Hệ thống Quản lý Bệnh viện

Bối cảnh:

Bệnh viện ABC là một cơ sở y tế hiện đại với số lượng bệnh nhân ngày càng tăng. Hiện tại, việc quản lý bệnh nhân, lịch khám, đơn thuốc và thông báo vẫn còn thủ công và thiếu hiệu quả. Để cải thiện dịch vụ và tối ưu hóa quản lý, ban giám đốc quyết định xây dựng một hệ thống quản lý bệnh viện theo kiến trúc **Microservices**.

Hệ thống này cần đảm bảo:

- Xử lý đồng thời nhiều yêu cầu từ bệnh nhân và bác sĩ.
- Giao tiếp giữa các bộ phận của bệnh viện (như quản lý bệnh nhân, quản lý lịch khám, đơn thuốc, và thông báo) một cách hiệu quả.
- Đảm bảo dữ liệu được lưu trữ và truy xuất an toàn, nhanh chóng.

Mục tiêu của hệ thống:

1. **Quản lý bệnh nhân:**
    - Đăng ký thông tin bệnh nhân mới.
    - Cập nhật và tra cứu thông tin bệnh nhân (hồ sơ bệnh án, lịch sử khám chữa bệnh).
2. **Quản lý lịch khám:**
    - Đặt lịch khám theo yêu cầu của bệnh nhân.
    - Cập nhật và xác nhận lịch khám cho từng bác sĩ.
3. **Quản lý đơn thuốc:**
    - Tạo đơn thuốc cho bệnh nhân sau khi khám.
    - Cập nhật tình trạng đơn thuốc (đã lấy, chưa lấy).
4. **Gửi thông báo:**
    - Nhắc nhở lịch khám qua email hoặc SMS.
    - Thông báo đơn thuốc đã sẵn sàng.
5. **Báo cáo và thống kê:**
    - Thống kê số lượng bệnh nhân theo tháng.
    - Báo cáo số lượng đơn thuốc đã cấp.

Yêu cầu hệ thống:

Kiến trúc hệ thống:

- **Phân hệ website**: giao diện các chức năng, tương tác với người dùng, thực hiện các chức năng thì gọi các services liên quan. Sử dụng kiến trúc MVC. Ngôn ngữ **php / python (được sử dụng framework)** – Không sử dụng các backend server khác e.g. nodejs, java web cho phân hệ website.
- Các Hệ thống microservice con: **Tuỳ chọn kỹ thuật, công nghệ**

Hiệu năng:

- Hệ thống phải xử lý ít nhất 100 yêu cầu/giây trong giờ cao điểm.
- Độ trễ tối đa khi gọi API không quá 1 giây.

Bảo mật:

- Cần có cơ chế xác thực (authentication) và phân quyền (authorization) cho nhân viên bệnh viện.
- Mã hóa dữ liệu nhạy cảm của bệnh nhân (số điện thoại, email).

Tính mở rộng:

- Có khả năng mở rộng từng phần của hệ thống khi số lượng bệnh nhân tăng.
- Các dịch vụ phải hoạt động độc lập, nếu một dịch vụ gặp lỗi, các dịch vụ khác vẫn phải hoạt động bình thường.

Yêu cầu đồ án:

1. **Phân tích nghiệp vụ:**
   - **Xác định các tác nhân (actors):** Bệnh nhân, bác sĩ, quản trị viên,....
   - **Xác định các trường hợp sử dụng (use cases):** Đặt lịch khám, quản lý bệnh nhân, quản lý đơn thuốc, gửi thông báo,....
   - **Xây dựng sơ đồ Use Case:** Giúp hình dung các chức năng hệ thống cần có.
2. **Thiết kế kiến trúc hệ thống:**
   - **Chọn mô hình microservices:**
     - Đề xuất chia nhỏ hệ thống theo chức năng (functional decomposition).
     - Sử dụng **RabbitMQ** để giao tiếp bất đồng bộ giữa các dịch vụ, đảm bảo tính nhất quán và hiệu năng.
   - **Các loại giao tiếp giữa dịch vụ:**
     - **Đồng bộ:** Khi cần phản hồi ngay (ví dụ: xác nhận đăng ký lịch khám).

- **Bất đồng bộ:** Khi không yêu cầu phản hồi tức thì (ví dụ: gửi thông báo nhắc lịch).

3. **Thiết kế cơ sở dữ liệu:**
   - **Đề xuất:** Chia thành các cơ sở dữ liệu riêng cho từng dịch vụ để tránh phụ thuộc.
   - **Các bảng có thể cần:**
     - Bệnh nhân (Patient): ID, tên, tuổi, giới tính, số điện thoại.
     - Lịch khám (Appointment): ID, mã bệnh nhân, bác sĩ, thời gian, trạng thái.
     - Đơn thuốc (Prescription): ID, mã bệnh nhân, thuốc, liều lượng.
     - .....

4. **Gợi ý công nghệ:**
   - **Web API:** Java Spring Boot.
   - **Giao tiếp bất đồng bộ:** RabbitMQ.
   - **Giao tiếp đồng bộ:** REST API.
   - **Cơ sở dữ liệu:**
     - **MySQL/PostgreSQL:** Dữ liệu quan hệ (bệnh nhân, lịch khám).
     - **MongoDB:** Dữ liệu không quan hệ (log, thông báo).

5. **Yêu cầu báo cáo và nộp bài:**
   1. **Báo cáo phân tích:**
      - Sơ đồ Use Case, sơ đồ tuần tự (Sequence Diagram).
      - Mô tả nghiệp vụ và các chức năng chính.
   2. **Báo cáo thiết kế:**
      - Sơ đồ kiến trúc microservices.
      - Cách sử dụng RabbitMQ để truyền thông điệp giữa các dịch vụ.
   3. **Mã nguồn:**
      - Đẩy lên GitHub với cấu trúc rõ ràng và README.md hướng dẫn.
   4. **Video demo:**
      - Giới thiệu tổng quan hệ thống (3-5 phút).
      - Demo các chức năng chính.