**1. Kiến trúc chung & Công nghệ**

Kiến trúc hệ thống được xây dựng dựa trên mô hình Microservices để đảm bảo tính độc lập, dễ bảo trì và khả năng mở rộng.

* Mô hình kiến trúc: Bao gồm 4 services lõi hoạt động độc lập: User Service, Product Service, Order Service, và Payment Service.
* API Gateway: Toàn bộ yêu cầu từ phía Frontend sẽ đi qua một API Gateway duy nhất. Gateway này chịu trách nhiệm xác thực người dùng, điều hướng yêu cầu đến đúng service, và kiểm soát lưu lượng truy cập.
* Giao tiếp giữa các Service:
  + Đồng bộ (Synchronous): Sử dụng REST API cho các tác vụ cần phản hồi ngay lập tức (ví dụ: Order Service gọi Product Service để kiểm tra giá).
  + Bất đồng bộ (Asynchronous): Sử dụng Message Broker (ví dụ: RabbitMQ, Kafka) cho các sự kiện không cần phản hồi tức thì, giúp giảm sự phụ thuộc và tăng tính ổn định (ví dụ: Payment Service thông báo thanh toán thành công).
* Công nghệ Backend: Có thể sử dụng các framework như FastAPI (Python) hoặc Node.js (Express) cho từng service.
* Cơ sở dữ liệu: Mỗi service sẽ quản lý schema hoặc database riêng trên PostgreSQL để đảm bảo tính độc lập về dữ liệu.
* Triển khai: Các service sẽ được đóng gói bằng Docker và triển khai trên nền tảng Kubernetes để quản lý và mở rộng linh hoạt.

**2. Thiết kế chi tiết từng Service**

**User Service (Dịch vụ Người dùng)**

* Mục đích: Quản lý toàn bộ thông tin và hành động liên quan đến tài khoản khách hàng.
* Dữ liệu quản lý: Các bảng Users và Addresses.
* Các API chính:
  + POST /auth/register: Đăng ký tài khoản mới.
  + POST /auth/login: Đăng nhập, trả về token xác thực (JWT).
  + GET /users/me/addresses: Lấy danh sách địa chỉ đã lưu của người dùng.

**Product Service (Dịch vụ Sản phẩm)**

* Mục đích: Quản lý thông tin về các món ăn trong thực đơn.
* Dữ liệu quản lý: Các bảng MenuItems và Categories.
* Các API chính:
  + GET /products: Trả về toàn bộ thực đơn.
  + GET /products/{id}: Lấy thông tin chi tiết của một món ăn.

**Order Service (Dịch vụ Đơn hàng)**

* Mục đích: Là trung tâm điều phối, quản lý toàn bộ vòng đời của một đơn hàng.
* Dữ liệu quản lý: Các bảng Orders và OrderItems.
* Các API chính:
  + POST /orders: Tạo một đơn hàng mới.
  + GET /orders/{id}: Lấy thông tin chi tiết và trạng thái của một đơn hàng.
  + GET /orders/history: Lấy lịch sử các đơn hàng đã đặt của người dùng.

**Payment Service (Dịch vụ Thanh toán)**

* Mục đích: Chịu trách nhiệm xử lý các giao dịch thanh toán.
* Dữ liệu quản lý: Bảng Payments để lưu trữ lịch sử giao dịch.
* Các API chính:
  + POST /payments/create-intent: Tạo một phiên thanh toán cho đơn hàng.
  + POST /payments/webhook: Nhận thông báo xác nhận thanh toán từ cổng thanh toán bên ngoài.

**3. Monitoring (Giám sát hệ thống)**

* Công cụ: Sử dụng Prometheus để thu thập metrics từ các service và Grafana để trực quan hóa dữ liệu trên các dashboard.

