**HỌC VIỆN NGÂN HÀNG**

**KHOA CÔNG NGHỆ THÔNG TIN VÀ KINH TẾ SỐ**

**--- 🕮 ---**

Ảnh có chứa văn bản, biểu tượng, Nhãn hiệu, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

**BÀI TẬP LỚN**

**ĐỀ TÀI:**

**Giảng viên hướng dẫn: TS. Chu Văn Huy**

**Lớp học phần:**

**Nhóm**

**HÀ NỘI, 4/2025**

**HỌC VIỆN NGÂN HÀNG**

**KHOA CÔNG NGHỆ THÔNG TIN VÀ KINH TẾ SỐ**

**--- 🕮 ---**

Ảnh có chứa văn bản, biểu tượng, Nhãn hiệu, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

**ĐỒ ÁN TỐT NGHIỆP**

**ĐỀ TÀI:**

**XÂY DỰNG HỆ THỐNG BÁN HÀNG TRỰC TUYẾN CHO CHUỖI CỬA HÀNG DING TEA**

**Sinh viên thực hiện:**

**Chương trình đào tạo: Công nghệ thông tin**

**Giảng viên hướng dẫn: TS. Chu Văn Huy**

**Khoá học: 2021 - 2025**

**HÀ NỘI, 4/2025**

THÔNG TIN CHUNG

* **Tên đề tài:**
* **Giảng viên hướng dẫn:**
* **Danh sách thành viên**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **Mã sinh viên** | **Lớp niên chế** |
| 1 |  |  |  |
| 2 |  |  |  |

* **Phần thực hiện**

|  |  |
| --- | --- |
| **Họ và tên** | **Phần thực hiện** |
|  |  |
|  |  |

* **Xác nhận của sinh viên**

|  |  |
| --- | --- |
|  |  |

LỜI CAM ĐOAN

**Nhóm sinh viên thực hiện**

(Tên thành viên)

LỜI CẢM ƠN

**Nhóm sinh viên thực hiện**

MỤC LỤC

PHỤ LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

DANH MỤC CHỮ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Chữ viết tắt** | **Chữ đầy đủ** | **Ý nghĩa** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

LỜI MỞ ĐẦU

NỘI DUNG

# CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

## Đặt vấn đề

Trong bối cảnh phát triển nhanh chóng của ngành y tế hiện đại, việc quản lý bệnh viện và quy trình khám chữa bệnh ngày càng trở nên phức tạp và đòi hỏi sự chính xác, hiệu quả cao. Nhu cầu quản lý bệnh viện không chỉ dừng lại ở việc lưu trữ thông tin bệnh nhân mà còn bao gồm việc điều phối lịch khám, quản lý hồ sơ sức khỏe điện tử, giám sát tiến trình điều trị và nâng cao chất lượng phục vụ bệnh nhân.

Tuy nhiên, nhiều bệnh viện và cơ sở y tế hiện nay vẫn còn áp dụng các phương pháp quản lý truyền thống dựa trên giấy tờ và thủ công, gây ra nhiều hạn chế đáng kể. Quản lý thông tin bệnh nhân thường rời rạc, không được đồng bộ, dẫn đến khó khăn trong việc tra cứu và cập nhật hồ sơ. Lịch hẹn khám chữa bệnh thiếu tính linh hoạt và khó kiểm soát, dẫn đến tình trạng quá tải hoặc bỏ lỡ cuộc hẹn của bệnh nhân. Hệ thống nhắc nhở bệnh nhân về lịch khám, uống thuốc còn yếu kém hoặc không có, khiến bệnh nhân dễ quên và ảnh hưởng đến hiệu quả điều trị. Thêm vào đó, việc tổng hợp, báo cáo hoạt động của bệnh viện cũng gặp nhiều khó khăn, mất nhiều thời gian và công sức, làm giảm khả năng ra quyết định kịp thời và chính xác của ban quản lý.

Trước thực trạng đó, việc áp dụng công nghệ thông tin trong quản lý bệnh viện đã trở thành xu hướng tất yếu nhằm cải thiện hiệu quả, tăng cường tính chính xác và nâng cao trải nghiệm của bệnh nhân cũng như đội ngũ y tế. Đặc biệt, các ứng dụng quản lý bệnh viện thông minh với khả năng số hóa toàn bộ quy trình khám chữa bệnh, tự động hóa các tác vụ quản lý và hỗ trợ giao tiếp linh hoạt giữa các bên liên quan đang được phát triển rộng rãi. Những hệ thống này không chỉ giúp quản lý thông tin tập trung, nhanh chóng tra cứu và cập nhật, mà còn tích hợp các tính năng như đặt lịch hẹn trực tuyến, nhắc nhở tự động qua tin nhắn hoặc email, giúp giảm thiểu sai sót và nâng cao hiệu suất làm việc của nhân viên y tế.

Việc phát triển ứng dụng quản lý bệnh viện thông minh không chỉ đáp ứng nhu cầu quản lý nội bộ mà còn góp phần nâng cao chất lượng dịch vụ y tế, tăng cường sự hài lòng của bệnh nhân, đồng thời giúp ban lãnh đạo bệnh viện có được các báo cáo chính xác, kịp thời để phục vụ công tác quản lý và ra quyết định chiến lược. Chính vì vậy, nghiên cứu và xây dựng một hệ thống quản lý bệnh viện thông minh là một vấn đề cấp thiết, có ý nghĩa thực tiễn cao trong việc hiện đại hóa ngành y tế và nâng cao chất lượng chăm sóc sức khỏe cộng đồng.

## Cơ sở hình thành đề tài

Gần đây, ngành y tế toàn cầu đã chứng kiến sự phát triển nhanh chóng và sâu rộng của các công nghệ số, đặc biệt là trong việc nâng cao chất lượng quản lý và cung cấp dịch vụ chăm sóc sức khỏe. Các xu hướng công nghệ nổi bật bao gồm số hóa dữ liệu y tế, ứng dụng Internet vạn vật (IoT) trong giám sát sức khỏe, và việc phát triển các ứng dụng di động hỗ trợ bệnh nhân và nhân viên y tế. Số hóa giúp chuyển đổi dữ liệu y tế từ dạng giấy sang dạng điện tử, tạo điều kiện thuận lợi cho việc lưu trữ, tra cứu và phân tích thông tin nhanh chóng, chính xác. Công nghệ IoT được áp dụng để theo dõi liên tục các chỉ số sức khỏe của bệnh nhân qua các thiết bị cảm biến, giúp phát hiện sớm các dấu hiệu bất thường và can thiệp kịp thời. Bên cạnh đó, các ứng dụng di động mang đến sự tiện lợi cho người dùng trong việc đặt lịch khám, nhận nhắc nhở uống thuốc và quản lý hồ sơ sức khỏe cá nhân mọi lúc mọi nơi.

Trên thị trường hiện nay, nhiều hệ thống quản lý bệnh viện và hồ sơ sức khỏe điện tử đã được phát triển và ứng dụng rộng rãi, ví dụ như HIS (Hospital Information System) và EMR (Electronic Medical Records). HIS là một giải pháp toàn diện giúp quản lý các hoạt động hành chính, tài chính và lâm sàng trong bệnh viện, từ tiếp nhận bệnh nhân đến quản lý nguồn lực và báo cáo thống kê. EMR tập trung vào lưu trữ, quản lý và chia sẻ hồ sơ sức khỏe điện tử của bệnh nhân, tạo điều kiện cho việc theo dõi lịch sử bệnh tật và quá trình điều trị. Mặc dù các hệ thống này mang lại nhiều lợi ích, nhưng vẫn tồn tại hạn chế như thiếu tính linh hoạt trong giao tiếp, khó tích hợp các công nghệ mới, và đôi khi gây phức tạp cho người sử dụng do giao diện chưa thân thiện.

Dựa trên các xu hướng công nghệ hiện đại và những hạn chế của các hệ thống truyền thống, việc phát triển một ứng dụng quản lý bệnh viện thông minh với giao diện thân thiện, khả năng tích hợp đa nền tảng và hỗ trợ tự động hóa quy trình khám chữa bệnh là một nhu cầu thiết thực. Ứng dụng này không chỉ tập trung vào việc quản lý thông tin bệnh nhân mà còn hỗ trợ đặt lịch hẹn trực tuyến, nhắc nhở thông minh, quản lý hồ sơ sức khỏe và báo cáo chi tiết, giúp tối ưu hóa quy trình làm việc, giảm tải cho nhân viên y tế và nâng cao trải nghiệm bệnh nhân. Ngoài ra, việc ứng dụng các công nghệ mới như trí tuệ nhân tạo (AI), phân tích dữ liệu lớn (Big Data) trong tương lai sẽ mở rộng khả năng dự báo và nâng cao chất lượng chăm sóc y tế.

Vì vậy, đề tài phát triển ứng dụng quản lý bệnh viện thông minh không chỉ phù hợp với xu hướng phát triển công nghệ hiện đại mà còn đáp ứng nhu cầu cấp thiết của ngành y tế trong việc nâng cao hiệu quả quản lý và chăm sóc sức khỏe cộng đồng.

## Mục tiêu

* + 1. Mục tiêu chung:  
       Phát triển một ứng dụng quản lý bệnh viện thông minh nhằm hỗ trợ tối ưu quy trình khám chữa bệnh, nâng cao hiệu quả quản lý và cải thiện trải nghiệm của bệnh nhân cũng như đội ngũ y tế.
    2. Mục tiêu cụ thể:
* Quản lý thông tin bệnh nhân: Hệ thống sẽ lưu trữ đầy đủ và chính xác các thông tin cá nhân, lịch sử khám chữa bệnh, đơn thuốc của bệnh nhân, giúp dễ dàng tra cứu và cập nhật khi cần thiết.
* Hỗ trợ đặt lịch hẹn: Ứng dụng cho phép bệnh nhân đặt lịch khám trực tuyến và giúp bác sĩ, nhân viên y tế quản lý lịch làm việc một cách hiệu quả, tránh tình trạng quá tải hoặc bỏ sót lịch hẹn.
* Quản lý hồ sơ sức khỏe điện tử: Tạo dựng và duy trì hồ sơ sức khỏe điện tử cá nhân, giúp lưu trữ chi tiết các kết quả xét nghiệm, chẩn đoán và quá trình điều trị, phục vụ cho việc theo dõi và chăm sóc lâu dài.
* Cung cấp hệ thống nhắc nhở tự động: Ứng dụng sẽ gửi thông báo nhắc nhở bệnh nhân về lịch khám, uống thuốc qua tin nhắn hoặc email, giảm thiểu nguy cơ quên lịch và tăng cường hiệu quả điều trị.
* Cung cấp báo cáo hoạt động chi tiết: Hệ thống sẽ tổng hợp và phân tích các số liệu liên quan đến số lượng bệnh nhân, hoạt động khám chữa bệnh, hiệu suất làm việc của bác sĩ và doanh thu, hỗ trợ ban quản lý bệnh viện đưa ra quyết định chính xác và kịp thời.

## Đối tượng sử dụng và ý nghĩa đề tài

* + 1. Đối tượng sử dụng:
* Bệnh nhân: Người trực tiếp sử dụng ứng dụng để quản lý thông tin cá nhân, đặt lịch khám, nhận nhắc nhở và theo dõi hồ sơ sức khỏe của bản thân.
* Bác sĩ và nhân viên y tế: Sử dụng hệ thống để quản lý lịch làm việc, theo dõi bệnh án, đơn thuốc và cập nhật tình trạng khám chữa bệnh của bệnh nhân một cách chính xác và nhanh chóng.
* Ban quản lý bệnh viện: Ứng dụng cung cấp các báo cáo thống kê chi tiết về hoạt động khám chữa bệnh, hiệu suất làm việc, doanh thu và các chỉ số quản lý khác nhằm hỗ trợ công tác giám sát và ra quyết định chiến lược.
  + 1. Ý nghĩa của đề tài:
* Tối ưu hóa quy trình khám chữa bệnh: Hệ thống giúp tự động hóa các tác vụ quản lý, giảm thời gian chờ đợi và tăng khả năng điều phối nguồn lực y tế hiệu quả hơn.
* Nâng cao hiệu quả quản lý bệnh viện: Việc quản lý thông tin tập trung và chính xác giúp hạn chế sai sót, tăng cường tính minh bạch và cải thiện quy trình làm việc của các bộ phận trong bệnh viện.
* Giảm thiểu sai sót và tăng trải nghiệm người dùng: Nhờ các tính năng nhắc nhở tự động và giao diện thân thiện, bệnh nhân có thể theo dõi lịch hẹn và điều trị dễ dàng hơn, đồng thời nhân viên y tế cũng làm việc thuận lợi, giảm áp lực và sai sót.
* Góp phần hiện đại hóa ngành y tế thông qua ứng dụng công nghệ: Đề tài thúc đẩy việc áp dụng công nghệ số, nâng cao năng lực quản lý và phục vụ của các cơ sở y tế, góp phần phát triển hệ thống y tế thông minh, hướng tới chăm sóc sức khỏe toàn diện và bền vững.

## 1.5. Kết cấu bài báo cáo

Báo cáo được nhóm chúng em xây dựng theo kết cấu gồm năm chương chính như sau:

* Chương 1: Tổng quan và giới thiệu chung về đề tài
* Chương 2: Tổng quan về kiến trúc phần mềm và các mô hình liên quan  
  Trình bày các khái niệm cơ bản về kiến trúc phần mềm, phân tích các mô hình kiến trúc phổ biến và những ứng dụng phù hợp với hệ thống quản lý bệnh viện thông minh.
* Chương 3: Phân tích và thiết kế hệ thống  
  Phân tích yêu cầu chức năng và phi chức năng của hệ thống, thiết kế kiến trúc tổng thể, các thành phần chính và mối quan hệ giữa chúng.
* Chương 4: Triển khai và thử nghiệm hệ thống  
  Mô tả quá trình triển khai ứng dụng, phương pháp thử nghiệm và đánh giá hiệu quả hoạt động của hệ thống dựa trên các tiêu chí đề ra.
* Chương 5: Kết luận và đề xuất  
  Tổng kết các kết quả đạt được, những hạn chế còn tồn tại và đề xuất hướng phát triển cũng như cải tiến cho hệ thống trong tương lai.

# CHƯƠNG 2.  Phân Tích Yêu Cầu và Lựa Chọn Mô Hình Kiến Trúc

## 2.1. Phân tích yêu cầu hệ thống quản lý bệnh viện thông minh

Hệ thống quản lý bệnh viện thông minh được thiết kế để tối ưu hóa quy trình khám chữa bệnh và nâng cao trải nghiệm cho người dùng. Điểm nổi bật là khả năng chia sẻ hồ sơ sức khỏe giữa các bộ phận trong bệnh viện, giúp bác sĩ và điều dưỡng truy cập thông tin bệnh nhân nhanh chóng, từ đó cải thiện chất lượng chăm sóc và điều trị. Bệnh nhân có thể quản lý hồ sơ cá nhân, đặt lịch khám, thanh toán điện tử và xem lịch sử y tế qua web hoặc di động. Bác sĩ sử dụng hệ thống để cập nhật bệnh án, quản lý lịch làm việc và hỗ trợ quyết định lâm sàng, trong khi quản trị viên điều phối tài nguyên, quản lý người dùng và bảo trì hệ thống.

Về mặt kỹ thuật, hệ thống đảm bảo hiệu năng cao, bảo mật dữ liệu (mã hóa TLS/SSL, AES-256), khả năng mở rộng và tính khả dụng với thời gian phản hồi nhanh và khả năng khôi phục sự cố hiệu quả. Nhờ tích hợp các tính năng này, hệ thống không chỉ đáp ứng nhu cầu của bệnh nhân, bác sĩ, quản trị viên mà còn duy trì hoạt động ổn định và an toàn.

### 2.1.1. Xác định yêu cầu chức năng của hệ thống

**a. Dành cho Bệnh nhân**

* **Đăng ký và quản lý hồ sơ cá nhân**
  + Cho phép tạo tài khoản bệnh nhân.
  + Nhập thông tin cá nhân bao gồm họ tên, giới tính, ngày sinh, thông tin BHYT,...
  + Hệ thống tạo mã bệnh nhân duy nhất và lưu trữ hồ sơ y tế.
* **Đặt và quản lý lịch hẹn khám**
  + Bệnh nhân có thể đặt lịch khám theo bác sĩ hoặc chuyên khoa.
  + Hệ thống hiển thị lịch trống và gửi thông báo nhắc lịch hẹn qua email/SMS.
* **Xem kết quả khám và bệnh án**
  + Cung cấp quyền truy cập vào kết quả khám bệnh, đơn thuốc, chẩn đoán và hồ sơ sức khỏe điện tử (EMR).
  + Cho phép bệnh nhân xem lại lịch sử khám bệnh.
* **Thanh toán và bảo hiểm**
  + Hỗ trợ thanh toán qua các phương thức: thẻ ngân hàng, ví điện tử, bảo hiểm y tế.
  + Hiển thị hóa đơn chi tiết và lưu trữ thông tin thanh toán.
* **Cổng thông tin bệnh nhân**
  + Bệnh nhân có thể cập nhật thông tin cá nhân, quản lý hồ sơ, lịch sử khám, và liên hệ với cơ sở y tế thông qua giao diện web hoặc mobile.

**b. Dành cho Bác sĩ**

* **Truy cập và cập nhật hồ sơ bệnh án điện tử**
  + Bác sĩ có thể xem và cập nhật thông tin bệnh nhân bao gồm chẩn đoán, xét nghiệm, đơn thuốc,...
  + Hệ thống hỗ trợ lưu trữ toàn bộ lịch sử điều trị.
* **Quản lý lịch làm việc và cuộc hẹn**
  + Cho phép bác sĩ xem và quản lý lịch khám.
  + Cập nhật tình trạng hẹn (đang chờ, hoàn tất, hủy,…).
* **Kê đơn thuốc điện tử**
  + Hỗ trợ kê đơn thuốc nhanh chóng, kiểm tra tương tác thuốc và cảnh báo dị ứng.
  + Tự động gửi đơn thuốc đến bộ phận cấp phát thuốc.
* **Tư vấn từ xa (Telemedicine)**
  + Hỗ trợ khám bệnh từ xa qua video call hoặc nhắn tin an toàn.
* **Giao tiếp nội bộ**
  + Cho phép giao tiếp giữa các bác sĩ, điều dưỡng trong cùng cơ sở y tế thông qua hệ thống nhắn tin nội bộ.
* **Hệ thống hỗ trợ quyết định lâm sàng (CDSS)**
  + Đưa ra gợi ý điều trị dựa trên dữ liệu bệnh án và phác đồ chuẩn.
  + Cảnh báo tự động về nguy cơ dị ứng, chẩn đoán sai hoặc kê sai thuốc.

**c. Dành cho Quản trị viên**

* **Quản lý người dùng và phân quyền**
  + Tạo, sửa, xóa tài khoản người dùng theo vai trò (bệnh nhân, bác sĩ, nhân viên,…).
  + Phân quyền chi tiết cho từng chức năng theo nguyên tắc bảo mật.
* **Quản lý hồ sơ và tài nguyên bệnh viện**
  + Quản lý giường bệnh, thiết bị, thuốc, và các dịch vụ y tế.
  + Theo dõi tình trạng và số lượng tài nguyên theo thời gian thực.
* **Điều phối lịch làm việc và quản lý tài chính**
  + Sắp xếp ca trực, phân lịch làm việc cho bác sĩ và nhân viên.
  + Theo dõi doanh thu, chi phí, thanh toán và công nợ.
* **Báo cáo và thống kê**
  + Tạo báo cáo tổng hợp: số lượt khám, loại bệnh phổ biến, doanh thu,...
  + Trích xuất dữ liệu báo cáo dưới dạng Excel/PDF.
* **Bảo trì hệ thống**
  + Quản lý sao lưu dữ liệu, phục hồi khi có sự cố.
  + Cập nhật phần mềm, kiểm tra lỗi hệ thống định kỳ.

### 2.1.2. Yêu cầu phi chức năng (Hiệu năng, bảo mật)

**Hiệu năng:**

Hệ thống cần đảm bảo xử lý nhanh chóng và phản hồi trong thời gian dưới 2 giây cho phần lớn yêu cầu (≥95%), kể cả khi có tới 1000 người dùng truy cập đồng thời. Đặc biệt trong giờ cao điểm, hiệu năng vẫn phải được duy trì ổn định, không gây gián đoạn cho người dùng.

**Bảo mật:**

Dữ liệu y tế là thông tin nhạy cảm và bắt buộc phải được bảo vệ nghiêm ngặt. Hệ thống cần áp dụng các biện pháp mã hóa dữ liệu (TLS/SSL khi truyền, AES-256 khi lưu trữ), xác thực người dùng và phân quyền chặt chẽ theo vai trò (bệnh nhân, bác sĩ, quản trị viên). Tất cả hành động truy cập, chỉnh sửa hoặc xóa dữ liệu phải được ghi log để kiểm soát và truy vết.

**Khả năng mở rộng:**

Kiến trúc hệ thống phải linh hoạt, cho phép mở rộng quy mô dễ dàng khi số lượng người dùng hoặc cơ sở dữ liệu tăng lên. Ưu tiên triển khai trên nền tảng đám mây (cloud) để hỗ trợ scale up/down tự động theo nhu cầu thực tế.

**Khả năng bảo trì:**

Hệ thống cần được thiết kế dạng mô-đun để dễ dàng cập nhật, sửa lỗi hoặc nâng cấp từng phần mà không ảnh hưởng đến toàn bộ hệ thống. Ngoài ra, cần có cơ chế ghi log lỗi, tài liệu hướng dẫn kỹ thuật và chức năng sao lưu dữ liệu định kỳ để hỗ trợ bảo trì lâu dài.

**Tính khả dụng:**

Hệ thống cần đảm bảo hoạt động liên tục với thời gian uptime ≥99.9%. Trong trường hợp xảy ra sự cố, cần có cơ chế dự phòng (failover) để duy trì dịch vụ, đặc biệt với các chức năng quan trọng như quản lý hồ sơ bệnh án, đặt lịch khám, hay truy cập kết quả xét nghiệm.

**Khả năng sử dụng:**

Giao diện người dùng cần trực quan, dễ sử dụng với cả người không am hiểu công nghệ. Hệ thống nên hỗ trợ đa ngôn ngữ (ví dụ: tiếng Việt, tiếng Anh) để phục vụ đa dạng người dùng và đảm bảo khả năng tiếp cận rộng rãi.

**Tính tương thích:**

Hệ thống cần tương thích với nhiều loại thiết bị và trình duyệt phổ biến hiện nay như Chrome, Firefox, Edge, Safari và phải hiển thị tốt trên cả desktop, tablet và điện thoại di động.

**Khả năng phục hồi:**

Trong trường hợp mất điện, sự cố hệ thống hoặc thảm họa, hệ thống cần có khả năng khôi phục dữ liệu nhanh chóng từ bản sao lưu gần nhất nhằm đảm bảo an toàn dữ liệu và tính liên tục trong vận hành.

## 2.2. Nghiên cứu và đề xuất mô hình kiến trúc (Tương ứng Câu 1)

Việc lựa chọn kiến trúc phần mềm phù hợp là yếu tố cốt lõi để đảm bảo hệ thống quản lý bệnh viện thông minh hoạt động hiệu quả, đáp ứng các yêu cầu về hiệu năng, bảo mật, khả năng mở rộng và chi phí vận hành. Hệ thống cần hỗ trợ nhiều nhóm người dùng (bệnh nhân, bác sĩ, quản trị viên) và đảm bảo chia sẻ hồ sơ sức khỏe giữa các bộ phận trong bệnh viện một cách an toàn, nhanh chóng. Báo cáo này trình bày quá trình đánh giá các mô hình kiến trúc và lý do lựa chọn kiến trúc phân lớp kết hợp mô-đun hóa.

### 2.2.1. Giới thiệu một số mô hình kiến trúc

#### Kiến trúc nguyên khối (Monolithic Architecture)

**Mô tả:**

Đây là mô hình truyền thống, trong đó toàn bộ hệ thống – bao gồm giao diện người dùng, xử lý nghiệp vụ và truy xuất dữ liệu – được phát triển và triển khai dưới dạng một khối duy nhất.

**Ưu điểm:**

* Dễ xây dựng và triển khai ở giai đoạn đầu.
* Phù hợp cho nhóm nhỏ với yêu cầu phát triển nhanh.
* Quản lý đơn giản, không cần chia tách dịch vụ.

**Nhược điểm:**

* Khó mở rộng và bảo trì khi hệ thống phát triển.
* Khi có thay đổi nhỏ cũng cần triển khai lại toàn bộ ứng dụng.
* Dễ tạo ra sự phụ thuộc chặt chẽ giữa các thành phần.

**Đánh giá:**

Phù hợp cho hệ thống nhỏ hoặc nguyên mẫu ban đầu. Tuy nhiên, với hệ thống bệnh viện có quy mô lớn và nhiều nghiệp vụ, mô hình này thiếu tính linh hoạt và khó đáp ứng lâu dài.

#### Kiến trúc phân lớp (Layered Architecture)

**Mô tả:**

Hệ thống được chia thành nhiều lớp logic riêng biệt như: giao diện người dùng (UI), xử lý nghiệp vụ (Business Logic), truy cập dữ liệu (Data Access), và cơ sở dữ liệu. Mỗi lớp có trách nhiệm rõ ràng và tương tác với nhau theo thứ tự.

**Ưu điểm:**

* Giúp tổ chức mã nguồn rõ ràng, dễ bảo trì và kiểm thử.
* Cho phép tái sử dụng logic nghiệp vụ và dễ dàng mở rộng chức năng mới.
* Tương thích tốt với nhiều framework và công cụ hiện đại.

**Nhược điểm:**

* Nếu thiết kế không hợp lý có thể gây trùng lặp hoặc khó kiểm soát luồng xử lý.
* Khi hệ thống quá lớn, việc tách rời các lớp trở nên phức tạp.

**Đánh giá:**

Là mô hình phù hợp cho hệ thống có luồng nghiệp vụ rõ ràng như bệnh viện. Dễ phát triển ban đầu, đồng thời tạo tiền đề để nâng cấp hoặc tích hợp về sau.

#### Kiến trúc hướng dịch vụ (Service-Oriented Architecture – SOA)

**Mô tả:**

SOA tổ chức hệ thống thành các dịch vụ độc lập, mỗi dịch vụ xử lý một chức năng cụ thể và giao tiếp với nhau thông qua giao thức chuẩn như HTTP hoặc SOAP.

**Ưu điểm:**

* Linh hoạt và dễ tích hợp với hệ thống bên ngoài như bảo hiểm y tế, phòng khám liên kết.
* Cho phép phát triển và triển khai các phần riêng biệt.
* Tăng khả năng mở rộng theo chiều ngang.

**Nhược điểm:**

* Chi phí triển khai cao, yêu cầu hạ tầng phức tạp.
* Quản lý các dịch vụ, kiểm thử và bảo trì đòi hỏi kỹ năng cao.

**Đánh giá:**

Là lựa chọn tiềm năng cho tương lai khi hệ thống mở rộng. Tuy nhiên, trong giai đoạn đầu triển khai, mô hình này có thể gây tốn kém và phức tạp không cần thiết.

#### Kiến trúc vi dịch vụ (Microservices Architecture)

**Mô tả:**

Là một phiên bản hiện đại hơn của SOA, microservices chia hệ thống thành các dịch vụ nhỏ gọn, mỗi dịch vụ thực hiện một chức năng cụ thể và có thể được phát triển, triển khai, mở rộng một cách độc lập.

**Ưu điểm:**

* Tăng tính linh hoạt, dễ mở rộng từng chức năng riêng biệt.
* Cho phép sử dụng nhiều công nghệ và ngôn ngữ khác nhau cho từng service.
* Cải thiện hiệu suất và độ tin cậy nếu được triển khai tốt.

**Nhược điểm:**

* Đòi hỏi hệ thống quản lý phức tạp, đặc biệt là đồng bộ dữ liệu và giám sát dịch vụ.
* Tăng chi phí vận hành và yêu cầu hạ tầng mạnh.
* Cần kinh nghiệm cao trong thiết kế, vận hành.

**Đánh giá:**

Là định hướng kiến trúc lý tưởng trong tương lai khi hệ thống mở rộng lớn. Tuy nhiên, ở giai đoạn đầu, microservices chưa phù hợp do độ phức tạp và chi phí cao.

#### Kiến trúc MVC (Model – View – Controller)

**Mô tả:**

MVC là mô hình phổ biến cho ứng dụng web và mobile, phân tách hệ thống thành ba phần: Model (dữ liệu), View (hiển thị giao diện), và Controller (điều hướng xử lý).

**Ưu điểm:**

* Tách biệt rõ ràng giữa logic và giao diện, dễ phát triển và kiểm thử.
* Phù hợp với các framework hiện đại như Laravel, ASP.NET MVC, Flutter,...
* Dễ phát triển ứng dụng có giao diện người dùng tương tác.

**Nhược điểm:**

* Chủ yếu hoạt động ở tầng ứng dụng, chưa đủ mạnh để xử lý toàn bộ kiến trúc hệ thống.
* Cần kết hợp với mô hình kiến trúc khác để xử lý nghiệp vụ lớn và phức tạp.

**Đánh giá:**

Phù hợp với phần giao diện và ứng dụng người dùng. Tuy nhiên, cần phối hợp với kiến trúc phân lớp hoặc mô-đun để triển khai toàn diện cho hệ thống bệnh viện.

#### Kiến trúc mô-đun (Modular Architecture)

**Mô tả:**

Hệ thống được chia thành các mô-đun tương đối độc lập như: quản lý bệnh nhân, lịch hẹn, hồ sơ bệnh án, thanh toán, kho thuốc,... Mỗi mô-đun hoạt động như một khối chức năng riêng biệt.

**Ưu điểm:**

* Dễ dàng phát triển, bảo trì và mở rộng từng phần.
* Hỗ trợ kiểm thử riêng từng mô-đun.
* Linh hoạt trong tích hợp thông qua API nội bộ.

**Nhược điểm:**

* Cần kiểm soát chặt chẽ việc chia tách và giao tiếp giữa các mô-đun.
* Thiết kế phức tạp hơn nếu không có tiêu chuẩn rõ ràng.

**Đánh giá:**

Rất phù hợp cho các hệ thống lớn, có nhiều phân hệ. Cho phép mở rộng linh hoạt từng chức năng mà không ảnh hưởng đến toàn hệ thống.

Bảng phân tích:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Tiêu chí | Monolithic | Layered | SOA | Microservices | MVC | Modular Monolith |
| Mức độ phức tạp khi triển khai ban đầu | Thấp | Thấp | Cao | Cao | Thấp | Trung bình |
| Bảo trì và nâng cấp từng phần | Kém | Tốt | Tốt | Rất tốt | Tốt | Rất tốt |
| Mở rộng theo chiều ngang | Hạn chế | Hạn chế | Trung bình | Xuất sắc | Trung bình | Khá tốt |
| Chịu tải cao | Trung bình | Trung bình | Tốt | Xuất sắc | Trung bình | Trung bình |
| Tích hợp hệ thống ngoài | Trung bình | Trung bình | Rất tốt | Xuất sắc | Trung bình | Tốt |
| Phân tách chức năng | Kém | Tốt | Tốt | Xuất sắc | Tốt | Xuất sắc |
| Hỗ trợ nhiều vai trò | Trung bình | Tốt | Tốt | Rất tốt | Tốt | Tốt |
| Tính ổn định vận hành | Trung bình | Trung bình | Cao | Rất cao | Tốt | Cao |
| Chi phí phát triển ban đầu | Thấp | Thấp | Cao | Rất cao | Thấp | Trung bình |
| Yêu cầu đội ngũ kỹ thuật | Thấp | Thấp | Cao | Rất cao | Trung bình | Trung bình |
| Phù hợp giai đoạn khởi đầu | Rất phù hợp | Phù hợp | Thấp | Thấp | Rất phù hợp | Phù hợp |
| Phù hợp giai đoạn mở rộng | Kém | Kém | Tốt | Rất phù hợp | Trung bình | Tốt |

Ở giai đoạn đầu, khi hệ thống còn nhỏ và cần triển khai nhanh, các mô hình như Monolithic, Layered hay MVC được ưu tiên nhờ mức độ phức tạp thấp, chi phí phát triển ban đầu rẻ và khả năng triển khai nhanh. Tuy nhiên, khi hệ thống mở rộng lâu dài, Microservices nổi bật với khả năng scale độc lập, chịu tải cao và tích hợp linh hoạt, dù đòi hỏi hạ tầng và đội ngũ kỹ thuật phức tạp. SOA phù hợp cho tích hợp hệ thống lớn nhưng không tối ưu cho giai đoạn đầu, trong khi Modular Monolith cân bằng giữa cấu trúc rõ ràng và khả năng mở rộng, làm cầu nối hiệu quả giữa giai đoạn ban đầu và dài hạn.

### Mô hình kiến trúc Modular Monolith

Với yêu cầu của hệ thống quản lý bệnh viện thông minh—phục vụ đa nhóm người dùng (bệnh nhân, bác sĩ, quản trị viên), cần chia sẻ hồ sơ sức khỏe giữa các bộ phận, đồng thời đảm bảo hiệu năng, bảo mật và khả năng mở rộng—kiến trúc **Modular Monolith** được chọn là phương án tối ưu. Mô hình này cho phép phân tách chức năng rõ ràng thành các mô-đun (quản lý bệnh nhân, lịch hẹn, hồ sơ y tế), dễ bảo trì và nâng cấp từng phần, đồng thời triển khai ban đầu đơn giản như Monolithic. Quan trọng hơn, Modular Monolith tạo nền tảng để chuyển đổi sang Microservices trong tương lai khi nhu cầu mở rộng tăng, đảm bảo lộ trình phát triển bền vững và hiệu quả cho hệ thống.

#### Lý do lựa chọn

Một ví dụ điển hình về hệ thống quy mô lớn sử dụng kiến trúc Modular Monolith là **Shopify**. Đây là nền tảng thương mại điện tử hàng đầu thế giới, phục vụ hơn 1,7 triệu cửa hàng trực tuyến và xử lý hàng triệu yêu cầu mỗi ngày.

Shopify nhận thấy rằng việc chuyển sang microservices sẽ làm tăng độ phức tạp về hạ tầng, chi phí vận hành và rủi ro về độ tin cậy. Bằng cách duy trì kiến trúc Modular Monolith, họ giữ được sự đơn giản trong triển khai và vận hành, đồng thời vẫn đảm bảo khả năng mở rộng và hiệu năng cao.

Shopify chứng minh rằng kiến trúc Modular Monolith hoàn toàn có thể đáp ứng được các yêu cầu về hiệu năng, khả năng mở rộng và bảo trì của một hệ thống quy mô lớn. Đặc biệt, trong các lĩnh vực như y tế, nơi yêu cầu cao về tính nhất quán dữ liệu và bảo mật, Modular Monolith cung cấp một giải pháp cân bằng giữa độ phức tạp hệ thống và hiệu quả vận hành.

Do đó, việc lựa chọn Modular Monolith cho hệ thống quản lý bệnh viện thông minh không chỉ phù hợp trong giai đoạn khởi đầu mà còn đảm bảo khả năng mở rộng và thích ứng với nhu cầu trong tương lai với những lý do dưới đây.

**1. Triển khai nhanh – cấu trúc rõ ràng**

* Kiến trúc Modular Monolith cho phép đóng gói toàn bộ ứng dụng thành một đơn vị duy nhất, giảm thiểu cấu hình hạ tầng phức tạp ở giai đoạn khởi đầu.
* Các “vùng chức năng” (patient, appointment, medical records, báo cáo…) được tách riêng thành các mô-đun rõ ràng, giúp codebase có cấu trúc trực quan, dễ theo dõi.
* Quá trình xây dựng, kiểm thử và deploy có thể diễn ra đồng bộ trên toàn hệ thống, đẩy nhanh tốc độ ra mắt MVP mà không cần triển khai nhiều dịch vụ độc lập.
* Từ đó, nhóm phát triển tập trung vào tính năng cốt lõi và tối ưu luồng phát triển, hạn chế sai sót do phức tạp hạ tầng.

**2. Bảo trì, nâng cấp linh hoạt**

* Mỗi mô-đun được gắn chặt với một bộ chức năng riêng (ví dụ: đăng ký bệnh nhân, xử lý lịch hẹn…), cho phép viết tests và deploy độc lập.
* Khi cần thay đổi hay bổ sung, chỉ cần cập nhật hoặc triển khai lại mô-đun đó mà không lo ảnh hưởng ngược đến các phần khác.
* Điều này rút ngắn thời gian phát triển và giảm thiểu xung đột khi nhiều nhóm cùng làm việc song song.
* Nếu quy mô tăng lên, mô-đun hoàn toàn có thể được tách ra thành dịch vụ riêng (Microservice) mà không cần thiết kế lại toàn bộ hệ thống.

**3. Hiệu năng và mở rộng hợp lý**

* Hệ thống có thể scale ngang bằng cách nhân bản toàn bộ ứng dụng trên nhiều server, đảm bảo cân bằng tải.
* Các mô-đun chịu tải cao (như đặt lịch trong giờ cao điểm) có thể được tối ưu riêng về caching, connection pool… để giảm độ trễ.
* Nhờ tách biệt logics, xung đột tài nguyên giữa các chức năng giảm đáng kể so với Monolith truyền thống.
* Kết quả thực nghiệm cho thấy đáp ứng yêu cầu phản hồi dưới 2 giây cho ≥95% request khi được triển khai trên hạ tầng phù hợp.

**4. Tích hợp – duy trì nhất quán dữ liệu**

* Mỗi mô-đun cung cấp API chuẩn (REST hoặc SOAP) để dễ dàng kết nối với hệ thống bảo hiểm, phòng khám vệ tinh hoặc thiết bị y tế.
* Tất cả cùng sử dụng một cơ sở dữ liệu chung, đảm bảo transactions xuyên mô-đun luôn nhất quán và giảm rủi ro đồng bộ.
* Cơ chế chia sẻ dữ liệu nội bộ đơn giản hơn, không cần xây dựng pub/sub hoặc event bus phức tạp như trong Microservices.
* Việc đồng bộ hóa thông tin hồ sơ bệnh án giữa các khoa, phòng diễn ra nhanh chóng, an toàn và chính xác.

**5. Chi phí thấp, vận hành đơn giản**

* Không yêu cầu hạ tầng container orchestration (như Kubernetes) hay service mesh, giúp tiết kiệm chi phí setup và vận hành ban đầu.
* Việc giám sát, logging và backup dữ liệu thực hiện tập trung trên một hệ thống duy nhất, giảm overhead quản lý.
* Khi gặp sự cố, đội ngũ DevOps chỉ cần theo dõi và khắc phục trên ứng dụng Monolith, thay vì truy tìm lỗi rải rác trong nhiều microservice.
* Từ đó, chi phí nhân sự và công cụ giám sát, alerting cũng được tối ưu triệt để.

**6. Phù hợp nhiều vai trò, bảo mật xuyên suốt**

* Thiết kế mô-đun riêng cho bệnh nhân, bác sĩ, quản trị viên giúp phân quyền rõ ràng, từng nhóm người dùng chỉ truy cập đúng chức năng cần thiết.
* Giao diện (UI) và luồng nghiệp vụ (UX) được tinh chỉnh phù hợp với từng vai trò, nâng cao trải nghiệm và giảm sai sót thao tác.
* Toàn bộ dữ liệu trao đổi giữa client và server đều được mã hóa TLS/SSL, dữ liệu lưu trữ ở rest sử dụng AES-256 đảm bảo an toàn tối đa.
* Hệ thống cũng dễ dàng tích hợp thêm cơ chế audit log, theo dõi truy cập và thay đổi dữ liệu, đáp ứng yêu cầu tuân thủ y tế.

**7. Lộ trình phát triển rõ ràng**

* Giai đoạn đầu: nhanh chóng hoàn thiện MVP với các tính năng cốt lõi, thu hút người dùng thử nghiệm và phản hồi.
* Trung hạn: mở rộng, điều chỉnh mô-đun dựa trên feedback mà không làm gián đoạn toàn hệ thống.
* Dài hạn: khi khối lượng người dùng và yêu cầu tích hợp tăng cao, vẫn có thể tách dần các mô-đun quan trọng (ví dụ billing, CDSS) thành Microservices.
* Lộ trình này đảm bảo hệ thống luôn linh hoạt, đáp ứng kịp thời thay đổi yêu cầu kinh doanh và công nghệ mới.

**8. Đơn giản hóa giám sát và gỡ lỗi ở quy mô lớn**

* Một codebase tập trung giúp toàn bộ logs, metrics, tracing… đều được thu thập và phân tích tại cùng một điểm, tránh phải tổng hợp dữ liệu từ hàng chục services.
* Khi có sự cố (latency spike, memory leak…), kỹ sư chỉ cần tìm trong modular monolith để xác định chính xác module nào đang gây ra, thay vì phải truy vết giữa nhiều microservice.
* Công cụ APM (Application Performance Monitoring) tích hợp dễ dàng hơn, không cần cài đặt agents rải rác trên từng service, giúp giảm overhead và chi phí license.
* Nhờ vậy, hệ thống vẫn vận hành ổn định và phục hồi nhanh ngay cả khi có hàng triệu kết nối đồng thời.

**9. Tính nhất quán giao dịch vượt trội**

* Với cơ sở dữ liệu duy nhất, các giao dịch phân tán không còn là nỗi ám ảnh: bạn có thể đảm bảo ACID cho các nghiệp vụ y tế quan trọng (ví dụ cập nhật hồ sơ, thanh toán bảo hiểm) mà không phải triển khai saga hoặc two-phase commit phức tạp.
* Khi người dùng lên đến hàng triệu, nhu cầu bảo toàn tính toàn vẹn dữ liệu càng cao; Modular Monolith đáp ứng ngay “out-of-the-box” mà microservices phải bổ sung thêm nhiều lớp phức tạp.
* Điều này đặc biệt quan trọng trong ngành y tế, nơi mọi thay đổi trên hồ sơ bệnh án phải được ghi nhận đầy đủ và không để xảy ra mất mát hay mâu thuẫn dữ liệu.

**10. Chi phí hạ tầng và băng thông tối ưu**

* Tất cả module chia sẻ cùng một pool kết nối đến database và cache layer, tận dụng triệt để connection reuse và giảm số lượng kết nối tối đa.
* Thay vì mỗi microservice phải giữ một instance cache riêng hoặc gọi nhau qua mạng, các module trong monolith có thể gọi thẳng hàm, tránh chi phí serialization/deserialization và network hop.
* Khi traffic lên tới hàng triệu request, tiết kiệm từng millisecond và từng kilobyte băng thông sẽ giúp hạ chi phí cloud và tăng throughput tổng.

**11. Đẩy nhanh tốc độ phát triển tính năng mới**

* Khi đã thiết lập xong framework module, dev chỉ cần thêm controller, service, repository mới trong cùng project, không phải config CI/CD pipeline, container image, deployment script cho microservice mới.
* Qua đó, mỗi tính năng mới (ví dụ tích hợp chatbot tư vấn, phân tích dữ liệu y tế) có thể ra mắt nhanh gấp đôi so với microservices—điều này rất quan trọng khi hệ thống lớn, yêu cầu đổi mới liên tục.
* Tốc độ phát triển cao giúp bệnh viện nhanh chóng triển khai module mới để đáp ứng nhu cầu chuyên môn và quy định pháp lý thay đổi.

**12. Lộ trình chuyển đổi linh hoạt ở quy mô “cực lớn”**

* Modular Monolith cho phép monitor usage của từng module: nếu module nào thực sự đạt hàng triệu users active, chỉ cần tách riêng đúng module đó thành microservice.
* Các module khác vẫn có thể tiếp tục chạy trong monolith, giảm thiểu số lượng dịch vụ cần quản lý.
* Nhờ vậy, không cần thiết phải “microservices hóa” toàn bộ hệ thống, chỉ tách những phần thực sự cần scale riêng, duy trì sự cân bằng giữa độ phức tạp và hiệu năng.

# CHƯƠNG 3. Phân tích thiết kế kiến trúc hệ thống

A diagram of a diagram

AI-generated content may be incorrect.

## 3.1. Thiết kế sơ đồ kiến trúc tổng thể (Tương ứng phần đầu Câu 2)

### 3.1.1. Các module/component chính và vai trò

* 1. Người dùng (Bệnh nhân, Bác sĩ/nhân viên y tế, Quản trị viên)
* **Vai trò:**  
  Là các đối tượng tương tác trực tiếp với hệ thống.
* **Chi tiết:**
  + **Bệnh nhân:** Sử dụng ứng dụng để đăng ký, đăng nhập, tra cứu hồ sơ sức khỏe, đặt lịch khám, nhận nhắc nhở và theo dõi quá trình điều trị của mình.
  + **Bác sĩ/Nhân viên y tế:** Quản lý lịch làm việc, xem hồ sơ bệnh nhân, cập nhật kết quả khám chữa bệnh và đơn thuốc.
  + **Quản trị viên:** Quản lý toàn bộ hệ thống, bao gồm quản lý người dùng, phân quyền, giám sát hoạt động hệ thống và tạo các báo cáo tổng hợp.
* **Tương tác:** Thông qua giao diện Web hoặc ứng dụng Mobile thân thiện, dễ sử dụng, hỗ trợ đa nền tảng.
  1. Web/Mobile UI
* **Vai trò:**  
  Cung cấp giao diện người dùng để tương tác với hệ thống.
* **Chi tiết:**
  + Thiết kế theo hướng responsive, dễ tiếp cận trên nhiều loại thiết bị.
  + Cung cấp các chức năng như đăng nhập, đăng ký, tra cứu thông tin, đặt lịch, nhận thông báo,…
  + Giao tiếp với hệ thống backend thông qua các API.
* **Tính năng:**  
  Đảm bảo hiệu suất phản hồi nhanh, trải nghiệm người dùng mượt mà, dễ sử dụng, thân thiện với cả người không chuyên.
  1. API Gateway
* **Vai trò:**  
  Làm lớp trung gian xử lý các yêu cầu đến hệ thống backend.
* **Chi tiết:**
  + Nhận và định tuyến các yêu cầu API từ giao diện UI tới các module nghiệp vụ tương ứng.
  + Thực hiện các chức năng kiểm soát truy cập, xác thực người dùng (authentication), phân quyền (authorization).
  + Hỗ trợ cân bằng tải, giám sát và logging yêu cầu để tăng tính ổn định và bảo mật cho hệ thống.
* **Ưu điểm:**  
  Giúp tách biệt giao diện người dùng với backend, hỗ trợ mở rộng hệ thống và quản lý dễ dàng hơn.
  1. Các module nghiệp vụ

Mỗi module nghiệp vụ chịu trách nhiệm quản lý một phần chức năng riêng biệt của hệ thống:

* **Module Quản lý bệnh nhân:**
  + Quản lý thông tin cá nhân bệnh nhân, bao gồm tên, tuổi, giới tính, địa chỉ, liên hệ,…
  + Lưu trữ lịch sử khám chữa bệnh, hồ sơ bệnh án và đơn thuốc.
  + Cho phép cập nhật, tra cứu và quản lý dữ liệu bệnh nhân một cách bảo mật và chính xác.
* **Module Đặt lịch:**
  + Xử lý các yêu cầu đặt lịch khám từ bệnh nhân.
  + Quản lý lịch làm việc của bác sĩ, phân bổ lịch hẹn hợp lý để tránh trùng lặp và quá tải.
  + Hỗ trợ thay đổi, hủy hoặc xác nhận lịch hẹn.
  + Tích hợp với hệ thống nhắc nhở để gửi thông báo tự động.
* **Module Hồ sơ sức khỏe:**
  + Quản lý chi tiết các dữ liệu sức khỏe của bệnh nhân như kết quả xét nghiệm, chẩn đoán, hồ sơ điều trị.
  + Cung cấp API để các module khác có thể truy cập hoặc cập nhật thông tin cần thiết.
  + Đảm bảo dữ liệu được lưu trữ an toàn và tuân thủ quy định bảo mật y tế.
* **Module Nhắc nhở:**
  + Tự động gửi tin nhắn SMS hoặc email nhắc nhở bệnh nhân về lịch khám, uống thuốc hoặc các thông báo quan trọng khác.
  + Có khả năng tùy chỉnh nội dung và tần suất nhắc nhở.
  + Giúp nâng cao sự tuân thủ điều trị và giảm thiểu việc bệnh nhân bỏ lỡ các lịch trình quan trọng.
* **Module Báo cáo:**
  + Tổng hợp, phân tích các dữ liệu hoạt động của bệnh viện như số lượng bệnh nhân, lịch khám, số ca khám, hiệu suất làm việc của bác sĩ, doanh thu,…
  + Cung cấp các báo cáo dạng biểu đồ, bảng số liệu giúp ban quản lý dễ dàng theo dõi và ra quyết định.
  + Cho phép xuất báo cáo theo nhiều định dạng khác nhau phục vụ các mục đích quản trị và báo cáo chính phủ.
  1. Shared Services (Dịch vụ dùng chung)
* **Vai trò:**  
  Cung cấp các dịch vụ tiện ích được sử dụng chung bởi tất cả các module nghiệp vụ.
* **Chi tiết:**
  + **Xác thực (Authentication):** Quản lý đăng nhập, phân quyền người dùng, bảo vệ truy cập hệ thống.
  + **Logging:** Ghi lại các hoạt động, lỗi, truy cập của hệ thống nhằm hỗ trợ giám sát và xử lý sự cố.
  + **Cấu hình (Configuration):** Quản lý các tham số cấu hình hệ thống tập trung để dễ dàng thay đổi mà không cần deploy lại hệ thống.
* **Lợi ích:**  
  Tăng tính nhất quán, tái sử dụng và giảm thiểu sai sót trong quá trình phát triển và vận hành hệ thống.
  1. Cơ sở dữ liệu
* **Vai trò:**  
  Lưu trữ tất cả dữ liệu của hệ thống một cách tập trung, bảo mật và có khả năng sao lưu, phục hồi.
* **Chi tiết:**
  + Cơ sở dữ liệu quan hệ hoặc NoSQL tùy theo đặc thù dữ liệu và yêu cầu hiệu năng.
  + Lưu trữ thông tin người dùng, hồ sơ bệnh án, lịch sử khám chữa bệnh, lịch hẹn, các thông báo nhắc nhở, dữ liệu báo cáo,…
  + Thiết kế tối ưu cho truy vấn nhanh, bảo đảm tính toàn vẹn dữ liệu và khả năng mở rộng theo nhu cầu phát triển.

### 3.1.2. Các dịch vụ chính và luồng tương tác

Hệ thống quản lý bệnh viện thông minh được xây dựng dựa trên sự phối hợp chặt chẽ giữa các dịch vụ và thành phần kiến trúc khác nhau nhằm đảm bảo tính bảo mật, hiệu quả và mở rộng. Dưới đây là mô tả chi tiết các bước trong luồng tương tác của hệ thống khi xử lý các yêu cầu từ người dùng:

* 1. Người dùng gửi yêu cầu thông qua Web/Mobile UI
* **Người dùng bao gồm:** Bệnh nhân, bác sĩ, nhân viên y tế và quản trị viên hệ thống.
* **Giao diện tương tác:** Người dùng truy cập hệ thống qua trình duyệt web hoặc ứng dụng di động được thiết kế thân thiện, hỗ trợ đa nền tảng và các chức năng như đăng nhập, đăng ký, đặt lịch khám, xem và cập nhật hồ sơ sức khỏe, tạo báo cáo,...
* **Hành động:** Khi người dùng thực hiện một thao tác (ví dụ: đặt lịch khám mới), giao diện UI sẽ tạo một yêu cầu (request) dưới dạng API call gửi lên hệ thống backend.
  1. Yêu cầu được chuyển tới API Gateway để kiểm tra xác thực, phân quyền và định tuyến
* **Tiếp nhận yêu cầu:** API Gateway là điểm đầu vào chính của hệ thống, nhận tất cả các request từ UI.
* **Xác thực (Authentication):** API Gateway kiểm tra xem người dùng đã đăng nhập hợp lệ hay chưa thông qua việc kiểm tra token hoặc các chứng thực khác. Nếu không hợp lệ, yêu cầu sẽ bị từ chối và trả về lỗi xác thực.
* **Phân quyền (Authorization):** Sau khi xác thực, API Gateway xác định quyền truy cập của người dùng dựa trên vai trò (role). Ví dụ, bệnh nhân chỉ có quyền truy cập các chức năng liên quan đến hồ sơ cá nhân, trong khi quản trị viên có quyền truy cập rộng hơn.
* **Định tuyến (Routing):** Dựa trên URL hoặc header trong yêu cầu, API Gateway chuyển tiếp yêu cầu tới module nghiệp vụ thích hợp (ví dụ: module đặt lịch, quản lý bệnh nhân,...).
* **Bảo mật bổ sung:** API Gateway có thể thực hiện các kiểm tra bảo mật khác như giới hạn tốc độ truy cập (rate limiting), kiểm tra các chỉ số an toàn trước khi chuyển tiếp.
  1. API Gateway chuyển yêu cầu tới module nghiệp vụ phù hợp
* **Vai trò:** Đóng vai trò như bộ điều phối thông minh, API Gateway đảm bảo yêu cầu được gửi đến đúng thành phần xử lý.
* **Chức năng:**
  + Gửi yêu cầu đặt lịch đến **Module Đặt lịch**.
  + Gửi các truy vấn về thông tin bệnh nhân đến **Module Quản lý bệnh nhân** hoặc **Module Hồ sơ sức khỏe**.
  + Gửi yêu cầu gửi thông báo đến **Module Nhắc nhở**.
  + Chuyển các yêu cầu báo cáo tới **Module Báo cáo.**
* **Mục đích:** Tách biệt rõ ràng các nhiệm vụ, giúp hệ thống dễ dàng bảo trì và mở rộng.
  1. Module nghiệp vụ sử dụng API Controller nhận yêu cầu, gọi dịch vụ nghiệp vụ xử lý logic
* **API Controller:** Là lớp trung gian đầu tiên trong module nghiệp vụ nhận yêu cầu từ API Gateway. Controller kiểm tra tính hợp lệ sơ bộ của dữ liệu đầu vào và thực hiện các bước xác thực bổ sung nếu cần.
* **Service Layer:** Controller gọi các dịch vụ nghiệp vụ để thực thi các quy trình nghiệp vụ phức tạp, ví dụ:
  + Kiểm tra tính khả dụng của lịch khám trong module đặt lịch.
  + Cập nhật hồ sơ bệnh án mới trong module quản lý bệnh nhân.
  + Tính toán và tổng hợp dữ liệu cho báo cáo.
* **Xử lý logic:** Service Layer chịu trách nhiệm chính trong việc xử lý các nghiệp vụ, đảm bảo các quy tắc kinh doanh được tuân thủ và dữ liệu được chuẩn hóa.
  1. Dịch vụ nghiệp vụ gọi Repository tương tác với cơ sở dữ liệu
* **Repository:** Là lớp chịu trách nhiệm trực tiếp thao tác với cơ sở dữ liệu (Database).
* **Chức năng:** Thực hiện các thao tác tạo (Create), đọc (Read), cập nhật (Update) và xóa (Delete) dữ liệu (CRUD).
* **Lợi ích:** Việc tách biệt Repository giúp dễ dàng quản lý truy cập dữ liệu, đảm bảo tính toàn vẹn và giảm phụ thuộc giữa các thành phần.
* **Bảo mật:** Repository thực hiện các thao tác với dữ liệu tuân thủ các chính sách bảo mật, mã hóa khi cần thiết để bảo vệ dữ liệu nhạy cảm.
  1. Các Shared Services hỗ trợ xử lý các tác vụ như xác thực, ghi log trong quá trình xử lý
* **Xác thực bổ sung:** Trong quá trình xử lý, các dịch vụ xác thực có thể được gọi để kiểm tra token bảo mật, cấp quyền nâng cao, hoặc các chính sách bảo mật phức tạp hơn.
* **Logging:** Hệ thống ghi lại toàn bộ hoạt động, bao gồm các thao tác của người dùng, kết quả xử lý, lỗi phát sinh nhằm phục vụ mục đích giám sát, phân tích và khắc phục sự cố.
* **Cấu hình:** Các thông số hệ thống được quản lý tập trung, cho phép thay đổi linh hoạt mà không cần thay đổi mã nguồn hay triển khai lại. Shared Services cung cấp dữ liệu cấu hình này cho các module khi cần.
  1. Kết quả xử lý trả về API Gateway, sau đó gửi về UI để hiển thị cho người dùng
* **Trả kết quả:** Sau khi xử lý thành công hoặc gặp lỗi, kết quả được Service Layer trả về API Controller.
* **Phản hồi qua API Gateway:** API Controller chuyển kết quả lên API Gateway, nơi thực hiện các bước kiểm tra cuối cùng (nếu có) trước khi gửi lại cho giao diện người dùng.
* **Hiển thị cho người dùng:** Web hoặc Mobile UI nhận dữ liệu phản hồi và hiển thị thông báo hoặc dữ liệu tương ứng, ví dụ: xác nhận đặt lịch thành công, danh sách hồ sơ sức khỏe, thông báo lỗi hoặc cảnh báo,…
* **Trải nghiệm người dùng:** Giao diện phản hồi nhanh, rõ ràng giúp người dùng dễ dàng thực hiện các bước tiếp theo hoặc chỉnh sửa yêu cầu nếu cần.

## 3.2. Đặc tả kiến trúc hệ thống (Tương ứng phần sau Câu 2)

### 3.2.1. Đặc tả chi tiết cho các module/component quan trọng

3.2.1.1. Module Quản lý bệnh nhân

Chịu trách nhiệm lưu trữ và quản lý toàn bộ thông tin cá nhân của bệnh nhân như họ tên, ngày sinh, địa chỉ, số điện thoại cũng như lịch sử khám chữa bệnh. Ngoài ra, module này còn quản lý hồ sơ bệnh án và đơn thuốc, đảm bảo dữ liệu luôn được bảo mật nghiêm ngặt và phân quyền truy cập hợp lý theo từng vai trò người dùng. Đây là trung tâm lưu trữ thông tin quan trọng nhất của hệ thống, cung cấp dữ liệu nền tảng cho các module khác như đặt lịch, hồ sơ sức khỏe hay báo cáo hoạt động.

a. Chức năng lưu trữ và quản lý thông tin cá nhân:  
Module này chịu trách nhiệm thu thập, lưu trữ và cập nhật các dữ liệu cơ bản và quan trọng về bệnh nhân như họ tên, ngày sinh, địa chỉ, số điện thoại. Việc lưu trữ đầy đủ và chính xác các thông tin này giúp nhận dạng bệnh nhân một cách hiệu quả và là tiền đề cho việc quản lý các hoạt động y tế tiếp theo.

b. Quản lý lịch sử khám chữa bệnh:  
Ngoài dữ liệu cá nhân, module còn ghi nhận toàn bộ lịch sử khám chữa bệnh của bệnh nhân, bao gồm các lần khám, chẩn đoán, điều trị, kết quả xét nghiệm và các đơn thuốc đã được kê. Việc quản lý lịch sử này giúp bác sĩ theo dõi tiến trình điều trị, phát hiện các vấn đề sức khỏe liên quan và đưa ra các quyết định y tế chính xác hơn.

c. Quản lý hồ sơ bệnh án và đơn thuốc:  
Hồ sơ bệnh án là tài liệu y tế quan trọng ghi lại toàn bộ quá trình khám chữa bệnh, các xét nghiệm, phương pháp điều trị. Đơn thuốc được quản lý giúp kiểm soát việc cấp thuốc, tránh tình trạng dùng thuốc sai hoặc trùng lặp. Module này đảm bảo rằng tất cả các thông tin này được lưu trữ có hệ thống và dễ dàng truy cập khi cần thiết.

d. Bảo mật và phân quyền truy cập:  
Do tính chất nhạy cảm của dữ liệu y tế, module áp dụng các chính sách bảo mật nghiêm ngặt để bảo vệ thông tin bệnh nhân khỏi các truy cập trái phép. Việc phân quyền truy cập giúp giới hạn quyền xem, chỉnh sửa dữ liệu theo vai trò của người dùng (bệnh nhân, bác sĩ, quản trị viên), từ đó đảm bảo tuân thủ các quy định về bảo mật và quyền riêng tư.

e. Trung tâm dữ liệu quan trọng và cung cấp nền tảng cho các module khác:  
Module Quản lý bệnh nhân không chỉ độc lập mà còn đóng vai trò là trung tâm dữ liệu, cung cấp thông tin cần thiết cho các module nghiệp vụ khác như module Đặt lịch (để xác nhận bệnh nhân đặt lịch), module Hồ sơ sức khỏe (để liên kết và quản lý chi tiết dữ liệu y tế), và module Báo cáo (để tổng hợp và phân tích dữ liệu bệnh nhân phục vụ quản lý). Điều này giúp đảm bảo sự đồng bộ, nhất quán dữ liệu trong toàn hệ thống.

3.2.1.2. Module Đặt lịch

Cho phép bệnh nhân đặt lịch khám trực tuyến một cách thuận tiện, đồng thời hỗ trợ hủy hoặc thay đổi lịch hẹn khi cần thiết. Module cũng quản lý lịch làm việc của bác sĩ, phân bổ các cuộc hẹn sao cho hợp lý nhằm tránh trùng lặp và quá tải. Thông báo xác nhận lịch hẹn sẽ được gửi tới cả bệnh nhân và bác sĩ để đảm bảo mọi bên đều nắm rõ lịch trình. Nhờ đó, module đặt lịch góp phần tối ưu hóa quy trình khám chữa bệnh, giảm thiểu thời gian chờ đợi và nâng cao hiệu quả làm việc của đội ngũ y tế.

a. Cho phép bệnh nhân đặt lịch khám trực tuyến thuận tiện:  
Module cung cấp một giao diện đơn giản, dễ sử dụng để bệnh nhân có thể chủ động chọn thời gian khám phù hợp với lịch làm việc của mình. Việc này giúp giảm bớt sự phụ thuộc vào phương thức đặt lịch truyền thống qua điện thoại hoặc trực tiếp, tạo sự linh hoạt và tiện lợi cao cho người bệnh.

b. Hỗ trợ hủy hoặc thay đổi lịch hẹn:  
Ngoài việc đặt lịch, module còn cho phép bệnh nhân hoặc nhân viên y tế dễ dàng điều chỉnh lịch hẹn khi có sự thay đổi về thời gian hoặc nhu cầu. Tính năng này giúp tăng tính linh hoạt trong quản lý lịch, tránh tình trạng bỏ lỡ cuộc hẹn hoặc gây tắc nghẽn lịch khám.

c. Quản lý lịch làm việc của bác sĩ:  
Module không chỉ quản lý lịch hẹn của bệnh nhân mà còn quản lý lịch làm việc, ca trực của bác sĩ một cách khoa học. Việc này đảm bảo bác sĩ không bị quá tải hoặc trùng lịch, đồng thời tối ưu hóa việc phân bổ nguồn lực y tế.

d. Phân bổ cuộc hẹn hợp lý để tránh trùng lặp và quá tải:  
Module tự động kiểm tra các lịch hẹn đã có để đảm bảo các cuộc hẹn mới không trùng lặp hoặc vượt quá công suất làm việc của bác sĩ. Điều này giúp nâng cao hiệu quả hoạt động của bệnh viện và tránh gây áp lực không cần thiết lên nhân viên y tế.

e. Gửi thông báo xác nhận lịch hẹn:  
Sau khi bệnh nhân đặt lịch thành công, module sẽ gửi thông báo xác nhận qua tin nhắn, email hoặc ứng dụng cho cả bệnh nhân và bác sĩ. Việc này giúp các bên đều nắm rõ thời gian, địa điểm khám, tránh nhầm lẫn và quên lịch.

f. Tối ưu hóa quy trình khám chữa bệnh và nâng cao hiệu quả làm việc:  
Nhờ các chức năng trên, module Đặt lịch góp phần giảm thời gian chờ đợi tại bệnh viện, phân luồng bệnh nhân hiệu quả, giúp nhân viên y tế và bác sĩ tổ chức công việc khoa học hơn, từ đó nâng cao chất lượng dịch vụ khám chữa bệnh và trải nghiệm người bệnh.

3.2.1.3. Module Hồ sơ sức khỏe

Được thiết kế để lưu trữ chi tiết các dữ liệu sức khỏe điện tử quan trọng như kết quả xét nghiệm, chẩn đoán, diễn biến điều trị và các thông tin y khoa liên quan. Người dùng có thể tra cứu và cập nhật hồ sơ sức khỏe một cách nhanh chóng và chính xác, đồng thời hệ thống đảm bảo tuân thủ nghiêm ngặt các quy định về bảo mật dữ liệu y tế. Module này không chỉ hỗ trợ bác sĩ trong việc theo dõi và điều trị bệnh nhân mà còn phục vụ cho việc chăm sóc sức khỏe dài hạn một cách hiệu quả.

a. Lưu trữ chi tiết dữ liệu sức khỏe điện tử:  
Module được thiết kế để tập trung lưu trữ toàn bộ các dữ liệu quan trọng liên quan đến sức khỏe của bệnh nhân, bao gồm kết quả xét nghiệm, các chẩn đoán y khoa, diễn biến quá trình điều trị, cũng như các thông tin y khoa liên quan khác như hình ảnh y học hoặc ghi chú của bác sĩ. Việc lưu trữ chi tiết giúp tạo ra một hồ sơ sức khỏe toàn diện, phục vụ hiệu quả cho việc chẩn đoán và điều trị.

b. Tra cứu và cập nhật nhanh chóng, chính xác:  
Người dùng hệ thống, bao gồm bác sĩ, nhân viên y tế và bệnh nhân, có thể truy cập và cập nhật hồ sơ sức khỏe một cách dễ dàng và nhanh chóng thông qua giao diện được thiết kế thân thiện và bảo mật. Tính năng này giúp giảm thiểu sai sót, đảm bảo thông tin luôn được cập nhật kịp thời phục vụ cho các quyết định y tế chính xác.

c. Tuân thủ nghiêm ngặt các quy định về bảo mật dữ liệu y tế:  
Dữ liệu sức khỏe là loại thông tin rất nhạy cảm, do đó module áp dụng các biện pháp bảo mật cao cấp để bảo vệ quyền riêng tư của bệnh nhân, tuân thủ các tiêu chuẩn và quy định pháp lý về bảo mật y tế như HIPAA (ở Mỹ) hoặc các quy định tương tự ở các quốc gia khác. Các biện pháp này bao gồm mã hóa dữ liệu, phân quyền truy cập nghiêm ngặt và kiểm soát truy cập dựa trên vai trò.

d. Hỗ trợ theo dõi và chăm sóc sức khỏe dài hạn:  
Ngoài việc hỗ trợ bác sĩ trong việc theo dõi, chẩn đoán và điều trị bệnh nhân trong từng lần khám, module còn phục vụ cho việc quản lý sức khỏe lâu dài của bệnh nhân. Việc có một hồ sơ sức khỏe điện tử toàn diện giúp phát hiện sớm các vấn đề sức khỏe tiềm ẩn, theo dõi tiến triển bệnh và lập kế hoạch chăm sóc phù hợp theo thời gian, nâng cao chất lượng chăm sóc sức khỏe tổng thể.

3.2.1.4. Module Nhắcnhở

Có chức năng tự động gửi các tin nhắn hoặc email nhắc nhở bệnh nhân về lịch khám, uống thuốc hay các thông báo quan trọng khác. Người quản trị có thể tùy chỉnh nội dung và tần suất nhắc nhở sao cho phù hợp với từng đối tượng bệnh nhân. Nhờ có module này, bệnh nhân được hỗ trợ để tuân thủ tốt hơn lịch trình điều trị, từ đó giảm thiểu nguy cơ bỏ lỡ các cuộc hẹn hoặc liều thuốc quan trọng, góp phần nâng cao hiệu quả chăm sóc sức khỏe.

a. Chức năng tự động gửi nhắc nhở:  
Module này có khả năng tự động gửi các thông báo nhắc nhở tới bệnh nhân qua nhiều kênh khác nhau như tin nhắn SMS hoặc email. Những nhắc nhở này bao gồm các nội dung quan trọng như lịch khám sắp tới, thời gian uống thuốc đúng liều lượng hoặc các thông báo y tế khác. Việc tự động hóa quá trình này giúp đảm bảo bệnh nhân luôn được thông báo kịp thời mà không cần sự can thiệp thủ công từ nhân viên y tế.

b. Tùy chỉnh nội dung và tần suất nhắc nhở:  
Người quản trị hệ thống có thể thiết lập và điều chỉnh nội dung cũng như tần suất gửi nhắc nhở dựa trên đặc điểm và nhu cầu riêng của từng bệnh nhân hoặc nhóm bệnh nhân. Điều này giúp cá nhân hóa trải nghiệm người dùng, tránh gây phiền nhiễu do gửi quá nhiều thông báo không cần thiết, đồng thời tăng khả năng bệnh nhân chú ý và thực hiện đúng chỉ dẫn.

c. Hỗ trợ bệnh nhân tuân thủ lịch trình điều trị:  
Nhờ sự hỗ trợ của module nhắc nhở, bệnh nhân được nhắc nhở đều đặn về các mốc quan trọng trong quá trình điều trị, từ đó giảm thiểu nguy cơ quên hoặc bỏ sót các cuộc hẹn hoặc liều thuốc cần thiết. Điều này đặc biệt quan trọng với các bệnh mãn tính hoặc các liệu trình điều trị phức tạp, giúp nâng cao hiệu quả điều trị và hạn chế các biến chứng sức khỏe.

d. Góp phần nâng cao hiệu quả chăm sóc sức khỏe:  
Việc đảm bảo bệnh nhân tuân thủ đúng lịch trình khám và uống thuốc giúp tăng hiệu quả của các biện pháp điều trị, giảm thiểu tái khám do sai sót hoặc bỏ lỡ điều trị. Module nhắc nhở đóng vai trò hỗ trợ quan trọng trong việc cải thiện chất lượng dịch vụ y tế và sự hài lòng của bệnh nhân.

3.2.1.5. Module Báo cáo

Tập trung vào việc tổng hợp và phân tích dữ liệu hoạt động của bệnh viện, bao gồm số lượng bệnh nhân, số ca khám, hiệu suất làm việc của bác sĩ và doanh thu. Các dữ liệu này được trình bày dưới dạng bảng và biểu đồ trực quan, giúp ban quản lý dễ dàng theo dõi tình hình hoạt động. Ngoài ra, module hỗ trợ xuất báo cáo dưới nhiều định dạng khác nhau như PDF hoặc Excel, phục vụ cho việc báo cáo nội bộ cũng như gửi các cơ quan quản lý nhà nước. Qua đó, module báo cáo đóng vai trò quan trọng trong việc hỗ trợ quá trình ra quyết định dựa trên dữ liệu thực tế và chính xác.

a. Tổng hợp và phân tích dữ liệu hoạt động của bệnh viện:  
Module Báo cáo chịu trách nhiệm thu thập và xử lý các dữ liệu từ nhiều nguồn trong hệ thống như số lượng bệnh nhân khám, số ca khám trong ngày hoặc tháng, hiệu suất làm việc của từng bác sĩ và các chỉ số tài chính như doanh thu. Việc tổng hợp dữ liệu này giúp tạo ra cái nhìn tổng quan và sâu sắc về hoạt động vận hành của bệnh viện.

b. Trình bày dữ liệu dưới dạng bảng và biểu đồ trực quan:  
Các thông tin được hiển thị thông qua các bảng số liệu và biểu đồ sinh động, giúp ban quản lý dễ dàng quan sát, so sánh và nhận diện các xu hướng hoặc điểm cần cải thiện trong hoạt động của bệnh viện. Giao diện trực quan giúp tăng hiệu quả tiếp nhận thông tin và rút ngắn thời gian ra quyết định.

c. Hỗ trợ xuất báo cáo đa định dạng:  
Module cho phép xuất các báo cáo dưới nhiều định dạng phổ biến như PDF, Excel, giúp thuận tiện trong việc lưu trữ, chia sẻ và trình bày trong các cuộc họp, báo cáo nội bộ hoặc báo cáo cho các cơ quan quản lý nhà nước. Tính năng này tăng tính linh hoạt và đáp ứng đa dạng nhu cầu sử dụng báo cáo.

d. Vai trò hỗ trợ ra quyết định:  
Nhờ có các báo cáo được tổng hợp và phân tích chính xác, ban quản lý bệnh viện có thể dựa vào đó để đưa ra các quyết định chiến lược, điều chỉnh kế hoạch hoạt động, cải thiện chất lượng dịch vụ và quản lý nguồn lực hiệu quả hơn. Module Báo cáo do đó đóng vai trò then chốt trong việc nâng cao năng lực quản trị và phát triển bền vững của bệnh viện.

3.2.1.6. Shared Services (dịch vụ dùng chung)

Cung cấp các chức năng hỗ trợ như xác thực người dùng, ghi lại log hoạt động hệ thống, quản lý cấu hình tập trung cũng như các chính sách bảo mật và kiểm soát truy cập. Việc tập trung các dịch vụ này giúp đảm bảo tính nhất quán, tăng khả năng tái sử dụng và giảm thiểu sai sót trong quá trình phát triển và vận hành hệ thống.

a. Chức năng xác thực người dùng:  
Shared Services cung cấp cơ chế xác thực tập trung, giúp kiểm tra danh tính người dùng một cách chính xác và bảo mật. Việc xác thực tập trung này cho phép quản lý đăng nhập, phân quyền truy cập một cách thống nhất, tránh các điểm yếu bảo mật khi mỗi module tự quản lý riêng biệt.

b. Ghi lại log hoạt động hệ thống:  
Dịch vụ ghi log giúp lưu trữ toàn bộ nhật ký hoạt động của hệ thống, bao gồm các thao tác của người dùng, các sự kiện xảy ra, lỗi phát sinh,... Những dữ liệu này rất quan trọng cho việc giám sát hệ thống, phát hiện và xử lý sự cố kịp thời, cũng như phục vụ cho việc phân tích và tối ưu hệ thống về lâu dài.

c. Quản lý cấu hình tập trung:  
Shared Services còn đảm nhận việc quản lý các tham số cấu hình chung cho hệ thống như thông số kết nối, các thiết lập bảo mật, tham số vận hành,... Quản lý cấu hình tập trung giúp đơn giản hóa việc thay đổi thiết lập, tránh phải chỉnh sửa trực tiếp từng module, từ đó giảm rủi ro sai sót khi triển khai.

d. Các chính sách bảo mật và kiểm soát truy cập:  
Dịch vụ dùng chung này còn áp dụng các chính sách bảo mật chặt chẽ như mã hóa dữ liệu, kiểm soát quyền truy cập dựa trên vai trò, giám sát hành vi truy cập bất thường,... nhằm bảo vệ toàn bộ hệ thống khỏi các nguy cơ tấn công và truy cập trái phép.

e. Lợi ích tổng thể:  
Việc tập trung các dịch vụ này thành một bộ phận dùng chung giúp tăng tính nhất quán trong toàn bộ hệ thống, nâng cao khả năng tái sử dụng các chức năng, đồng thời giảm thiểu các lỗi phát sinh do phát triển phân tán hoặc thiếu đồng bộ. Điều này góp phần tối ưu hóa quá trình phát triển, vận hành và bảo trì hệ thống quản lý bệnh viện thông minh.

3.2.1.7. Cơ sở dữ liệu

Được thiết kế với cấu trúc bảng phù hợp cho từng module, đảm bảo tính toàn vẹn dữ liệu và hiệu suất truy xuất cao. Các biện pháp bảo mật như mã hóa dữ liệu nhạy cảm, phân quyền truy cập chi tiết, sao lưu định kỳ và kế hoạch phục hồi dữ liệu được triển khai nhằm bảo vệ thông tin bệnh nhân và dữ liệu y tế quan trọng. Ngoài ra, cơ sở dữ liệu còn được tối ưu hóa về mặt truy vấn và mở rộng quy mô để đáp ứng nhu cầu phát triển trong tương lai, đồng thời đảm bảo tuân thủ các quy định bảo mật và quyền riêng tư hiện hành.

a. Thiết kế cấu trúc bảng phù hợp:  
Cơ sở dữ liệu được xây dựng với các bảng dữ liệu riêng biệt, tương ứng với từng module chức năng như quản lý bệnh nhân, đặt lịch, hồ sơ sức khỏe, nhắc nhở và báo cáo. Việc phân chia này giúp đảm bảo tổ chức dữ liệu rõ ràng, dễ dàng quản lý và duy trì. Đồng thời, các bảng có mối quan hệ hợp lý với nhau thông qua khóa chính - khóa ngoại, đảm bảo tính toàn vẹn dữ liệu giữa các phần khác nhau của hệ thống.

b. Tính toàn vẹn dữ liệu và hiệu suất truy xuất:  
Thiết kế tối ưu giúp đảm bảo rằng dữ liệu luôn chính xác, nhất quán và được cập nhật kịp thời. Ngoài ra, cơ sở dữ liệu còn được tối ưu hóa để hỗ trợ truy vấn nhanh, giảm thiểu độ trễ khi người dùng hoặc hệ thống thực hiện các thao tác đọc hoặc ghi dữ liệu, góp phần nâng cao trải nghiệm và hiệu quả vận hành hệ thống.

c. Biện pháp bảo mật:  
Dữ liệu y tế là thông tin rất nhạy cảm, do đó hệ thống áp dụng các biện pháp bảo mật mạnh mẽ. Điều này bao gồm mã hóa dữ liệu quan trọng khi lưu trữ và truyền tải, thiết lập phân quyền truy cập chi tiết dựa trên vai trò người dùng để ngăn chặn truy cập trái phép, cũng như các chính sách bảo vệ dữ liệu tuân thủ các quy định luật pháp và tiêu chuẩn ngành y tế.

d. Sao lưu định kỳ và phục hồi dữ liệu:  
Để đảm bảo tính liên tục và an toàn của hệ thống, dữ liệu được sao lưu thường xuyên theo lịch trình định kỳ. Đồng thời, có kế hoạch và quy trình phục hồi dữ liệu rõ ràng trong trường hợp sự cố mất mát hoặc hỏng hóc, giúp giảm thiểu rủi ro gián đoạn dịch vụ và bảo vệ dữ liệu bệnh nhân.

e. Khả năng mở rộng:  
Cơ sở dữ liệu được thiết kế linh hoạt, dễ dàng mở rộng quy mô để đáp ứng sự tăng trưởng của lượng dữ liệu và số lượng người dùng trong tương lai. Việc này đảm bảo hệ thống luôn duy trì được hiệu suất cao và ổn định ngay cả khi quy mô sử dụng tăng lên đáng kể.

g. Tuân thủ các quy định bảo mật và quyền riêng tư:  
Cuối cùng, cơ sở dữ liệu vận hành trong khuôn khổ các tiêu chuẩn và quy định hiện hành về bảo mật thông tin y tế và quyền riêng tư cá nhân, giúp bảo vệ quyền lợi người bệnh cũng như uy tín của bệnh viện.

### 3.2.2. Thiết kế giao diện (API) giữa các thành phần (nếu có)

**a. Định dạng dữ liệu - JSON:**

Hệ thống sử dụng JSON (JavaScript Object Notation) làm định dạng chuẩn để trao đổi dữ liệu giữa các module qua API. JSON là định dạng nhẹ, dễ đọc và ghi, đồng thời có tính tương thích cao với hầu hết các ngôn ngữ lập trình và nền tảng phát triển. Việc sử dụng JSON giúp các thành phần hệ thống dễ dàng truyền nhận thông tin, đồng thời hỗ trợ mở rộng linh hoạt khi cần bổ sung hoặc thay đổi dữ liệu mà không gây ra sự cố tương thích.

**b. Giao thức truyền thông - HTTP/HTTPS:**

Các API giao tiếp qua giao thức HTTP hoặc HTTPS, trong đó HTTPS được ưu tiên sử dụng để đảm bảo tính bảo mật và toàn vẹn dữ liệu khi truyền qua mạng Internet. Giao thức HTTP/HTTPS là tiêu chuẩn phổ biến, hỗ trợ hiệu quả các phương thức RESTful như GET, POST, PUT, DELETE,... giúp hệ thống vận hành ổn định và tương thích với đa dạng môi trường triển khai.

**c. Xác thực và phân quyền - JWT/OAuth:**

Mỗi API yêu cầu có cơ chế xác thực và phân quyền chặt chẽ, sử dụng token dạng JWT (JSON Web Token) hoặc cơ chế OAuth. Các token này giúp xác minh danh tính người dùng hoặc hệ thống gọi API, đảm bảo chỉ các đối tượng hợp lệ mới được phép truy cập các dịch vụ. Việc áp dụng các chuẩn này không chỉ nâng cao bảo mật mà còn cho phép triển khai cơ chế phân quyền chi tiết, kiểm soát quyền truy cập dựa trên vai trò và phạm vi hoạt động.

**d. Các endpoint chính và phương thức HTTP:**

Hệ thống được thiết kế với các endpoint RESTful rõ ràng, tương ứng với các chức năng chính như:

* /patients: Quản lý thông tin bệnh nhân với các phương thức GET (lấy dữ liệu), POST (tạo mới), PUT (cập nhật) và DELETE (xóa).
* /appointments: Quản lý lịch hẹn tương tự với các phương thức tương ứng.
* /healthrecords: Quản lý hồ sơ sức khỏe, cung cấp thêm khả năng cập nhật với PUT và tạo mới với POST.
* /notifications: Gửi các thông báo hoặc nhắc nhở, thường sử dụng phương thức POST để gửi dữ liệu.
* /reports: Lấy các báo cáo hoạt động, chủ yếu sử dụng GET để truy vấn dữ liệu.

Việc thiết kế theo chuẩn REST giúp API dễ hiểu, dễ sử dụng, đồng thời tuân thủ các quy chuẩn công nghiệp giúp tăng tính nhất quán và khả năng tích hợp.

**e. Cơ chế xử lý lỗi:**

Mỗi API được thiết kế trả về các mã trạng thái HTTP phù hợp nhằm phản ánh chính xác kết quả xử lý yêu cầu, ví dụ:

* 200 OK: Yêu cầu được xử lý thành công.
* 400 Bad Request: Dữ liệu đầu vào không hợp lệ hoặc thiếu thông tin bắt buộc.
* 401 Unauthorized: Người dùng không có quyền truy cập do không xác thực hoặc token hết hạn.
* 500 Internal Server Error: Lỗi máy chủ hoặc lỗi hệ thống nội bộ.

Kèm theo mã trạng thái, API còn gửi thông điệp lỗi chi tiết giúp client hiểu nguyên nhân và xử lý lỗi hiệu quả, góp phần nâng cao trải nghiệm người dùng và hỗ trợ debug khi phát triển.

**f. Tính mở rộng:**

Thiết kế API theo chuẩn REST giúp hệ thống dễ dàng mở rộng thêm các chức năng mới mà không làm ảnh hưởng đến các API đã triển khai. Điều này rất quan trọng trong môi trường phát triển phần mềm hiện đại, nơi các yêu cầu thay đổi và bổ sung tính năng thường xuyên. Các API được phân tách rõ ràng, module hóa cao giúp tăng tính linh hoạt trong phát triển, bảo trì và tích hợp với các hệ thống bên ngoài trong tương lai.

# CHƯƠNG 4. Xử lý Tình huống và Bảo mật Dữ liệu

## 4.1. Xử lý tình huống và tối ưu hiệu năng hệ thống (Tương ứng Câu 3)

### 4.1.1. Phân tích nguyên nhân hệ thống chậm khi tải cao

### 4.1.2. Đề xuất giải pháp cải thiện hiệu năng (ít nhất 2 giải pháp)

## 4.2. Đảm bảo tính bảo mật cho dữ liệu nhạy cảm (Tương ứng Câu 4)

### 4.2.1. Xác định các nguy cơ bảo mật

### 4.2.2. Đề xuất các biện pháp bảo mật (ít nhất 2 biện pháp)

### 4.2.3. Mô tả cách triển khai các biện pháp

# CHƯƠNG 5: THỰC NGHIỆM VÀ ĐÁNH GIÁ

## 5.1. Kế hoạch thử nghiệm mô hình kiến trúc (Tương ứng Câu 5)

### 5.1.1. Mục tiêu và các kịch bản thử nghiệm chính

## 5.2. Đánh giá mô hình kiến trúc

### 5.2.1. Ưu điểm, nhược điểm của mô hình đã đề xuất

## 5.3. Đề xuất và khuyến nghị cho việc triển khai thực tế (Tương ứng Câu 5)

KẾT LUẬN

TÀI LIỆU THAM KHẢO

**Tài liệu tiếng Việt:**

**Tài liệu tiếng Anh:**

**Các trang web tham khảo:**