

## BỘ THÔNG TIN VÀ TRUYỀN THÔNG

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

oOo

Môn Phân tích & thiết kế hệ thống

**Dự án: Hệ thống quản lý**

**nhà hàng**

**Họ Và Tên: Phan Thuận Thành  
 Lớp: HN-KS24-CNTT2**

MỤC LỤC

**[CHƯƠNG I. GIỚI THIỆU 3](#_Toc1240)**

**[1.1. Mục đích 3](#_Toc12339)**

**[1.2. Bối cảnh và Vấn đề cần giải quyết 3](#_Toc20174)**

**[1.3. Phạm vi của Hệ thống 4](#_Toc18662)**

**[1.4. Định nghĩa, Từ viết tắt 5](#_Toc28950)**

**[CHƯƠNG 2. MÔ TẢ TỔNG QUAN 5](#_Toc11408)**

**[2.1. Bối cảnh sản phẩm (Restaurant Management System) 5](#_Toc14258)**

**[2.2. Kiến trúc 3 tầng (Mô tả + Sơ đồ) 6](#_Toc4235)**

**[2.3. Biểu đồ Use Case tổng thể 7](#_Toc15986)**

**[2.4. Đặc điểm Người dùng (Actors) 8](#_Toc21981)**

**[2.5. Các ràng buộc 9](#_Toc3306)**

**[CHƯƠNG 3. YÊU CẦU CỤ THỂ 10](#_Toc3526)**

**[3.1. Yêu cầu về Giao diện Người dùng 10](#_Toc32598)**

**[3.1.1. Bản phác thảo giao diện bằng tay 10](#_Toc23464)**

**[3.1.2. Giao diện chính được làm trên Figma 13](#_Toc19292)**

**[3.2. Yêu cầu Chức năng 15](#_Toc26419)**

**[3.2.1. Use Case: Đặt bàn 15](#_Toc10244)**

**[3.2.2. Use Case: Gọi món 16](#_Toc26883)**

**[3.2.3. Use Case: Chế biến món 17](#_Toc6758)**

**[3.2.4. Use Case: Thanh toán hóa đơn 18](#_Toc27914)**

**[3.2.5. Use Case: Quản lý thực đơn 19](#_Toc13538)**

**[3.2.6. Use Case: Xem báo cáo doanh thu 20](#_Toc388)**

**[3.2.7. Use Case: Quản lý nhân viên 21](#_Toc17616)**

**[3.2.8. Use Case: Quản lý bàn ăn 22](#_Toc32323)**

**[3.3. Yêu cầu Phi chức năng 23](#_Toc2836)**

**[3.4. Yêu cầu về Cơ sở dữ liệu Logic 24](#_Toc12176)**

**[PHỤ LỤC 29](#_Toc15871)**

**[A. Biểu đồ Lớp (Class Diagram) 29](#_Toc18686)**

**[B. Biểu đồ Tuần Tự (Sequence Diagram) 33](#_Toc18256)**

**[1. Biểu đồ tuần tự – UC: Đặt bàn 33](#_Toc26585)**

**[2. Biểu đồ tuần tự – UC: Gọi món 36](#_Toc1899)**

**[3. Biểu đồ tuần tự – UC: Thanh toán đơn hàng 38](#_Toc27169)**

**[4. Biểu đồ tuần tự – UC: Quản lý bàn ăn 41](#_Toc28398)**

**[5. Biểu đồ tuần tự – UC: Xem báo cáo doanh thu 45](#_Toc17958)**

**[C. BIỂU ĐỒ TRẠNG THÁI ( State Diagram) 46](#_Toc2643)**

**[1. Biểu đồ trạng thái – UC: Đặt bàn 46](#_Toc12355)**

**[2. Biểu đồ trạng thái – UC: Thanh toán đơn hàng 49](#_Toc14143)**

**[3. Biểu đồ trạng thái – UC: Chế biến món 53](#_Toc1875)**

# CHƯƠNG I. GIỚI THIỆU

## 1.1. Mục đích

- Tài liệu này (Software Requirements Specification - SRS) đặc tả các yêu cầu chức năng và phi chức năng cho dự án **“Hệ thống Quản lý Nhà hàng” (Restaurant Management System - RMS).  
-** Tài liệu này đóng vai trò là **nguồn tham chiếu chính thức** cho toàn bộ các bên liên quan bao gồm **Chủ nhà hàng, Quản lý, Nhân viên phục vụ, Bộ phận bếp, và đội ngũ phát triển hệ thống**.

- Tài liệu giúp xác định rõ:

+ Những chức năng hệ thống cần thực hiện.

+ Các tiêu chuẩn về hiệu năng, bảo mật và tính ổn định.

+ Cơ sở cho các giai đoạn thiết kế, lập trình, kiểm thử, triển khai và bảo trì sau này.

- Hệ thống RMS hướng tới việc **tự động hóa toàn bộ quy trình vận hành** trong nhà hàng — từ đặt bàn, gọi món, chuyển order đến bếp, theo dõi tình trạng chế biến, cho đến thanh toán và báo cáo doanh thu — nhằm **tối ưu năng suất làm việc, giảm sai sót thủ công, và nâng cao trải nghiệm khách hàng**.

## 1.2. Bối cảnh và Vấn đề cần giải quyết

- Hiện nay, nhiều nhà hàng vẫn đang vận hành dựa trên **quy trình bán thủ công**: ghi order bằng giấy, nhập hóa đơn trên máy tính tiền riêng lẻ, và tổng hợp doanh thu bằng Excel.  
- Cách làm này dẫn đến nhiều **bất cập** trong quản lý và phục vụ:

+ **Quy trình phục vụ chậm**: Nhân viên phục vụ phải chạy qua lại giữa bàn ăn và quầy, ghi order bằng tay rồi chuyển cho bếp, dễ gây sai sót và chậm trễ trong giờ cao điểm.

+ **Thiếu đồng bộ giữa các bộ phận**: Phục vụ, bếp và thu ngân không có hệ thống liên kết dữ liệu, dẫn đến nhầm lẫn trong order, trùng món hoặc bỏ sót yêu cầu.

+ **Khó theo dõi hoạt động**: Quản lý không thể theo dõi tức thời số lượng bàn đang phục vụ, doanh thu theo ca hoặc hiệu suất của nhân viên.

+ **Thống kê thủ công**: Việc tổng hợp doanh thu, chi phí, lượng tiêu thụ món ăn phải làm bằng tay, mất thời gian và dễ sai sót.

+ **Trải nghiệm khách hàng kém**: Khách phải chờ lâu để được gọi món, nhận hóa đơn hoặc thanh toán.

- Hệ thống **RMS** được đề xuất để **giải quyết triệt để** các vấn đề trên bằng cách:

+ Cung cấp **một nền tảng quản lý tập trung** cho toàn bộ nhà hàng.

+ Cho phép **các bộ phận hoạt động đồng bộ** qua mạng nội bộ.

+ Tự động hóa quy trình từ **đặt bàn → gọi món → chế biến → thanh toán.**

+ Cung cấp **báo cáo doanh thu, hiệu suất và thống kê món ăn** theo thời gian thực.

## 1.3. Phạm vi của Hệ thống

**- Các chức năng có trong hệ thống (In-scope):**

+ **Module Quản lý Bàn ăn (Table Management):**

* Quản lý thông tin bàn ăn: số bàn, khu vực, tình trạng (trống, đang phục vụ, đã đặt trước).
* Hỗ trợ đặt bàn trước và cập nhật trạng thái theo thời gian thực.

+ **Module Gọi món & Order (Ordering System):**

* Cho phép nhân viên phục vụ chọn món cho bàn qua tablet hoặc máy tính.
* Gửi order trực tiếp đến bếp qua hệ thống nội bộ.
* Cập nhật trạng thái món (đang chế biến, hoàn thành, đã phục vụ).

+ **Module Quản lý Món ăn (Menu Management):**

* Quản lý danh sách món ăn, nguyên liệu, giá, mô tả.
* Phân loại theo danh mục (món chính, món phụ, nước uống...).
* Cập nhật nhanh giá bán hoặc tình trạng món hết hàng.

+ **Module Quản lý Thanh toán & Hóa đơn (Billing System):**

* Tạo và in hóa đơn tự động khi hoàn tất bữa ăn.
* Hỗ trợ chia hóa đơn, giảm giá, và áp dụng khuyến mãi.
* Quản lý phương thức thanh toán (tiền mặt, thẻ, ví điện tử...).

**+ Module Quản lý Nhân viên (Staff Management):**

* Lưu thông tin nhân viên, phân quyền (phục vụ, bếp, thu ngân, quản lý).
* Theo dõi ca làm việc và hiệu suất nhân viên.

**+ Module Báo cáo & Thống kê (Reporting & Analytics):**

* Báo cáo doanh thu theo ngày, tuần, tháng.
* Thống kê món bán chạy, ca làm hiệu quả, lượng khách hàng.
* Xuất file Excel/PDF phục vụ quản lý và kế toán.

**- Các chức năng không có trong hệ thống (Out-of-scope):**

* Hệ thống **không tích hợp quản lý kho nguyên liệu** chi tiết (sẽ xem xét ở giai đoạn 2).
* Hệ thống **không hỗ trợ đặt món hoặc thanh toán trực tuyến từ xa.**
* Hệ thống **không bao gồm quản lý nhân sự, chấm công và tính lương.**
* Hệ thống k**hông hỗ trợ quản lý khách hàng thân thiết hoặc điểm thưởng** ở giai đoạn đầu.

## 1.4. Định nghĩa, Từ viết tắt

|  |  |
| --- | --- |
| **Từ viết tắt** | **Định nghĩa** |
| **RMS** | Restaurant Management System – Hệ thống Quản lý Nhà hàng |
| **Bàn ăn (Table)** | Một vị trí phục vụ cụ thể trong nhà hàng, có thể dành cho 1 hoặc nhiều khách. |
| **Order (Đơn gọi món)** | Danh sách món ăn mà khách yêu cầu trong một lần phục vụ. |
| **Món ăn (Dish / Menu Item)** | Một sản phẩm được phục vụ trong nhà hàng, gồm tên, giá và mô tả. |
| **Hóa đơn (Bill / Receipt)** | Tài liệu tổng hợp toàn bộ order của khách, dùng để thanh toán. |
| **Nhân viên phục vụ (Waiter/Waitress)** | Người tiếp nhận order và phục vụ khách. |
| **Bếp (Kitchen)** | Bộ phận tiếp nhận order và chế biến món ăn. |
| **Quản lý (Manager)** | Người theo dõi, thống kê và điều hành hoạt động nhà hàng. |
| **POS** | Point of Sale – hệ thống tính tiền và xử lý thanh toán. |

# CHƯƠNG 2. MÔ TẢ TỔNG QUAN

2.1. Bối cảnh sản phẩm (Restaurant Management System)- Hệ thống Quản lý Nhà hàng (Restaurant Management System - RMS) là một **ứng dụng web nội bộ** (web-based internal system), được triển khai nhằm h**ỗ trợ hoạt động vận hành, phục vụ và thanh toán** tại nhà hàng. Hệ thống cho phép **quản lý bàn ăn, thực đơn, đơn hàng, nhân viên và hóa đơn** trên một nền tảng duy nhất, giúp tối ưu quy trình phục vụ khách hàng và giảm thiểu sai sót trong thao tác thủ công.

- Ứng dụng được truy cập thông qua **trình duyệt web** trên các thiết bị như **máy tính tại quầy thu ngân, máy tính bảng của nhân viên phục vụ hoặc laptop của quản lý**. Tất cả dữ liệu của hệ thống (bàn ăn, món ăn, đơn hàng, nhân viên, doanh thu,...) được lưu trữ và xử lý trong **cơ sở dữ liệu tập trung**, đặt tại **máy chủ nội bộ của nhà hàng** hoặc trên **máy chủ đám mây (cloud server)** tùy quy mô triển khai.

- Hệ thống được thiết kế hướng tới mục tiêu:

+ Tự động hóa quy trình phục vụ và thanh toán.

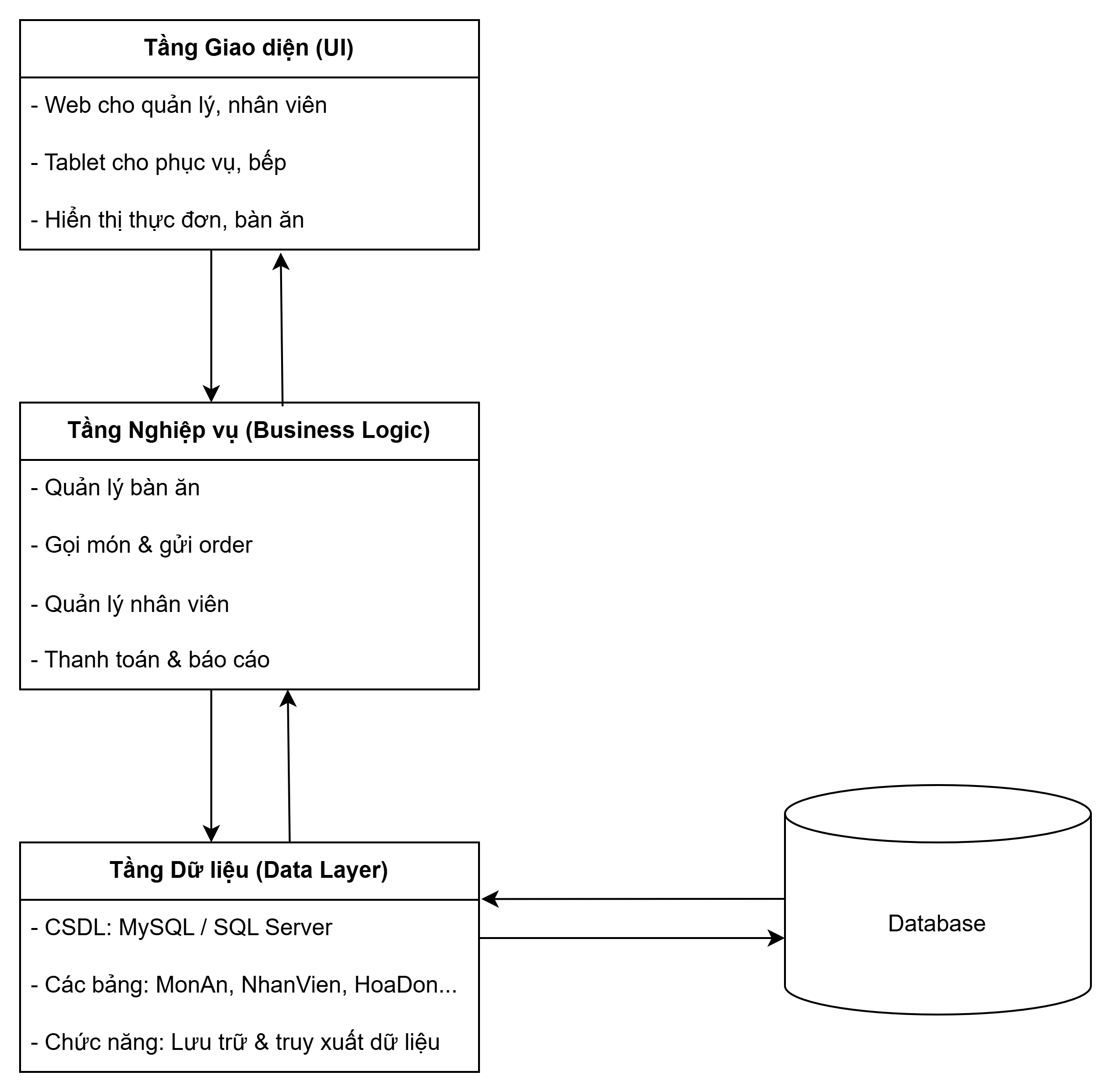
+ Giảm thời gian chờ đợi của khách hàng.

+ Tăng tính minh bạch và hiệu quả trong quản lý doanh thu.

+ Dễ dàng mở rộng, tích hợp thêm các chức năng mới (đặt bàn online, quản lý kho, ưu đãi…).

## 2.2. Kiến trúc 3 tầng (Mô tả + Sơ đồ)

|  |  |
| --- | --- |
| **Tầng** | **Mô tả** |
| **Tầng Giao diện (Presentation Layer)** | Đây là tầng trực tiếp tương tác với người dùng, bao gồm giao diện web cho quản lý và nhân viên, cùng giao diện trên tablet cho bộ phận phục vụ và bếp. Tầng này hiển thị các thông tin như danh sách bàn ăn, thực đơn, order món và hóa đơn, giúp người dùng thao tác dễ dàng, nhanh chóng và trực quan trong quá trình làm việc. |
| **Tầng Nghiệp vụ (Business Logic Layer)** | Tầng này đóng vai trò trung tâm trong hệ thống, chịu trách nhiệm xử lý toàn bộ logic nghiệp vụ. Các chức năng chính gồm: quản lý bàn ăn, tiếp nhận và gửi order món đến bếp, xử lý thanh toán, quản lý thông tin nhân viên và tổng hợp báo cáo doanh thu. Tầng này đảm bảo các quy trình vận hành diễn ra chính xác, thống nhất và hiệu quả giữa các bộ phận. |
| **Tầng Dữ liệu (Data Layer)** | Tầng dữ liệu quản lý toàn bộ thông tin của hệ thống thông qua cơ sở dữ liệu tập trung. Các bảng chính bao gồm bàn ăn, món ăn, hóa đơn, nhân viên và doanh thu. Dữ liệu được lưu trữ an toàn, đảm bảo tính toàn vẹn, bảo mật và có thể truy xuất nhanh chóng khi cần thiết, giúp hỗ trợ tối đa cho các hoạt động của nhà hàng. |



## 2.3. Biểu đồ Use Case tổng thể

- **Hệ thống Quản lý Nhà hàng (Restaurant Management System)** được thiết kế xoay quanh **năm nhóm người dùng chính**, cung cấp các chức năng phù hợp với vai trò nghiệp vụ của từng đối tượng. Các chức năng này được thể hiện tổng quan qua **biểu đồ Use Case tổng thể**:

**- Actor:**

**+ Nhân viên phục vụ**

**+ Đầu bếp**

**+ Thu ngân**

**+ Quản lý**

**+ Khách hàng**

**- Use Case: Nhóm 1 – Quản lý Hệ thống (do Quản lý thực hiện) **

**+ Quản lý bàn ăn:** Thêm, sửa, xóa và cập nhật trạng thái bàn (đang trống, đã đặt, đang phục vụ). 

**+ Quản lý thực đơn:** Quản lý danh sách món ăn, giá bán, nguyên liệu và tình trạng sẵn sàng của món. 

**+ Quản lý nhân viên:** Thêm mới, phân quyền và cập nhật thông tin nhân viên. 

**+ Xem báo cáo doanh thu:** Tổng hợp doanh thu theo ngày, tuần, tháng:

Include → Kiểm tra quyền truy cập (chỉ Quản lý được phép xem).

**- Nhóm 2 – Phục vụ khách (do Nhân viên phục vụ thực hiện) **

**+ Đặt bàn:** Đặt bàn theo yêu cầu khách hàng, kiểm tra bàn trống, đề xuất bàn khác nếu khu vực đã đầy.

Include → Kiểm tra bàn trống

Extend → Gợi ý bàn khác 

**+ Gọi món:** Ghi nhận order của khách và gửi thông tin đến bếp. Include → Gửi Order cho đầu bếp 

**+ Quản lý bàn ăn:** Cập nhật trạng thái bàn, thêm ghi chú, hoặc hủy bàn nếu khách đổi ý.

**+ Thanh toán hóa đơn:** Hỗ trợ khách yêu cầu thanh toán, liên kết với bộ phận thu ngân.

**- Nhóm 3 – Bếp (do Đầu bếp thực hiện) **

**+ Chế biến món:** Tiếp nhận order, kiểm tra nguyên liệu, chế biến và cập nhật trạng thái món.

Include → Thông báo món hoàn thành

Extend → Kiểm tra nguyên liệu

**- Nhóm 4 – Thanh toán (do Thu ngân thực hiện) **

**+ Thanh toán hóa đơn**: Xử lý thanh toán cho khách hàng sau khi dùng bữa, in hóa đơn, và ghi nhận doanh thu.

Include → Tạo hóa đơn

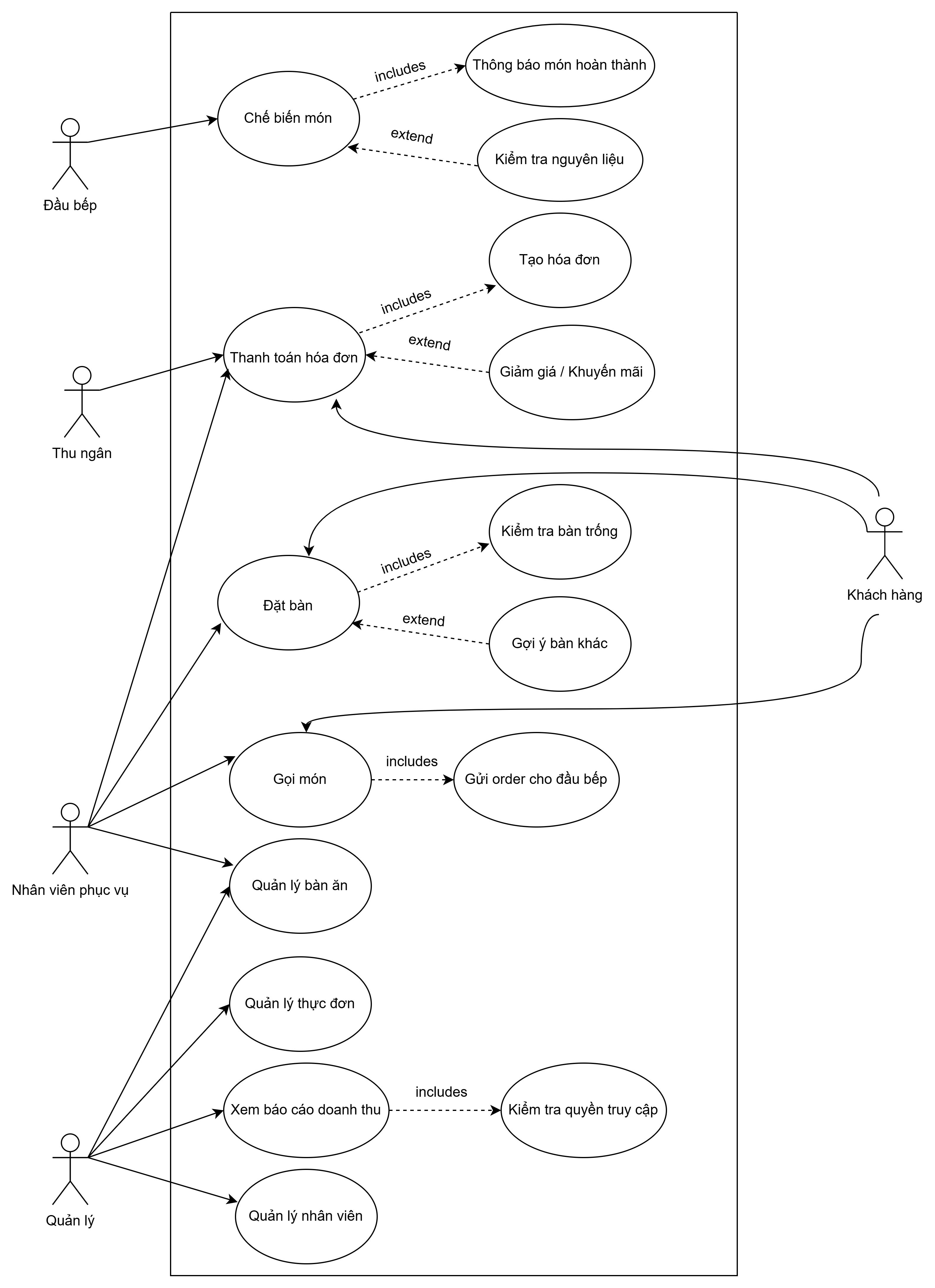
Extend → Áp dụng khuyến mãi/giảm giá

**- Nhóm 5 – Khách hàng **

**+ Đặt bàn:** Khách có thể yêu cầu đặt bàn thông qua nhân viên phục vụ. 

**+ Gọi món:** Khách trực tiếp yêu cầu món với nhân viên phục vụ. 

**+ Thanh toán hóa đơn:** Khách yêu cầu thanh toán và nhận hóa đơn từ thu ngân.



## 2.4. Đặc điểm Người dùng (Actors)

|  |  |
| --- | --- |
| **Actor** | **Mô tả** |
| **Quản lý** | - Là người có quyền truy cập cao nhất trong hệ thống.  - Có thể thao tác toàn bộ chức năng quản lý bàn, thực đơn, nhân viên, và xem báo cáo doanh thu.  - Trước khi thực hiện các chức năng quan trọng, hệ thống sẽ kiểm tra quyền truy cập để đảm bảo bảo mật.  - Thường sử dụng hệ thống trên máy tính hoặc laptop tại văn phòng quản lý |
| **Nhân viên phục vụ** | - Là người tương tác trực tiếp với khách hàng.  - Sử dụng hệ thống trên máy tính bảng để đặt bàn, gọi món, và gửi order đến đầu bếp.  - Có thể kiểm tra tình trạng bàn trống, cập nhật trạng thái bàn hoặc hỗ trợ khách thanh toán.  - Cần thao tác nhanh và chính xác, đặc biệt trong giờ cao điểm. |
| **Đầu bếp** | - Tiếp nhận order từ hệ thống, chế biến món ăn tương ứng.  - Có thể kiểm tra nguyên liệu trước khi bắt đầu và gửi thông báo “món hoàn thành” khi xong.  - Không tương tác với khách hàng mà chỉ làm việc qua giao diện nội bộ tại khu vực bếp. |
| **Thu ngân** | - Thực hiện quy trình thanh toán cho khách hàng sau bữa ăn.  - Tạo hóa đơn tự động, áp dụng giảm giá hoặc khuyến mãi nếu có.  - Ghi nhận thanh toán và lưu thông tin hóa đơn phục vụ cho báo cáo doanh thu.  - Sử dụng hệ thống tại quầy thu ngân, kết nối với máy in hóa đơn. |
| **Khách hàng** | - Là người sử dụng dịch vụ của nhà hàng.  - Thực hiện các yêu cầu thông qua nhân viên phục vụ: đặt bàn, gọi món, yêu cầu thanh toán.  - Nhận hóa đơn thanh toán sau khi hoàn tất bữa ăn. |

## 2.5. Các ràng buộc

- **RB-01**: Mỗi nhân viên phục vụ chỉ được thao tác trên các bàn thuộc khu vực được phân công.  
- **RB-02**: Khi đặt bàn, hệ thống phải kiểm tra bàn trống. Nếu bàn đầy, hệ thống tự động gợi ý bàn khác.  
- **RB-03**: Khi gọi món, hệ thống bắt buộc gửi order đến đầu bếp để xác nhận chế biến.  
- **RB-04**: Đầu bếp có thể kiểm tra nguyên liệu trước khi chế biến. Nếu thiếu, hệ thống sẽ gửi thông báo cho Quản lý.  
- **RB-05**: Sau khi chế biến, đầu bếp phải gửi thông báo món hoàn thành để nhân viên phục vụ nhận và mang ra cho khách.  
- **RB-06**: Khi thanh toán, hệ thống sẽ tự động tạo hóa đơn và cho phép áp dụng khuyến mãi/giảm giá nếu có.  
- **RB-07**: Chỉ Quản lý mới được truy cập các chức năng quản lý nhân viên, bàn ăn, thực đơn và báo cáo doanh thu.  
- **RB-08**: Hệ thống phải hoạt động ổn định trên máy tính bảng (khu vực phục vụ), máy tính (khu vực thu ngân và quản lý), và thiết bị bếp nội bộ.  
- **RB-09**: Dữ liệu được lưu trữ tập trung trên máy chủ nội bộ, có cơ chế phân quyền người dùng theo vai trò.  
- **RB-10**: Hệ thống cần đảm bảo xử lý song song nhiều order trong giờ cao điểm mà không bị treo hoặc mất dữ liệu.

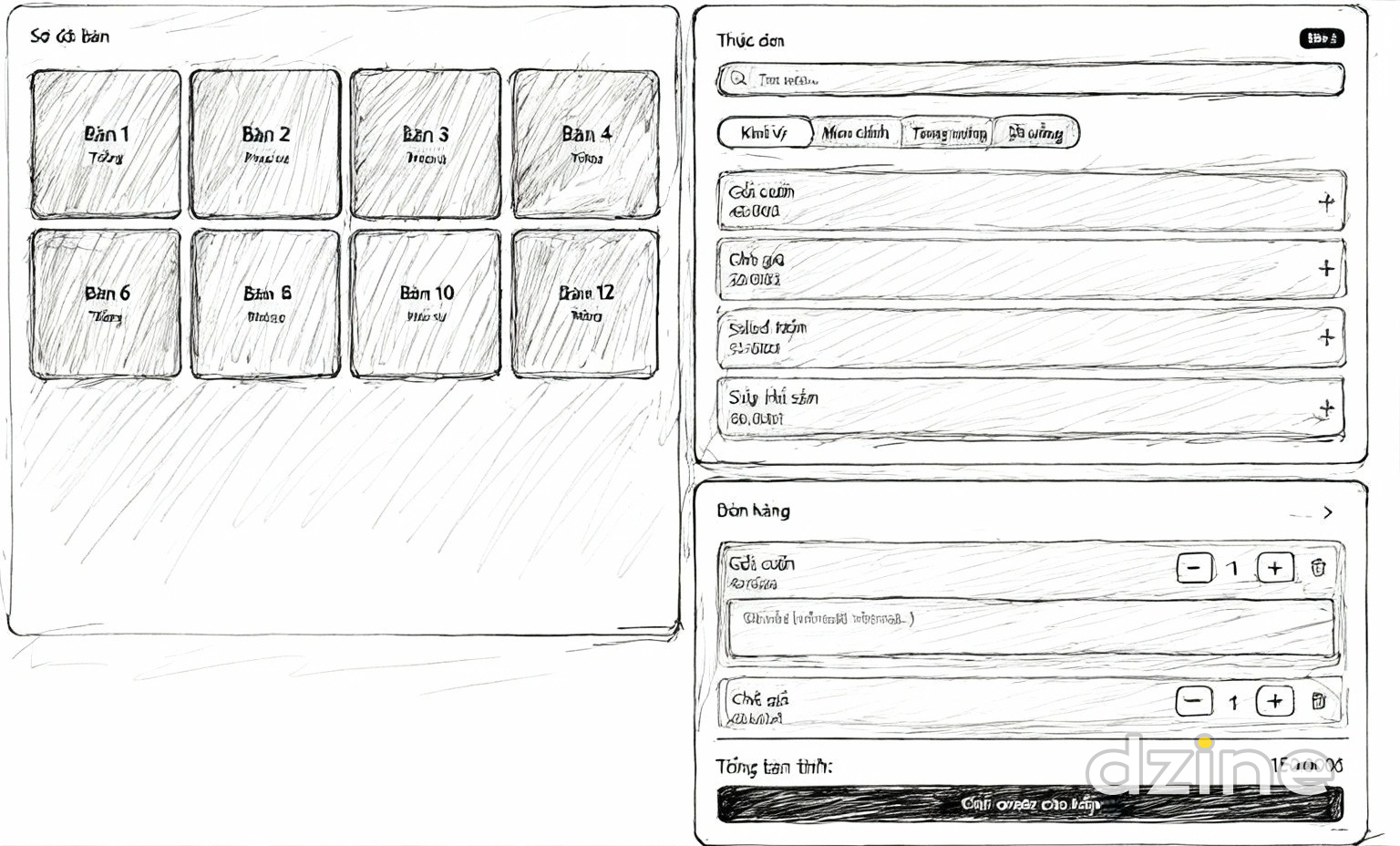
CHƯƠNG 3. YÊU CẦU CỤ THỂ

## 3.1. Yêu cầu về Giao diện Người dùng

- Giao diện của hệ thống RMS được thiết kế theo phong cách hiện đại, trực quan và tối ưu cho nhiều loại thiết bị, bao gồm máy tính, máy tính bảng và thiết bị POS.  
- Mục tiêu chính là giúp nhân viên và quản lý thao tác nhanh, chính xác trong môi trường thực tế nhà hàng.

## 3.1.1. Bản phác thảo giao diện bằng tay

**- Wireframe 1: Màn hình Gọi món (Order Screen)**



**+ Mô tả:**

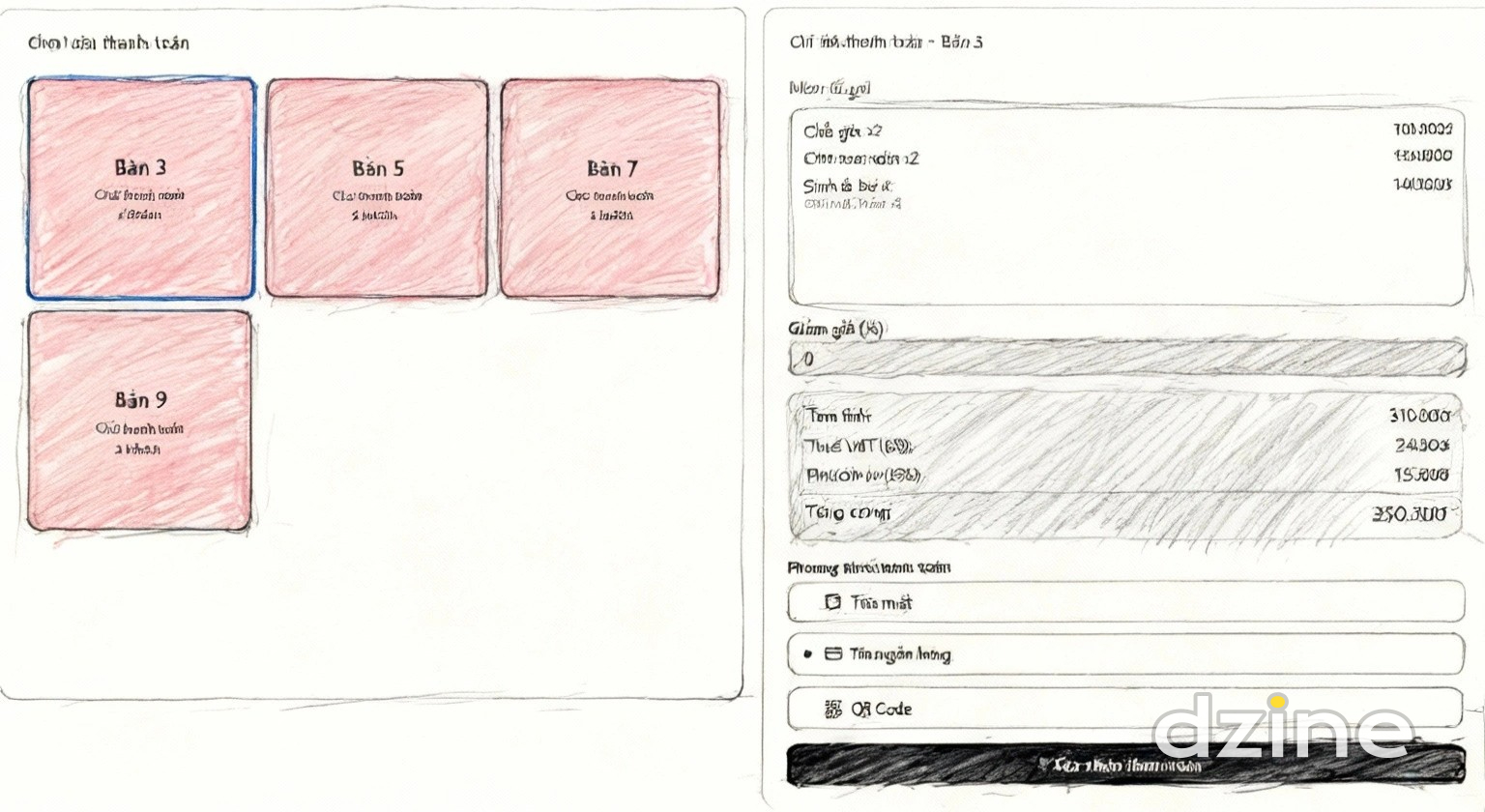
* Màn hình chia thành 2 phần:

Bên trái: Hiển thị sơ đồ bàn ăn (dạng grid hoặc floor map), thể hiện trạng thái bàn (Trống, Đang phục vụ, Đã đặt trước).

Bên phải: Hiển thị danh sách món ăn theo danh mục (Khai vị, Món chính, Tráng miệng, Đồ uống). Nhân viên có thể chọn món nhanh hoặc tìm kiếm bằng tên.

* Khi chọn món, hệ thống sẽ hiển thị số lượng, ghi chú (ví dụ: ít cay, thêm đá) và tự động tính tổng tiền tạm tính.
* Cuối cùng, nhân viên nhấn “Gửi order cho bếp”, hệ thống gửi danh sách món sang màn hình bếp.

**- Wireframe 2: Màn hình Thanh toán (Payment Screen)**

****

**+ Mô tả:**

* Sau khi khách yêu cầu thanh toán, nhân viên chọn bàn → hệ thống hiển thị danh sách món đã gọi, tổng tiền, giảm giá (nếu có), thuế và phí dịch vụ.
* Cho phép chọn phương thức thanh toán (tiền mặt, thẻ, QR code).
* Nút “Xác nhận thanh toán” thực hiện tạo hóa đơn và in biên lai tự động.
* Hệ thống chuyển trạng thái bàn từ “Đang phục vụ” sang “Trống”.

**- Wireframe 3: Dashboard Quản lý**

****

**+ Mô tả:**

* Trang tổng quan dành cho Quản lý hiển thị:

Thống kê nhanh: Số bàn đang hoạt động, tổng doanh thu trong ngày, số đơn đã hoàn tất.

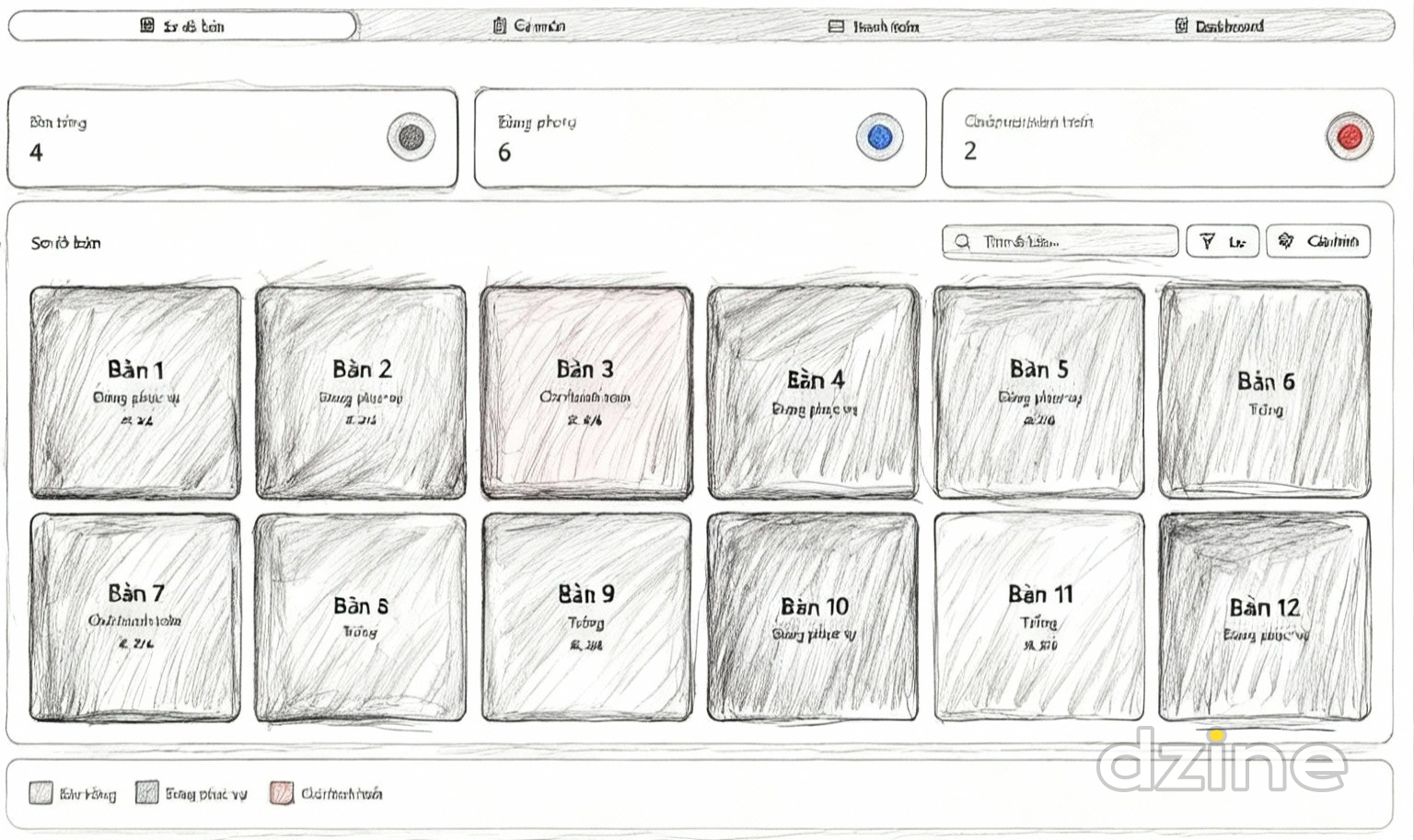
Biểu đồ doanh thu (theo ngày, tuần, tháng).

Danh sách nhân viên đang làm việc, cùng thời gian ca làm.

Top 5 món bán chạy nhất trong ngày.

* Quản lý có thể chuyển nhanh sang các trang chi tiết như “Quản lý nhân viên”, “Báo cáo doanh thu”, “Quản lý menu”.

- **Wireframe 4: Màn hình Sơ đồ bàn (Table Overview Screen)**



**+ Mô tả:**

* Màn hình hiển thị toàn bộ danh sách bàn theo dạng lưới (grid) hoặc sơ đồ mặt bằng (floor map), giúp nhân viên quan sát tổng quan.
* Mỗi bàn được biểu diễn dạng ô vuông kèm số bàn, và được tô màu theo trạng thái:

Màu xám → Bàn trống.

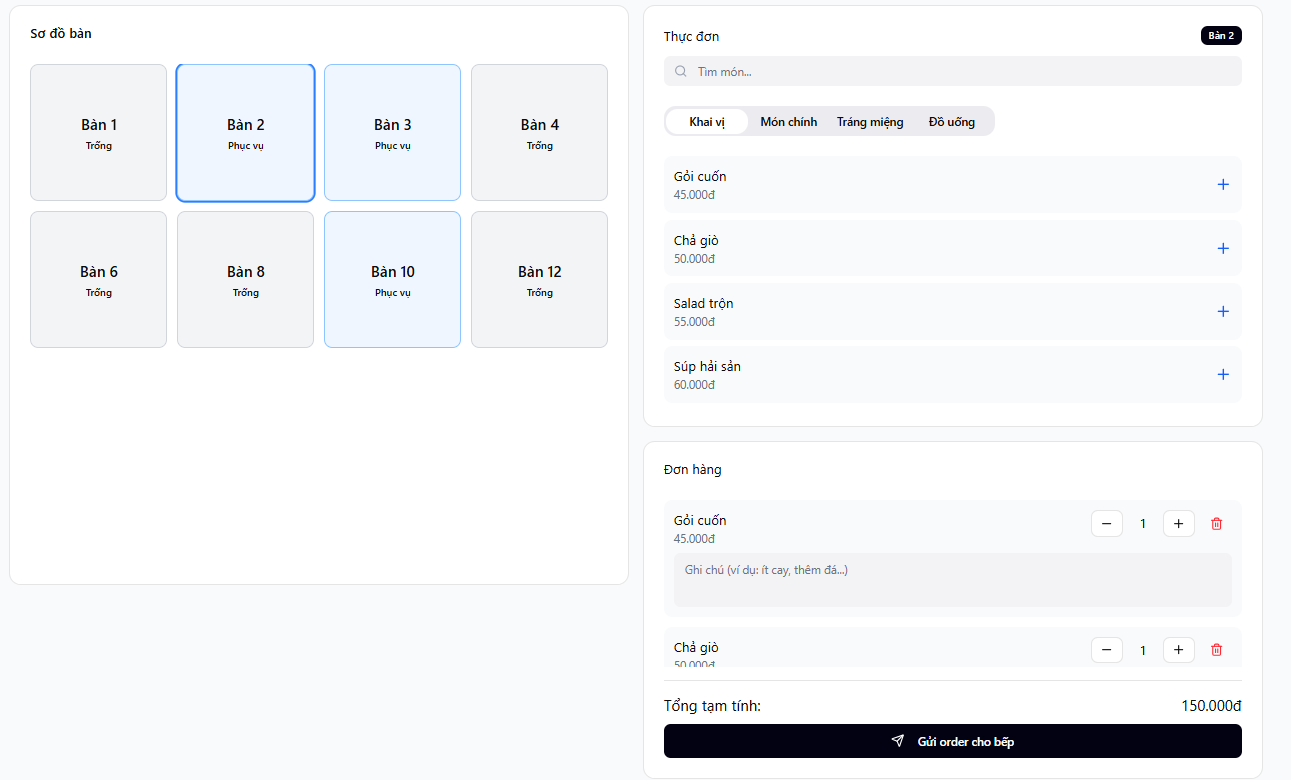
Màu xanh → Bàn đang phục vụ.

Màu đỏ → Bàn chờ thanh toán.

* Nhân viên có thể nhấn chọn một bàn để mở popup thông tin chi tiết, bao gồm:  
   Số bàn  
   Trạng thái hiện tại  
   Số lượng khách  
   Thời gian bắt đầu phục vụ  
   Nhân viên đang phụ trách
* Trong popup, hệ thống hiển thị các nút thao tác tùy theo trạng thái bàn:  
   Bàn trống → Nút “Nhận khách” hoặc “Đặt bàn”.  
   Bàn đang phục vụ → Nút “Gọi món”, “Xem order hiện tại”.  
   Bàn chờ thanh toán → Nút “Thanh toán”.
* Phía trên cùng có thanh công cụ:  
   Ô tìm kiếm theo số bàn  
   Bộ lọc theo trạng thái (Trống / Đang phục vụ / Chờ thanh toán)  
   Nút “Cấu hình sơ đồ bàn” dành cho quản lý (thêm bàn, đổi vị trí bàn…).
* Màn hình này cập nhật theo thời gian thực (real-time) khi bàn được gọi món, thanh toán hoặc thay đổi trạng thái từ các màn hình khác.

## 3.1.2. Giao diện chính được làm trên Figma

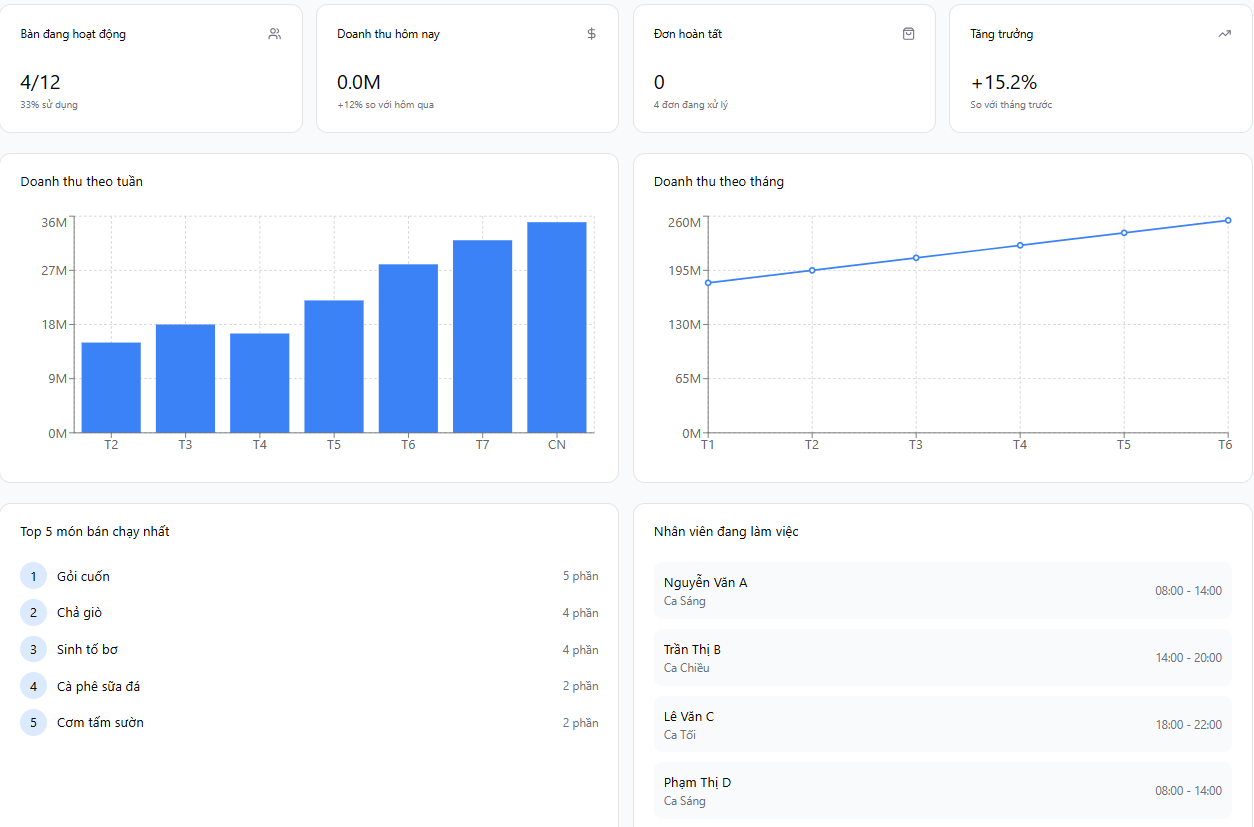
**- Figma 1: Màn hình Gọi món (Order Screen)**

****

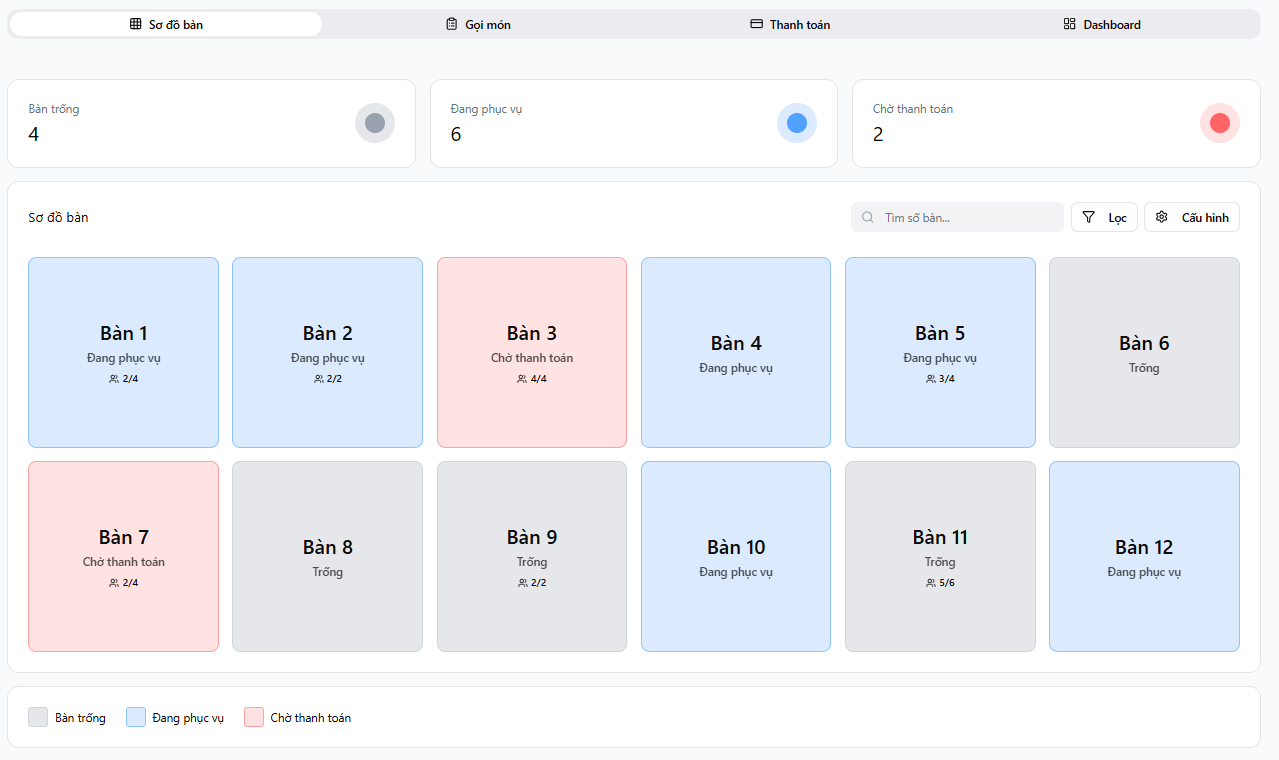
**- Figma2: Màn hình Thanh toán (Payment Screen)**

****

**- Figma3: Dashboard Quản lý**

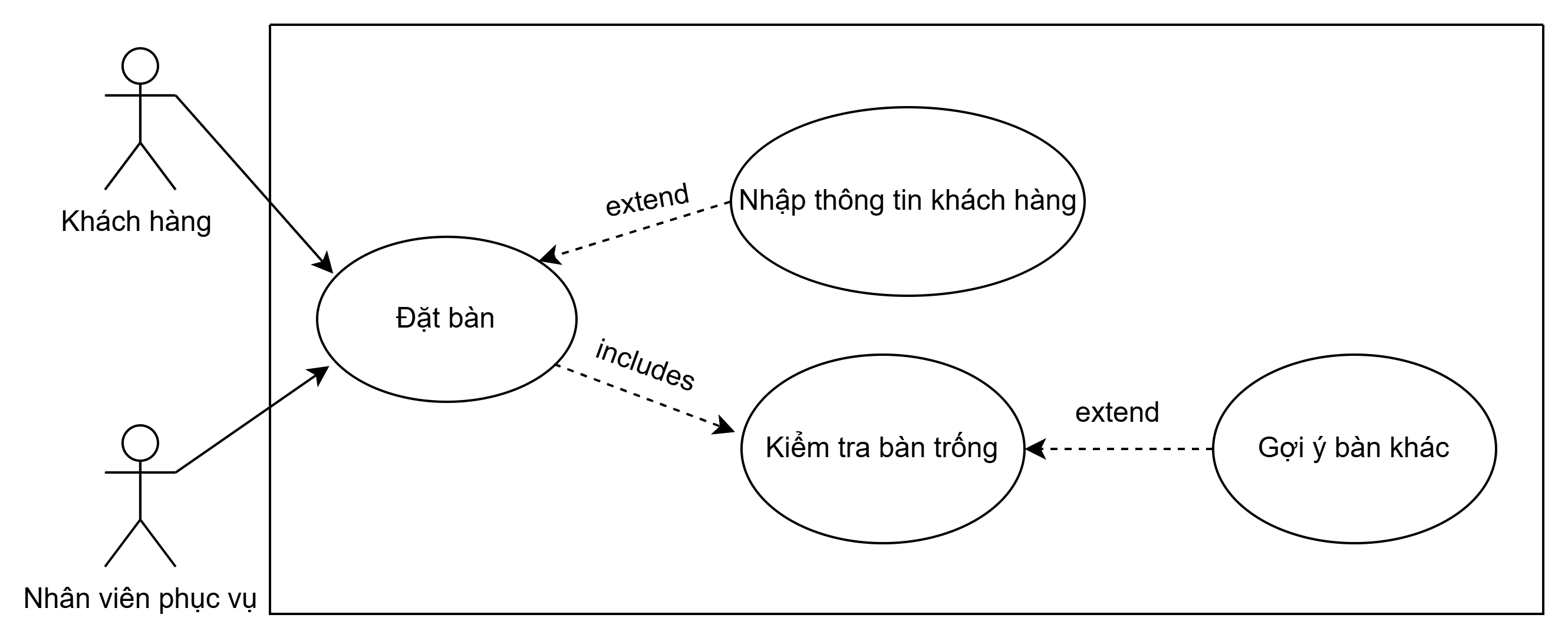
****

**- Figma 4: **Màn hình Sơ đồ bàn (Table Overview Screen)****

********

## 3.2. Yêu cầu Chức năng

## 3.2.1. Use Case: Đặt bàn



**- Mô tả ngắn:** Cho phép Nhân viên phục vụ tiếp nhận yêu cầu đặt bàn của Khách hàng, kiểm tra bàn trống và đề xuất bàn khác nếu cần.

**- Actor:** Nhân viên phục vụ, Khách hàng

**- Điều kiện tiên quyết:** Nhân viên phục vụ đã đăng nhập.

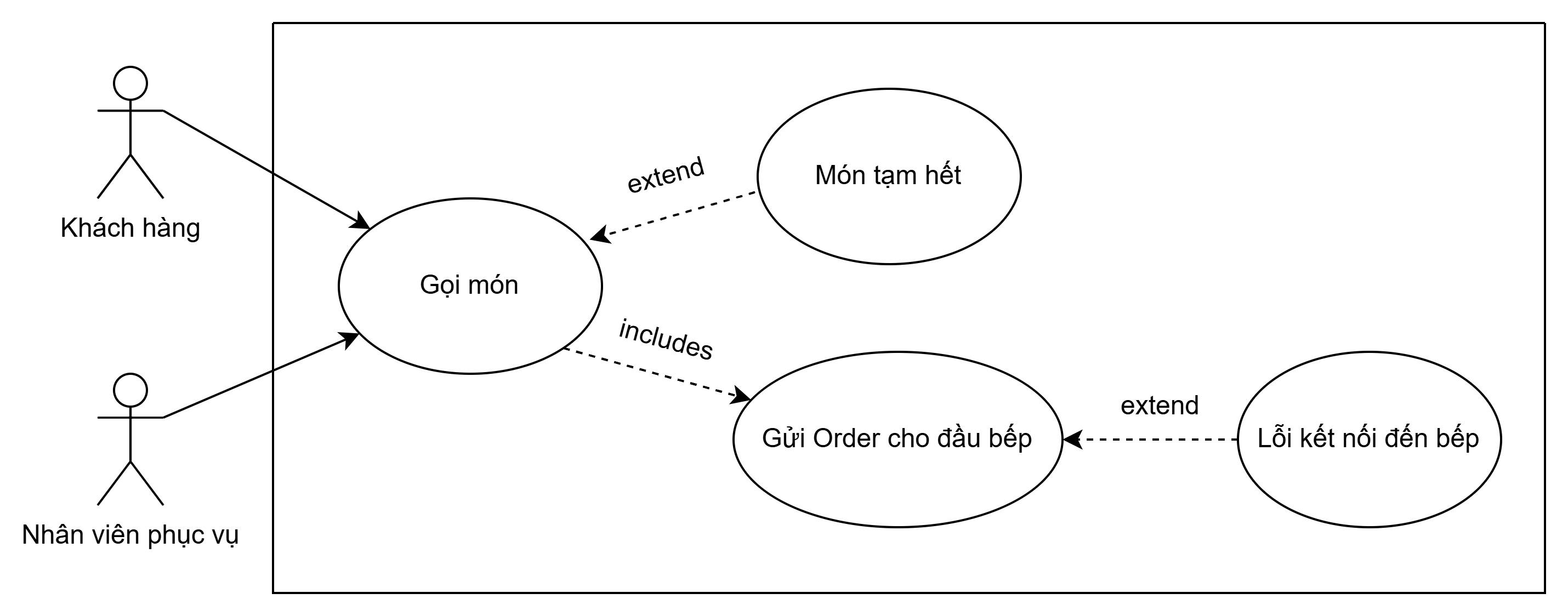
**- Luồng sự kiện chính (Main Flow):**

1. Khách hàng yêu cầu đặt bàn.
2. Nhân viên chọn chức năng “Đặt bàn”.
3. Hệ thống kiểm tra bàn trống. (Include → Kiểm tra bàn trống)
4. Nếu có bàn trống, nhân viên xác nhận đặt bàn.
5. Hệ thống lưu thông tin đặt bàn và cập nhật trạng thái.

**- Luồng sự kiện thay thế:**

* **A1: Không còn bàn trống (Tại bước 3):**  
  Hệ thống tự động Extend → Gợi ý bàn khác trong khu vực khác.
* **A2: Thông tin khách không hợp lệ (Tại bước 2):**  
  Nếu thiếu thông tin liên hệ, hệ thống yêu cầu nhập lại.

## 3.2.2. Use Case: Gọi món



**- Mô tả ngắn**: Chức năng cho phép Nhân viên phục vụ ghi nhận order của Khách hàng và gửi đến Đầu bếp.

**- Actor:** Nhân viên phục vụ, Khách hàng

**- Điều kiện tiên quyết:** Bàn đã được đặt hoặc đang phục vụ.

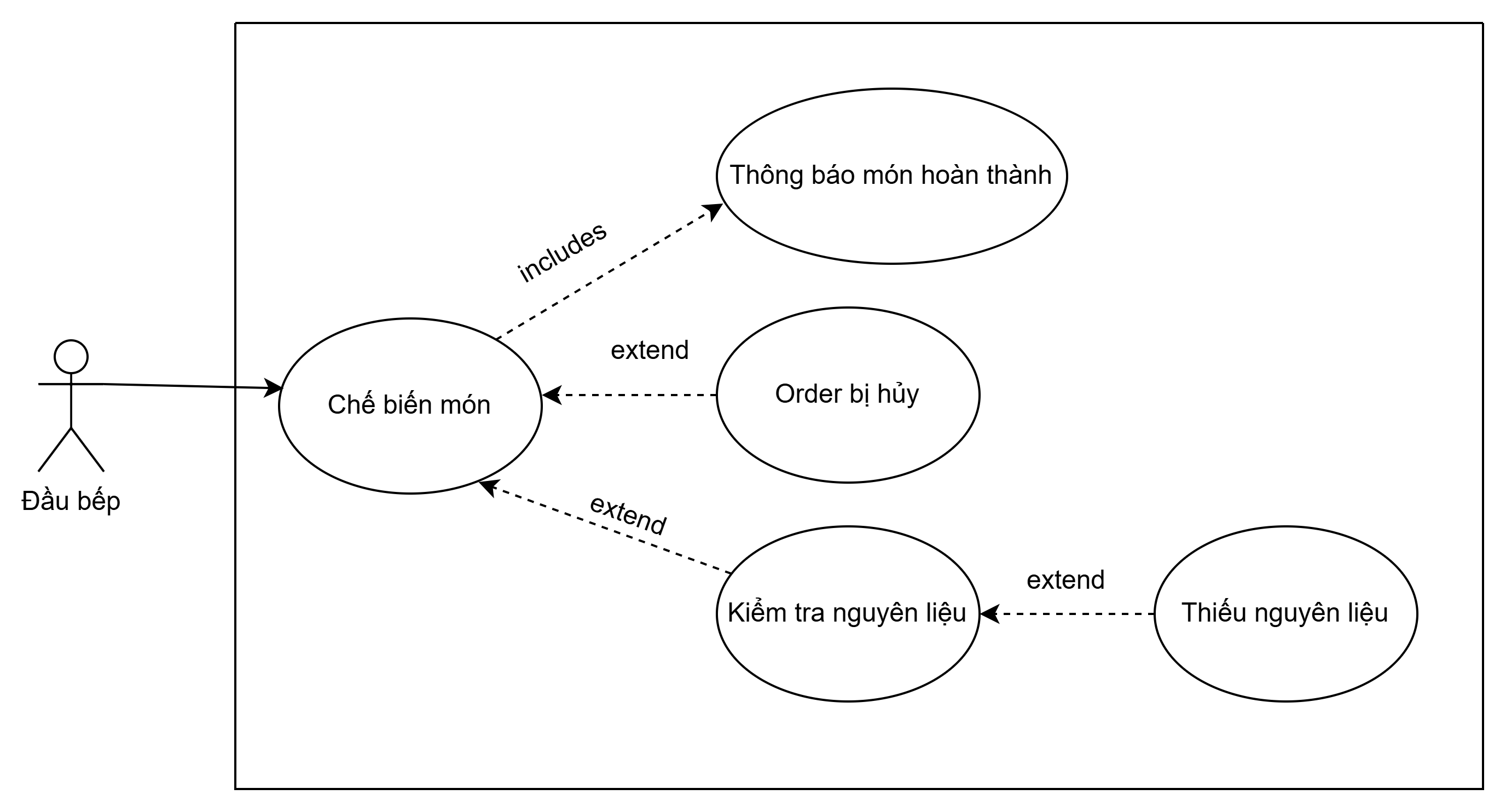
**- Luồng sự kiện chính (Main Flow):**

1. Khách hàng gọi món qua nhân viên.
2. Nhân viên chọn chức năng “Gọi món”.
3. Nhân viên nhập danh sách món khách yêu cầu.
4. Hệ thống lưu order và gửi thông tin đến bếp. (Include → Gửi order cho đầu bếp)
5. Hệ thống hiển thị xác nhận “Order đã gửi thành công.”

**- Luồng sự kiện thay thế:**

* **A1: Món tạm hết (Tại bước 3):**  
  Hệ thống thông báo “Món này tạm hết. Vui lòng chọn món khác.”
* **A2: Kết nối bếp lỗi (Tại bước 4):**  
  Hệ thống hiển thị “Không thể gửi order đến bếp, vui lòng thử lại.”

## 3.2.3. Use Case: Chế biến món



**- Mô tả ngắn:** Chức năng cho phép Đầu bếp tiếp nhận order, chế biến món, kiểm tra nguyên liệu và cập nhật trạng thái món.

**- Actor:** Đầu bếp

**- Điều kiện tiên quyết:** Order đã được gửi từ nhân viên phục vụ.

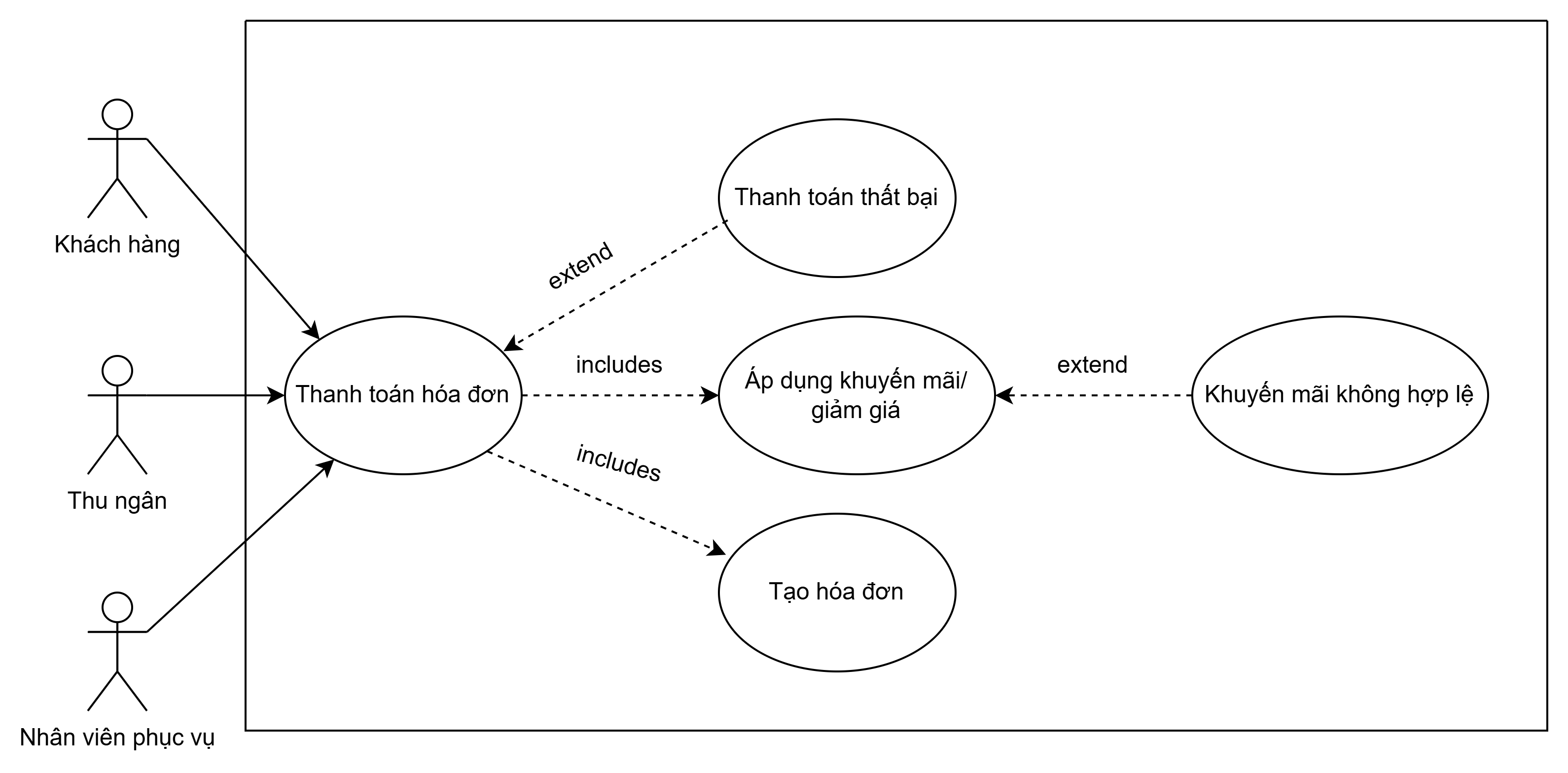
**- Luồng sự kiện chính (Main Flow):**

1. Hệ thống hiển thị danh sách order mới.
2. Đầu bếp chọn order cần chế biến.
3. Kiểm tra nguyên liệu trong kho. (Extend → Kiểm tra nguyên liệu)
4. Thực hiện chế biến món ăn.
5. Khi hoàn tất, đầu bếp cập nhật trạng thái món. (Include → Thông báo món hoàn thành)

**- Luồng sự kiện thay thế:**

* **A1: Thiếu nguyên liệu (Tại bước 3):**  
  Hệ thống cảnh báo “Nguyên liệu không đủ, vui lòng bổ sung kho.”
* **A2: Order bị hủy (Tại bước 2):**  
  Nếu order bị nhân viên hủy, hệ thống hiển thị thông báo “Order này đã bị hủy.”

## 3.2.4. Use Case: Thanh toán hóa đơn



**- Mô tả ngắn:** Chức năng cho phép Thu ngân xử lý thanh toán cho Khách hàng sau khi dùng bữa, in hóa đơn và ghi nhận doanh thu.

**- Actor:** Thu ngân, Khách hàng, Nhân viên phục vụ

**- Điều kiện tiên quyết:** Hóa đơn đã được tạo từ order của bàn.

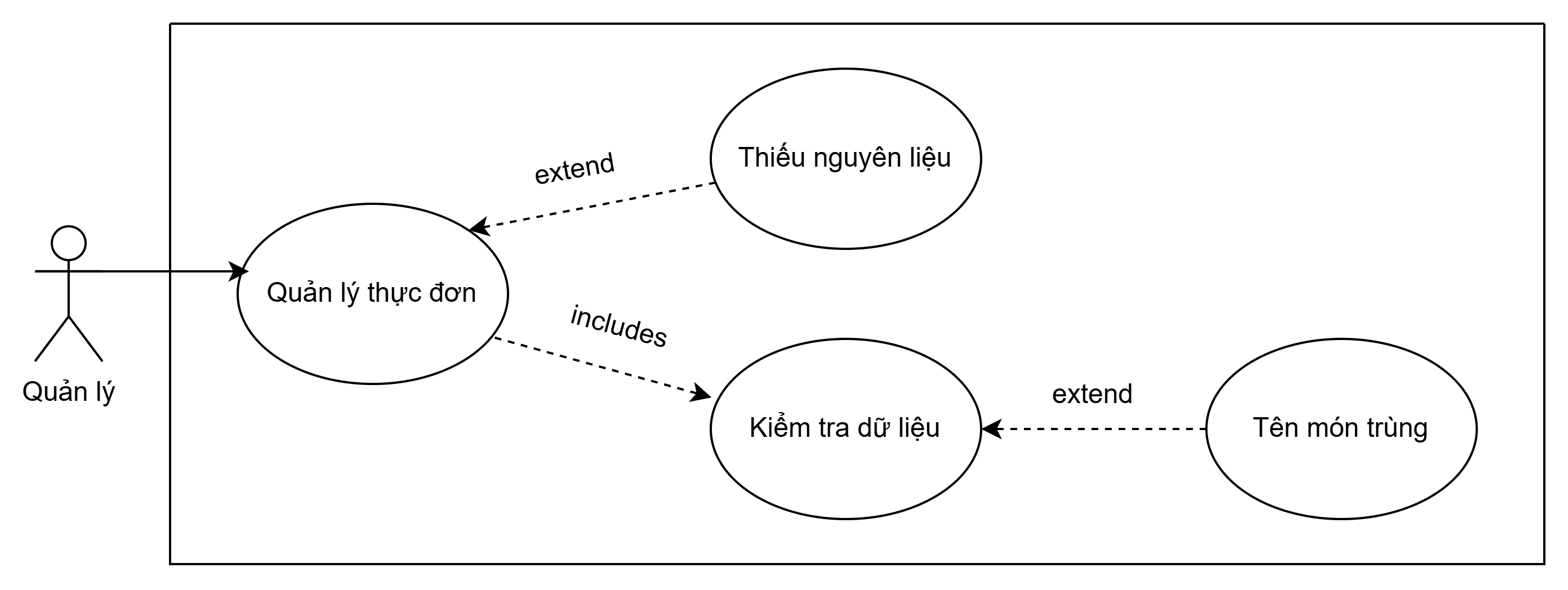
**- Luồng sự kiện chính (Main Flow):**

1. Khách hàng yêu cầu thanh toán.
2. Thu ngân chọn chức năng “Thanh toán hóa đơn”.
3. Hệ thống hiển thị chi tiết hóa đơn. (Include → Tạo hóa đơn)
4. Thu ngân xác nhận phương thức thanh toán và áp dụng ưu đãi nếu có. (Include → Áp dụng khuyến mãi/giảm giá)
5. Hệ thống xử lý giao dịch, in hóa đơn và lưu doanh thu.
6. Hệ thống hiển thị thông báo “Thanh toán thành công.”

**- Luồng sự kiện thay thế:**

* **A1: Thanh toán thất bại (Tại bước 5):**  
  Nếu lỗi kết nối hoặc hết tiền, hệ thống hiển thị “Thanh toán không thành công, vui lòng thử lại.”
* **A2: Không áp dụng được khuyến mãi (Tại bước 4):**  
  Hệ thống thông báo “Mã khuyến mãi không hợp lệ hoặc đã hết hạn.”

## 3.2.5. Use Case: Quản lý thực đơn



**- Mô tả ngắn:** Cho phép Quản lý thêm, sửa, xóa món ăn và cập nhật giá, nguyên liệu, tình trạng món.

- **Actor:** Quản lý

**- Điều kiện tiên quyết:** Quản lý đã đăng nhập và có quyền truy cập chức năng quản lý thực đơn.

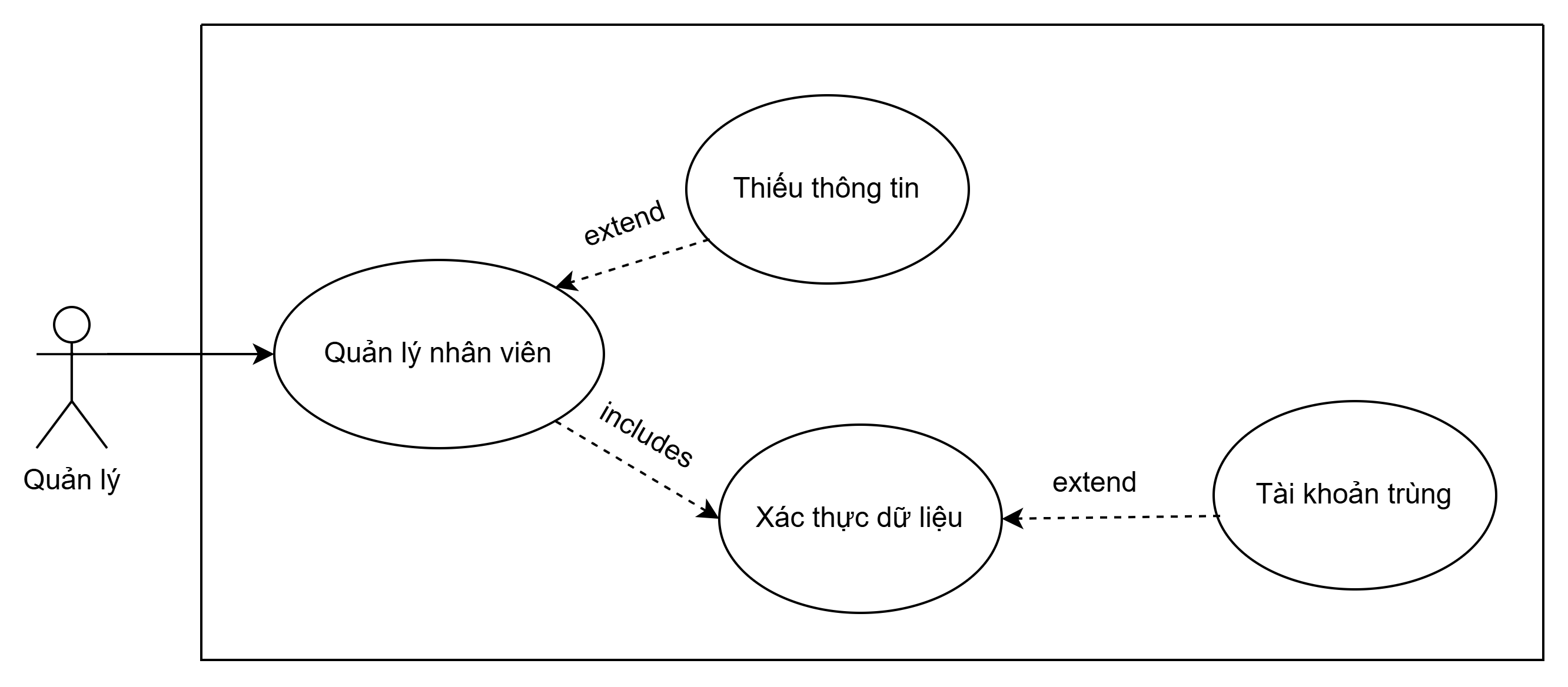
**- Luồng sự kiện chính (Main Flow):**

1. Quản lý chọn “Quản lý thực đơn”.
2. Hệ thống hiển thị danh sách các món ăn hiện có.
3. Quản lý chọn thêm mới hoặc chỉnh sửa món.
4. Nhập thông tin món ăn: tên, giá, nguyên liệu, tình trạng.
5. Hệ thống kiểm tra dữ liệu.
6. Lưu thông tin và cập nhật thực đơn.
7. Hệ thống thông báo “Cập nhật thực đơn thành công.”

**- Luồng sự kiện thay thế:**

* **A1: Tên món trùng (Tại bước 5):**  
  Nếu tên món đã tồn tại, hệ thống báo “Tên món này đã có trong thực đơn.”
* **A2: Thiếu thông tin nguyên liệu (Tại bước 4):**  
  Nếu người dùng không nhập nguyên liệu, hệ thống cảnh báo “Vui lòng bổ sung nguyên liệu.”

## 3.2.6. Use Case: Xem báo cáo doanh thu



**- Mô tả ngắn:** Cho phép Quản lý xem tổng hợp doanh thu theo ngày, tuần hoặc tháng.

**- Actor:** Quản lý

**- Điều kiện tiên quyết:** Quản lý đã đăng nhập.

**- Luồng sự kiện chính (Main Flow):**

1. Quản lý chọn chức năng “Xem báo cáo doanh thu”.
2. Hệ thống kiểm tra quyền truy cập. (Include → Kiểm tra quyền truy cập)
3. Quản lý chọn khoảng thời gian cần xem.
4. Hệ thống tổng hợp và hiển thị doanh thu tương ứng.
5. Quản lý có thể in hoặc xuất file báo cáo.

**- Luồng sự kiện thay thế:**

* **A1: Không có dữ liệu (Tại bước 4):**  
  Nếu trong khoảng thời gian chọn không có giao dịch, hệ thống hiển thị “Không có dữ liệu doanh thu.”
* **A2: Không đủ quyền (Tại bước 2):**  
  Nếu người dùng không phải quản lý, hệ thống báo “Bạn không có quyền truy cập chức năng này.”

## 3.2.7. Use Case: Quản lý nhân viên

**- Mô tả ngắn:** Chức năng cho phép Quản lý thêm mới, phân quyền và cập nhật thông tin nhân viên.

**- Actor:** Quản lý

**- Điều kiện tiên quyết:** Quản lý đã đăng nhập và có quyền quản trị nhân sự.

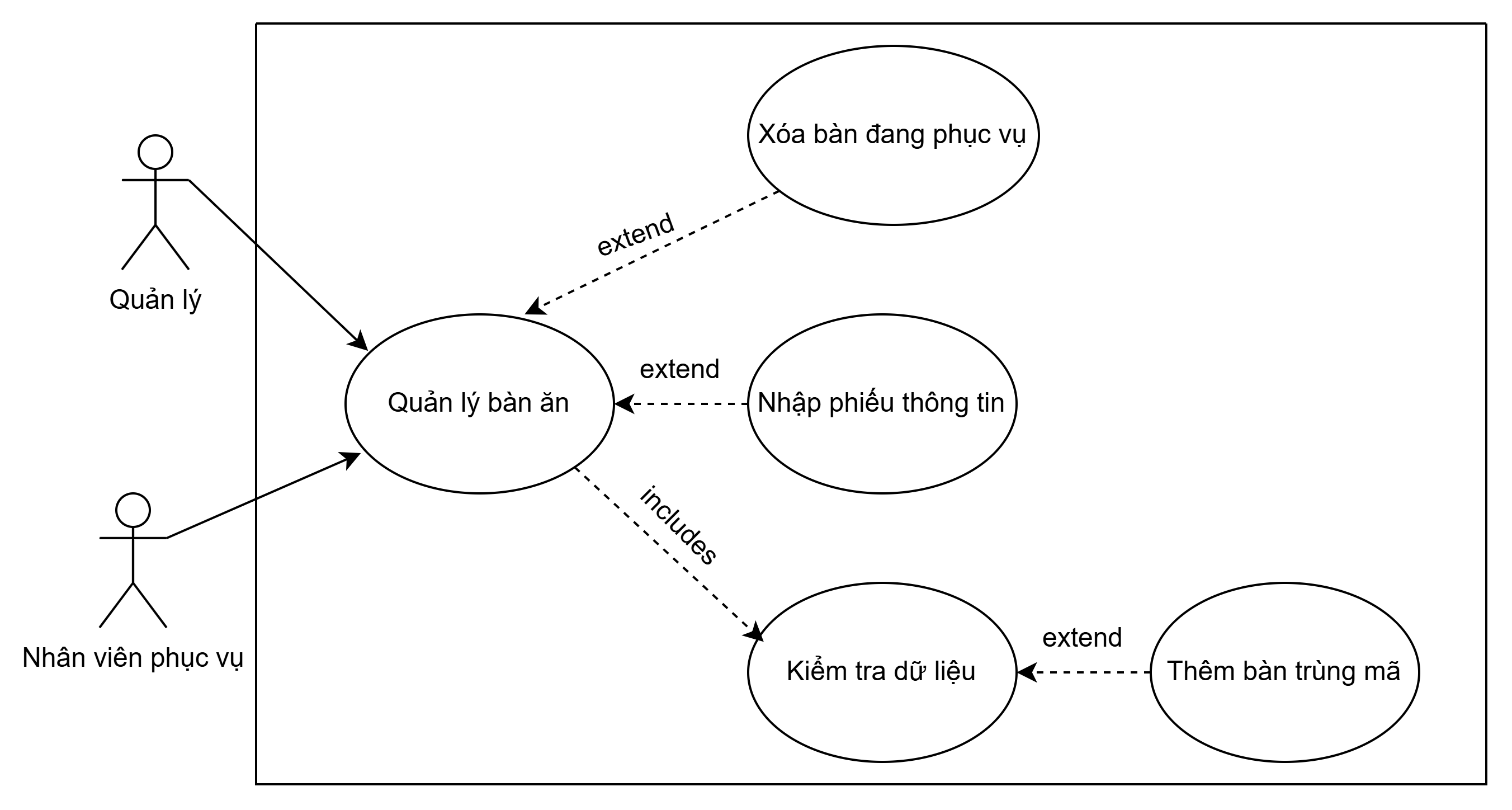
**- Luồng sự kiện chính (Main Flow):**

1. Quản lý chọn chức năng “Quản lý nhân viên”.
2. Hệ thống hiển thị danh sách nhân viên.
3. Quản lý thêm mới hoặc sửa thông tin nhân viên.
4. Nhập dữ liệu nhân viên (tên, chức vụ, tài khoản, phân quyền...).
5. Hệ thống xác thực dữ liệu.
6. Hệ thống lưu thông tin và hiển thị thông báo “Cập nhật thành công.”

**- Luồng sự kiện thay thế:**

* **A1: Thiếu thông tin (Tại bước 4):**  
  Nếu bỏ trống trường bắt buộc, hệ thống báo “Vui lòng nhập đầy đủ thông tin.”
* **A2: Tài khoản trùng (Tại bước 5):**  
  Nếu tên tài khoản đã tồn tại, hệ thống thông báo “Tên đăng nhập đã được sử dụng.”

## 3.2.8. Use Case: Quản lý bàn ăn



**- Mô tả ngắn:** Chức năng cho phép Quản lý và Nhân viên phục vụ thêm, sửa, xóa và cập nhật trạng thái bàn (trống, đã đặt, đang phục vụ).

**- Actor:** Quản lý, Nhân viên phục vụ

**- Điều kiện tiên quyết:** Người dùng đã đăng nhập thành công và có quyền truy cập chức năng quản lý bàn.

**- Luồng sự kiện chính (Main Flow):**

1. Actor chọn chức năng "Quản lý bàn ăn".
2. Hệ thống hiển thị danh sách các bàn hiện có cùng trạng thái.
3. Actor chọn thêm mới, chỉnh sửa hoặc xóa bàn.
4. Hệ thống hiển thị form tương ứng để nhập thông tin.
5. Actor nhập đầy đủ thông tin bàn (số bàn, khu vực, trạng thái...).
6. Hệ thống kiểm tra tính hợp lệ dữ liệu.
7. Hệ thống lưu thay đổi và cập nhật danh sách bàn.
8. Hệ thống hiển thị thông báo “Cập nhật bàn thành công”.

**- Luồng sự kiện thay thế (Alternative Flows):**

* **A1: Nhập thiếu thông tin (Tại bước 5):**  
  Nếu người dùng để trống các trường bắt buộc, hệ thống hiển thị thông báo “Vui lòng nhập đầy đủ thông tin bàn.”
* **A2: Thêm bàn trùng mã (Tại bước 6):**  
  Nếu mã bàn đã tồn tại, hệ thống hiển thị thông báo “Số bàn này đã tồn tại trong hệ thống.”
* **A3: Xóa bàn đang phục vụ (Tại bước 3):**  
  Nếu bàn đang phục vụ khách, hệ thống hiển thị cảnh báo “Bàn đang được sử dụng, không thể xóa.”

## 3.3. Yêu cầu Phi chức năng

|  |  |  |
| --- | --- | --- |
| **Mã** | **Yêu cầu** | **Mô tả** |
| **PER-01** | Hiệu năng | Thời gian phản hồi cho các thao tác như tìm kiếm bàn trống, tra cứu món ăn, xem hóa đơn phải dưới 2 giây trên tập dữ liệu khoảng 200 bàn, 500 món ăn và 10,000 giao dịch. |
| **PER-02** | Hiệu năng | Hệ thống có thể xử lý đồng thời tối thiểu 20 người dùng (bao gồm nhân viên phục vụ, đầu bếp, thu ngân, quản lý) mà không bị treo hoặc mất dữ liệu. |
| **USA-01** | Tính khả dụng | Một nhân viên mới, sau khi được hướng dẫn không quá 15 phút, phải có thể thực hiện thành thạo các nghiệp vụ chính (đặt bàn, gọi món, thanh toán) mà không cần trợ giúp. |
| **USA-02** | Tính thân thiện người dùng | Giao diện hệ thống phải đơn giản, trực quan, các nút chức năng được bố trí rõ ràng, dễ thao tác trên máy tính bảng hoặc máy POS cảm ứng. |
| **SEC-01** | Bảo mật | Chỉ Quản lý mới được truy cập các chức năng quản lý nhân viên, xem báo cáo doanh thu và cấu hình hệ thống. Các vai trò khác bị giới hạn trong phạm vi quyền hạn của mình. |
| **SEC-02** | Bảo mật dữ liệu | Mật khẩu người dùng phải được mã hóa trước khi lưu trữ, không hiển thị ở dạng văn bản gốc. Các thao tác đăng nhập/đăng xuất phải được ghi log. |
| **REL-01** | Độ tin cậy | Hệ thống phải có thời gian hoạt động (uptime) đạt tối thiểu 99.5% trong giờ làm việc. Dữ liệu không được mất trong trường hợp hệ thống tạm thời gián đoạn. |
| **REL-02** | Sao lưu dữ liệu | Dữ liệu phải được sao lưu tự động hàng ngày và có thể phục hồi trong vòng 10 phút khi xảy ra sự cố. |
| **MAI-01** | Bảo trì | Hệ thống phải cho phép cập nhật dữ liệu thực đơn, nhân viên, bàn ăn mà không ảnh hưởng đến hoạt động đang diễn ra. |
| **POR-01** | Khả năng mở rộng | Hệ thống phải có khả năng mở rộng để hỗ trợ nhiều chi nhánh nhà hàng trong tương lai, với dữ liệu và tài khoản được quản lý độc lập theo từng chi nhánh. |

## 3.4. Yêu cầu về Cơ sở dữ liệu Logic

- Hệ thống sử dụng cơ sở dữ liệu quan hệ (RDBMS) để đảm bảo tính toàn vẹn và nhất quán dữ liệu.

**- Mô tả các thực thể chính:**

**+ CUSTOMER**

* CustomerID (PK): Mã định danh duy nhất của khách hàng.
* Name: Tên khách hàng.
* Phone: Số điện thoại của khách hàng, dùng để liên hệ hoặc xác nhận đặt bàn.
* LoyaltyPoints: Điểm thưởng khách hàng tích lũy, sử dụng trong chương trình khách hàng thân thiết.

**+ TABLE**

* TableID (PK): Mã định danh duy nhất của bàn ăn.
* TableNumber: Số thứ tự của bàn, dùng hiển thị trong nhà hàng.
* Capacity: Sức chứa tối đa của bàn (số người).
* Location: Vị trí bàn trong nhà hàng (ví dụ: gần cửa, phòng riêng).
* Status: Trạng thái bàn: "Trống", "Đang phục vụ", "Đặt trước".

**+ RESERVATION**

* ReservationID (PK): Mã định danh duy nhất của mỗi đặt bàn.
* TableID (FK): Khóa ngoại liên kết bàn được đặt.
* CustomerID (FK): Khóa ngoại liên kết khách hàng đặt bàn.
* ReservationTime: Thời gian đặt bàn.
* Status: Trạng thái đặt bàn: "Đang chờ", "Đã xác nhận", "Hủy".

**+ EMPLOYEE**

* EmployeeID (PK): Mã định danh duy nhất của nhân viên.
* Name: Tên nhân viên.
* Role: Vai trò nhân viên: "Thu ngân", "Phục vụ", "Quản lý".
* Phone: Số điện thoại của nhân viên.
* Status: Trạng thái làm việc: "Đang hoạt động", "Đã nghỉ".
* Username: Tài khoản đăng nhập hệ thống.
* Password: Mật khẩu đăng nhập.

**+ DISH**

* DishID (PK): Mã định danh duy nhất của món ăn.
* Name: Tên món ăn.
* Price: Giá món ăn.
* Description: Mô tả món ăn, bao gồm nguyên liệu và cách chế biến.
* Availability: Trạng thái món ăn: "Có sẵn", "Hết món".
* Category: Loại món ăn: "Khai vị", "Món chính", "Tráng miệng"...

**+ ORDER**

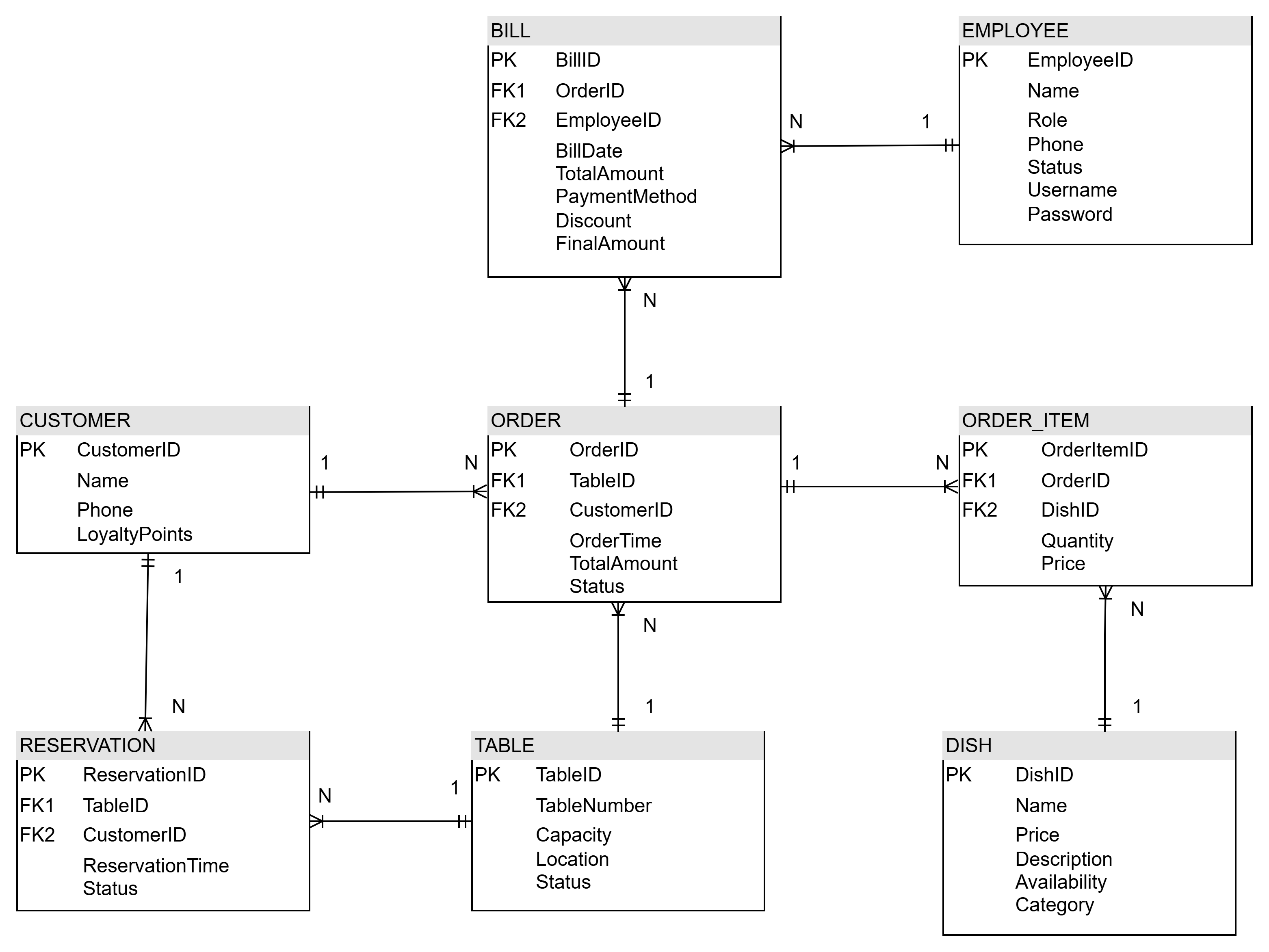
* OrderID (PK): Mã định danh duy nhất của order.
* TableID (FK): Khóa ngoại liên kết bàn phục vụ order này.
* CustomerID (FK): Khóa ngoại liên kết khách hàng tạo order.
* OrderTime: Thời gian order được tạo.
* TotalAmount: Tổng tiền của order.
* Status: Trạng thái order: "Đang phục vụ", "Hoàn thành", "Hủy".

**+ ORDER\_ITEM**

* OrderItemID (PK): Mã định danh chi tiết món trong order.
* OrderID (FK): Khóa ngoại liên kết order chứa món này.
* DishID (FK): Khóa ngoại liên kết món ăn.
* Quantity: Số lượng món gọi.
* Price: Giá món tại thời điểm order (dùng để tính tổng).

**+ BILL**

* BillID (PK): Mã định danh hóa đơn.
* OrderID (FK): Khóa ngoại liên kết order được thanh toán.
* EmployeeID (FK): Khóa ngoại liên kết nhân viên lập hóa đơn.
* BillDate: Ngày lập hóa đơn.
* TotalAmount: Tổng tiền order trước khi áp dụng chiết khấu.
* Discount: Mức giảm giá (nếu có).
* FinalAmount: Số tiền thanh toán sau chiết khấu.
* PaymentMethod: Hình thức thanh toán: "Tiền mặt", "Thẻ", "Ví điện tử".



**- Mối quan hệ và khóa ngoại**

**+ CUSTOMER – RESERVATION**

* Mối quan hệ: Một khách hàng có thể đặt nhiều bàn (1:N).
* Điều này có nghĩa một khách hàng có thể thực hiện nhiều lượt đặt bàn khác nhau, nhưng mỗi lượt đặt bàn chỉ thuộc về một khách hàng duy nhất.
* Khóa ngoại: CustomerID trong RESERVATION tham chiếu CUSTOMER.CustomerID.
* Giải thích: Khóa ngoại này đảm bảo rằng mỗi lượt đặt bàn luôn gắn với một khách hàng hợp lệ trong hệ thống, tránh việc đặt bàn “vô chủ”.

**+ TABLE – RESERVATION**

* Mối quan hệ: Một bàn có thể được đặt nhiều lần theo thời gian (1:N).
* Một bàn có thể xuất hiện trong nhiều lượt đặt bàn khác nhau (ví dụ: trưa đặt khách A, tối đặt khách B).
* Khóa ngoại: TableID trong RESERVATION tham chiếu TABLE.TableID.
* Giải thích: Khóa ngoại này giúp hệ thống biết bàn nào được đặt trong từng lượt và kiểm soát tình trạng bàn (trống, đặt trước).

**+ CUSTOMER – ORDER**

* Mối quan hệ: Một khách hàng có thể tạo nhiều order (1:N).
* Khách hàng có thể gọi nhiều order trong một ngày hoặc nhiều ngày khác nhau.
* Khóa ngoại: CustomerID trong ORDER tham chiếu CUSTOMER.CustomerID.
* Giải thích: Mỗi order phải thuộc về một khách hàng cụ thể để quản lý lịch sử mua hàng, tính điểm thưởng, hoặc liên kết với hóa đơn.

**+ TABLE – ORDER**

* Mối quan hệ: Một bàn có thể phục vụ nhiều order theo thời gian (1:N).
* Một bàn có thể phục vụ nhiều lượt order, nhưng mỗi order chỉ thuộc một bàn duy nhất.
* Khóa ngoại: TableID trong ORDER tham chiếu TABLE.TableID.
* Giải thích: Giúp theo dõi lịch sử order theo bàn, kiểm soát tình trạng bàn đang phục vụ hay trống.

**+ ORDER – ORDER\_ITEM**

* Mối quan hệ: Một order có thể chứa nhiều món (1:N).
* Mỗi order có thể gọi nhiều món khác nhau, nhưng mỗi món trong chi tiết order chỉ thuộc một order duy nhất.
* Khóa ngoại: OrderID trong ORDER\_ITEM tham chiếu ORDER.OrderID.
* Giải thích: Giúp liên kết từng món gọi với order, để tính tổng tiền và quản lý chế biến.

**+ DISH – ORDER\_ITEM**

* Mối quan hệ: Một món ăn có thể xuất hiện trong nhiều order\_item (1:N).
* Một món như “Phở bò” có thể được gọi trong nhiều order khác nhau.
* Khóa ngoại: DishID trong ORDER\_ITEM tham chiếu DISH.DishID.
* Giải thích: Giúp hệ thống biết món nào được gọi và quản lý tồn kho, trạng thái món (“còn món” hay “hết món”).

**+ ORDER – BILL**

* Mối quan hệ: Một order có thể có một hóa đơn (1:1).
* Mỗi order khi thanh toán sẽ tạo ra một hóa đơn duy nhất.
* Khóa ngoại: OrderID trong BILL tham chiếu ORDER.OrderID.
* Giải thích: Đảm bảo mỗi hóa đơn liên kết chính xác với order cần thanh toán.

**+ EMPLOYEE – BILL**

* Mối quan hệ: Một nhân viên có thể lập nhiều hóa đơn (1:N).
* Mỗi hóa đơn được lập bởi một nhân viên, nhưng một nhân viên có thể lập nhiều hóa đơn trong ca làm việc.
* Khóa ngoại: EmployeeID trong BILL tham chiếu EMPLOYEE.EmployeeID.
* Giải thích: Giúp theo dõi nhân viên tạo hóa đơn, phục vụ báo cáo doanh thu hoặc phân quyền quản lý.

**- Cardinality & Optionality**

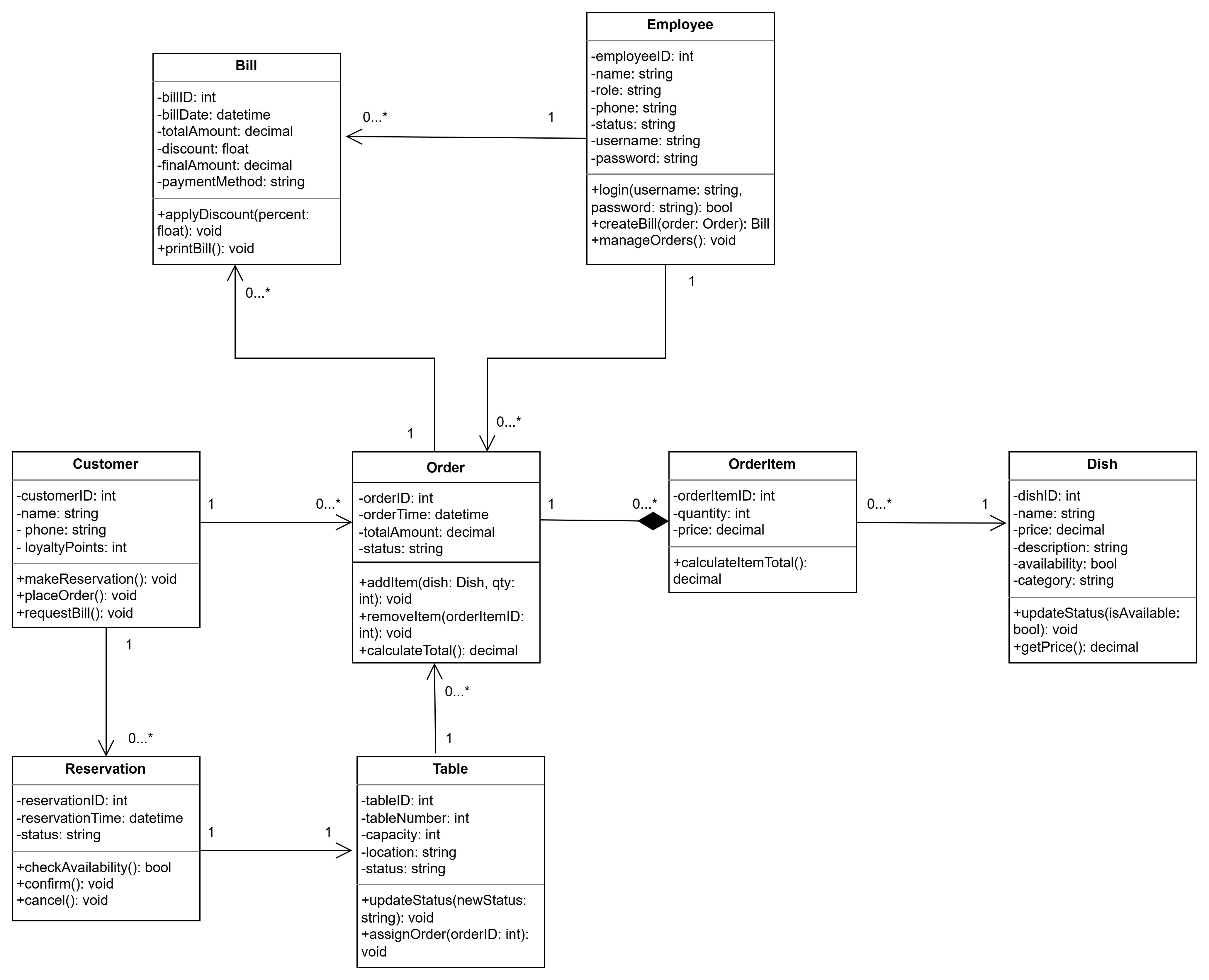
|  |  |  |  |
| --- | --- | --- | --- |
| **Quan hệ** | **Cardinality** | **Optionality** | **Giải thích** |
| **CUSTOMER → RESERVATION** | 1..\* | Optional | Một khách hàng có thể chưa đặt bàn hoặc có thể đặt nhiều lần. |
| **TABLE → RESERVATION** | 1..\* | Optional | Một bàn có thể chưa từng được đặt hoặc được đặt nhiều lần theo thời gian. |
| **CUSTOMER → ORDER** | 1..\* | Optional | Một khách hàng có thể chưa tạo order nào hoặc có thể tạo nhiều order. |
| **TABLE → ORDER** | 1..\* | Optional | Một bàn có thể chưa phục vụ order nào hoặc có thể phục vụ nhiều order. |
| **ORDER → ORDER\_ITEM** | 1..\* | Mandatory | Mỗi order phải có ít nhất một món trong chi tiết order. |
| **DISH → ORDER\_ITEM** | 1..\* | Optional | Một món ăn có thể chưa từng được order hoặc xuất hiện trong nhiều chi tiết order. |
| **ORDER → BILL** | 1..1 | Mandatory | Khi thanh toán, mỗi order tạo duy nhất một hóa đơn. |
| **EMPLOYEE → BILL** | 1..\* | Mandatory | Một nhân viên có thể lập một hoặc nhiều hóa đơn trong ca làm việc. |

# PHỤ LỤC

## Biểu đồ Lớp (Class Diagram)

**- Mô tả các Class**

|  |  |  |
| --- | --- | --- |
| **Class** | **Thuộc tính** | **Phương thức** |
| **Customer** | * customerID: int – Mã định danh duy nhất cho mỗi khách hàng. * name: string – Họ tên khách hàng. * phone: string – Số điện thoại liên hệ. * loyaltyPoints: int – Điểm tích lũy thành viên | * makeReservation() – Đặt bàn. * placeOrder() – Gọi món. * requestBill() – Yêu cầu thanh toán. |
| **Reservation** | * reservationID: int – Mã đặt bàn. * reservationTime: datetime – Thời gian đặt. * status: string – Trạng thái (Pending, Confirmed, Canceled…). | * checkAvailability() – Kiểm tra bàn trống. * confirm() – Xác nhận đặt bàn. * cancel() – Hủy đặt bàn. |
| **Table** | * tableID: int – Mã bàn. * tableNumber: int – Số thứ tự bàn. * capacity: int – Số chỗ ngồi. * location: string – Khu vực/ vị trí bàn. * status: string – Trạng thái (Trống, Đang phục vụ, Đã đặt…). | * updateStatus() – Cập nhật trạng thái bàn. * assignOrder() – Gán đơn hàng vào bàn. |
| **Order** | * orderID: int – Mã đơn hàng. * orderTime: datetime – Thời gian tạo. * totalAmount: double – Tổng giá trị đơn hàng. * status: string – Trạng thái (Pending, In Progress, Completed). | * addItem() – Thêm món vào đơn. * removeItem() – Xóa món khỏi đơn. * calculateTotal() – Tính tổng tiền. |
| **OrderItem** | * orderItemID: int – Mã chi tiết đơn hàng. * quantity: int – Số lượng món. * price: double – Giá món tại thời điểm đặt. | * calculateItemTotal() – Tính tổng tiền cho món. |
| **Dish** | * dishID: int – Mã món ăn. * name: string – Tên món. * price: double – Giá bán. * description: string – Mô tả món. * availability: string – Tình trạng (Còn / Hết hàng). * category: string – Loại món (Khai vị, Món chính, Tráng miệng…). | * updateStatus() – Cập nhật tình trạng món. * getPrice() – Lấy giá hiện tại. |
| **Bill** | * billID: int – Mã hóa đơn. * billDate: datetime – Ngày lập hóa đơn. * totalAmount: double – Tổng tiền trước giảm giá. * discount: double – Mức giảm giá. * finalAmount: double – Tổng tiền sau giảm giá. * paymentMethod: string – Hình thức thanh toán (tiền mặt, thẻ, ví điện tử…). | * applyDiscount() – Áp dụng khuyến mãi. * printBill() – In hóa đơn. |
| **Employee** | * employeeID: int – Mã nhân viên. * name: string – Họ tên. * role: string – Vai trò (Phục vụ, Đầu bếp, Thu ngân, Quản lý). * phone: string – Số điện thoại. * status: string – Trạng thái làm việc. * username: string – Tên đăng nhập. * password: string – Mật khẩu. | * login() – Đăng nhập hệ thống. * createBill() – Lập hóa đơn. * manageOrders() – Quản lý đơn hàng. |



**- Các quan hệ chính**

|  |  |  |
| --- | --- | --- |
| **Mối quan hệ** | **Multiplicity** | **Mô tả** |
| **Customer – Reservation** | 1 – 0..\* | Một khách hàng có thể đặt nhiều bàn, nhưng có thể chưa có đặt bàn nào. Mối quan hệ này là association, vì Reservation tồn tại độc lập với Customer. |
| **Reservation – Table** | 1 – 1 | Mỗi đặt bàn liên kết với đúng một bàn. Bàn ăn không bị xóa khi đặt bàn bị hủy. |
| **Customer – Order** | 1 – 0..\* | Một khách hàng có thể có nhiều đơn đặt món, nhưng cũng có thể chưa có đơn nào. |
| **Table – Order** | 1 – 0..\* | Một bàn có thể có nhiều đơn hàng trong cùng hoặc nhiều lượt phục vụ khác nhau. |
| **Order – OrderItem** | 1 – 0..\* (Composition) | Một đơn hàng bao gồm nhiều chi tiết món (OrderItem). Khi Order bị xóa, các OrderItem cũng bị xóa theo. |
| **OrderItem – Dish** | 0..\* – 1 | Mỗi chi tiết món chỉ liên kết với một món ăn, nhưng một món có thể xuất hiện trong nhiều chi tiết khác nhau. |
| **Order – Bill** | 1 – 1 | Mỗi đơn hàng tạo ra một hóa đơn. Hóa đơn tồn tại độc lập với đơn hàng nếu được lưu trữ riêng. |
| **Employee – Bill** | 1 – 0..\* | Một nhân viên có thể tạo nhiều hóa đơn trong quá trình làm việc. |
| **Employee – Order** | 1 – 0..\* | Một nhân viên có thể tiếp nhận hoặc phục vụ nhiều đơn hàng khác nhau. |

# Biểu đồ Tuần Tự (Sequence Diagram)

## Biểu đồ tuần tự – UC: Đặt bàn

**- Actors & Layers**

**+ Actors**

* **Khách hàng:** Đưa ra yêu cầu đặt bàn (miệng hoặc trực tiếp tại quầy), cung cấp thông tin số người và thời gian mong muốn, chọn bàn khi có danh sách bàn trống.
* **Nhân viên phục vụ:** Nhận yêu cầu của khách → nhập thông tin → thao tác trên hệ thống → thông báo kết quả cho khách.

**+ UI Layer**

* Giao diện nơi nhân viên phục vụ nhập số người, thời gian.
* Hiển thị danh sách bàn trống.
* Hiển thị kết quả đặt bàn thành công hoặc gợi ý phương án khác.

**+ Business Layer**

* Tiếp nhận yêu cầu kiểm tra bàn trống từ UI.
* Gửi yêu cầu truy vấn xuống Data Layer.
* Nhận dữ liệu bàn trống, xử lý logic:

Nếu có bàn trống → tiếp nhận lựa chọn bàn từ UI → lưu reservation.

Nếu không có bàn trống → tạo gợi ý bàn/thời gian khác.

* Trả nghiệp vụ kết quả về UI.

**+ Data Layer**

* Truy vấn danh sách bàn trống theo thời gian.
* Lưu thông tin reservation khi khách chọn bàn.
* Trả lại kết quả xác nhận cho Business Layer.

**- Luồng thông điệp chính**

1. **Khách hàng → Nhân viên phục vụ:** Yêu cầu đặt bàn (số người, thời gian mong muốn).
2. **Nhân viên phục vụ → UI Layer:** Nhập thông tin vào hệ thống.
3. **UI → Business Layer:** Gửi yêu cầu kiểm tra bàn trống.
4. **Business → Data Layer:** Truy vấn các bàn trống theo thời gian yêu cầu.
5. **Data Layer → Business:** Trả danh sách bàn trống.
6. **Hệ thống phân nhánh kết quả:**

**+ Nhánh 1 – Có bàn trống:**

* Business gửi UI danh sách bàn trống.
* UI hiển thị danh sách → Nhân viên phục vụ thông báo cho khách.
* Khách hàng chọn bàn.
* UI gửi Business yêu cầu lưu reservation cho bàn được chọn.
* Business gửi Data lưu reservation.
* Data xác nhận thành công → trả về Business.
* Business phản hồi UI: "Đặt bàn thành công".

**+ Nhánh 2 – Không có bàn trống:**

* Business gửi UI gợi ý bàn/thời gian khác.
* UI hiển thị gợi ý.
* Nhân viên phục vụ thông báo lại cho khách phương án thay thế.

1. **UI → Nhân viên phục vụ**: Hiển thị kết quả cuối cùng.
2. **Nhân viên phục vụ → Khách hàng:** Truyền đạt lại kết quả đặt bàn.

**- Combined Fragment**

+ Sử dụng **alt** để mô tả hai tình huống loại trừ nhau:

* Nhánh 1: Có bàn trống → Khách chọn bàn → Lưu reservation → Báo thành công.
* Nhánh 2: Không có bàn trống → Gợi ý thời gian/bàn khác.

+ Đây là cách thể hiện chính xác nhất nghiệp vụ đặt bàn (chỉ có 1 kết quả xảy ra).

**- Activation Bar**

+ **UI Layer:** bật khi nhập thông tin & hiển thị kết quả.

**+ Business Layer:** bật trong suốt quy trình xử lý nghiệp vụ.

**+ Data Layer:** bật khi truy vấn bàn trống & khi lưu reservation.

**+ Actors:** Không dùng activation bar (vì chỉ kích hoạt thao tác chứ không xử lý nội bộ).

**- Điểm nổi bật**

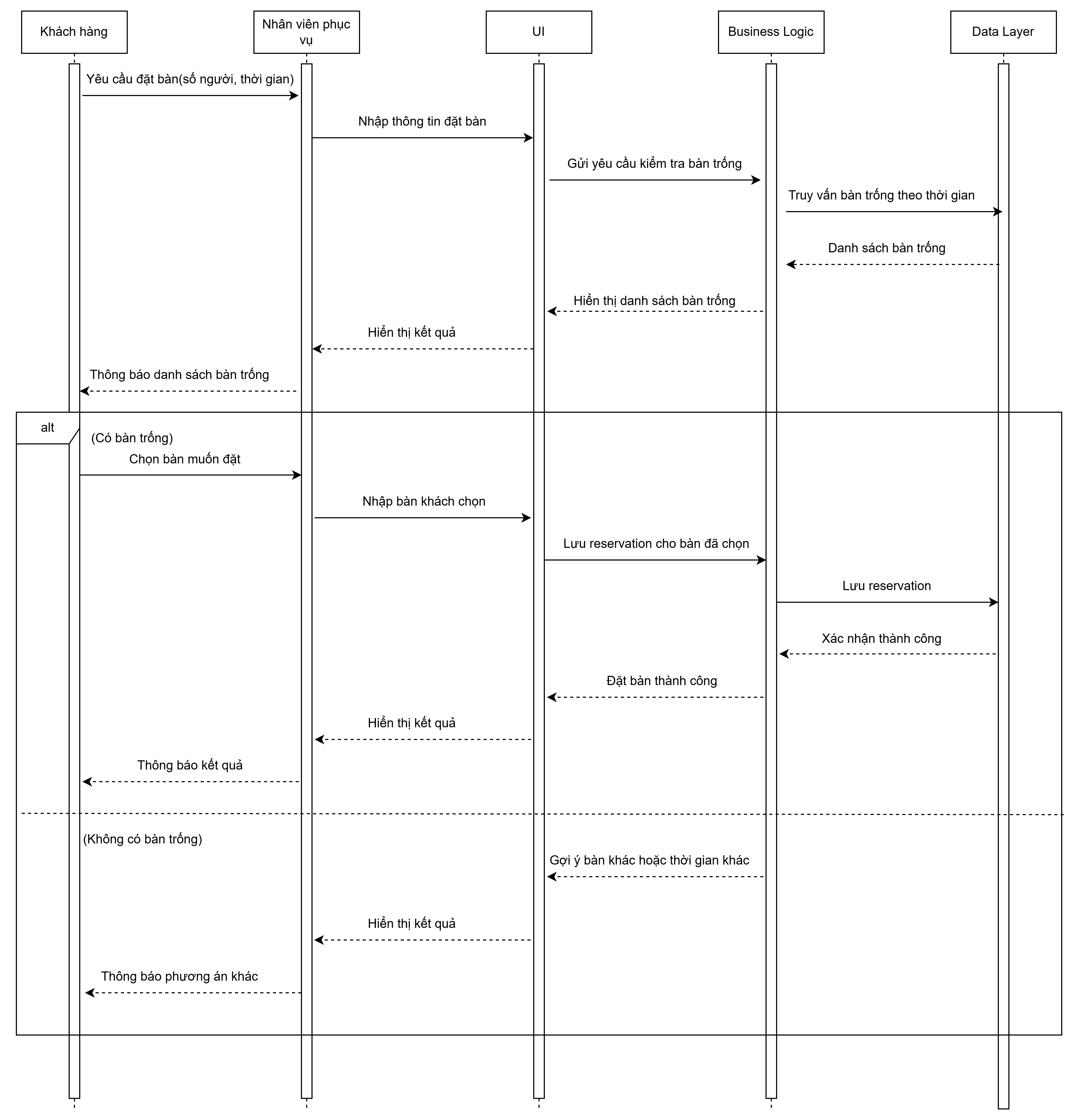
+ Sử dụng đúng 2 actor đại diện cho quá trình thực tế tại nhà hàng.

+ Logic chia thành **4 layer rõ ràng** (Actor – UI – Business – Data).

+ Áp dụng **alt fragment** để thể hiện 2 nhánh nghiệp vụ diễn ra trong thực tế.

+ Luồng thông điệp đầy đủ, có thêm bước quan trọng: **Khách hàng chọn bàn trong danh sách bàn trống.**

+ Sơ đồ đảm bảo tính chính xác trong xử lý nghiệp vụ đặt bàn của nhà hàng.



## Biểu đồ tuần tự – UC: Gọi món

**- Actors & Layers**

**+ Actors:**

* Khách hàng: chọn món.
* Nhân viên phục vụ**:** nhập món khách chọn vào hệ thống.

+ **UI Layer:** giao diện nhận món từ khách và nhân viên, hiển thị kết quả.

+ **Business Layer:** xử lý logic kiểm tra món có trong menu, lưu order.

+ **Data Layer:** lưu trữ thông tin menu và order.

**- Luồng thông điệp chính**

1. Khách hàng chọn món → Nhân viên phục vụ nhập món vào tablet.
2. UI Layer gửi danh sách order tới Business Layer.
3. **Loop**: Business Layer kiểm tra từng món trong order:

* Nếu món có trong menu → xác nhận hợp lệ.
* Nếu món không có → thông báo món không có.

1. Lưu toàn bộ order vào Data Layer.
2. Trả về UI Layer hiển thị kết quả thành công/không hợp lệ cho cả nhân viên và khách hàng.

**- Combined Fragment**

+ **loop**: lặp lại kiểm tra từng món trong order.

+ **alt**: xử lý 2 tình huống món hợp lệ hoặc món không có trong menu.

**- Activation Bar**

+ UI, Business, Data Layer đều có activation bar thể hiện thời gian thực thi.

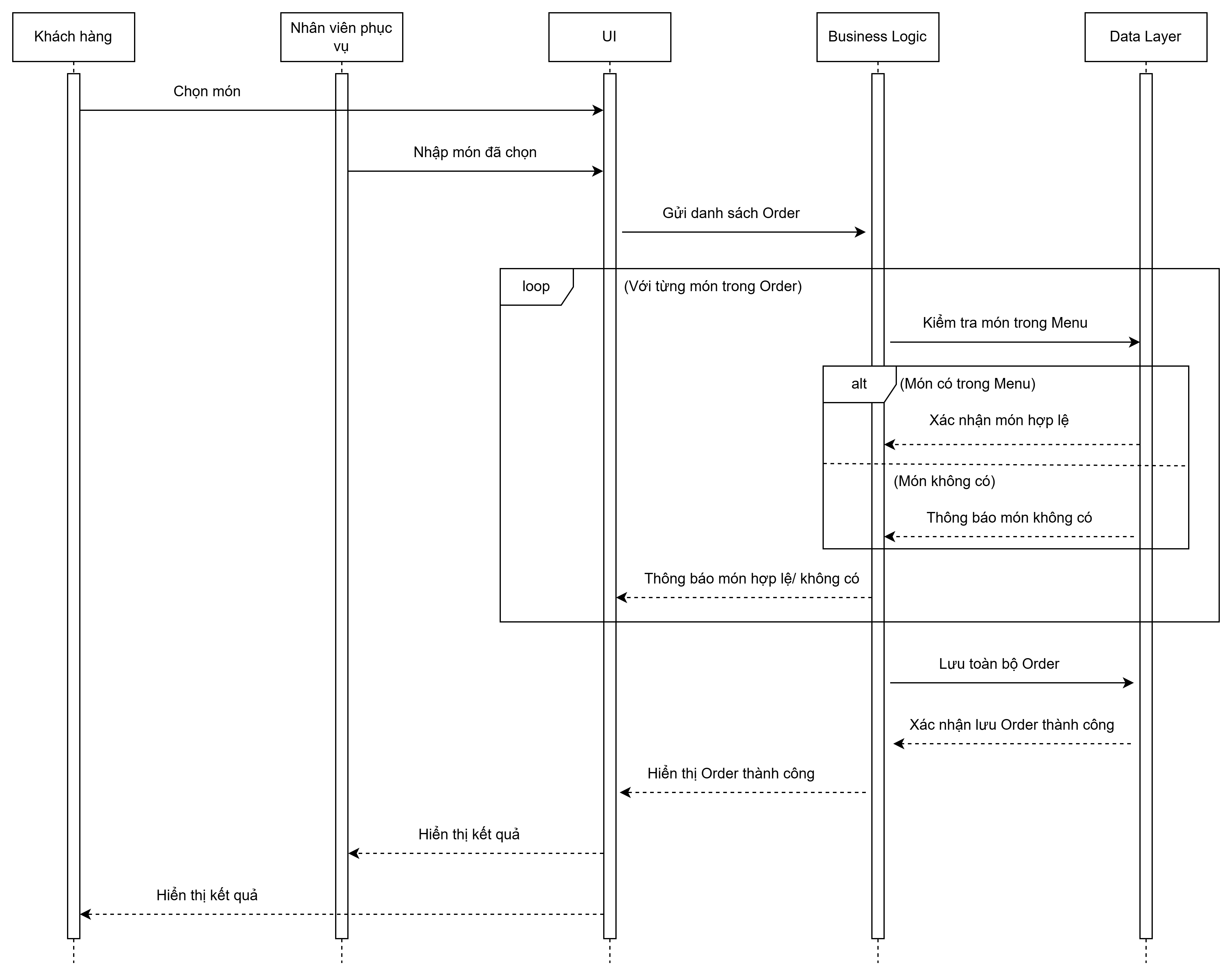
+ Actors không có activation bar vì chỉ thao tác nhập chọn món.

**- Điểm nổi bật**

+ Hỗ trợ nhiều món cùng lúc nhờ **loop**.

+ Xử lý tình huống món không có trong menu bằng **alt**.

+ Hai actor tương tác với UI, giúp biểu diễn thực tế nghiệp vụ nhà hàng.



## Biểu đồ tuần tự – UC: Thanh toán đơn hàng

**- Actors & Layers**

**+ Actors**

* **Khách hàng:** Chọn phương thức thanh toán (tiền mặt / thẻ / QR ngân hàng / ví điện tử) và thực hiện thao tác tương ứng (quẹt thẻ, quét QR…).
* **Nhân viên phục vụ:** Xác nhận phương thức thanh toán, thao tác trên hệ thống, thông báo kết quả cho khách hàng.

**+ UI Layer**

* Giao diện để nhân viên thao tác thanh toán.
* Hiển thị QR (nếu dùng QR Banking hoặc Ví điện tử).
* Nhận kết quả từ POS/QR và trả lại cho người dùng.
* Hiển thị hóa đơn sau khi thanh toán thành công.

**+ Business Layer**

* Xử lý toàn bộ logic thanh toán:

Lấy thông tin order.

Kiểm tra mã khuyến mãi (nếu có).

Áp dụng giảm giá.

Điều phối logic theo từng phương thức thanh toán.

Lưu giao dịch vào Data Layer.

**+ Data Layer**

* Lưu trữ:

Thông tin order.

Mã khuyến mãi.

Giao dịch thanh toán.

* Trả dữ liệu về cho Business Layer để xử lý.

**- Luồng thông điệp chính**

1. Khách hàng → Nhân viên phục vụ

* Khách chọn phương thức thanh toán (tiền mặt / thẻ / QR / ví điện tử), cung cấp mã giảm giá nếu có.

1. Nhân viên phục vụ → UI Layer

* Nhân viên xác nhận thanh toán và nhập thông tin vào giao diện.

1. UI Layer → Business Layer

* Gửi yêu cầu xử lý thanh toán.

1. Business Layer → Data Layer

* Lấy thông tin order của bàn (các món, tổng tiền, phí dịch vụ…).

1. Data Layer → Business Layer

* Trả về dữ liệu order.

1. Kiểm tra mã khuyến mãi (tùy chọn – opt fragment)

* Nếu khách nhập mã giảm giá, Business gửi yêu cầu đến Data để kiểm tra:

Còn hiệu lực?

Đúng điều kiện?

Đạt giá trị tối thiểu?

* Nếu hợp lệ → Business áp dụng giảm giá.
* Nếu không hợp lệ → thông báo lỗi (nhưng vẫn có thể thanh toán không giảm giá).

1. Hệ thống phân nhánh theo phương thức thanh toán (alt fragment)

* Nhánh 1: Thanh toán tiền mặt

Nhân viên thu tiền trực tiếp.

Business ghi nhận giao dịch vào Data Layer.

Data xác nhận thành công.

* Nhánh 2: Thanh toán bằng thẻ (POS)

UI yêu cầu khách đưa thẻ vào máy POS.

Khách quẹt thẻ → POS trả kết quả.

UI gửi kết quả thanh toán đến Business.

Business lưu giao dịch vào Data Layer.

Data xác nhận thành công.

* Nhánh 3: Thanh toán QR Banking

UI hiển thị mã QR ngân hàng.

Khách dùng app ngân hàng quét QR và thanh toán.

Sau khi giao dịch thành công, hệ thống nhận callback/báo trạng thái.

UI gửi kết quả đến Business.

Business lưu giao dịch vào Data Layer.

* Nhánh 4: Thanh toán bằng ví điện tử (Momo / ZaloPay / ShopeePay)

UI hiển thị QR ví điện tử.

Khách quét QR để thanh toán trên ví.

Khi ví điện tử xác nhận giao dịch → UI nhận callback.

Business lưu giao dịch vào Data Layer.

1. Hoàn tất thanh toán

* Business → UI: Trả lại hóa đơn, tổng tiền, thông tin giảm giá (nếu có), phương thức thanh toán.
* UI → Nhân viên phục vụ: Hiển thị hóa đơn cuối.
* Nhân viên phục vụ → Khách hàng: Thông báo thanh toán thành công.

**- Combined Fragments**

* **opt (optional fragment)**: Dùng cho phần mã khuyến mãi (chỉ xảy ra nếu khách nhập mã).
* **alt (alternative fragment):** Dùng để tách rõ 4 phương thức thanh toán, đảm bảo mỗi luồng được mô tả độc lập:

Tiền mặt

Quẹt thẻ POS

QR Banking

Ví điện tử

**- Activation Bars**

**+ UI Layer:** bật khi nhận input từ nhân viên, hiển thị QR, nhận callback và xuất kết quả.

**+ Business Layer:** bật liên tục trong toàn bộ quá trình xử lý thanh toán và logic khuyến mãi.

**+ Data Layer:** bật mỗi khi hệ thống truy vấn/ghi giao dịch.

**+ Actors:** không có activation bar (vì chỉ kích hoạt thao tác, không xử lý logic).

**- Điểm nổi bật**

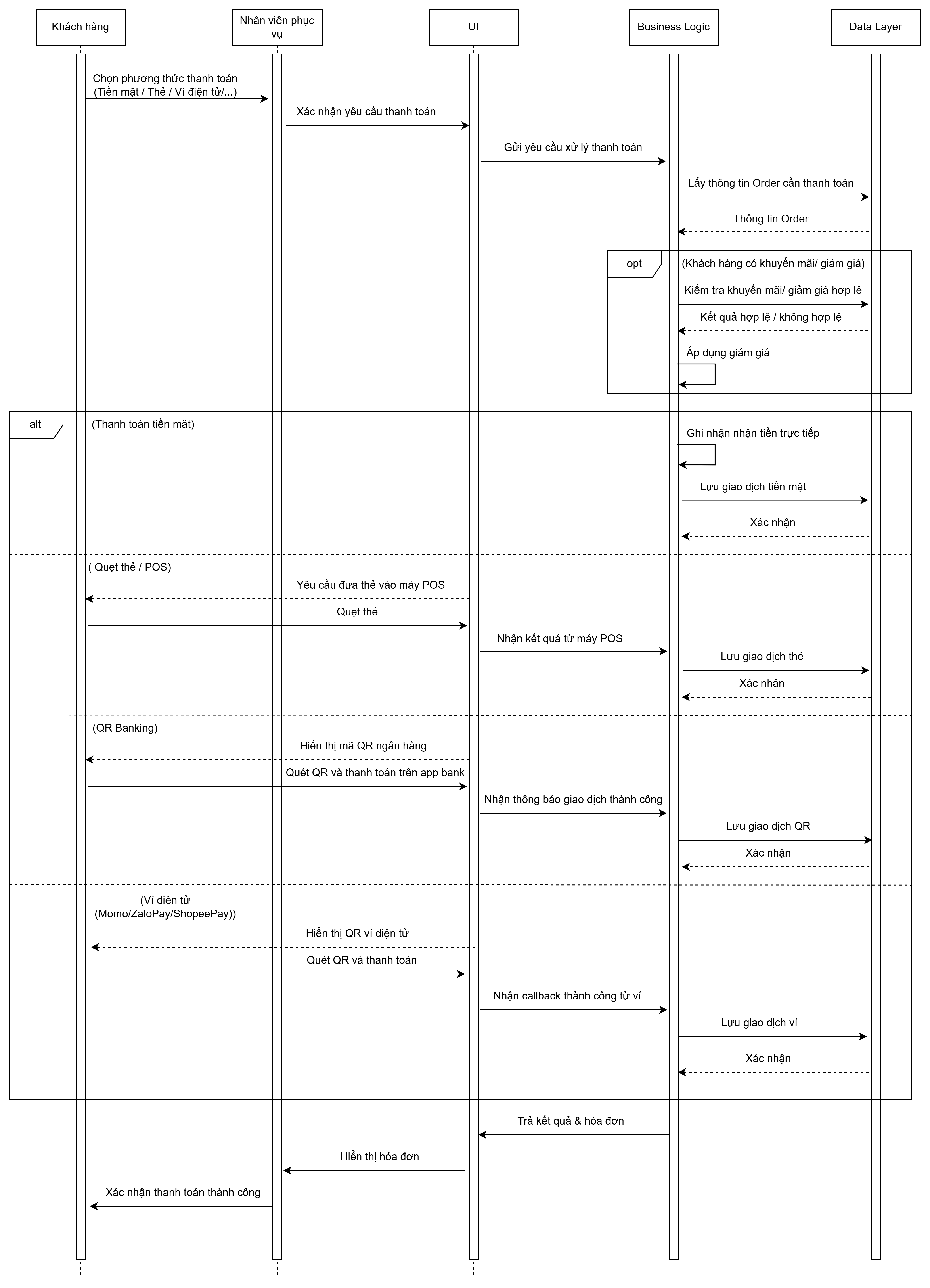
+ Mô tả chi tiết 4 phương thức thanh toán thực tế.

+ Có phân nhánh logic rõ ràng bằng alt.

+ Có xử lý mã khuyến mãi bằng opt.

+ Tách 3 layer (UI–Business–Data) chuẩn kiến trúc.

+ Hệ thống thanh toán thực tế được tái hiện đầy đủ: POS, QR Bank, e-Wallet.



4. Biểu đồ tuần tự – UC: Quản lý bàn ăn   
**- Actors & Layers**

**+ Actors**

* Quản lý / Nhân viên phục vụ:  
  Có quyền thao tác trên danh sách bàn ăn gồm:  
  → Thêm bàn  
  → Sửa thông tin bàn  
  → Xóa bàn  
  → Cập nhật trạng thái bàn (Trống / Đã đặt / Đang phục vụ)

**+ UI Layer**

* Giao diện chạy trên máy tính hoặc tablet.
* Nơi Actor chọn thao tác bàn, nhập thông tin và xem kết quả.

**+ Business Layer**

* Xử lý toàn bộ logic cho từng loại thao tác:  
  → Validate dữ liệu bàn  
  → Kiểm tra trạng thái hợp lệ  
  → Điều phối lưu/xóa/sửa giữa UI và Data Layer

**+ Data Layer**

* Lưu trữ dữ liệu bàn ăn (table), trạng thái bàn và các thuộc tính liên quan.
* Thực hiện các thao tác database: insert, update, delete.

**- Luồng thông điệp chính (Main Flow)**

**+ Actor chọn thao tác**

* Actor → UI: Chọn thao tác (Thêm / Sửa / Xóa / Cập nhật trạng thái).
* UI → Business: Gửi yêu cầu loại thao tác để xử lý.

**+ Business Layer xử lý theo từng nhánh**

* NHÁNH 1 – Thêm bàn

BL → UI: Yêu cầu nhập thông tin bàn mới.

UI → Actor: Hiển thị form nhập.

Actor → UI: Nhập thông tin bàn (số ghế, tên bàn, vị trí…).

UI → BL: Gửi thông tin bàn mới.

BL → Data: Lưu bàn mới vào database.

Data → BL: Trả kết quả thành công.

BL → UI: Thông báo “Thêm bàn thành công”.

* NHÁNH 2 – Sửa bàn

BL → UI: Yêu cầu nhập bàn cần sửa + thông tin mới.

UI → Actor: Hiển thị form sửa.

Actor → UI: Nhập dữ liệu sửa.

UI → BL: Gửi dữ liệu sửa.

BL → Data: Cập nhật thông tin bàn.

Data → BL: Xác nhận sửa thành công.

BL → UI: “Sửa bàn thành công”.

* NHÁNH 3 – Xóa bàn

BL → UI: Yêu cầu xác nhận xóa.

UI → Actor: Hiển thị popup xác nhận.

Actor → UI: Xác nhận xóa bàn.

UI → BL: Gửi yêu cầu xóa.

BL → Data: Xóa bàn khỏi DB.

Data → BL: Trả về thành công.

BL → UI: “Xóa bàn thành công”.

* NHÁNH 4 – Cập nhật trạng thái bàn

BL → UI: Yêu cầu chọn bàn và trạng thái mới.

UI → Actor: Hiển thị danh sách trạng thái.

Actor → UI: Chọn trạng thái (Trống / Đã đặt / Đang phục vụ).

UI → BL: Gửi trạng thái mới lên hệ thống.

BL → Data: Lưu trạng thái mới vào DB.

Data → BL: Xác nhận cập nhật.

BL → UI: “Cập nhật trạng thái thành công”.

**+ Kết thúc**

* UI → Actor: Hiển thị kết quả cuối cùng của thao tác.

**- Combined Fragment**

+ Diagram sử dụng alt để tách 4 kịch bản khác nhau:

* Thêm bàn
* Sửa bàn
* Xóa bàn
* Cập nhật trạng thái

+ Mỗi nhánh có xử lý riêng, giúp sơ đồ rõ ràng và dễ mở rộng.

**- Activation Bar**

* UI, Business, Data Layer đều hiển thị activation bar vì:  
  → UI xử lý nhập liệu và hiển thị thông tin  
  → BL xử lý logic, điều phối  
  → DL thao tác đọc/ghi dữ liệu
* Actor không cần activation bar vì chỉ thực hiện thao tác trigger.

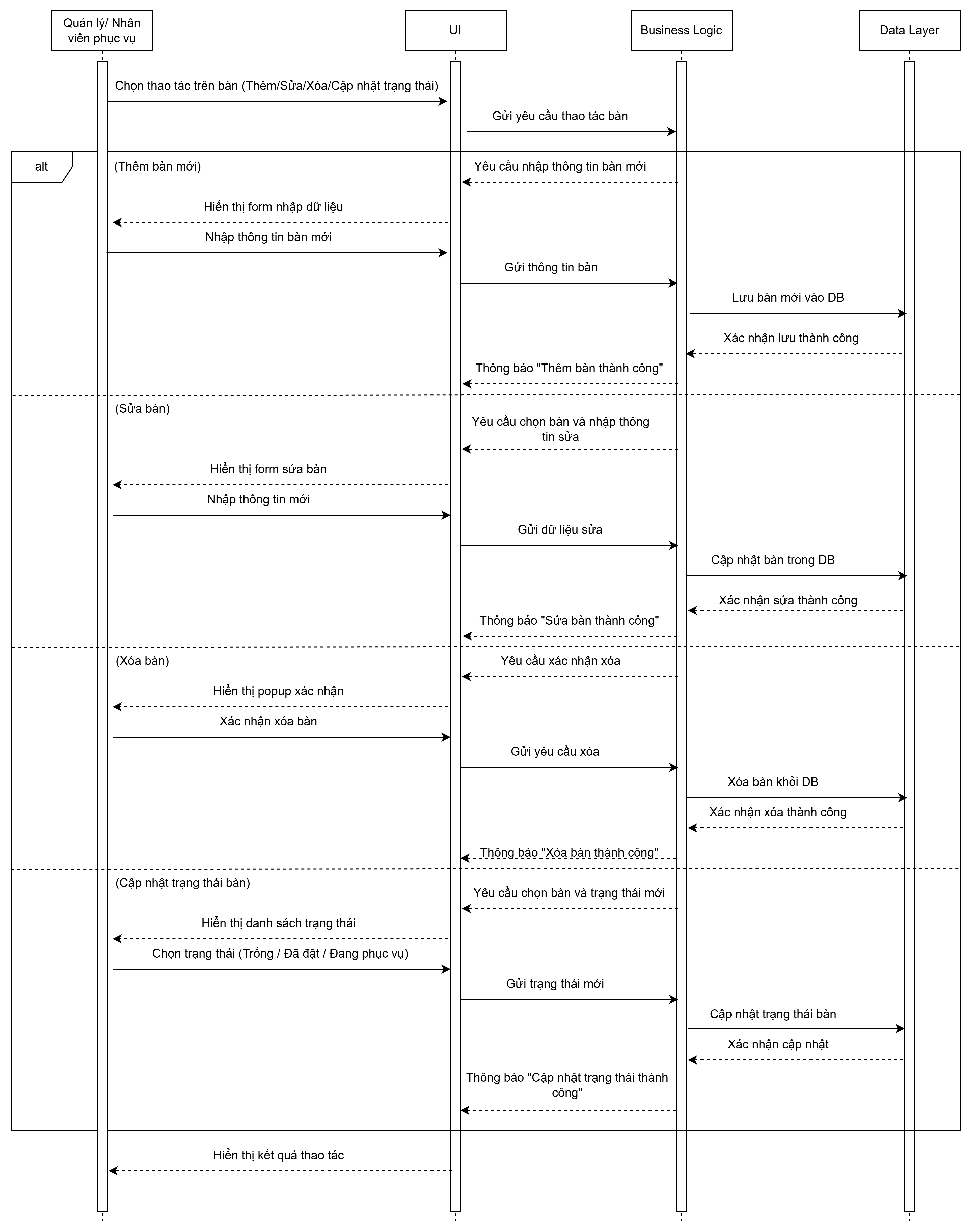
**- Điểm nổi bật**

+ Rõ nghiệp vụ quản lý bàn của nhà hàng.

+ Tách biệt 4 layer, đúng chuẩn mô hình 3-tier + UI.

+ alt fragment chi tiết giúp diagram dễ hiểu, dễ bảo trì.

+ Các bước đều cụ thể, gần sát logic thực tế hệ thống quản lý nhà hàng.



5. Biểu đồ tuần tự – UC: Xem báo cáo doanh thu  
**- Actors & Layers**

**+ Actor:** Quản lý – yêu cầu xem các loại báo cáo.

**+ UI Layer:** Giao diện hiển thị các báo cáo.

**+ Business Layer:** Xử lý logic tạo các báo cáo từ dữ liệu.

**+ Data Layer:** Lưu trữ dữ liệu bàn, đặt bàn, gọi món, doanh thu.

**- Luồng thông điệp chính**

1. Quản lý chọn loại báo cáo trên UI.
2. UI gửi yêu cầu tới Business Layer.
3. Business Layer xử lý song song (par) nhiều loại báo cáo:

* Lấy dữ liệu doanh thu.
* Lấy dữ liệu đặt bàn.
* Lấy dữ liệu món ăn.

1. Data Layer trả về dữ liệu tương ứng.
2. Business Layer tổng hợp kết quả và trả về UI.
3. UI hiển thị tất cả báo cáo cho quản lý.

**- Combined Fragment**

+ Sử dụng **par** để biểu diễn **xử lý đồng thời** nhiều loại báo cáo, giúp minh họa rằng các truy vấn báo cáo có thể diễn ra song song.

**- Activation Bar**

**+ UI Layer, Business Layer, Data Layer** có activation bar thể hiện thời gian thực thi.

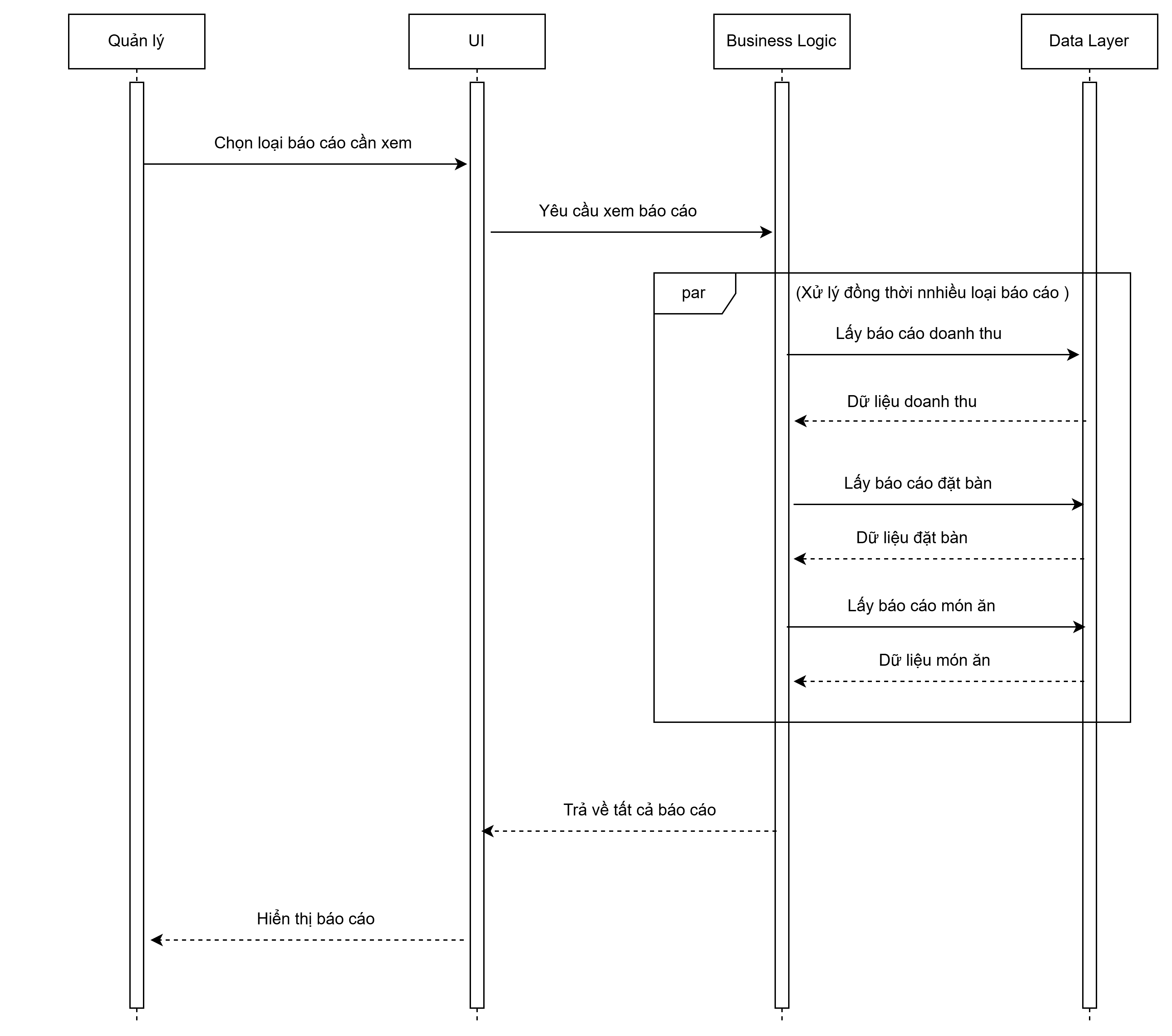
+ Actor **không có activation bar**, chỉ thao tác trên UI.

**- Điểm nổi bật**

+ Xử lý song song nhiều loại báo cáo bằng **par fragment** → thể hiện trực quan luồng dữ liệu đồng thời.

+ Dễ mở rộng khi thêm các loại báo cáo mới.

+ Tách biệt 4 lớp rõ ràng: UC → UI → Business → Data.



## BIỂU ĐỒ TRẠNG THÁI ( State Diagram)

1. Biểu đồ trạng thái – UC: Đặt bàn  
**- Các trạng thái (States)**

**+ Mới\_Tạo**

* Reservation vừa được tạo khi nhân viên nhập thông tin khách hàng.
* Chưa kiểm tra bàn trống.

**+ Đang\_Xác\_Nhận**

* Hệ thống đang xử lý thông tin đặt bàn.
* Kiểm tra bàn trống, thời gian, số lượng khách.

**+ Đã\_Xác\_Nhận**

* Đã tìm được bàn phù hợp.
* Bàn được giữ cho khách trong một khoảng thời gian.

**+ Từ\_Chối**

* Hệ thống không tìm được bàn phù hợp hoặc thông tin không hợp lệ.
* Reservation bị từ chối.

**+ Hủy**

* Khách yêu cầu hủy hoặc quá giờ không đến.
* Reservation bị hủy.

**+ Check\_In**

* Khách đã đến nhà hàng và được đưa vào bàn.
* Reservation chuyển sang phục vụ.

**+ Hoàn\_Tất**

* Khách kết thúc bữa ăn.
* Bàn được giải phóng hoàn toàn → Reservation kết thúc.

**- Mô tả**

**+ Mới\_Tạo → Đang\_Xác\_Nhận**

* **Event**: nhapThongTin()
* **Action**: kiemTraDuLieu()  
  → Khi nhân viên nhập tên khách, SĐT, số lượng người, thời gian… hệ thống bắt đầu kiểm tra tính hợp lệ.

**+ Đang\_Xác\_Nhận → Đã\_Xác\_Nhận**

* **Event**: xacNhan()
* **Guard**: [coBanTrong]
* **Action**: luuReservation()  
  → Nếu có bàn phù hợp thì reservation được lưu và bàn được giữ.

**+ Đang\_Xác\_Nhận → Từ\_Chối**

* **Event**: tuChoi()
* **Guard**: [khongDuBan]
* **Action**: guiGoiYBanKhac()  
  → Không có bàn phù hợp → hệ thống từ chối.

**+ Đã\_Xác\_Nhận → Check\_In**

* **Event**: khachDen()
* **Guard**: [trongThoiGian]
* **Action**: capNhatTrangThaiBan()  
  → Khách đến đúng giờ → bắt đầu phục vụ bàn.

**+ Đã\_Xác\_Nhận → Hủy**

* **Trường hợp 1**: khachHuy() → khách chủ động hủy
* **Trường hợp 2**:quaGio() → khách không đến → no-show
* **Action**: xoaReservation() hoặc danhDauKhachKhongDen()

**+ Check\_In → Hoàn\_Tất**

* **Event**: ketThucPhucVu()
* **Action**: dongPhien()  
  → Khi khách ăn xong.

**+ Từ\_Chối, Hủy, Hoàn\_Tất → [\*]**

→ Hoàn thành vòng đời của reservation.

**- Điểm nổi bật của biểu đồ**

+ Dùng **guard** đúng chuẩn UML: [duBanTrong], [khongDuBan], [trongThoiGian].

+ Dùng **action** ngay sau dấu /: luuReservation(), guiGoiYBanKhac(),…

+ Trạng thái rõ ràng, mô phỏng đúng vòng đời của một lần đặt bàn trong thực tế nhà hàng.

+ Có nhánh **Check-In, Hủy, Từ chối** thể hiện thực tế hoạt động.

# IMG_256

2. Biểu đồ trạng thái – UC: Thanh toán đơn hàng   
**- Các trạng thái (States)**

**+ Chờ\_Thanh\_Toán**

* Hóa đơn vừa được tạo, chưa thanh toán.
* Chưa chọn phương thức thanh toán.

**+ Tính\_Tổng\_Tiền**

* Hệ thống tính tổng tiền của hóa đơn: món ăn, phụ thu, thuế.
* Chuẩn bị cho bước áp dụng khuyến mãi hoặc chọn phương thức thanh toán.

**+ Áp\_Dụng\_Khuyến\_Mãi**

* Hệ thống kiểm tra và áp dụng các khuyến mãi hợp lệ (nếu có).
* Có thể cập nhật tổng tiền hóa đơn hoặc giữ nguyên nếu không hợp lệ.

**+ Chọn\_Phương\_Thức**

* Khách hàng hoặc thu ngân chọn phương thức thanh toán: Tiền mặt, Thẻ, QR, Ví điện tử…
* Chưa xác thực tính hợp lệ của phương thức.

**+ Xác\_Thực\_Phương\_Thức**

* Hệ thống kiểm tra tính hợp lệ của phương thức thanh toán đã chọn.
* Nhánh thành công → tiếp tục thanh toán.
* Nhánh lỗi → chuyển sang trạng thái Lỗi\_Thanh\_Toán.

**+ Thanh\_Toán**

* Hệ thống gửi lệnh thanh toán đến cổng thanh toán hoặc ngân hàng.
* Có thể thành công hoặc thất bại.

**+ Lỗi\_Thanh\_Toán**

* Thanh toán thất bại do: hết tiền, thẻ lỗi, mạng lỗi, timeout…
* Người dùng có thể thử lại hoặc hủy giao dịch.

**+ Thành\_Công**

* Thanh toán hoàn tất thành công.
* Hệ thống cập nhật trạng thái hóa đơn và lưu lịch sử giao dịch.

**+ In\_Hóa\_Đơn**

Hệ thống in hóa đơn hoặc xuất PDF cho khách hàng.

**+ Hoàn\_Tất**

* Giao dịch kết thúc.
* Hóa đơn được đánh dấu hoàn tất, kết thúc vòng đời giao dịch.

**+ Hủy**

* Khách hàng hoặc thu ngân hủy giao dịch trước khi thanh toán thành công.

**- Mô tả**

**+ Chờ\_Thanh\_Toán → Tính\_Tổng\_Tiền**

* **Event**: taoHoaDon()
* **Action**: tinhTien()

→ Khi hóa đơn được tạo, hệ thống bắt đầu tính tổng tiền.

**+ Tính\_Tổng\_Tiền → Áp\_Dụng\_Khuyến\_Mãi**

* **Event**: kiemTraKM()
* **Guard**: [maHopLe]
* **Action**: capNhatGia()

→ Kiểm tra mã khuyến mãi, áp dụng nếu hợp lệ.

**+ Áp\_Dụng\_Khuyến\_Mãi → Tính\_Tổng\_Tiền**

* **Event**: khongApDung()
* **Action**: giuNguyenGia()

→ Nếu không có khuyến mãi hợp lệ, giữ nguyên tổng tiền.

**+ Tính\_Tổng\_Tiền → Chọn\_Phương\_Thức**

* **Event: boQuaKM()**

→ Bỏ qua bước áp dụng khuyến mãi → chọn phương thức thanh toán.

**+ Chọn\_Phương\_Thức → Xác\_Thực\_Phương\_Thức**

* **Event**: chonPTTT()
* **Action**: xacThucPT()

→ Kiểm tra phương thức thanh toán đã chọn có hợp lệ không.

**+ Xác\_Thực\_Phương\_Thức → Thanh\_Toán**

* **Event**: xacThucOK()
* **Action**: taoGiaoDich()

→ Phương thức hợp lệ → bắt đầu thanh toán.

**+ Xác\_Thực\_Phương\_Thức → Lỗi\_Thanh\_Toán**

* Event: xacThucFail()
* Action: thongBaoLoiPT()

→ Phương thức không hợp lệ → thông báo lỗi.

**+ Thanh\_Toán → Thành\_Công**

* **Event**: thanhCong()
* **Action**: capNhatHoaDon()

→ Thanh toán thành công → cập nhật trạng thái hóa đơn.

**+ Thanh\_Toán → Lỗi\_Thanh\_Toán**

* **Event**: thatBai()
* **Action**: hienThiLoi()

→ Thanh toán thất bại → hiển thị lỗi.

**+ Lỗi\_Thanh\_Toán → Chọn\_Phương\_Thức**

* **Event**: chonLaiPTTT()

→ Thử lại phương thức thanh toán khác.

**+ Lỗi\_Thanh\_Toán → Hủy**

* **Event**: huyGD()
* **Action**: xoaGiaoDich()

→ Hủy giao dịch.

**+ Thành\_Công → In\_Hóa\_Đơn**

* **Event**: inHD()
* **Action**: taoPDF()

→ In hóa đơn hoặc xuất PDF.

**+ In\_Hóa\_Đơn → Hoàn\_Tất**

* **Event**: hoanTat()
* **Action**: dongGiaoDich()

→ Hoàn tất giao dịch.

**+ Hủy, Hoàn\_Tất → [\*]**

→ Kết thúc vòng đời giao dịch.

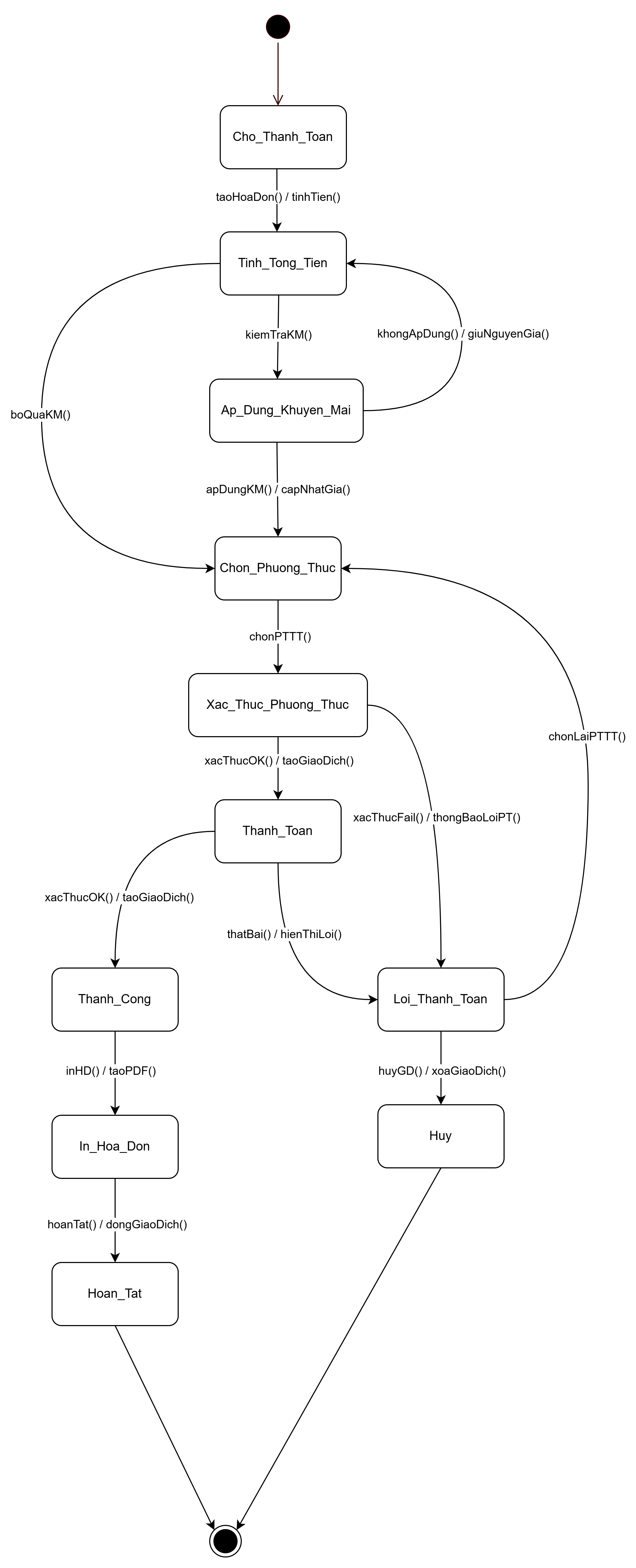
**- Điểm nổi bật của biểu đồ**

+ Dùng **guard** đúng chuẩn UML: [maHopLe], [xacThucOK].

+ Dùng **action** ngay sau dấu /: tinhTien(), capNhatGia(), xoaGiaoDich().

+ Trạng thái rõ ràng, mô phỏng đúng vòng đời giao dịch thanh toán trong nhà hàng.

+ Xử lý đầy đủ các nhánh: thành công, thất bại, hủy, in hóa đơn, thử lại phương thức thanh toán.



## 3. Biểu đồ trạng thái – UC: Chế biến món

**- Các trạng thái (States)**

**+ Mới\_Tiếp\_Nhận**

* Nhận đơn hàng từ hệ thống hoặc nhân viên phục vụ.
* Chưa kiểm tra nguyên liệu, chưa bắt đầu chuẩn bị món.

**+ Chuẩn\_Bị**

* Hệ thống/bếp chuẩn bị nguyên liệu cho món ăn.
* Kiểm tra nguyên liệu có đầy đủ, rửa, sơ chế.

**+ Lỗi\_Nguyên\_Liệu**

* Nguyên liệu không đủ hoặc bị hỏng.
* Hệ thống thông báo lỗi, cần cập nhật nguyên liệu.

**+ Đang\_Nấu**

* Món đang được nấu.
* Có thể hủy nấu nếu cần.

**+ Đang\_Kiểm\_Tra\_Chất\_Lượng**

* Kiểm tra món đã nấu xong có đạt tiêu chuẩn.
* Nếu đạt → Hoàn\_Tất, nếu không → Sửa\_Lại.

**+ Sửa\_Lại**

* Nấu lại món ăn nếu lần đầu kiểm tra không đạt.
* Sau khi nấu lại → trở về kiểm tra chất lượng.

**+ Hoàn\_Tất**

* Món đã nấu xong, đạt chuẩn, chuyển đến phục vụ khách.
* Kết thúc vòng đời món ăn trong bếp.

**+ Hủy**

* Món bị hủy do lỗi nấu, hủy từ khách hoặc từ bếp.
* Kết thúc vòng đời món ăn.

**- Mô tả**

**+ Mới\_Tiếp\_Nhận → Chuẩn\_Bị**

* **Event**: nhanDonHang()
* **Action**: kiemTraNguyenLieu()  
  → Khi nhận đơn, hệ thống bắt đầu kiểm tra nguyên liệu.

**+ Chuẩn\_Bị → Đang\_Nấu**

* **Event**: nguyenLieuDayDu()
* **Action**: chuanBiNguyenLieu()  
  → Nguyên liệu đầy đủ, bếp bắt đầu nấu món.

**+ Chuẩn\_Bị → Lỗi\_Nguyên\_Liệu**

* Event: nguyenLieuThieu()
* Action: thongBaoKhongDuNL()  
  → Nguyên liệu không đủ → thông báo lỗi.

**+ Đang\_Nấu → Đang\_Kiểm\_Tra\_Chất\_Lượng**

* **Event**: nauXong()
* **Action**: kiemTraChuan()  
  → Món nấu xong → kiểm tra chất lượng.

**+ Đang\_Nấu → Hủy**

* **Event**: huyNau()
* **Action**: dungQuaTrinh()  
  → Hủy quá trình nấu món.

**+ Đang\_Kiểm\_Tra\_Chất\_Lượng → Hoàn\_Tất**

* **Event**: kiemTraOK()
* **Action**: chuyenPhucVu()  
  → Món đạt chuẩn → chuyển phục vụ.

**+ Đang\_Kiểm\_Tra\_Chất\_Lượng → Sửa\_Lại**

* **Event**: kiemTraFail()
* **Action**: nauLai()  
  → Món không đạt chuẩn → nấu lại.

**+ Sửa\_Lại → Đang\_Kiểm\_Tra\_Chất\_Lượng**

* **Event**: nauXongLai()
* **Action**: kiemTraChuan()  
  → Sau khi nấu lại → kiểm tra chất lượng tiếp.

**+ Lỗi\_Nguyên\_Liệu → Mới\_Tiếp\_Nhận**

* **Event**: capNhatNL()
* **Action**: thuLai()  
  → Nguyên liệu được bổ sung → quay lại tiếp nhận món.

**+ Hủy, Hoàn\_Tất → [\*]**  
 → Kết thúc vòng đời món ăn trong bếp.

**- Điểm nổi bật của biểu đồ**

+ Trạng thái mô phỏng đầy đủ vòng đời món ăn: tiếp nhận → chuẩn bị → nấu → kiểm tra → phục vụ hoặc hủy.

+ Transition đầy đủ, rõ ràng event + action, đúng chuẩn UML.

+ Có nhánh xử lý lỗi nguyên liệu và nấu lại, mô phỏng thực tế bếp nhà hàng.

+ Dễ dàng mở rộng nếu muốn thêm các trạng thái khác như “Đang trang trí”, “Chờ khách nhận món”…

