**Product Requirements Document (PRD) – FoodFast Delivery**

**1. Problem Alignment (Xác định vấn đề)**

* Người dùng gặp khó khăn khi đặt món ăn qua nhiều nền tảng khác nhau (web, mobile), trải nghiệm không đồng nhất.
* Khách hàng thiếu công cụ theo dõi đơn hàng minh bạch, thời gian thực.
* Các hệ thống giao đồ ăn hiện nay có quá nhiều bước, gây mất thời gian và dễ nhầm lẫn.
* Quản lý thanh toán và dịch vụ chưa tích hợp đồng bộ, khó mở rộng khi có nhiều đơn hàng.

**2. High Level Approach (Giải pháp tổng thể)**

* Xây dựng **ứng dụng web + mobile** sử dụng **React.js + React Native** để người dùng có thể đặt món thuận tiện.
* Backend theo **kiến trúc 3 lớp** và chia thành **4 microservices** (User, Product, Order, Payment).
* Tích hợp **CI/CD pipeline** để đảm bảo cập nhật nhanh chóng, ổn định.
* Tích hợp **giám sát hệ thống real-time** để theo dõi hiệu năng từng service.

**3. Narrative (Kịch bản trải nghiệm)**

1. Người dùng mở ứng dụng (web/mobile), đăng nhập hoặc tạo tài khoản.
2. Họ duyệt menu, chọn món và thêm vào giỏ hàng.
3. Người dùng xác nhận đơn hàng, chọn phương thức thanh toán (Momo, VNPay, thẻ tín dụng).
4. Hệ thống xử lý qua **Payment Service**, xác nhận thành công.
5. **Order Service** cập nhật trạng thái đơn hàng → người dùng theo dõi tiến trình giao hàng theo thời gian thực.
6. Người dùng nhận được thông báo khi đơn hàng hoàn tất.

**4. Goals (Mục tiêu)**

* Cung cấp trải nghiệm đặt hàng nhanh, tiện lợi, nhất quán trên web và mobile.
* Tích hợp giỏ hàng, thanh toán, và theo dõi đơn hàng trong một hệ thống duy nhất.
* Dễ dàng mở rộng để hỗ trợ nhiều đối tác (nhà hàng, dịch vụ vận chuyển).
* Đảm bảo hệ thống luôn ổn định, có giám sát real-time và khả năng phục hồi khi gặp sự cố.

**5. Solution Alignment (Giải pháp cụ thể)**

**Key Features**

1. **Frontend (FE)**:
   * React.js (web), React Native (mobile).
   * Xử lý đặt món, thanh toán, theo dõi đơn hàng.
2. **Backend (BE)**:
   * Kiến trúc 3 lớp: Presentation – Business – Data.
   * Kết nối với DB (MySQL/PostgreSQL).
3. **4 Microservices**:
   * **User Service**: quản lý tài khoản, xác thực JWT/OAuth2.
   * **Product Service**: quản lý món ăn, danh mục, tồn kho.
   * **Order Service**: xử lý giỏ hàng, trạng thái đơn.
   * **Payment Service**: tích hợp cổng thanh toán (Momo, VNPay, PayPal).
4. **CI/CD**:
   * Pipeline tự động build, test, deploy cho cả hệ thống và từng service.
   * Dùng Docker + Kubernetes để triển khai.
5. **Monitoring**:
   * Giám sát real-time bằng Prometheus + Grafana.
   * Dashboard theo dõi cho từng service (User, Product, Order, Payment).

**6. Key Flows (Luồng hoạt động)**

* **Operational Flow**: Người dùng → FE → BE → Microservices → DB.
* **Payment Flow**: Người dùng xác nhận đơn → Payment Service xử lý → cập nhật Order Service.

**7. Key Logic (Logic chính)**

1. **Xử lý đặt hàng**: Giỏ hàng lưu tạm → xác nhận → Order Service tạo đơn → gọi Payment Service.
2. **Thanh toán**: Payment Service liên kết API của Momo/VNPay → phản hồi kết quả → cập nhật Order Service.
3. **Thông báo trạng thái**: Order Service gửi event → FE cập nhật theo thời gian thực (WebSocket/SignalR).

**8. Launch Plan (Kế hoạch triển khai)**

**9. Appendix (Phụ lục)**

* **Changelog**: cập nhật các thay đổi về service, API.
* **Open Questions**:
  + Tích hợp vận chuyển qua Grab/Gojek hay tự xây dựng?
  + Hình thức khuyến mãi/giảm giá sẽ xử lý ở đâu (Product Service hay Order Service)?
* **FAQs**:
  + Làm sao đổi phương thức thanh toán?
  + Làm sao theo dõi shipper?