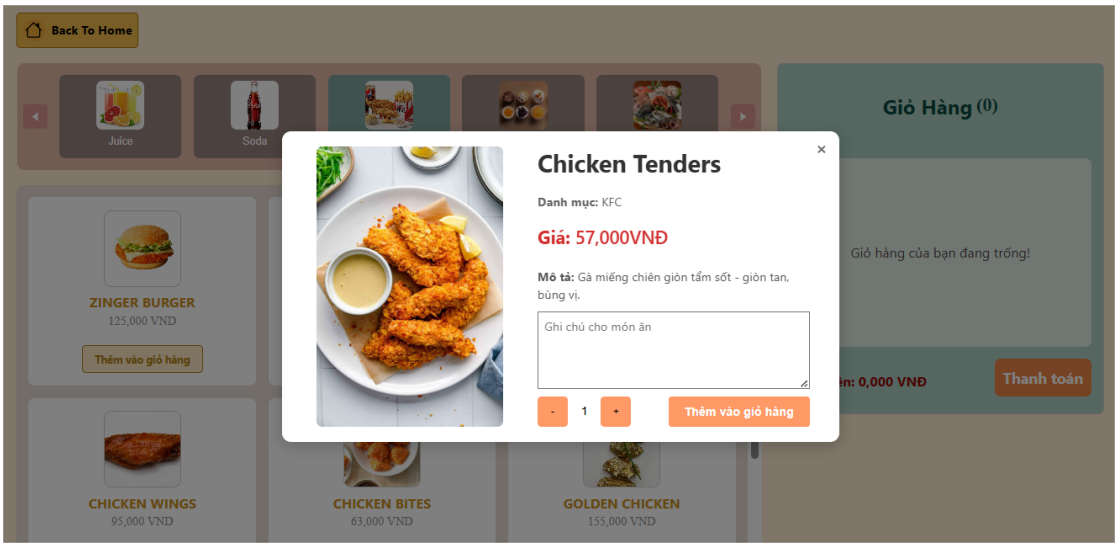
Hình 13: Dọn dẹp giỏ hàng

5 Triển khai - Sprint 2

5.1 Triển Khai MVP cho Màn Hình chức năng hiển thị chi tiết sản phẩm và thanh toán

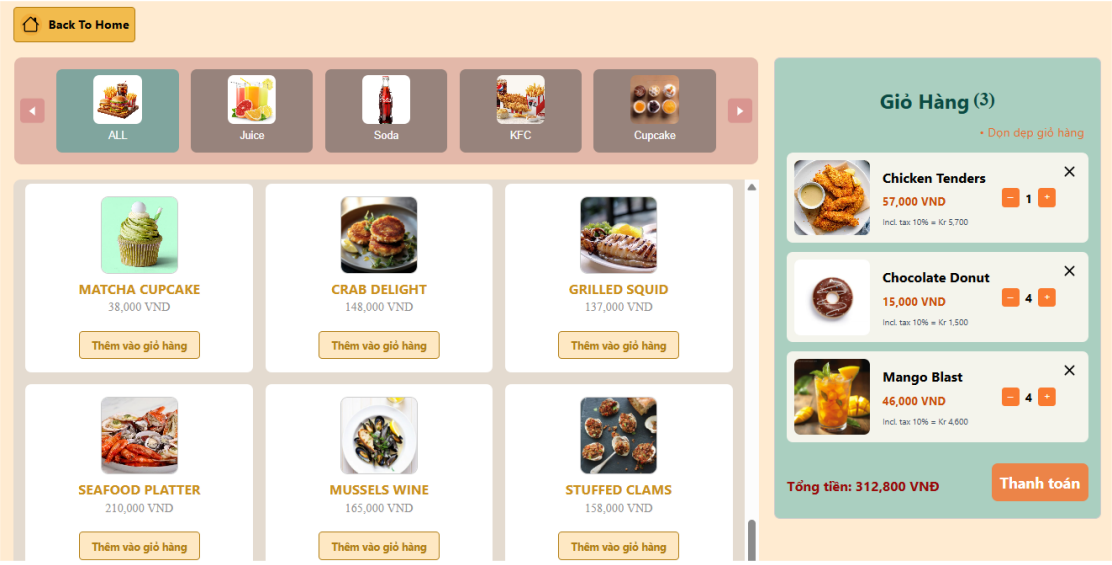
5.1.1 Màn Hình Chi Tiết Món Ăn

- Hiển thị mô tả, giá, và hình ảnh món ăn.



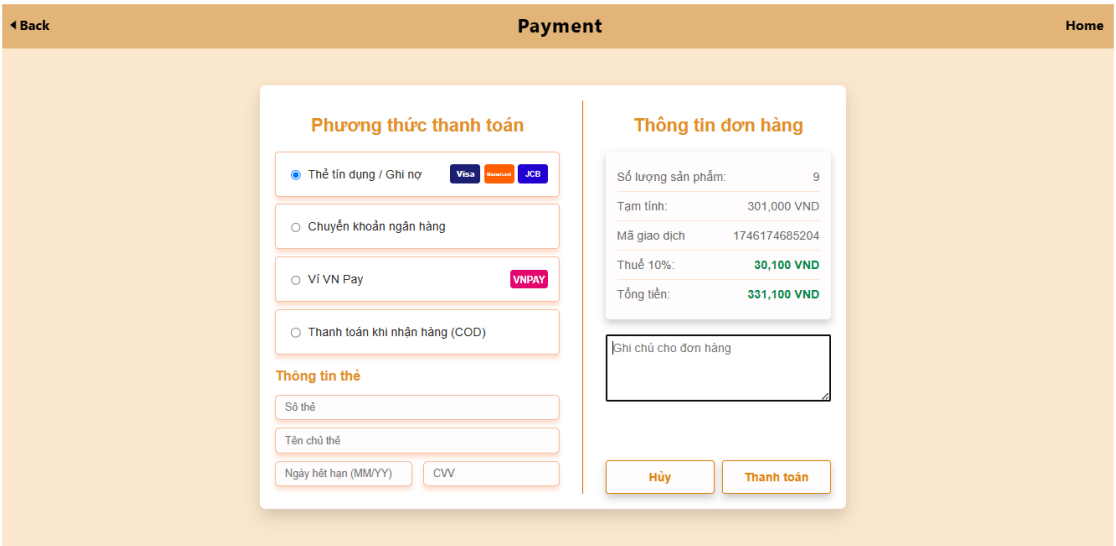
Hình 14: Thông tin chi tiết món ăn

- Cập nhật số lượng sản phẩm, yêu cầu khách hàng và hiển thị tổng số tiền.

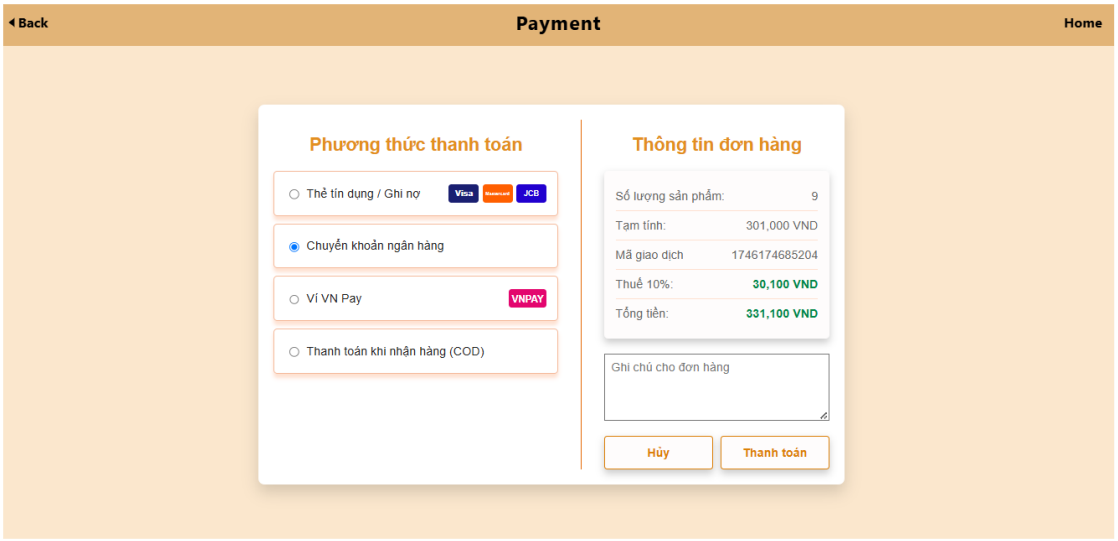


Hình 15: Cập nhật giỏ hàng

- Hiển thị chức năng thanh toán.



Hình 16: Hiển thị lựa chọn thanh toán



Hình 17: Thanh toán bằng phương thức chuyển khoản

Tài liệu

SCVX Introduction "**link: http://cvxr.com/cvx/doc/intro.html/**", *What is CVX*, lần truy cập cuối: 15/04/2017. CVX Introduction2 "**link: http://cvxr.com/cvx/doc/intro.html/**", *What is CVX*, lần truy cập cuối: 15/04/2017.