Contents

[User Stories cho hệ thống quản lý nhà sách 2](#_Toc192763962)

[Các microservice có thể có trong hệ thống quản lý nhà sách: 2](#_Toc192763963)

[Quy trình Phát triển và Triển khai 2](#_Toc192763964)

[Công nghệ dự kiến 3](#_Toc192763965)

[Kiến trúc chi tiết cho hệ thống quản lý nhà sách 6](#_Toc192763966)

[Cấu trúc dự án và thư mục dự kiến 7](#_Toc192763967)

[Kiến trúc chi tiết mới nếu tích hợp Resilience, Observability và Monitoring 8](#_Toc192763968)

## User Stories cho hệ thống quản lý nhà sách

**Khách hàng:**

* Tôi muốn tìm kiếm sách theo tên, thể loại, tác giả để dễ dàng tìm được sách mình quan tâm
* Tôi muốn xem thông tin chi tiết về sách (giá cả, tình trạng còn hàng, nội dung tóm tắt)
* Tôi muốn thêm sách vào giỏ hàng và thanh toán
* Tôi muốn theo dõi trạng thái đơn hàng của mình
* Tôi muốn đánh giá và viết nhận xét về sách đã mua

**Quản lý:**

* Tôi muốn thêm/sửa/xóa thông tin sách trong kho
* Tôi muốn quản lý đơn hàng và cập nhật trạng thái giao hàng
* Tôi muốn xem báo cáo doanh thu theo ngày/tháng/năm
* Tôi muốn quản lý thông tin khách hàng
* Tôi muốn theo dõi hàng tồn kho và nhận cảnh báo khi sắp hết hàng

## Các microservice có thể có trong hệ thống quản lý nhà sách:

1. **User Service**: Quản lý người dùng, xác thực, phân quyền
2. **Inventory Service**: Quản lý sách và hàng tồn kho
3. **Order Service**: Quản lý đơn hàng và thanh toán
4. **Catalog Service**: Quản lý danh mục, tìm kiếm và hiển thị sách
5. **Review Service**: Đánh giá và bình luận về sách
6. **Notification Service**: Thông báo đến khách hàng
7. **Analytics Service**: Thống kê và báo cáo doanh số

## Quy trình Phát triển và Triển khai

**Giai đoạn 1: Monolithic với Docker (1-2 tháng)**

1. Thiết kế cơ sở dữ liệu và các domain model
2. Phát triển ứng dụng monolithic với module rõ ràng
3. Đóng gói trong Docker container
4. Triển khai trên Kubernetes đơn giản (single service)
5. Thiết lập CI/CD pipeline cơ bản

**Giai đoạn 2: Chuyển đổi sang Microservices (3-6 tháng)**

1. Tách module Catalog và Inventory thành microservices riêng
2. Thiết lập API Gateway và Service Discovery
3. Mở rộng CI/CD pipeline cho multiple services
4. Thiết lập monitoring với Prometheus và Grafana

**Giai đoạn 3: Hoàn thiện Microservices (6-12 tháng)**

1. Tách tất cả các module thành microservices
2. Cải tiến hệ thống message broker cho giao tiếp giữa các service
3. Thiết lập Kubernetes auto-scaling và self-healing
4. Triển khai blue/green deployment

## Công nghệ dự kiến

**1. Kiến trúc Microservice**

* **Service Discovery & Configuration:**
  + **Spring Cloud Netflix Eureka** hoặc **Kubernetes native service discovery**
  + **Spring Cloud Config Server** hoặc **Kubernetes ConfigMaps & Secrets**
* **API Gateway:**
  + **Spring Cloud Gateway** - Tích hợp tốt với hệ sinh thái Spring
  + Thay thế: Kong hoặc Nginx Ingress Controller (cho Kubernetes)
* **Circuit Breaker & Resilience:**
  + **Resilience4j** (thay thế cho Hystrix đã không còn được phát triển)

**2. Giao tiếp giữa các Service**

* **Đồng bộ:**
  + **Spring Cloud OpenFeign** - Client HTTP khai báo cho gọi API giữa các services
  + **Spring WebFlux** - Cho các API reactive
* **Bất đồng bộ (Event Driven):**
  + **Apache Kafka** - Cho các luồng dữ liệu lớn và phức tạp
  + **RabbitMQ** - Cho messaging thông thường, đơn giản hơn Kafka

**3. Containerization & Orchestration**

* **Docker:**
  + Docker Compose cho môi trường development
  + Dockerfile tối ưu với multi-stage builds
* **Kubernetes:**
  + **Helm** - Quản lý Kubernetes packages
  + **Kustomize** - Quản lý YAML manifests
  + **Ingress Controller** - NGINX hoặc Traefik

**4. CI/CD & DevOps**

* **CI/CD Pipeline:**
  + **GitHub Actions** - Tích hợp tốt với GitHub repositories
  + **Jenkins** hoặc **GitLab CI/CD** - Cho các pipeline phức tạp hơn
* **Container Registry:**
  + **Docker Hub** hoặc **GitHub Container Registry** (ghcr.io)
  + **Harbor** cho private registry trong doanh nghiệp

**5. Monitoring & Logging**

* **Monitoring:**
  + **Prometheus** - Thu thập metrics
  + **Grafana** - Visualization
  + **Spring Boot Actuator** - Endpoints cho health checks và metrics
* **Logging:**
  + **ELK Stack** (Elasticsearch, Logstash, Kibana)
  + **Fluentd** - Thu thập logs từ Kubernetes pods
  + **Loki** + Grafana - Giải pháp logging nhẹ hơn

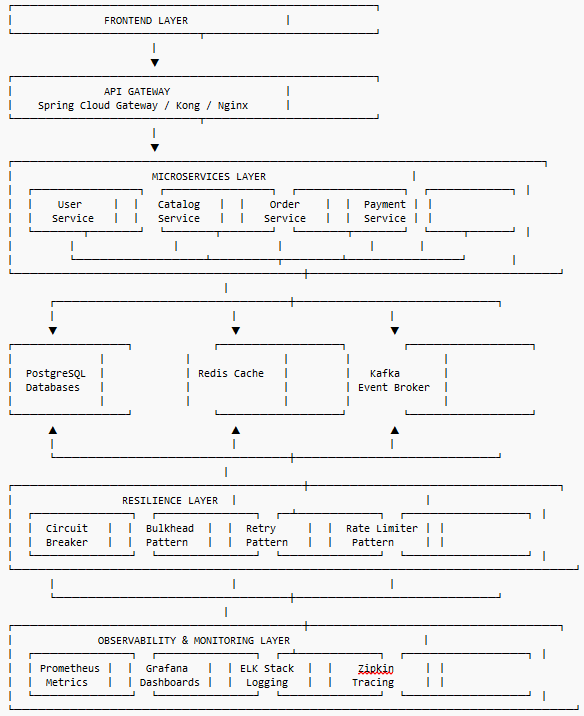
**6. Database & Caching**

* **Database Access:**
  + **Spring Data JPA** - ORM cho PostgreSQL
  + **Flyway** hoặc **Liquibase** - Database migrations
* **Caching:**
  + **Redis** - In-memory cache và session storage
  + **Spring Cache** - Abstraction layer cho caching

**7. Bảo mật**

* **Authentication & Authorization:**
  + **Spring Security** + **OAuth2** / **JWT**
  + **Keycloak** - Identity và Access Management
* **API Security:**
  + **Spring Security** configs
  + **API Key**  cho xác thực API giữa các services

## Kiến trúc chi tiết cho hệ thống quản lý nhà sách



## Cấu trúc dự án và thư mục dự kiến

A close-up of a form

AI-generated content may be incorrect.

## Kiến trúc chi tiết mới nếu tích hợp Resilience, Observability và Monitoring

A diagram of a computer

AI-generated content may be incorrect.