

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  
**KHOA HỆ THỐNG THÔNG TIN**

**NGUYỄN XUÂN SANG**

**KHÓA LUẬN TỐT NGHIỆP**  
**ỨNG DỤNG BLOCKCHAIN VÀO HỆ THỐNG QUẢN**  
**LÝ BỆNH ÁN ĐIỆN TỬ**

**Blockchain application to electronic medical management system**

**KỸ SƯ NGÀNH HỆ THỐNG THÔNG TIN**

**TP. HỒ CHÍ MINH, 2019**

**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**  
**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**  
**KHOA HỆ THỐNG THÔNG TIN**

**NGUYỄN XUÂN SANG – 15520720**

**KHÓA LUẬN TỐT NGHIỆP**  
**ỨNG DỤNG BLOCKCHAIN VÀO HỆ THỐNG QUẢN**  
**LÝ BỆNH ÁN ĐIỆN TỬ**

**Blockchain application to electronic medical management system**

**KỸ SƯ NGÀNH HỆ THỐNG THÔNG TIN**

**GIẢNG VIÊN HƯỚNG DẪN**

**ThS. THÁI BẢO TRÂN**

**TP. HỒ CHÍ MINH, 2019**

## **THÔNG TIN HỘI ĐỒNG CHẤM KHÓA LUẬN TỐT NGHIỆP**

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo Quyết định số .....  
ngày ..... của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC

CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc Lập - Tự Do - Hạnh Phúc

TP. HCM, ngày.....tháng.....năm.....

## NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP CỦA CÁN BỘ HƯỚNG DẪN

**Tên khóa luận:**

**ỨNG DỤNG BLOCKCHAIN**

**VÀO HỆ THỐNG QUẢN LÝ BỆNH ÁN ĐIỆN TỬ**

**Nhóm SV thực hiện:**

Nguyễn Xuân Sang

**Cán bộ hướng dẫn:**

15520720 ThS. Thái Bảo Trân

**Đánh giá Khóa luận**

1. Về cuốn báo cáo:

Số trang \_\_\_\_\_ Số chương \_\_\_\_\_

Số bảng số liệu \_\_\_\_\_ Số hình vẽ \_\_\_\_\_

Số tài liệu tham khảo \_\_\_\_\_ Sản phẩm \_\_\_\_\_

Một số nhận xét về hình thức cuốn báo cáo:

2. Về nội dung nghiên cứu:

3. Về chương trình ứng dụng:

4. Về thái độ làm việc của sinh viên:

**Đánh giá chung:**

**Điểm từng sinh viên:**

Nguyễn Xuân Sang:...../10

**Người nhận xét**  
(Ký tên và ghi rõ họ tên)

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

TRƯỜNG ĐẠI HỌC  
CÔNG NGHỆ THÔNG TIN

Độc Lập - Tự Do - Hạnh Phúc

TP. HCM, ngày.....tháng.....năm.....

## NHẬN XÉT KHÓA LUẬN TỐT NGHIỆP CỦA CÁN BỘ PHẢN BIỆN

**Tên khóa luận:**

### ỨNG DỤNG BLOCKCHAIN VÀO HỆ THỐNG QUẢN LÝ BỆNH ÁN ĐIỆN TỬ

**Nhóm SV thực hiện:**

**Cán bộ phản biện:**

Nguyễn Xuân Sang

15520720

**Đánh giá Khóa luận**

1. Về cuốn báo cáo:

Số trang \_\_\_\_\_ Số chương \_\_\_\_\_

Số bảng số liệu \_\_\_\_\_ Số hình vẽ \_\_\_\_\_

Số tài liệu tham khảo \_\_\_\_\_ Sản phẩm \_\_\_\_\_

Một số nhận xét về hình thức cuốn báo cáo:

2. Về nội dung nghiên cứu:

3. Về chương trình ứng dụng:

4. Về thái độ làm việc của sinh viên:

**Đánh giá chung:**

**Điểm từng sinh viên:**

Nguyễn Xuân Sang:...../10

**Người nhận xét**  
(Ký tên và ghi rõ họ tên)

## LỜI CẢM ƠN

Khóa luận tốt nghiệp là kết quả làm việc, nghiên cứu và học tập của tác giả trong suốt thời gian ngồi trên ghế giảng đường của Trường Đại học Công nghệ Thông tin. Tác giả không thể hoàn thành tốt khóa luận tốt nghiệp nếu như không có sự quan tâm và giúp đỡ tận tình của quý thầy cô, bạn bè và gia đình.

Đầu tiên, tác giả xin gửi lời cảm ơn chân thành đến tập thể quý Giảng viên và Cán bộ Trường Đại học Công nghệ Thông tin – Đại học Quốc gia TP. Hồ Chí Minh và quý giảng viên Khoa Hệ Thống Thông Tin đã tạo môi trường và điều kiện thuận lợi, giúp cho tác giả có những kiến thức nòng cốt làm nền tảng để thực hiện đề tài.

Đặc biệt, tác giả xin gửi lời cảm ơn sâu sắc và lòng biết ơn nhất đến giảng viên hướng dẫn – ThS. Thái Bảo Trân. Cô đã trực tiếp dõi theo, hướng dẫn tận tình, sửa chữa và đóng góp nhiều ý kiến quý báu giúp tác giả hoàn thành tốt khóa luận tốt nghiệp của em.

Bên cạnh đó, tác giả xin được cảm ơn gia đình, người thân và bạn bè đã động viên, khích lệ và hỗ trợ tác giả về mặt tinh thần trong suốt thời gian thực hiện khóa luận tốt nghiệp.

Trong thời gian một học kỳ thực hiện đề tài, tác giả đã vận dụng những kiến thức nền tảng đã tích lũy được, đồng thời kết hợp với những kiến thức chuyên sâu nhờ học hỏi và nghiên cứu trong thời gian qua nhằm hoàn thành một báo cáo đồ án một cách tốt nhất. Tuy nhiên, trong quá trình thực hiện tác giả không tránh khỏi những thiếu sót. Chính vì vậy, tác giả rất mong nhận được những sự góp ý từ phía các thầy cô nhằm hoàn thiện những kiến thức mà tác giả đã học tập và là hành trang để tác giả thực hiện tiếp các đề tài khác trong tương lai.



Trong suốt quá trình hoàn thiện báo cáo, nếu em còn nhiều thiếu sót và nhiều chỗ mơ hồ, khó hiểu, em mong muốn nhận được sự góp ý chân thành từ quý thầy/cô để em có thể hoàn thiện báo cáo của mình một cách trọn vẹn nhất.

Xin chân thành cảm ơn quý thầy cô!

Tác giả

Nguyễn Xuân Sang

**TRƯỜNG ĐẠI HỌC**  
**CÔNG NGHỆ THÔNG TIN**

**Độc Lập - Tự Do - Hạnh Phúc**

**ĐỀ CƯƠNG CHI TIẾT**

**TÊN ĐỀ TÀI:**

**Ứng dụng Blockchain vào hệ thống quản lý bệnh án điện tử**  
**Blockchain application to electronic medical management system**

**Cán bộ hướng dẫn:** ThS. Thái Bảo Trân

**Thời gian thực hiện:** Từ 1/9/2019 đến 12/12/2019

**Sinh viên thực hiện:**

Nguyễn Xuân Sang – 15520720

**NỘI DUNG ĐỀ TÀI**

**1. Lý do chọn đề tài**

Ngày nay, với sự phát triển vượt bậc của khoa học kỹ thuật ở thời đại cách mạng công nghiệp 4.0. Công nghệ thông tin là một lĩnh vực có nhiều ứng dụng thiết thực nhất trong mọi lĩnh vực của cuộc sống, đặc biệt nó là công cụ hỗ trợ đắc lực không thể thiếu trong công tác quản lý. Dễ dàng thấy rằng cơ sở dữ liệu là một trong ứng dụng quan trọng của công tác tin học hóa trong quản lý kinh doanh của các tổ chức doanh nghiệp.

Nhờ vào công tác tin học hoá mà công tác quản lý, điều hành của các doanh nghiệp tỏ ra có hiệu quả, nhanh chóng, chính xác, lưu trữ gọn, bảo mật cao và dễ dàng.

Đối với lĩnh vực ICT, Blockchain đã được áp dụng trong thanh toán tiền điện tử (đồng tiền mã hóa) giúp đem lại tính minh bạch, bảo mật và nhanh nhẹn trong giao dịch số. Ngoài ra, việc ứng dụng công nghệ Blockchain còn mang lại lợi ích thiết thực trong việc quản lý Y Khoa với tính năng lưu trữ hồ sơ bệnh án một cách bảo mật, an toàn và chia sẻ.

Đề tài giúp ứng dụng công nghệ Blockchain vào trong việc quản lý điện tử. Giảm thiểu công việc cho người quản lý. Ngoài ra còn bảo mật và an toàn dữ liệu hơn.

## **2. Mục tiêu đề tài**

- Nghiên cứu về công nghệ Blockchain. Tìm hiểu các tính năng của Blockchain. Từ đó rút ra các ưu và nhược điểm của công nghệ này.
- Cài đặt hệ thống Blockchain.
- Xây dựng một ứng dụng chạy để kiểm chứng tính năng của Blockchain.
- Mục tiêu đề tài ứng dụng Blockchain để quản lý các vấn đề chính như sau:
  - Quản lý bệnh nhân
  - Quản lý quy trình khám chữa bệnh
- Ngoài những yêu cầu xử lý mà bài toán đã đặt ra, chương trình này sẽ hỗ trợ đắc lực cho công tác khám và điều trị bệnh cho bệnh nhân tại các bệnh viện và phòng khám, giảm khối lượng công việc tăng năng suất cho người sử dụng.

## **3. Phạm vi đề tài**

Đề tài tập trung vào các nội dung chính sau:

- Nghiên cứu công nghệ Blockchain

- Nghiên cứu platform Blockchain trong chứng thực và các quy trình ứng dụng Blockchain trong quản lý bệnh án điện tử.
- Nghiên cứu các tính năng của Blockchain.
- Đưa ra các ưu điểm và nhược điểm của các loại Blockchain.
- Quản lý thông tin bệnh nhân, bệnh án điện tử.

#### **4. Đối tượng**

- Công nghệ Blockchain và hệ thống quản lý bệnh án điện tử.
- Hyperledger Fabric, Hyperledger Composer.
- Quy trình khám chữa bệnh và lưu hồ sơ y tế.

#### **5. Phương pháp thực hiện**

- Nội dung 1: Nghiên cứu công nghệ Blockchain
- Nội dung 2: So sánh các hệ thống Blockchain
- Nội dung 3: Xây dựng hệ thống quản lý bệnh án điện tử kiểm chứng.
  - Đăng ký
  - Đăng nhập.
  - Tạo và lưu thông tin bệnh án lên Blockchain.
  - Lấy thông tin bệnh án.

#### **6. Kết quả mong đợi của đề tài**

- Nghiên cứu, áp dụng công nghệ Blockchain vào việc quản lý hồ sơ y tế.
- Hoàn thiện ứng dụng hệ thống quản lý bệnh án điện tử.

### **KẾ HOẠCH THỰC HIỆN**

<b>Công việc cần thực hiện</b>	<b>Thời gian</b>
- Gặp giáo viên hướng dẫn trao đổi về ý tưởng thực hiện luận văn.	1/9/2019 – 8/9/2019

<ul style="list-style-type: none"> <li>- Tìm các nghiên cứu, bài báo khoa học liên quan.</li> <li>- Tìm hiểu các chức năng cần triển khai.</li> </ul>	
<ul style="list-style-type: none"> <li>- Gặp giáo viên hướng dẫn đăng ký đề tài</li> <li>- Trao đổi các nội dung khả thi được thực hiện trong đề tài.</li> </ul>	9/9/2019 – 14/9/2019
<ul style="list-style-type: none"> <li>- Tập trung nghiên cứu công nghệ Blockchain</li> <li>- Nghiên cứu các tính năng của Blockchain và so sánh các hệ thống Blockchain.</li> </ul>	15/9/2019 – 22/9/2019
<ul style="list-style-type: none"> <li>- Tìm hiểu các vấn đề gây khó khăn trong việc quản lý bệnh án truyền thống.</li> <li>- Tìm hiểu quy trình khám chữa bệnh được áp dụng thực tế.</li> </ul>	23/9/2019 – 23/10/2019
<ul style="list-style-type: none"> <li>- Nghiên cứu lập trình trên Hyperledger.</li> <li>- Nghiên cứu triển khai ứng dụng trên Hyperledger.</li> <li>- Xây dựng hệ thống để kiểm chứng các tính năng của hệ thống Blockchain.</li> </ul>	24/10/2019 – 1/11/2019
<ul style="list-style-type: none"> <li>- Hoàn thiện ứng dụng, đề xuất bổ sung nếu có.</li> </ul>	2/11/2019 – 19/12/2019
<ul style="list-style-type: none"> <li>- Viết báo cáo hoàn chỉnh cho đề tài.</li> </ul>	20/12/2019 – 28/12/2019

<p><b>Xác nhận của CBHD</b></p> <p>(Ký tên và ghi rõ họ tên)</p> <p>Thái Bảo Trân</p>	<p><b>TP. HCM, 18/8/2019</b></p> <p><b>Sinh viên</b></p> <p>(Ký tên và ghi rõ họ tên)</p>
---	---

## MỤC LỤC

Chương 1. MỞ ĐẦU .....	3
1.1. Tổng quan.....	3
1.2. Bài toán .....	4
1.3. Mục tiêu đề tài .....	5
1.4. Đối tượng và phạm vi nghiên cứu .....	5
1.5. Thách thức của bài toán .....	5
1.6. Kết chương .....	7
Chương 2. CƠ SỞ LÝ THUYẾT .....	8
2.1. Các khái niệm chung.....	8
2.1.1. Tìm hiểu về Blockchain.....	8
2.1.2. Tìm hiểu về Hyperledger Blockchain và Hyperledger Fabric.....	35
2.2. Lý do sử dụng Hyperledger.....	51
2.3. Kết chương .....	53
Chương 3. MÔ HÌNH BÀI TOÁN .....	55
3.1. Giới thiệu mô hình .....	55
3.2. Các quy trình trong hệ thống.....	56
3.2.1. Quy trình hoạt động của một hệ thống cơ bản.....	56
3.2.2. Cập nhật thông tin Profile.....	57
3.2.3. Các quy trình cấp quyền .....	58
3.2.4. Khởi tạo và cập nhật thông tin bệnh án Health Record.....	60
3.3. Kết chương .....	61

Chương 4. CÀI ĐẶT VÀ THỬ NGHIỆM HỆ THỐNG .....	62
4.1. Môi trường và công cụ xây dựng.....	62
4.2. Sơ đồ USE-CASE.....	64
4.2.1. USE-CASE tổng quát .....	64
4.2.2. USE-CASE đăng ký .....	66
4.2.3. USE-CASE đăng nhập.....	68
4.2.4. USE-CASE quản lý thông tin cá nhân .....	69
4.2.5. USE-CASE quản lý thông tin bệnh án .....	70
4.2.6. USE-CASE cập nhật thông tin bệnh án.....	71
4.2.7. USE-CASE quản lý yêu cầu (Request) .....	72
4.2.8. USE-CASE xem danh sách bác sĩ.....	73
4.2.9. USE-CASE xem danh sách bệnh nhân.....	74
4.2.10. USE-CASE đăng xuất .....	75
4.3. Xây dựng ứng dụng.....	75
4.3.1. Cài đặt môi trường.....	75
4.3.2. Định nghĩa các Participant.....	78
4.3.3. Định nghĩa các Enum .....	79
4.3.4. Định nghĩa các Asset của Doctor (Bác sĩ).....	81
4.3.5. Định nghĩa các Asset của Patient (Bệnh nhân).....	83
4.3.6. Định nghĩa các Access Rule Control.....	86
4.3.7. Định nghĩa các câu Query.....	93
4.3.8. Định nghĩa các Transaction và Chaincode .....	95



4.3.9.	Xây dựng tầng Web Application.....	97
4.4.	Kết chương .....	117
Chương 5.	KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....	118
5.1.	Đánh giá sản phẩm.....	118
5.1.1.	Đã làm được .....	118
5.1.2.	Chưa làm được .....	118
5.2.	Hướng phát triển.....	119
TÀI LIỆU THAM KHẢO .....		120

## DANH MỤC HÌNH ẢNH

Hình 2.1. Hình ảnh số người tham gia vào Blockchain .....	10
Hình 2.2. Hình ảnh #2 chuyển cho #9 \$10 .....	11
Hình 2.3. Người trong mạng lưới Blockchain tham gia tìm số niêm phong.....	13
Hình 2.4. Ledger L bao gồm Blockchain B và World State W .....	22
Hình 2.5. World State trong Blockchain .....	23
Hình 2.6. Minh họa về Blockchain .....	23
Hình 2.7. Minh họa về Transaction.....	25
Hình 2.8. Client bắt đầu một Transaction.....	27
Hình 2.9. Các Endorsing Peers sẽ kiểm tra chữ ký và thực thi Transaction với bản sao Ledger được lưu trên nó .....	28
Hình 2.10. Transaction Value cùng với chữ ký của Endorsing Peer đã tạo ra nó sẽ được gửi trở lại đến SDK dưới dạng một proposal response .....	29
Hình 2.11. Đối chiếu các Proposal Response .....	29
Hình 2.12. Client tập hợp các Endorsement vào trong một Transaction .....	30
Hình 2.13. Từng Transaction trong block sẽ được gắn tag là hợp lệ hoặc không hợp lệ. ....	30
Hình 2.14. Blockchain và cơ sở dữ liệu truyền thống.....	31
Hình 2.15. Một Blockchain là một danh sách liên kết bao gồm con trỏ.....	33
Hình 2.16. Giới thiệu Hyperledger Fabric.....	35
Hình 2.17. Kiến trúc của Hyperledger .....	37
Hình 2.18. Các loại tập tin trong Hyperledger.....	38
Hình 2.19. Transaction Flow trong Hyperledger Fabric .....	40

Hình 2.20. Đề xuất giao dịch trong mạng Hyperledger Fabric .....	41
Hình 2.21. Các Endorsers nắm bắt tập hợp các dữ liệu Read và Written, tạo RW Sets, và trả về cho ứng dụng khách sau khi được kí bởi Endorsing Peer .....	42
Hình 2.22. Ứng dụng gửi giao dịch được chứng thực và RW Sets đến Ordering Service .....	43
Hình 2.23. Orderer gửi các giao dịch đã được sắp xếp thông tin thành một khối đến tất cả các Committing Peer .....	43
Hình 2.24. Committing Peers xác nhận mỗi Transaction trong khối có khớp với Blockchain không.....	44
Hình 2.25. Committing Peer thông báo không đồng bộ cho ứng dụng khách về sự thành công hay thất bại của giao dịch.....	45
Hình 2.26. Chaincode trong Hyperledger Fabric.....	46
Hình 2.27. Các thành phần trong mạng Hyperledger Fabric.....	47
Hình 2.28. Mô tả Channel trong Hyperledger Fabric .....	48
Hình 2.29. State Database trong Hyperledger Fabric .....	51
Hình 3.1. Mô hình bài toán quản lý bệnh viện .....	55
Hình 3.2. Quy trình hoạt động của hệ thống và mã hóa dữ liệu.....	56
Hình 3.3. Quy trình cập nhật thông tin Doctor Profile.....	57
Hình 3.4. Quy trình cập nhật thông tin Patient Profile.....	58
Hình 3.5. Patient yêu cầu cấp quyền xem thông tin cá nhân của Doctor.....	58
Hình 3.6. Doctor yêu cầu xem thông tin cá nhân của Doctor khác .....	59
Hình 3.7. Doctor yêu cầu cấp quyền xem thông tin cá nhân của Patient.....	59
Hình 3.8. Doctor yêu cầu cấp quyền cập nhật thông tin bệnh án .....	60

Hình 3.9. Patient khởi tạo bệnh án .....	60
Hình 3.10. Doctor cập nhật thông tin bệnh án.....	61
Hình 4.1. Xây dựng hệ thống với NodeJS, Hyperledger Composer và Hyperledger Fabric .....	63
Hình 4.2. Sơ đồ USE-CASE tổng quát.....	64
Hình 4.3. USE-CASE Patient đăng ký vào hệ thống .....	66
Hình 4.4. USE-CASE Doctor đăng ký vào hệ thống .....	67
Hình 4.5. USE-CASE đăng nhập .....	68
Hình 4.6. USE-CASE quản lý thông tin cá nhân.....	69
Hình 4.7. USE-CASE Patient quản lý bệnh án.....	70
Hình 4.8. USE-CASE Doctor cập nhật thông tin bệnh án .....	71
Hình 4.9. USE-CASE quản lý yêu cầu.....	72
Hình 4.10. USE-CASE xem danh sách Doctor .....	73
Hình 4.11. USE-CASE Doctor xem danh sách Patient.....	74
Hình 4.12. USE-CASE đăng xuất.....	75
Hình 4.13. Khởi tạo một Project mới thông qua Hyperledger Composer.....	78
Hình 4.14. Kiến trúc hệ thống của ứng dụng.....	97
Hình 4.15. Cách thức hoạt động của JSON Web Token.....	101
Hình 4.16. Giao diện trang Login .....	107
Hình 4.17. Giao diện trang Đăng ký .....	108
Hình 4.18. Giao diện trang Profile của Doctor.....	109
Hình 4.19. Giao diện trang danh sách bệnh nhân .....	109

Hình 4.20. Giao diện trang danh sách bác sĩ .....	110
Hình 4.21. Giao diện trang danh sách bệnh án .....	110
Hình 4.22. Giao diện trang cập nhật thông tin bệnh án.....	111
Hình 4.23. Giao diện trang Request .....	112
Hình 4.24. Giao diện trang Profile của Patient.....	112
Hình 4.25. Giao diện trang danh sách bác sĩ .....	113
Hình 4.26. Giao diện trang Request của Patient.....	113
Hình 4.27. Giao diện trang chi tiết Health Record của bệnh nhân .....	114
Hình 4.28. Hình ảnh Transaction trong hệ thống.....	115
Hình 4.29. Hình ảnh chi tiết về một Transaction.....	116

## **DANH MỤC BẢNG**

Bảng 1.1. Thời gian khám chữa bệnh của 3 tuyến theo từng loại hình khám bệnh .....	6
Bảng 2.1. So sánh Blockchain và cơ sở dữ liệu.....	33
Bảng 2.2. So sánh Public Blockchain, Private Blockchain và Permissioned Blockchain .....	52
Bảng 2.3. So sánh Bitcoin, Ethereum và Hyperledger.....	53
Bảng 4.1. Danh sách các Enum và mô tả .....	81
Bảng 4.2. Danh sách các Asset của Doctor và mô tả.....	83
Bảng 4.3. Danh sách các Asset của Patient và mô tả.....	86
Bảng 4.4. Transaction và Chaincode của Doctor.....	96
Bảng 4.5. Transaction và Chaincode của Patient.....	96
Bảng 4.6. Danh sách các Route và mô tả .....	106

## DANH MỤC CHỮ VIẾT TẮT

STT	Chữ viết tắt	Ý nghĩa
1	API	Application Programming Interface - giao diện lập trình ứng dụng
2	CDHA	Chuẩn đoán hình ảnh
3	CSDL	Cơ sở dữ liệu
4	ĐT	Điện tử
5	HSSK	Hồ sơ sức khỏe
6	IBM	International Business Machines (Tên một công ty)
7	ICT	Information & Communication Technologies (Công nghệ thông tin và truyền thông)
8	ID	Mã số định danh
9	KT	Kiểm tra
10	PoW	Proof of Work (Bằng chứng công việc)
11	PoS	Proof of Stake (Bằng chứng cổ phần)
12	P2P	Peer-to-Peer
13	UBND	Ủy ban nhân dân

## **TÓM TẮT KHÓA LUẬN**

Trong thời đại của cuộc Cách mạng công nghệ 4.0, công nghệ thông tin đã trở thành một phần không thể thiếu trong cuộc sống, được áp dụng rộng rãi trong nhiều lĩnh vực, đặc biệt hỗ trợ đắc lực trong công tác quản lý, nhất là quản lý công. Nhờ vào công tác tin học hóa mà công tác quản lý, điều hành tỏ ra có hiệu quả nhanh chóng, chính xác, lưu trữ gọn, bảo mật cao và dễ dàng.

Đề tài giúp ứng dụng công nghệ Blockchain vào trong việc quản lý bệnh án điện tử. Việc ứng dụng Blockchain mang lại lợi ích thiết thực trong việc quản lý hồ sơ y khoa. Giảm thiểu công việc cho người quản lý. Ngoài ra còn bảo mật và an toàn dữ liệu hơn. Hồ sơ Blockchain sau khi được tạo bởi người nắm giữ khóa riêng tư sẽ không thể bị làm giả. Tránh việc sai sót trong y khoa sau đó tiến hành sửa đổi hồ sơ nhằm thoát tội.

Trong phạm vi khóa luận, tôi tiến hành phát triển hệ thống quản lý bệnh án điện tử cho một bệnh viện. Do thời gian nghiên cứu có hạn, đề tài được thực hiện bởi một người nên trong quá trình phát triển không tránh có sai sót. Kính mong quý Thầy Cô, bạn bè đóng góp ý kiến để đề tài được hoàn thiện hơn, mang nhiều ý nghĩa thực tiễn hơn.

Khóa luận này chia làm 5 chương bao gồm:

### **Chương 1: MỞ ĐẦU**

Giới thiệu tổng quan nhất về nội dung đề tài, bao gồm tổng quan về đề tài, bài toán, mục tiêu và phạm vi nghiên cứu của đề tài, thách thức của bài toán.

### **Chương 2: CƠ SỞ LÝ THUYẾT**

Giới thiệu về những lý thuyết sử dụng trong đề tài, bao gồm các khái niệm chung về Blockchain và Hyperledger Fabric.



### **Chương 3: MÔ HÌNH BÀI TOÁN**

Giới thiệu về mô hình và các quy trình trong hệ thống

### **Chương 4: CÀI ĐẶT VÀ THỬ NGHIỆM HỆ THỐNG**

Trình bày về môi trường và công cụ xây dựng hệ thống, cũng như các kỹ thuật để triển khai hệ thống và một số giao diện của hệ thống.

### **Chương 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Đánh giá về sản phẩm và hướng phát triển của sản phẩm

## **Chương 1. MỞ ĐẦU**

Trong chương này, tác giả trình bày về tổng quan của đề tài, bài toán cần giải quyết, mục tiêu đề tài, đối tượng và phạm vi của nghiên cứu cũng như thách thức của bài toán.

### **1.1. Tổng quan**

Hiện nay các cơ sở y tế, bệnh viện vẫn đang lưu trữ hồ sơ bệnh án dạng văn bản, ước tính tiêu tốn khoảng 2.300 - 2.500 tỷ đồng hằng năm, là khoản chi phí lớn có thể dành để đầu tư thiết bị, cải tạo hạ tầng hoặc cải thiện chất lượng dịch vụ..., chưa kể các vấn đề lo ngại trong bảo toàn thông tin.

Ứng dụng sẽ giúp người dân tiết kiệm thời gian và chi phí, giúp cơ sở y tế lưu trữ hồ sơ toàn bộ lịch sử khám, chữa bệnh của người dân. Những thông tin cá nhân cơ bản, nhóm máu, tiền sử bệnh, những lần khám, chữa bệnh... được lưu giữ và tái sử dụng để người bệnh chủ động theo dõi tình trạng sức khỏe trong suốt cuộc đời.

Phó chủ tịch UBND TP.HCM vừa ký ban hành kế hoạch triển khai hồ sơ sức khỏe (HSSK) điện tử (ĐT) tại TP giai đoạn 2019 - 2025.<sup>1</sup>

Mục tiêu là xây dựng mẫu HSSK có mã số định danh duy nhất (ID) cho mỗi người dân, thu thập đầy đủ các dữ liệu cơ bản theo mẫu HSSK cho từng người dân theo địa bàn dân cư. Chọn lựa phần mềm tin học về HSSK ĐT đáp ứng các yêu cầu theo quy định của Bộ Y tế. Xây dựng và lưu trữ cơ sở dữ liệu về HSSK ĐT. Đảm bảo chính sách bảo mật, chính sách sao lưu dự phòng, quy định khai thác và sử dụng cơ sở dữ liệu phục vụ người dân và quản lý; tích hợp dữ liệu ngành y tế vào kho dữ liệu dùng chung của TP.

UBND TP yêu cầu trước mắt thu thập dữ liệu ban đầu về tình hình sức khỏe người dân. Các dữ liệu bắt buộc phải có khi lập HSSK ĐT, gồm: dữ liệu hành chính (họ tên, giới tính, ngày sinh, địa chỉ, số CMND/căn cước công dân, mã định danh HSSK ĐT...); dữ liệu tiền sử và các yếu tố liên quan sức khỏe (tiền sử bệnh tật, dị ứng...); dữ liệu chỉ số

---

<sup>1</sup> <https://thanhnien.vn/suc-khoe/xay-dung-ho-so-suc-khoe-dien-tu-phai-dam-bao-tinh-bao-mat-1162473.html>

sinh tồn, chỉ số nhân trắc học (cân nặng, chiều cao, BMI, huyết áp...). Giao Sở Y tế tham mưu UBND TP thành lập ban chỉ đạo, tổ giúp việc thực hiện kế hoạch triển khai HSSK ĐT tại TP... Phấn đấu đến năm 2025 khi hạ tầng của các cơ sở y tế hoàn chỉnh thì đưa vào sử dụng HSSK ĐT.

Qua đó ta thấy sự cấp thiết cần xây dựng một hệ thống quản lý là một điều hoàn toàn cấp bách. Đây cũng chính là lý do chính cho sự ra đời của đề tài này.

## **1.2. Bài toán**

Công nghệ thông tin là một lĩnh vực có nhiều ứng dụng thiết thực nhất trong mọi lĩnh vực của cuộc sống, đặc biệt nó là công cụ hỗ trợ đắc lực không thể thiếu trong công tác quản lý. Nhờ vào công tác tin học hoá mà công tác quản lý, điều hành tỏ ra có hiệu quả, nhanh chóng, chính xác, lưu trữ gọn, bảo mật cao và dễ dàng. Sức mạng của Blockchain rất to lớn, tạo sự đột phá trong mọi ngành của xã hội.

Việc ứng dụng công nghệ Blockchain còn mang lại lợi ích thiết thực trong việc quản lý y khoa với tính năng lưu trữ hồ sơ bệnh án một cách bảo mật, an toàn và chia sẻ.

Đề tài giúp ứng dụng công nghệ Blockchain vào trong việc quản lý điện tử. Giảm thiểu công việc cho người quản lý. Ngoài ra còn bảo mật và an toàn dữ liệu hơn.

Công nghệ Blockchain giúp giảm thiểu các thủ tục giấy tờ, vì vậy làm giảm sự quá tải của các cơ sở khám chữa bệnh hiện nay.

Ngoài ra, khi một dữ liệu trong Blockchain được tạo ra bởi người giữ khóa riêng tư, thì dữ liệu đó sẽ không thể nào bị làm giả. Mọi thao tác trên dữ liệu đó sẽ được ghi lại lịch sử giao dịch. Vì vậy có thể chống được việc ngụy tạo bệnh án, bao che cho người trong ngành khi có sai sót xảy ra. Vì vậy, người bệnh luôn được bảo vệ trong bất cứ tình huống nào, kể cả khi có sai sót xảy ra. Hai tác nhân bác sĩ và bệnh nhân kiểm tra qua lại lẫn nhau tạo nên một sự cân bằng trong hệ thống.

### **1.3. Mục tiêu đề tài**

Nghiên cứu về công nghệ Blockchain. Tìm hiểu các tính năng của Blockchain. Từ đó rút ra các ưu và nhược điểm của công nghệ này.

Cài đặt hệ thống Blockchain.

Xây dựng một ứng dụng đơn giản để kiểm chứng tính năng của Blockchain.

Mục tiêu đề tài ứng dụng Blockchain để quản lý các vấn đề chính như sau:

- Quản lý bệnh nhân
- Quản lý quy trình khám chữa bệnh

Ngoài những yêu cầu xử lý mà bài toán đã đặt ra, chương trình này sẽ hỗ trợ đắc lực cho công tác khám và điều trị bệnh cho bệnh nhân tại các bệnh viện và phòng khám, giảm khối lượng công việc tăng năng suất cho người sử dụng.

### **1.4. Đối tượng và phạm vi nghiên cứu**

- Công nghệ Blockchain và hệ thống quản lý bệnh án điện tử.
- Hyperledger Fabric, Hyperledger Composer.
- Quy trình khám chữa bệnh và lưu hồ sơ y tế.

### **1.5. Thách thức của bài toán**

Quy trình quản lý bệnh án hiện tại còn nhiều phức tạp, qua nhiều khâu trung gian. Điều này làm cho quy trình trở nên vô cùng rườm rà và trở nên khó khăn với người tham gia khám chữa bệnh.

Ngoài ra, một trong những thách thức cần giải quyết trong quá trình xây dựng hệ thống quản lý bệnh án là phải đảm bảo dữ liệu y tế chia sẻ cho các bên liên quan nhưng vẫn đảm bảo tính riêng tư của bệnh án và thông tin của bệnh nhân.

Việc thuyết phục để nhận được sự đồng ý của các bệnh viện sử dụng hệ thống quản lý bệnh án điện tử cũng là một vấn đề đáng cân nhắc.

Mỗi hồ sơ bệnh án được lưu trữ thông qua các quy trình làm việc khác nhau, nếu không sử dụng Blockchain sẽ rất khó để biết được rằng ai đã sửa cũng như cập nhật thông tin gì vào bệnh án của bệnh nhân. Vì thế thách thức cần xây dựng một môi trường làm việc đáng tin cậy là một thách thức lớn cho người phát triển hệ thống.

STT	Loại hình khám bệnh	Thời gian khám	Giảm so với thời gian quy định	Thời gian quy định
1	Khám lâm sàng đơn thuần trung bình	66.5 phút (Trong đó thời gian chờ khám bệnh là 45.4 phút)	54.5 phút	< 2 giờ
2	Khám lâm sàng có làm thêm 01 KT xét nghiệm/CĐHA, thăm dò chức năng	125.6 phút (Trong đó thời gian chờ khám bệnh là 56.5 phút)	54.5 phút	< 3 giờ
3	Khám lâm sàng có làm thêm 02 KT phối hợp cả xét nghiệm và CĐHA hoặc xét nghiệm và thăm dò chức năng	176.2 phút (Trong đó thời gian chờ khám bệnh là 71.4 phút)	33.8 phút	< 3.5 phút
4	Khám lâm sàng có làm thêm 02 KT phối hợp cả xét nghiệm và CĐHA hoặc xét nghiệm và thăm dò chức năng	199.9 phút (Trong đó thời gian chờ khám bệnh là 92.6 phút)	40.1 phút	< 4 giờ

*Bảng 1.1. Thời gian khám chữa bệnh của 3 tuyến theo từng loại hình khám bệnh*

Thông qua bảng trên, ta thấy quy trình khám bệnh còn nhiều phức tạp và tốn thời gian nhiều. Vì thế sự cấp bách cần một quy trình khám bệnh tốt hơn cũng là một bài toán cần giải quyết.

## **1.6. Kết chương**

Trong chương 1, đề tài giới thiệu tổng quan về thực trạng quản lý bệnh án tại Việt Nam cũng như những bất cập của nó mang lại. Từ đó đưa ra được bài toán cấp bách cần xây dựng hệ thống quản lý bệnh án điện tử, góp phần tránh việc sửa bệnh án một cách trái phép.

Từ đây, khóa luận rút ra được đối tượng, phạm vi của đề tài mà khóa luận trình bày, đưa ra hướng giải quyết đúng đắn mà tác giả cần thực hiện cho đề tài này, giới hạn được công việc cần thực hiện, rút ra được thách thức mà tác giả và đề tài gặp phải, làm căn cứ đề xuất hướng giải quyết.

Tóm lại, chương này đưa ra các luận cứ ban đầu cho khóa luận của tác giả, định ra hướng đi cho tác giả trong suốt quá trình giải quyết vấn đề của đề tài.

## **Chương 2. CƠ SỞ LÝ THUYẾT**

Chương này tác giả nói về các khái niệm chung về Blockchain giúp người dùng có một cái nhìn tổng quát về Blockchain để có thể hiểu sâu hơn các chương sau. Ngoài ra chương này còn nói về Hyperledger Blockchain và Hyperledger Fabric. Qua đó người đọc sẽ hiểu được các ưu điểm và nhược điểm của công nghệ Blockchain và lý do sử dụng công nghệ này cho đề tài.

### **2.1. Các khái niệm chung**

#### **2.1.1. Tìm hiểu về Blockchain**

##### ***a. Giới thiệu***

Người ta biết đến Blockchain với một ứng dụng khá nổi trong giới truyền thông cũng như công nghệ hiện nay (Bitcoin). Tiền kỹ thuật số đã trở thành một vấn đề được bàn luận khá sôi nổi trên rất nhiều phương tiện truyền thông đại chúng, báo chí, báo mạng...

Công nghệ Blockchain theo định nghĩa của IBM: Blockchain là một cuốn sổ cái phân tán được chia sẻ, bất biến để ghi lại các giao dịch, theo dõi tài sản và xây dựng niềm tin. Thông tin được lưu trữ trong các khối mã hóa được liên kết với nhau và mở rộng theo thời gian. Mỗi khối chứa băm (hash) của khối trước đó, dấu thời gian (timestamp) và dữ liệu giao dịch.

Các Block này hoạt động độc lập và có thể mở rộng theo thời gian. Chúng được quản lý bởi những người tham gia hệ thống chứ không phải qua đơn vị trung gian. Blockchain sở hữu tính năng vô cùng đặc biệt đó là việc truyền tải dữ liệu không đòi hỏi một đơn vị trung gian để xác nhận thông tin. Hệ thống Blockchain tồn tại rất nhiều nút độc lập có khả năng xác thực thông tin mà không đòi hỏi “dấu hiệu của niềm tin”. Thông tin trong Blockchain không thể bị thay đổi và chỉ được bổ sung thêm khi có sự đồng thuận của tất cả các nút trong hệ thống. Đây là một hệ thống bảo mật an toàn cao trước khả năng bị đánh cắp dữ liệu. Ngay cả khi một phần của hệ thống Blockchain bị sụp đổ, những máy tính và các nút khác sẽ tiếp tục bảo vệ thông tin và giữ cho mạng tiếp tục hoạt động.

Các Block trao đổi thông tin với nhau thông qua các Transaction. Transaction là một giao dịch được gửi bởi người tham gia để tác động đến tài sản (Asset) được giữ trong sổ đăng ký tài sản trên Blockchain. Giao dịch với mạng doanh nghiệp (Business Network) được xác định trong mô hình mạng doanh nghiệp và các hoạt động của chúng được xác định trong tập chức năng của bộ xử lý giao dịch. Các chức năng của bộ xử lý giao dịch tác động lên tài sản và người tham gia để tạo, cập nhật hoặc xóa các thuộc tính trên tài sản và người tham gia.

Tuy nhiên, không có nhiều người thực sự hiểu Blockchain là gì. Nó được ứng dụng vào cuộc sống hiện tại như thế nào mà được các phương tiện truyền thông bàn luận nhiều đến như vậy.

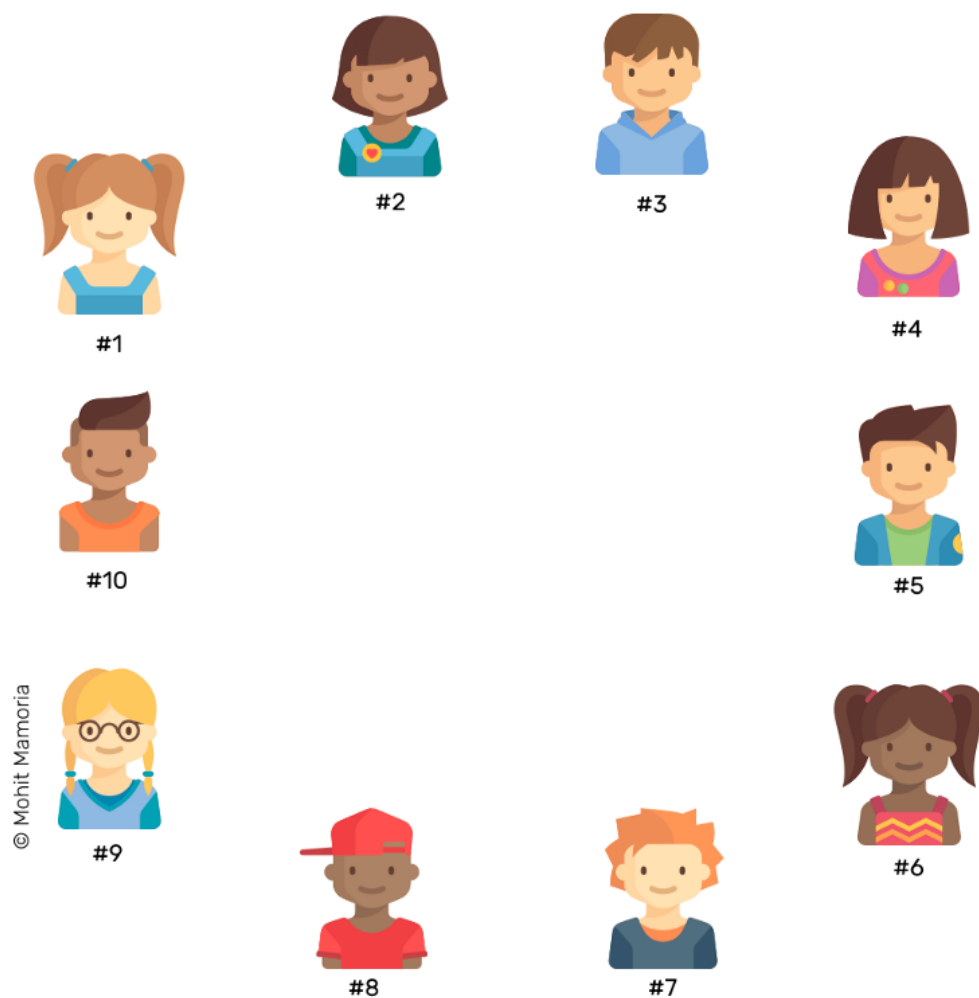
### **Ví dụ đơn giản về Blockchain:<sup>2</sup>**

Giả sử có 10 người muốn không phải phụ thuộc vào ngân hàng hoặc bất kỳ tổ chức thứ ba nào. Theo như thỏa thuận ban đầu, họ sẽ có thông tin chi tiết về tài khoản của những người khác trong suốt thời gian thỏa thuận có hiệu lực, ngoại trừ một thông tin quan trọng - danh tính (identity) của người đó.

---

<sup>2</sup> <https://kipalog.com/posts/Blockchain-la-cai-quai-gi-vay--->





*Hình 2.1. Hình ảnh số người tham gia vào Blockchain*

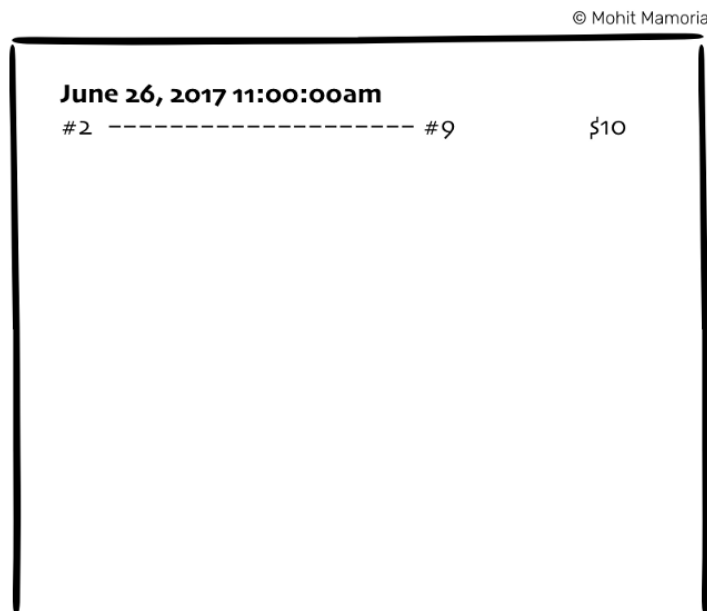
Tất cả mọi người đều được cung cấp một thư mục rỗng (empty folder) để bắt đầu. Khi quá trình bắt đầu diễn ra, tất cả 10 cá nhân độc lập này sẽ cùng thêm các trang giấy vào thư mục rỗng hiện tại của họ. Và tập hợp các trang giấy này sẽ tạo thành một cuốn sổ cái để theo dõi các giao dịch.

Bây giờ, hãy tưởng tượng tất cả 10 người trong mạng lưới đều có một tờ giấy trắng và một chiếc bút chì trong tay. Mọi người đều đã sẵn sàng để ghi ra bất cứ giao dịch nào xảy ra trong hệ thống.

Giả sử có một giao dịch xảy ra, #2 muốn chuyển 10\$ cho #9.

Để thực hiện giao dịch, #2 thông báo với tất cả mọi người: "Tôi muốn chuyển 10\$ cho #9. Mọi người xin hãy note lại vào trang giấy của mình".

Tiếp đó, tất cả mọi người sẽ cùng kiểm tra xem #2 có đủ số dư trong tài khoản để chuyển 10\$ cho #9 hay không. Nếu ok, tất cả mọi người sẽ cùng note lại thông tin giao dịch trên trang giấy của mình.



*Hình 2.2. Hình ảnh #2 chuyển cho #9 \$10*

Giao dịch sau đó được coi là hoàn thành.

Công việc này sẽ được tiếp tục, cho đến khi trang giấy của mọi người đều đã kín chỗ trống. Giả sử rằng mỗi trang giấy chỉ có đủ chỗ trống cho 10 giao dịch.

Khi trang giấy của mọi người đều đã đầy, đã đến lúc cất trang giấy đó vào tập tài liệu cá nhân và lấy ra một trang giấy trắng mới

Trước khi cất trang giấy ghi lại các giao dịch vào tập tài liệu, chúng ta cần niêm phong nó bằng một khoá unique mà tất cả mọi người trong mạng (network) đều đồng ý. Bằng cách niêm phong trang giấy này, chúng ta sẽ đảm bảo rằng không ai có thể thực hiện bất kỳ thay đổi về nội dung nào khi mà bản sao của nó được lưu giữ trong tập tài liệu của từng người - không phải hôm nay, ngày mai hay thậm chí là rất nhiều năm về sau.

Một khi đã ở trong tập tài liệu, nó sẽ mãi ở trong đó, và luôn được niêm phong. Hơn nữa, nếu mọi người tin tưởng vào dấu niêm phong, mọi người cũng sẽ tin tưởng vào nội dung của trang giấy ghi lại giao dịch. Dấu niêm phong chính là điểm mấu chốt của phương pháp này.

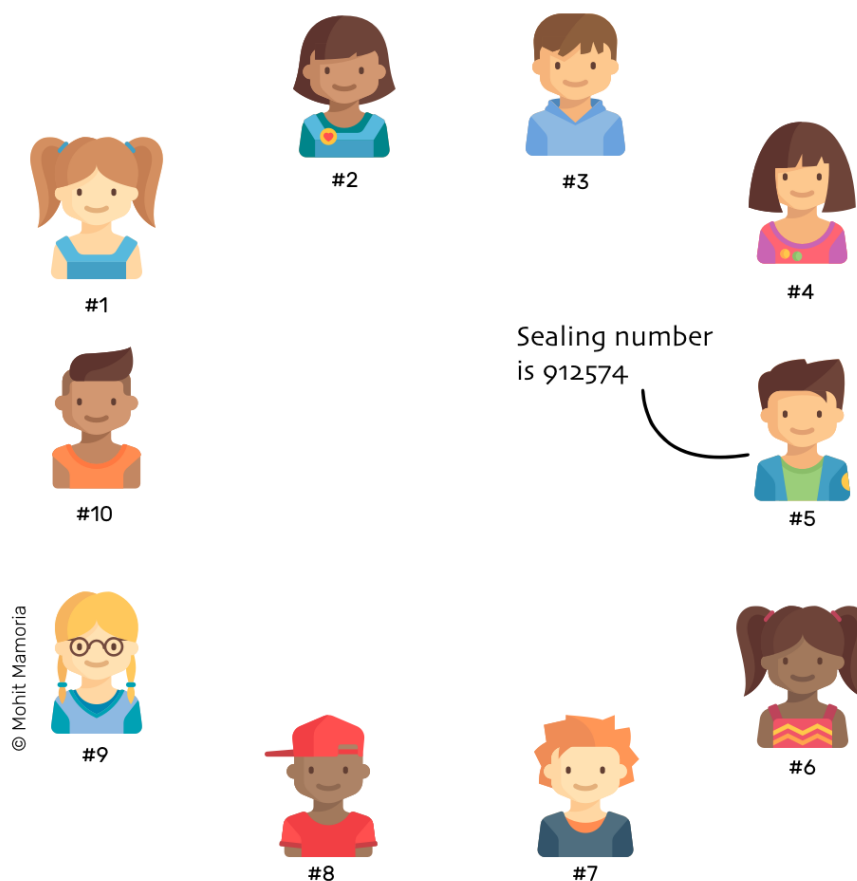
Trước kia, các bên thứ 3, bên trung gian (ngân hàng, các tổ chức tài chính) tạo cho chúng ta niềm tin rằng bất cứ điều gì họ viết trong cuốn sổ cái (register) sẽ không bao giờ thay đổi. Trong một hệ thống phân tán (distributed) và không tập trung (decentralized) như hệ thống tôi đang mô tả, dấu niêm phong sẽ cung cấp niềm tin tương tự như thế.

Để niêm phong trang giấy ghi các thông tin giao dịch của toàn bộ network, chúng ta cần tìm ra một con số mà khi nối vào danh sách giao dịch và đưa qua hàm Hash, chúng ta sẽ thu được một chuỗi ký tự bắt đầu bằng điều kiện mong muốn.

Khi chuỗi ký tự được tính toán sau khi đã tiêu tốn một khoảng thời gian (và cả điện năng), nó sẽ được dùng để niêm phong trang thông tin giao dịch. Nếu có một ai đó thử thay đổi nội dung của trang giấy này, số niêm phong sẽ cho phép bất kỳ ai cũng có thể kiểm tra tính toàn vẹn của trang giấy.

Sau khi đã hiểu về cơ chế niêm phong, chúng ta sẽ cùng quay lại thời điểm khi trang ghi thông tin ghi giao dịch đã ghi đầy 10 giao dịch và hết chỗ để viết thêm.

Lúc này, tất cả mọi người trong mạng sẽ cùng thực hiện việc tính toán để tìm ra con số niêm phong. Người đầu tiên tìm ra được số niêm phong sẽ thông báo cho tất cả những người còn lại.



*Hình 2.3. Người trong mạng lưới Blockchain tham gia tìm số niêm phong*

Ngay khi nhận được số niêm phong, tất cả mọi người trong mạng sẽ xác nhận xem nếu dùng Hash Function với đầu vào là số niêm phong và trang ghi thông tin giao dịch, chuỗi ký tự trả về có thoả mãn yêu cầu đưa ra hay không. Nếu có, tất cả mọi người sẽ đóng dấu trang thông tin giao dịch của mình với số niêm phong này và đặt nó vào tập tài liệu.

Nhưng nếu có ai đó, giả sử là #7, nói rằng số niêm phong này không thoả mãn yêu cầu đầu ra. Những trường hợp như vậy không phải là điều bất thường.

Những lý do có thể gây ra điều này bao gồm :

- Có lỗi trong quá trình tiếp nhận thông tin (nhiều,...) do đó số niêm phong #7 nhận được không phải là số niêm phong đã được thông báo ra.
- Anh ta có thể đã viết sai các giao dịch được thông báo trong mạng
- Anh ta có thể đã cố gắng lừa dối mọi người hoặc không trung thực khi viết các giao dịch

Không quan trọng lý do là gì, #7 chỉ có duy nhất một lựa chọn, đó là loại bỏ trang ghi thông tin giao dịch của mình và copy lại từ một người khác, để sau đó có thể đặt nó vào tập thư mục của mình.

Nếu #7 không đưa trang ghi thông tin giao dịch của mình vào tập thư mục, anh ta không thể tiếp tục viết thêm các giao dịch khác nữa, và như vậy anh ta sẽ bị cấm tham gia vào mạng lưới.

### ***b. Lịch sử hình thành và phát triển<sup>3</sup>***

- Năm 1991: Stuart Haber và W. Scott Stornetta lần đầu tiên miêu tả về công nghệ chuỗi các khối được bảo mật (Cryptographically secured chain of blocks).
- Năm 1992: Haber và Stornetta cùng Bayer tiếp tục bước thêm một bước bằng cách thêm Merkle Tree (còn gọi là Hash tree) để thiết kế khối chuỗi giúp lưu trữ dữ liệu hiệu quả hơn trong một khối đơn lẻ thay vì cần nhiều khối để lưu trữ.
- Năm 2008: Satoshi Nakamoto lần đầu tiên đưa ra khái niệm Blockchain, từ “khối” (Block) và “chuỗi” (Chain) được sử dụng riêng trong bài báo gốc của Satoshi Nakamoto.

---

<sup>3</sup> <https://medium.com/lumiwallet/the-history-of-blockchain-pt-1-first-concepts-and-their-creators-a406225f8488>

- Năm 2009, Blockchain trở thành phần cốt lõi của tiền điện tử Bitcoin.
- Từ năm 2014, thuật ngữ Blockchain 2.0 được sử dụng cùng với việc phát triển hợp đồng thông minh (Smart Contract). Hợp đồng thông minh ứng dụng trong phạm vi rộng hơn trong kinh tế, tài chính, chứng khoán, ngân hàng gồm những việc có liên quan đến thỏa thuận hay hợp đồng.
- Từ năm 2017, Blockchain 3.0 được phát triển không chỉ ứng dụng trong lĩnh vực kinh tế, tài chính mà còn mở rộng ra các lĩnh vực như giáo dục, y tế, chính phủ và nghệ thuật.

### ***c. Khái quát Blockchain***

Công nghệ Blockchain theo định nghĩa của IBM: Blockchain là một cuốn sổ cái phân tán được chia sẻ, bất biến để ghi lại các giao dịch, theo dõi tài sản và xây dựng niềm tin. Thông tin được lưu trữ trong các khối mã hóa được liên kết với nhau và mở rộng theo thời gian. Mỗi khối chứa băm (hash) của khối trước đó, dấu thời gian (timestamp) và dữ liệu giao dịch.

Cuốn sổ cái này được chia sẻ bởi một mạng lưới. Mỗi thành viên của mạng này giữ bản sao của cuốn sổ cái đó và tất cả thành viên phải xác thực mọi bản cập nhật chung để tất cả bản sao của cuốn sổ cái đó có nội dung giống nhau.

Thông tin lưu trong cuốn sổ này là những giao dịch, hợp đồng, tài sản hay bất kì điều gì khác có lưu trữ ở dạng kỹ thuật số.

Toàn bộ các thông tin lưu trữ trong Blockchain minh bạch vì thông tin không thể sửa, các thành viên trong cộng đồng có thể xem, truy xuất thông tin.

### ***d. Cách thức hoạt động***

Công nghệ Blockchain có thể nói là sự kết hợp giữa 3 loại công nghệ sau:<sup>4</sup>

---

<sup>4</sup> <https://bitcoinvietnamnews.com/blockchain-la-gi>

- Mật mã học: sử dụng public key và hàm hash để đảm bảo tính minh bạch, toàn vẹn dữ liệu và đảm bảo tính riêng tư.
- Mạng ngang hàng: mỗi một nút trong mạng được xem như một client và cũng là server để lưu trữ bản sao của ứng dụng.
- Lý thuyết trò chơi: tất cả các nút tham gia vào hệ thống đều phải tuân thủ luật chơi đồng thuận (Proof of work - PoW, Proof of stake - PoS...) và được thúc đẩy bởi động lực kinh tế.

**Cơ chế đồng thuận:** Đối với các mạng Blockchain, đây là một bước quan trọng để đảm bảo rằng các thông tin được sử dụng trong mạng lưới là đã được chứng thực một cách kỹ càng với sự khớp lệnh của hơn 50% các Peer trong mạng lưới này. Để đảm bảo cho bất kỳ một giao dịch nào được thực hiện đều là chính xác, mạng lưới sẽ sử dụng các nút của mạng lưới để thực hiện khớp lệnh, chỉ khi hơn 50% số nút khớp lệnh thì giao dịch đó mới được thực hiện. Cũng vì cơ chế này mà thông tin trên blockchain muốn thực hiện tấn công là một việc khá khó khăn. Bởi vì nếu bạn muốn tấn công và lấy được Blockchain đó bạn phải đảm bảo 2 yếu tố:

- Bạn phải chiếm được quyền kiểm soát được ít nhất là 1 nút trong mạng lưới.
- Bạn phải gần như phải cùng lúc tấn công trên 50% các nút khác của mạng lưới để làm chủ khớp lệnh của các nút này.

**Tính bất biến:** Tất cả các thông tin giao dịch được thực hiện bởi mạng lưới sẽ được lưu trữ xuống Blockchain. Và những dữ liệu này hoàn toàn không thể sửa hay xóa được, trừ khi bạn xóa luôn hoàn toàn mạng Blockchain này. Điều này là một trở ngại rất lớn vì chỉ cần chúng ta không hủy hết toàn bộ Blockchain lưu trữ trên tất cả các Peer, chỉ cần 1 Peer còn sống sót nó sẽ phục hồi lại Blockchain cho các Peer đã mất trong mạng lưới. Như vậy khả năng xóa bỏ hoàn toàn mạng lưới là việc khó khăn vô cùng.

Để hoạt động, Blockchain cần ba nhóm chính bao gồm: người dùng, các nút và các thợ đào:

- **Người dùng:** là những người tham gia vào hệ thống tạo ra các giao dịch.
- **Nút:** là tất cả các máy tính kết nối với mạng có thể đọc và viết từ một Blockchain. Đây được coi là xương sống của Blockchain. Các nút luôn được kết nối và đồng bộ với mạng, chứa một bản sao đầy đủ của tất cả các giao dịch đã từng xảy ra.
- **Thợ đào:** là các thành viên trong mạng Blockchain với các máy tính có cấu hình cao, đóng vai trò xác nhận các giao dịch và thêm một khối vào mạng khi khối đã được xác nhận.

#### ***e. Phân loại Blockchain***

Có 3 loại Blockchain gồm:<sup>5</sup>

- **Blockchain công cộng (Public Blockchain):** là một mạng mở, bất kỳ ai có kết nối internet cũng có thể tham gia vào mạng để xem các thông tin, thực hiện các giao dịch. Do đó việc xác thực vào mạng Blockchain này đòi hỏi phải có hàng vạn người và hàng vạn nút tham gia. Việc tấn công vào mạng Blockchain này là điều bất khả thi (nếu muốn tấn công vào mạng này thì cần chi phí rất lớn và công nghệ cực kỳ tối tân). Blockchain công cộng mang đặc tính của Blockchain một cách rõ ràng như tính phi tập trung, tính phân tán, bỏ qua trung gian, phân quyền toàn mạng lưới,... Một số Blockchain ở dạng này là Bitcoin, Ethereum,...
- **Blockchain riêng (Private Blockchain):** là một mạng do một tổ chức tư nhân mở và có quyền quản trị. Khi một người tham gia phải được quản trị viên cấp phép một hay một số quyền cụ thể như kết nối vào mạng, xem thông tin, gửi, nhận, lưu trữ dữ liệu. Người dùng chỉ được quyền đọc dữ liệu, không được quyền ghi dữ liệu. Quyền ghi dữ liệu thuộc thẩm quyền của bên tổ chức thứ ba tuyệt đối tin cậy. Bên thứ ba được

---

<sup>5</sup> <https://www.hashcashconsultants.com/media/differences-between-public-private-and-permissioned-blockchain-network/>



toàn quyền quyết định mọi thay đổi trên mạng Blockchain. Vì là Private Blockchain nên thời gian xác nhận giao dịch khá nhanh vì chỉ cần một lượng nhỏ các nút tham gia vào quá trình xác nhận.

- **Blockchain riêng một phần (Consortium Blockchain/Hybrid Blockchain/Permissioned Blockchain):** tương tự như Blockchain riêng, nhưng thay vì một tổ chức kiểm soát mạng Blockchain thì sẽ có một số tổ chức cùng kiểm soát Blockchain đó. Và những người mới khi tham gia vào mạng thì chỉ cần một trong các quản trị viên cấp phép quyền là có thể tham gia. Ví dụ: các ngân hàng hay tổ chức tài chính liên doanh sẽ sử dụng Blockchain cho riêng mình.

#### ***f. Các phương pháp lưu trữ trong Blockchain***

Có 3 phương pháp lưu trữ gồm:<sup>6</sup>

**Giao dịch tài sản (Asset Transaction):** giao dịch bằng tài sản gồm có 2 dạng:

- **Tiền tệ:** ở đây muốn nói đến chính là tiền ảo được sử dụng trong công nghệ Blockchain. Giao dịch của tiền ảo được Blockchain lưu trữ lại.
- **Chứng từ về quyền sở hữu:** chứng từ về quyền sở hữu cho tài sản hữu hình như đất, nhà, xe cộ,...hay các tài sản vô hình như quyền sở hữu trí tuệ.

**Hợp đồng thông minh (Smart Contract):** là các chương trình máy tính nhỏ được lưu trữ trên một Blockchain, các giao dịch được thực hiện với điều kiện được nêu trong hợp đồng. Do đó hợp đồng thông minh thường là một dạng câu điều kiện như sau “thực hiện B nếu A xảy ra” hoặc “thực hiện C nếu A xảy ra, ngược lại thì thực hiện B”,... Và không giống với hợp đồng bình thường, hợp đồng thông minh sẽ tự động thực hiện. Trong khi bình thường, sau khi đạt được thỏa thuận, các bên tham gia trong hợp đồng thực hiện điều được ghi trong hợp đồng.

---

<sup>6</sup> A TRACEABILITY SYSTEM FOR PRODUCT BASED ON BLOCKCHAIN TECHNOLOGY – Bui Xuan Tai

**g. Đặc điểm và tính chất của Blockchain <sup>7</sup>**

**Tính bảo mật:** các dữ liệu, thông tin được lưu trữ trên Blockchain được phân tán khắp nơi và đảm bảo an toàn vì chỉ có người nắm giữ khóa riêng tư mới có quyền truy xuất dữ liệu đó.

**Tính toàn vẹn:** đảm bảo các dữ liệu giao dịch không thể bị làm giả, không thể phá hủy các chuỗi Blockchain. Theo như lý thuyết thì chỉ có máy tính lượng tử mới có thể can thiệp vào giải mã các chuỗi Blockchain, đồng thời nó chỉ có thể bị phá hủy khi không có Internet trên toàn cầu.

**Tính bất biến:** một khi những dữ liệu hoặc giao dịch đã được ghi vào Blockchain bởi người nắm giữ khóa riêng tư hay còn gọi là private key (khóa này chỉ riêng người khởi tạo Blockchain mới sở hữu) thì dữ liệu đó sẽ không thể sửa chữa và nó sẽ được lưu lại mãi mãi trên Blockchain

**Tính phân tán:** thông tin được tổ chức trên Blockchain tồn tại dưới dạng cơ sở dữ liệu được chia sẻ và hòa hợp liên tục. Không có một phiên bản tập trung nào của cơ sở dữ liệu này tồn tại do chúng được phân tán khắp nơi.

**Tính minh bạch:** ai cũng có thể xem được đường đi của các giao dịch trên Blockchain từ địa chỉ này tới địa chỉ khác và có thể thống kê được toàn bộ lịch sử trên địa chỉ đó.

**h. Các cơ chế đồng thuận trong Blockchain <sup>8</sup>**

Proof of work (Bằng chứng công việc): đây chính là nguồn tài nguyên máy tính và năng lượng điện, những thứ giúp các thợ đào giải quyết các bài toán phức tạp. Được sử dụng phổ biến trong Ethereum hay Bitcoin. Việc này tiêu tốn khá nhiều năng lượng điện.

---

<sup>7 8</sup> BUILDING AN ELECTRONIC HEALTH RECORD MANAGEMENT SYSTEM USING BLOCKCHAIN – Vu Tan Phong, Vo Minh Ngoc

Proof of stake (Bằng chứng cổ phần): Cơ chế đồng thuận này phân cấp hơn, ít tốn năng lượng hơn. Phổ biến trong Peercoin và nhiều loại tiền mã hóa khác.

Proof of authority (Bằng chứng ủy nhiệm): Đây là mô hình tập trung, hiệu suất cao, có khả năng mở rộng tốt.

Proof of weight (Bằng chứng khối lượng): Có thể tùy chỉnh và mở rộng tốt, thường thấy trong Filecoin...

Byzantine Fault Tolerance (Đồng thuận chống gian lận): phổ biến trong Hyperledger, Ripple... Năng suất cao, chi phí thấp, có khả năng mở rộng. Hiện nay Hyperledger đang được IBM phát triển khá tốt.

### ***i. Ledger trong Blockchain <sup>9</sup>***

Sổ cái là khái niệm quan trọng trong Blockchain, nó lưu trữ thông tin thực tế quan trọng về các đối tượng kinh doanh; cả giá trị hiện tại của các thuộc tính của các đối tượng và lịch sử giao dịch dẫn đến các giá trị hiện tại này.

Một sổ cái chứa trạng thái hiện tại của một doanh nghiệp như một tập chí giao dịch. Các sổ cái đầu tiên của châu Âu và Trung Quốc có từ gần 1000 năm trước, và người Sumer đã có các sổ cái bằng đá cách đây 4000 năm - nhưng hãy để bắt đầu với một ví dụ cập nhật hơn!

Bạn có thể sử dụng để xem tài khoản ngân hàng của bạn. Điều quan trọng nhất đối với bạn là số dư khả dụng - đó là những gì bạn có thể chi tiêu vào thời điểm hiện tại. Nếu bạn muốn xem số dư của mình được lấy như thế nào, thì bạn có thể xem qua các khoản tín dụng và ghi nợ giao dịch đã xác định số dư đó. Đây là một ví dụ thực tế của một sổ cái - một trạng thái (số dư ngân hàng của bạn) và một tập hợp các giao dịch được đặt hàng (tín dụng và ghi nợ) xác định nó.

---

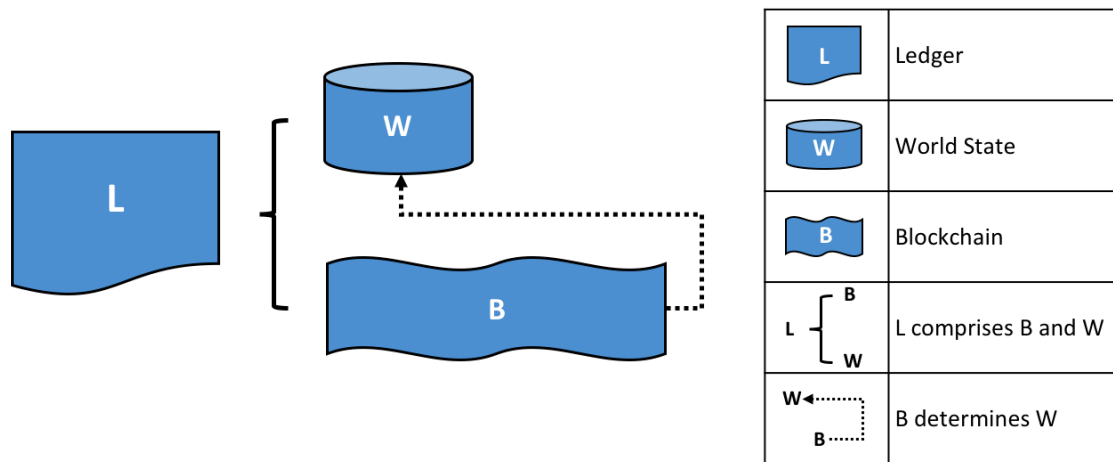
<sup>9</sup> <https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html>

### ***Sổ cái (ledger), sự kiện (fact) và trạng thái (state)***

Một sổ cái không có nghĩa đen là lưu trữ các đối tượng kinh doanh - thay vào đó nó lưu trữ sự kiện về các đối tượng đó. Khi chúng tôi nói, chúng tôi lưu trữ một đối tượng kinh doanh trong một sổ cái, điều chúng tôi thực sự muốn nói là chúng tôi ghi lại sự thật về tình trạng hiện tại của một đối tượng (state) và sự kiện về lịch sử của các giao dịch dẫn đến trạng thái hiện tại (current state). Trong một thế giới ngày số hóa, có thể có cảm giác như chúng ta đang nhìn vào một vật thể, hơn là sự kiện về một vật thể. Trong trường hợp của một đối tượng kỹ thuật số, nó có khả năng nó tồn tại trong kho dữ liệu bên ngoài; các sự kiện chúng tôi lưu trữ trong sổ cái cho phép chúng tôi xác định vị trí của nó cùng với các thông tin quan trọng khác về nó.

Mặc dù sự thật về tình trạng hiện tại của một đối tượng kinh doanh có thể thay đổi, lịch sử của sự kiện về nó là bất biến (immutable), nó có thể được thêm vào, nhưng nó không thể được thay đổi hồi cứu. Chúng tôi sẽ thấy cách nghĩ về Blockchain như một lịch sử bất biến về các sự kiện về các đối tượng kinh doanh là một cách đơn giản nhưng mạnh mẽ để hiểu về nó.

Một sổ cái bao gồm hai phần riêng biệt, mặc dù có liên quan - một trạng thái thế giới và một chuỗi khối. Mỗi trong số này đại diện cho một tập hợp các sự kiện về một tập hợp các đối tượng kinh doanh.



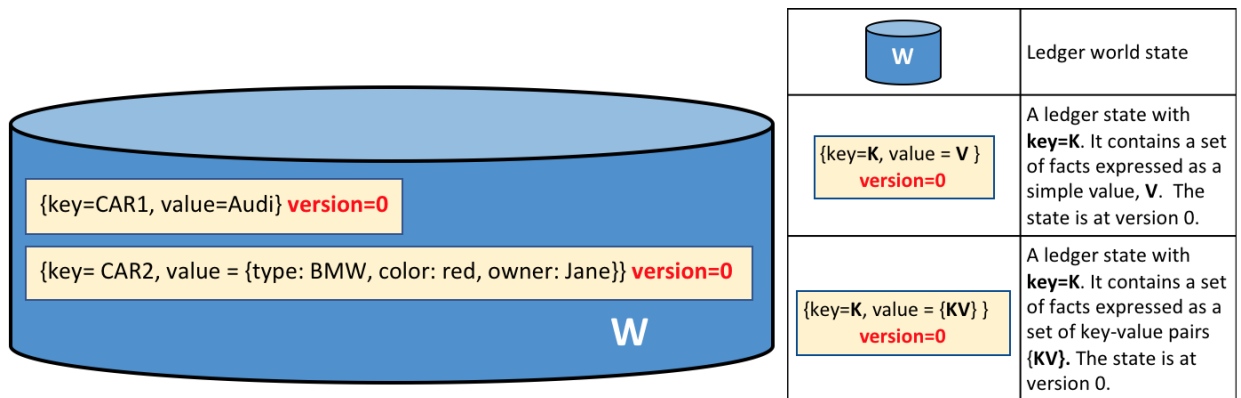
Hình 2.4. Ledger *L* bao gồm Blockchain *B* và World State *W*

Một Ledger *L* bao gồm Blockchain *B* và trạng thái thế giới *W*, trong đó Blockchain *B* xác định trạng thái thế giới *W*. Chúng ta cũng có thể nói rằng trạng thái *W* thế giới có nguồn gốc từ Blockchain *B*.

### ***World State***

Trạng thái thế giới giữ giá trị hiện tại của các thuộc tính của đối tượng kinh doanh dưới dạng trạng thái sô cái duy nhất. Điều đó rất hữu ích vì các chương trình thường yêu cầu giá trị hiện tại của một đối tượng; sẽ rất khó khăn khi đi qua toàn bộ Blockchain để tính toán giá trị hiện tại của đối tượng - bạn chỉ cần lấy nó trực tiếp từ trạng thái thế giới

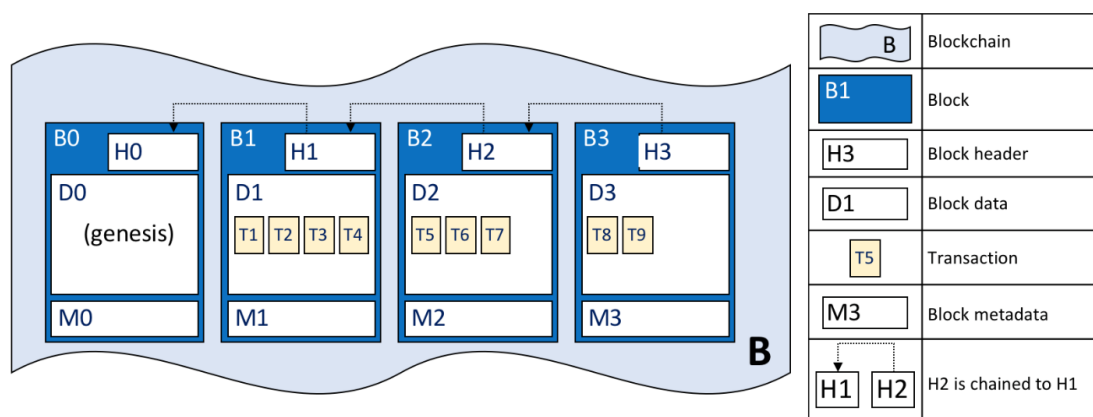
### ***Ví dụ:***



Hình 2.5. World State trong Blockchain

Một World State bao gồm hai trạng thái. Trạng thái đầu tiên là: key = CAR1 và value = Audi. Trạng thái thứ hai có giá trị phức tạp hơn: key = CAR2 và value = {model: BMW, color = red, own = Jane}. Cả hai trạng thái đều ở phiên bản 0.

#### j. Blockchain



Hình 2.6. Minh họa về Blockchain

Blockchain là một bản ghi lịch sử về các sự kiện về cách các đối tượng này đến trạng thái hiện tại của họ. Blockchain đã ghi lại mọi phiên bản trước đó của mỗi trạng thái sổ cái và cách nó được thay đổi.

Blockchain được cấu trúc như nhật ký tuần tự của các khối được liên kết với nhau, trong đó mỗi khối chứa một chuỗi các giao dịch, mỗi giao dịch đại diện cho một truy vấn hoặc cập nhật lên trạng thái thế giới. Cơ chế chính xác theo đó các giao dịch được đặt hàng được thảo luận ở nơi khác; Điều quan trọng nhất là thứ tự khối đó, cũng như trình tự giao dịch trong các khối, được thiết lập khi các khối được tạo lần đầu tiên bởi một thành phần được gọi là dịch vụ đặt hàng (Ordering services).

Mỗi Block header bao gồm một hàm băm của các giao dịch khối khối, cũng như một bản sao của hàm băm của tiêu đề Block header trước đó. Theo cách này, tất cả các giao dịch trên sổ cái được sắp xếp theo trình tự và liên kết mật mã với nhau. Băm và liên kết này làm cho dữ liệu sổ cái rất an toàn. Ngay cả khi một nút lưu trữ sổ cái bị giả mạo, nó sẽ không thể thuyết phục tất cả các nút khác rằng nó có Blockchain chính xác vì sổ cái được phân phối trên toàn mạng lưới các nút độc lập.

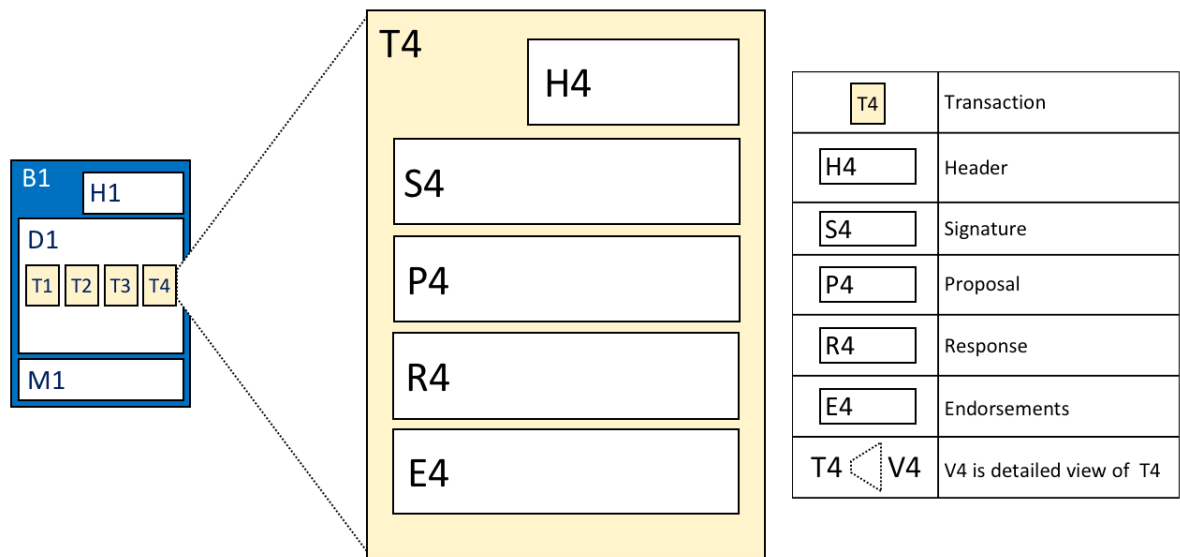
Blockchain luôn được triển khai dưới dạng tập, trái ngược với trạng thái thế giới, sử dụng cơ sở dữ liệu. Đây là một lựa chọn thiết kế hợp lý vì cấu trúc dữ liệu Blockchain bị thiên lệch rất nhiều đối với một tập hợp rất nhỏ các hoạt động đơn giản. Áp dụng vào cuối Blockchain là hoạt động chính và truy vấn hiện là hoạt động tương đối không thường xuyên.

#### ***k. Transaction trong Blockchain<sup>10</sup>***

Như đã thấy, một giao dịch nắm bắt các thay đổi đối với World State. Hãy xem xét cấu trúc dữ liệu khối chi tiết có chứa các giao dịch trong một khối.

---

<sup>10</sup> <https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html>



*Hình 2.7. Minh họa về Transaction*

Chi tiết giao dịch. Giao dịch T4 trong blockdata D1 của khối B1 bao gồm tiêu đề giao dịch, H4, chữ ký giao dịch, S4, đề xuất giao dịch P4, phản hồi giao dịch, R4 và danh sách chứng thực, E4.

#### **Header:**

Phần này, được minh họa bởi H4, nắm bắt một số siêu dữ liệu cần thiết về giao dịch - ví dụ: tên của mã chuỗi có liên quan và phiên bản của nó.

#### **Signature:**

Phần này, được minh họa bởi S4, chứa chữ ký điện tử, được tạo bởi ứng dụng khách. Trường này được sử dụng để kiểm tra xem các chi tiết giao dịch chưa bị giả mạo, vì nó yêu cầu khóa riêng của ứng dụng này để tạo ra nó.

#### **Proposal:**

Trường này, được minh họa bởi P4, mã hóa các tham số đầu vào được cung cấp bởi một ứng dụng cho hợp đồng thông minh tạo ra bản cập nhật sổ cái được đề xuất. Khi hợp



đồng thông minh chạy, đề xuất này cung cấp một tập hợp các tham số đầu vào, kết hợp với trạng thái thế giới hiện tại, xác định trạng thái thế giới mới.

***Response:***

Phần này, được minh họa bởi R4, ghi lại các giá trị trước và sau của trạng thái thế giới, dưới dạng một bộ Đọc ghi (bộ RW). Nó có kết quả đầu ra của một hợp đồng thông minh và nếu giao dịch được xác thực thành công, nó sẽ được áp dụng vào sổ cái để cập nhật trạng thái thế giới.

***Endorsements:***

Như thể hiện trong E4, đây là danh sách các phản hồi giao dịch đã ký từ mỗi tổ chức được yêu cầu đủ để đáp ứng chính sách chứng thực. Bạn sẽ nhận thấy rằng, trong khi chỉ có một phản hồi giao dịch được bao gồm trong giao dịch, có nhiều chứng thực. Điều đó bởi vì mỗi chứng thực mã hóa một cách hiệu quả tổ chức của nó, phản ứng giao dịch cụ thể của nó - có nghĩa là không cần bao gồm bất kỳ phản hồi giao dịch nào không phù hợp với chứng thực vì nó sẽ bị từ chối là không hợp lệ và không cập nhật trạng thái thế giới.

***Ví dụ:<sup>11</sup>***

Chúng ta giải định rằng chúng ta đã hoàn thiện xong phần network, trong network đó có 1 Channel gồm Orderer và 2 tổ chức A, B. Mỗi client A, B đã được đăng ký một identity với Certificate Authority (CA) của tổ chức tương ứng.

Chaincode đã được cài trên các Peer và được instantiate trên Channel. Chaincode chứa logic để thực hiện một Transaction mua-bán củ cải. Endorsement Policy (Chính sách chứng thực) cũng đã được đặt ra cho chaincode này, với yêu cầu là 1 Transaction phải được endorsing (chứng thực) bởi cả 2 tổ chức.

---

<sup>11</sup> <https://viblo.asia/p/bai-4-transaction-trong-hyperledger-fabric-63vKjjDAK2R>

### Client A bắt đầu 1 Transaction



Hình 2.8. Client bắt đầu một Transaction

Client A gửi một yêu cầu muốn mua củ cải. Request này được gửi đến PeerA, PeerB, là các Peer của 2 tổ chức A, B. Do ở trên mình đã giả sử là Endorsement Policy quy định cả 2 tổ chức A và B phải chứng thực một Transaction nên bắt buộc yêu cầu phải được gửi đến cả 2 Peer như thế.

Tiếp theo, với sự hỗ trợ bởi SDK (NodeJS, Java, Python) một Transaction Proposal (đề xuất giao dịch) sẽ được xây dựng từ yêu cầu ở trên. Transaction Proposal là một yêu cầu để invoke (gọi) một function nào đó trong chaincode với các tham số đầu vào nhất định, function này có thể là write hoặc read Ledger.

SDK servers sẽ có nhiệm vụ đóng gói Transaction Proposal thành một "Properly Architected Format" (bộ đệm giao thức trên gRPC) và lấy thông-tin-số (hiệu đơn giản đây là khi client dùng identity của mình để tạo request thì hệ thống sẽ phân tích và lấy được thông-tin-số từ identity đó) của client để tạo signature (chữ ký) cho Transaction Proposal.

**Các Endorsing Peers sẽ kiểm tra chữ ký và thực thi Transaction với bản sao Ledger được lưu trên nó.**



*Hình 2.9. Các Endorsing Peers sẽ kiểm tra chữ ký và thực thi Transaction với bản sao Ledger được lưu trên nó*

Các Endorsing Peers sẽ xác nhận rằng Transaction Proposal được tạo đúng chuẩn và nó chưa từng được gửi đi trước đây (mục đích chống tấn công theo kiểu replay-attack) và chữ ký trên Transaction Proposal là đúng (sử dụng MSP để verify - xem lại bài trước), và người gửi Transaction - client A có quyền "write" trên Channel.

Các Endorsing Peers sẽ lấy đầu vào của Transaction Proposal làm đối số cho function trong chaincode mà được gọi đến. Chaincode sẽ được thực thi đối với bản sao Ledger lưu tại mỗi Peer để tạo ra một Transaction Value bao gồm giá trị trả về sau khi thực thi, cặp key-value cho một đối tượng cần được tạo hoặc cập nhật. Tại thời điểm này, Ledger sẽ bị "đóng băng" lại.

Transaction Value ở trên, cùng với chữ ký của Endorsing Peer đã tạo ra nó sẽ được gửi trở lại đến SDK dưới dạng một "proposal response"



Hình 2.10. Transaction Value cùng với chữ ký của Endorsing Peer đã tạo ra nó sẽ được gửi trở lại đến SDK dưới dạng một proposal response

### Đối chiếu các Proposal Response

Application sẽ kiểm tra chữ ký của các Endorsing Peers và so sánh các Proposal Response xem chúng có giống nhau không. Nếu đây chỉ là một hành động truy vấn thì Application sẽ không submit Transaction đến ordering service.

Nếu đây là một Transaction cập nhật Ledger, Application sẽ kiểm tra xem Endorsement Policy đã chỉ định trước có được thực hiện hay không trước khi submit lên Ordering Service (tức là cả PeerA và PeerB đều đã endorsing chưa).

Ngay cả khi Application "không kiểm tra Proposal Response" hoặc chuyển trực tiếp Transaction chưa được chứng thực đến Ordering Service, thì Endorsement Policy vẫn sẽ được thực thi bởi các Peer và ở giai đoạn commit validation (bước 5), do đó Endorsement Policy luôn được đảm bảo.



Hình 2.11. Đối chiếu các Proposal Response

### Client sẽ tập hợp các endorsement vào trong một Transaction

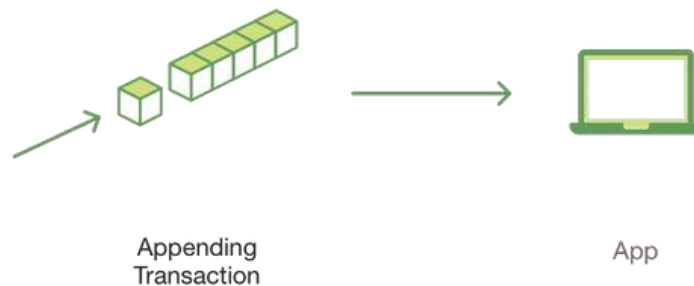
Application sẽ truyền Transaction Proposal và Transaction Response trong một "Transaction Message" đến Ordering Service. Transaction sẽ chứa cặp key-value cho đối tượng cần cập nhật, các chữ ký của các Endorsing Peer. Ordering Service sẽ không cần phải kiểm tra nội dung của Transaction, nó chỉ cần nhận tất cả các Transaction và sắp xếp nó theo thứ tự thời gian.



Hình 2.12. Client tập hợp các Endorsement vào trong một Transaction

### Transaction được validated

Các blocks chứa các Transactions sẽ được chuyển đến tất cả các Peers trên Channel. Các Transaction trong block sẽ được kiểm tra để đảm bảo rằng Endorsement Policy đã được thực hiện và Ledger được "đóng băng" trong lúc tạo cặp key-value. Từng Transaction trong block sẽ được gắn tag là hợp lệ hoặc không hợp lệ.

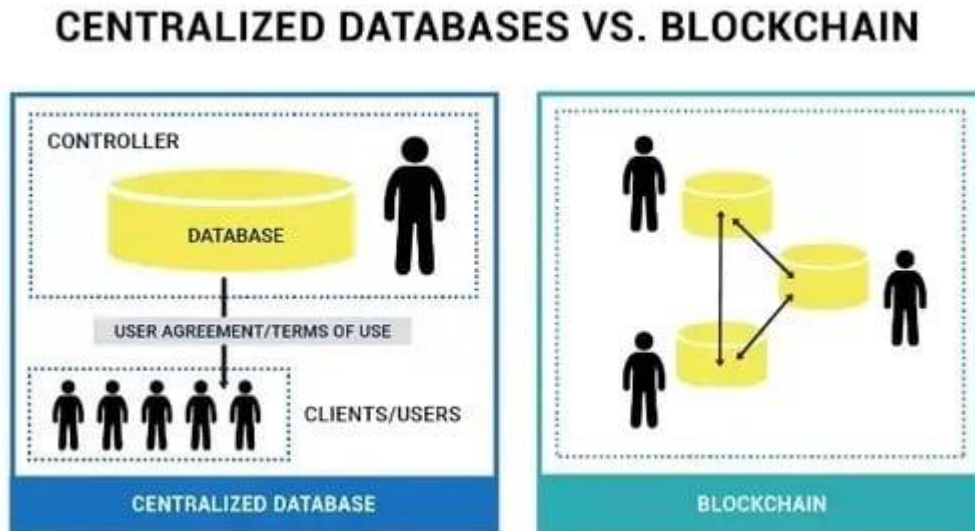


Hình 2.13. Từng Transaction trong block sẽ được gắn tag là hợp lệ hoặc không hợp lệ.

## Cập nhật Ledger

Mỗi Peer sẽ nối thêm Block vào chuỗi và với mỗi Transaction hợp lệ, các cặp key-value sẽ được thêm vào với Ledger. Một sự kiện được phát đi, để thông báo cho Client Application rằng Transaction đã được thêm vào chuỗi, cũng như thông báo về việc Transaction hợp lệ hay không hợp lệ.

### *1. So sánh Blockchain và Cơ sở dữ liệu truyền thống (Centralized Databases)<sup>12</sup>*



*Hình 2.14. Blockchain và cơ sở dữ liệu truyền thống*

Một cơ sở dữ liệu truyền thống (CSDL) là một cấu trúc dữ liệu được sử dụng để lưu trữ thông tin. Điều này bao gồm dữ liệu có thể được truy vấn để thu thập thông tin chuyên sâu về báo cáo có cấu trúc được sử dụng bởi các thực thể để hỗ trợ các quyết định kinh

<sup>12</sup> <https://intellipaat.com/blog/tutorial/blockchain-tutorial/blockchain-vs-database/>

doanh, tài chính và quản lý. Chính phủ cũng sử dụng cơ sở dữ liệu để lưu trữ các bộ dữ liệu lớn có quy mô lên tới hàng triệu hồ sơ.

Thông tin được lưu trữ trong cơ sở dữ liệu có thể được tổ chức bằng hệ thống quản lý cơ sở dữ liệu. Một cơ sở dữ liệu đơn giản được lưu trữ trong các phần tử dữ liệu được gọi là bảng. Các bảng chứa các trường, xác định loại bản ghi, lưu trữ dữ liệu được gọi là thuộc tính. Mỗi trường chứa các cột mô tả trường và các hàng xác định một bản ghi được lưu trữ trong cơ sở dữ liệu.

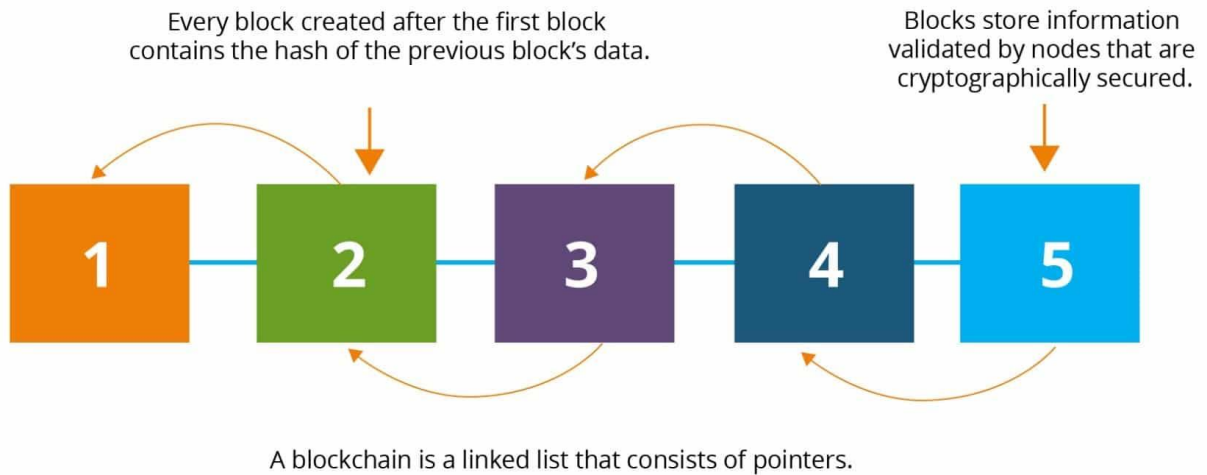
Sự khác biệt chính giữa Blockchain và CSDL truyền thống là tính phi tập trung. Trong khi các bản ghi trên CSDL đều được tập trung, mỗi người trên Blockchain có một bản sao bảo mật của tất cả các bản ghi và tất cả các thay đổi để mỗi người dùng có thể xem nguồn gốc của dữ liệu. Mỗi người tham gia duy trì một bản sao của các hồ sơ, công nghệ Blockchain sẽ ngay lập tức xác định và sửa bất kỳ thông tin không đáng tin cậy nào.

Blockchain là một cấu trúc dữ liệu mà người dùng chỉ có thể ghi, chứ không thể xóa hay chỉnh sửa dữ liệu được. Trong đó, các mục mới sẽ được thêm vào cuối sổ cái. Mỗi khối mới được gắn vào Blockchain bằng cách liên kết với khối trước đó.

Khi dữ liệu có thể tự động xác định và sửa lỗi dựa trên logic kinh doanh được mã hóa (hợp đồng thông minh) và sự đồng thuận, về cơ bản, người tham gia có thể tin tưởng vào nó. Khi hai doanh nghiệp làm việc cùng nhau, họ gần như không bao giờ chia sẻ một cơ sở dữ liệu với một bộ hồ sơ, bởi vì cơ sở dữ liệu đang được duy trì và cập nhật bởi quản trị viên cơ sở dữ liệu (DBA).

Một Blockchain lưu trữ thông tin trong các khối có kích thước đồng đều. Mỗi khối chứa thông tin băm từ khối trước để cung cấp bảo mật mật mã. Băm sử dụng SHA256 là hàm băm một chiều. Thông tin băm này là dữ liệu và chữ ký số từ khối trước đó và băm của các khối trước đó quay trở lại khối đầu tiên được tạo ra trong chuỗi khối được gọi là khối genesis của nhóm Drake. Thông tin đó được chạy qua hàm băm sau đó trở đến địa chỉ

của khối trước đó. Cấu trúc dữ liệu Blockchain là một ví dụ về Cây Merkle, được sử dụng như một cách hiệu quả để xác minh dữ liệu.



*Hình 2.15. Một Blockchain là một danh sách liên kết bao gồm con trỏ*

Trong cơ sở dữ liệu truyền thống, dữ liệu có thể dễ dàng bị xóa hay sửa đổi. Thông thường, sẽ có những quản trị viên có thể thay đổi bất kỳ phần nào của dữ liệu hoặc cấu trúc của nó.

<b>Blockchain</b>	<b>Database</b>
Phi tập trung	Tập trung
Không cần cấp phép	Cần cấp phép
Không cần Administrator	Cần Administrator
Kiến trúc kiểu P2P (Peer to Peer)	Kiến trúc máy Client/Server

*Bảng 2.1. So sánh Blockchain và cơ sở dữ liệu*



### ***m. Ứng dụng của Blockchain***

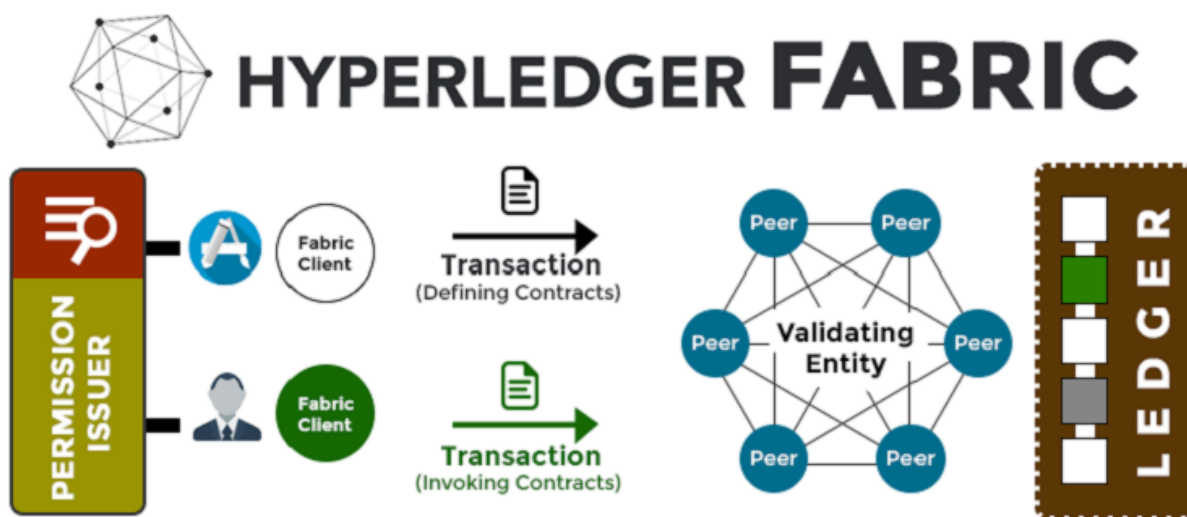
Hợp đồng thông minh được xây dựng trên nền tảng công nghệ Blockchain có thể loại bỏ sự cần thiết của nhân lực trung gian. Lưới năng lượng có thể sử dụng công nghệ này nhằm tăng cường an ninh mạng. Các nhà cung cấp vận tải hàng không vũ trụ có thể tìm đến Blockchain như một khoản đầu tư tiềm năng nhằm theo dõi chuỗi cung ứng và tăng hiệu quả kinh doanh.

Blockchain được ứng dụng thích hợp nhất trong các lĩnh vực:

- Giao dịch tiền
- Giao dịch có giá trị lớn
- Xác minh dữ liệu đáng tin cậy (danh tính, danh tiếng, uy tín, tính toàn vẹn, v.v.)
- Xác minh Public Key
- Các ứng dụng phân tán (Dapps)
- Hệ thống bầu cử
- Y tế
- Giáo dục...

## 2.1.2. Tìm hiểu về Hyperledger Blockchain và Hyperledger Fabric

### a. Giới thiệu về Hyperledger<sup>13</sup>



Hình 2.16. Giới thiệu Hyperledger Fabric

Hyperledger là dự án mã nguồn mở do Linux Foundation khởi xướng (12-2015), được xây dựng nhằm hướng đến việc thúc đẩy ứng dụng công nghệ Blockchain phát triển mạnh mẽ hơn nữa trong việc lưu trữ và xác nhận các giao dịch để phục vụ cho nhiều nền công nghiệp khác nhau. Với Hyperledger, Linux Foundation nhằm mục đích tạo ra một môi trường để các nhà phát triển phần mềm phối hợp với nhau cùng xây dựng nên các framework Blockchain, tăng cường sự cộng tác phát triển giữa các tổ chức toàn cầu một cách xuyên suốt và hiệu quả, cung cấp cơ sở hạ tầng trung lập, cởi mở được hỗ trợ bởi cộng đồng công nghệ, giáo dục công chúng về cơ hội thị trường cho công nghệ Blockchain, tạo ra sổ cái phân tán để hỗ trợ các giao dịch kinh doanh. Hyperledger được

<sup>13</sup> <https://www.hyperledger.org/projects/fabric>

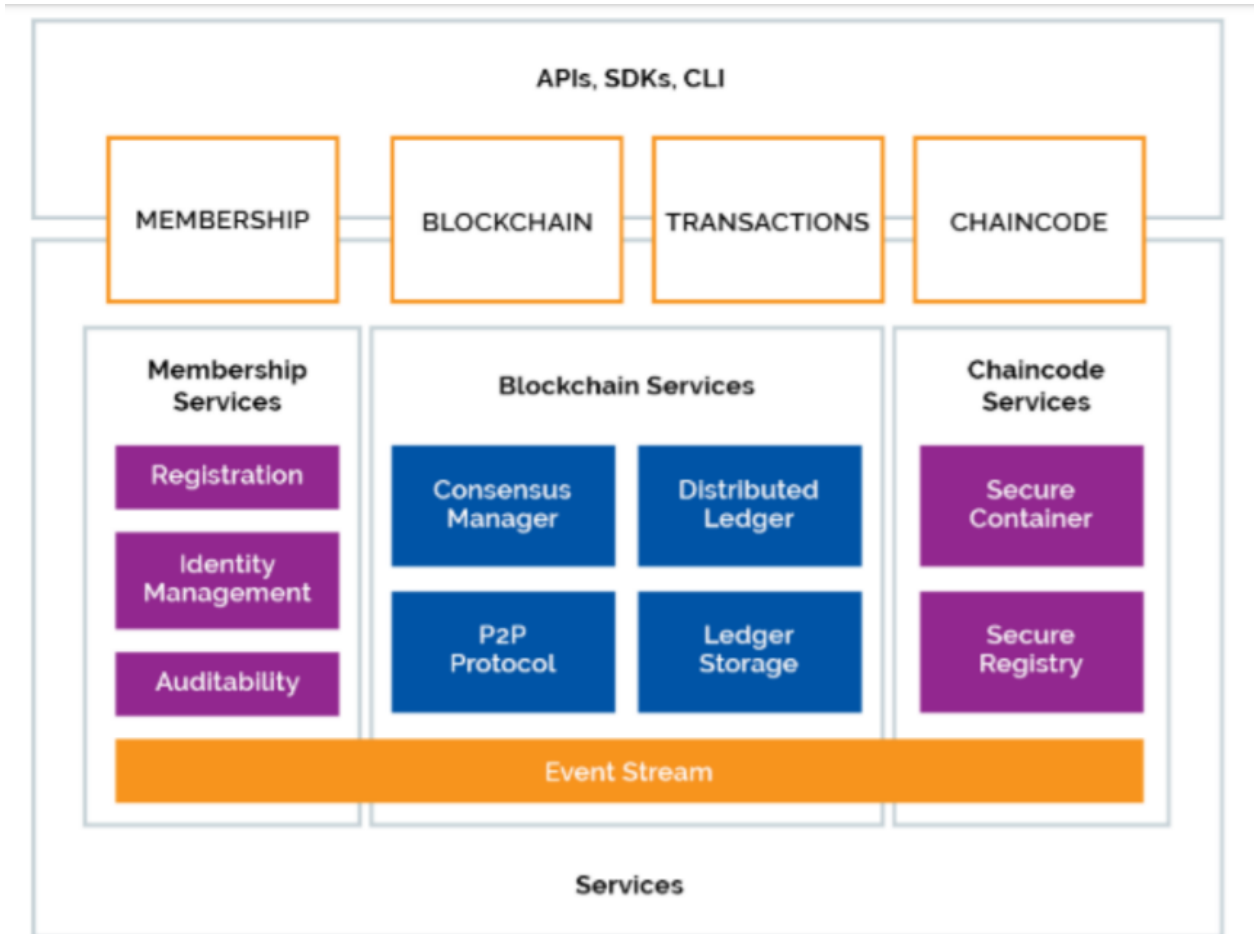
<https://viblo.asia/p/bai-2-gioi-thieu-hyperledger-fabric-cau-truc-cua-mot-mang-hyperledger-fabric-LzD5djE0ZjY>

đóng góp bởi rất nhiều hãng công nghệ lớn, nổi bật nhất là IBM. Khác với các nền tảng Blockchain khác như Bitcoin

**Hyperledger Fabric** là dự án đầu tiên được xây dựng bởi IBM và Digital Asset, một nền tảng cho phép các components như thuật toán đồng thuận, membership service có thể dễ dàng plug and play. Ở phiên bản 1.0 Chaincode của Hyperledger hay còn gọi là Smart Contract (hợp đồng thông minh) được viết bằng ngôn ngữ Go. Với bản nâng cấp 1.1, Chaincode có thể được viết bằng Javascript. Chaincode tương tự như hợp đồng thông minh của Ethereum, nó hoạt động trên docker container. Một điểm mới và khác biệt được Hyperledger mang tới đó là khái niệm Channels. Channels là một mạng con private nằm trong một mạng chung hay cộng đồng chung. Channels chứa hai hoặc nhiều members với mục đích thực hiện các giao dịch mang tính bảo mật và riêng tư. Chỉ các thành viên của Channels này mới được thấy các giao dịch của họ.

**Hyperledger Composer:** cung cấp các công cụ để xây dựng mạng Blockchain, cho phép tạo ra các mạng Blockchain theo từng nhu cầu cụ thể. Hỗ trợ phát triển và tương tác với smartcontract (trong Hyperledger được gọi là chaincode).

*b. Kiến trúc của Hyperledger<sup>14</sup>*



*Hình 2.17. Kiến trúc của Hyperledger*

**Membership:** cung cấp các dịch vụ quản lý danh tính, quyền riêng tư, bảo mật và kiểm toán trên mạng.

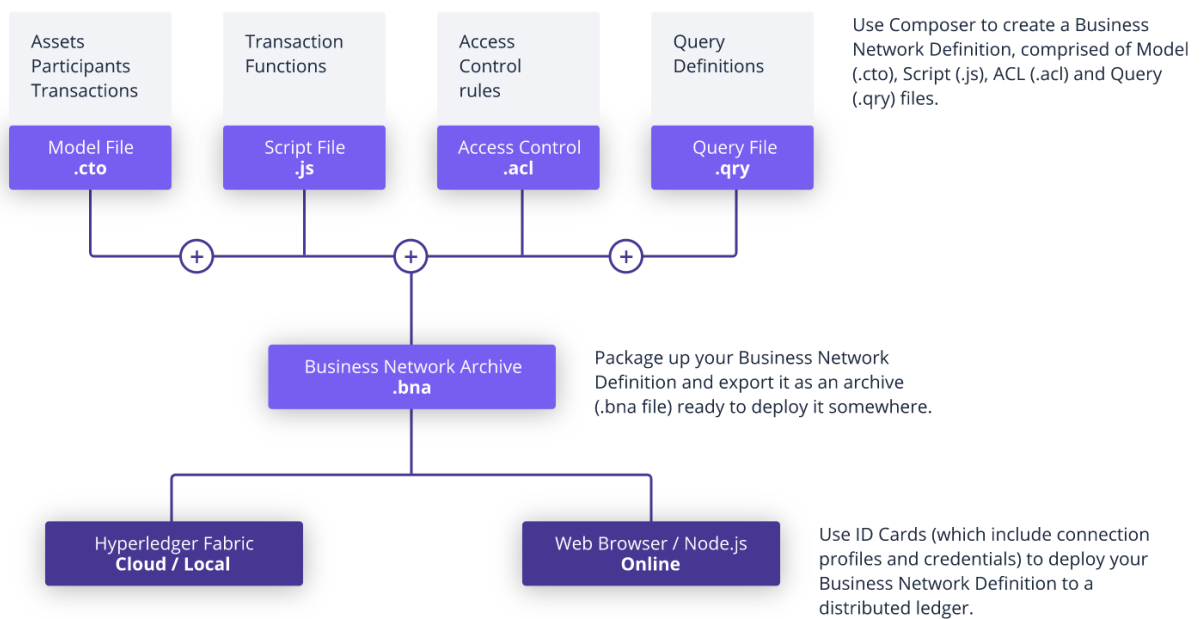
**Blockchain Services:** các dịch vụ Blockchain bao gồm ba thành phần chính: trình quản lý đồng thuận, giao thức Peer-to-Peer (P2P) và sổ cái phân phối (Distributed Ledger & Ledger Storage).

<sup>14</sup> <https://hyperledger-fabric.readthedocs.io/en/release-1.4/whatis.html>

**Transactions:** Các giao dịch được thực hiện và lưu trữ trên Blockchain.

**Chaincode:** Hyperledger lưu các hợp đồng thông minh dưới dạng các chaincode (ngôn ngữ lập trình để phát triển smartcontract), chaincode được sử dụng trong Hyperledger viết bằng Golang hoặc JavaScript. Có thể hiểu chaincode là một ứng dụng phân tán, chạy trên các nút xác nhận tính hợp lệ và được đóng gói trong các docker.

*c. Các loại tập tin trong mạng Hyperledger<sup>15</sup>*



*Hình 2.18. Các loại tập tin trong Hyperledger*

**Model File (.cto):** tập tin này dùng để mô tả các participants, assets và các Transactions trong mạng Blockchain. Định nghĩa ra các thuộc tính. Hyperledger Composer sẽ cung cấp một loại ngôn ngữ là model language để mô tả chúng.

<sup>15</sup> <https://hyperledger.github.io/composer/v0.19/introduction/introduction>

**Script File (.js):** thường được tạo ra bởi các lập trình viên, đây là tập tin JavaScript thực hiện các business logic trong mạng Blockchain.

**Access Control File (.acl):** các tập tin này dùng để quy định những participants nào trong mạng Blockchain được quyền truy cập vào loại tài sản nào. Ví dụ: admin thì có quyền xem toàn bộ các Transaction, nhưng user thì chỉ được xem Transaction của chính user đó. Hyperledger cung cấp Access Control Language (ACL) để định nghĩa các permission này.

**Query File (.qry):** định nghĩa các truy vấn trong Blockchain. Tương tự như các truy vấn CSDL.

Sử dụng Hyperledger Composer để định nghĩa ra mạng Blockchain, bao gồm Model file (.cto), Script file (.js), Access Control file (.acl) và Query file (.qry).

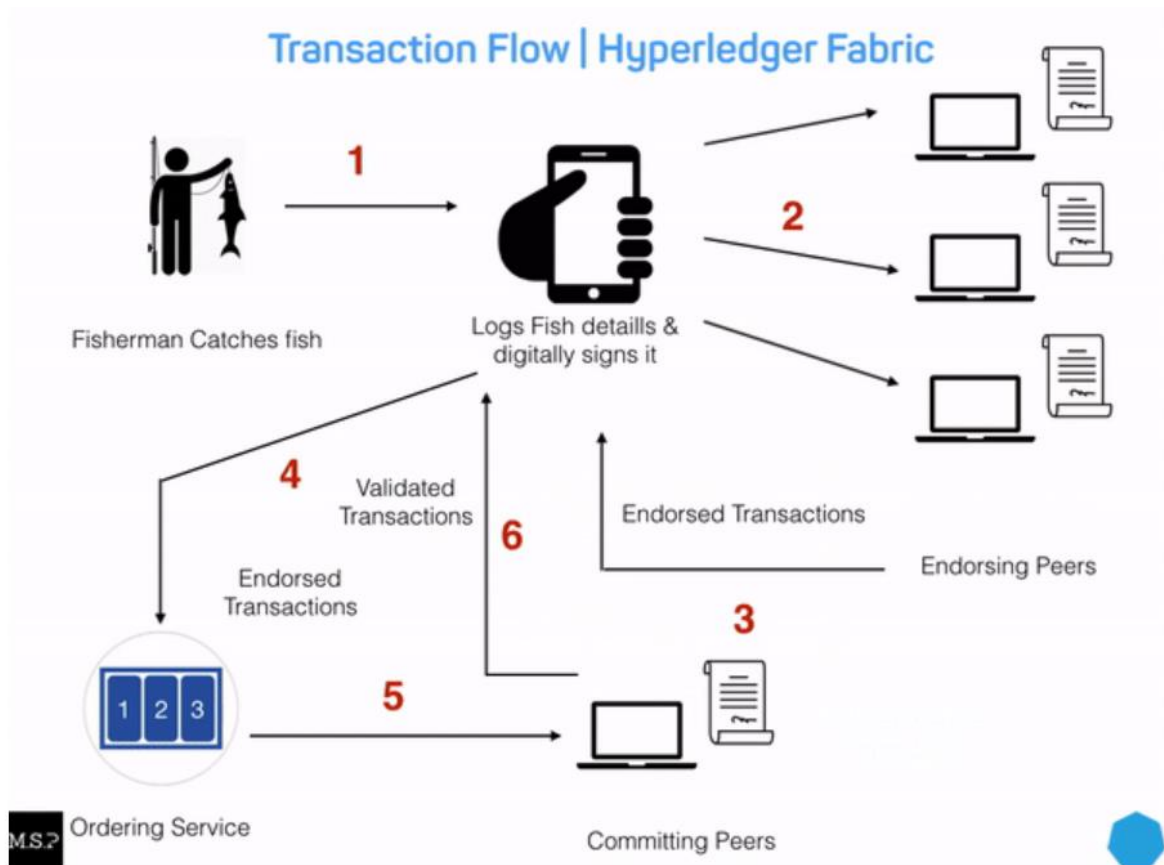
Sau đó, đóng gói mạng Blockchain đã định nghĩa thành tập tin Business Network Archive (.bna) để có thể triển khai mạng Blockchain này đến sổ cái phân tán ở trên Hyperledger Fabric (chạy trên Cloud/ local) hoặc trên Web Browser/ Node.js (chạy online) bằng cách sử dụng thẻ ID Cards (thẻ này bao gồm thông tin đăng nhập và các hồ sơ kết nối).

#### ***d. Các Role trong Hyperledger Fabric Network***

- **Client:** một người đề xuất các giao dịch trong mạng Blockchain
- **Peers:** là các nút ngang hàng có nhiệm vụ duy trì trạng thái của mạng Blockchain và chứa một bản sao của sổ cái (Ledger). Có 2 loại Peer trong mạng Blockchain:
- + **Endorser:** mô phỏng và chứng thực các giao dịch, các Peer này nhận các yêu cầu giao dịch từ các ứng dụng của người dùng, sau đó thực hiện:
  - Xác thực giao dịch: Kiểm tra chi tiết các Certificate của người yêu cầu thực hiện giao dịch

- Thực thi Chaincode: mô phỏng kết quả của giao dịch nhưng không cập nhật vào sổ cái
- + **Committers:** xác minh xác nhận và xác thực kết quả giao dịch, trước khi cam kết giao dịch với Blockchain.
- **Ordering Service:** chấp nhận các giao dịch được chứng thực, sắp xếp chúng thành một khối và phân chúng vào các Committing Peer.

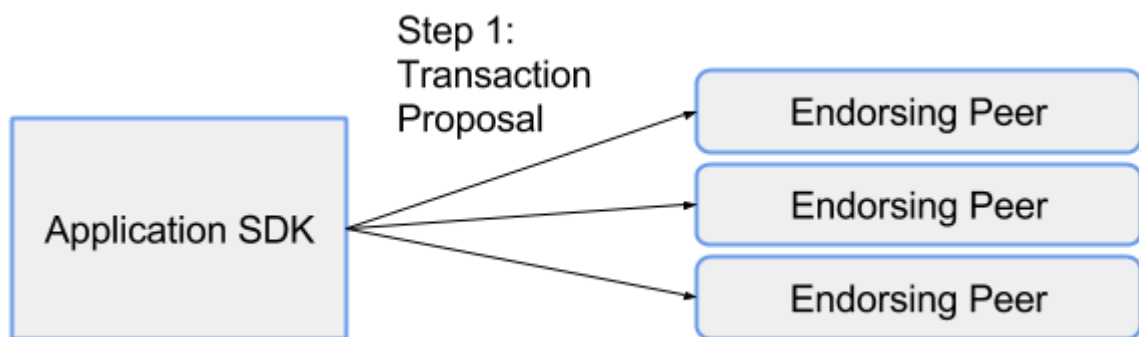
*e. Luồng của Transaction<sup>16</sup>*



Hình 2.19. Transaction Flow trong Hyperledger Fabric

<sup>16</sup> <https://medium.com/swlh/hyperledger-chapter-8-what-is-hyperledger-fabric-chaincode-a74778dff2ae>  
<https://medium.com/swlh/hyperledger-chapter-6-hyperledger-fabric-components-technical-context-767985f605dd>

**Bước 1:** Trong mạng Hyperledger Fabric, các giao dịch được khởi tạo bởi các ứng dụng của người dùng nhằm gửi đi các đề xuất giao dịch hay nói cách khác là đề xuất giao dịch cho các Endorsing Peer.

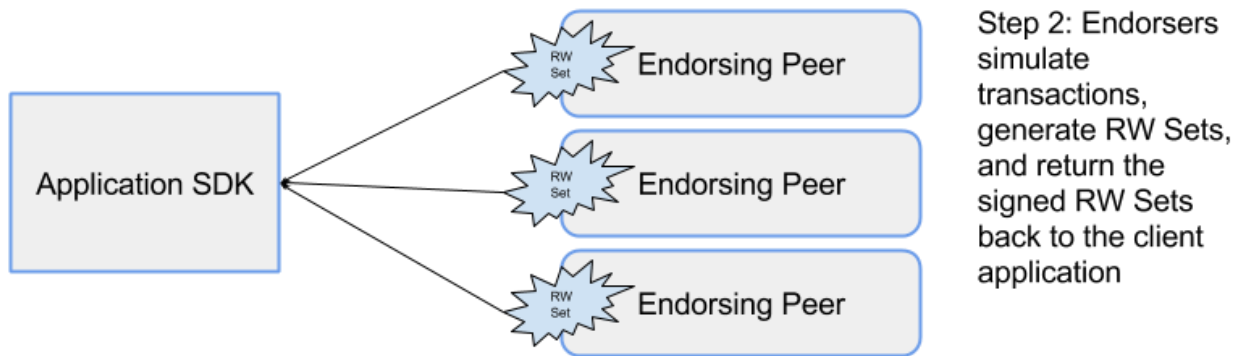


*Hình 2.20. Đề xuất giao dịch trong mạng Hyperledger Fabric*

Các ứng dụng của người dùng (ứng dụng khách), cho phép mọi người giao tiếp với mạng Blockchain.

**Bước 2:** Mỗi Endorsing Peer làm nhiệm vụ mô phỏng giao dịch được đề xuất nhưng không cập nhật vào trong sổ cái (ledger). Các Endorsing Peer sẽ nắm bắt tập hợp các dữ liệu Read và Written, được gọi là RW Sets. Các RW Sets này nắm bắt những gì đã đọc từ trạng thái hiện tại của Blockchain trong khi mô phỏng giao dịch, cũng như những gì sẽ được thêm vào trong Blockchain khi giao dịch được thực hiện. Các RW sets này sau đó được ký bởi các Endorsing Peer và được trả về cho ứng dụng khách để sử dụng cho các bước tiếp theo của quá trình làm việc.



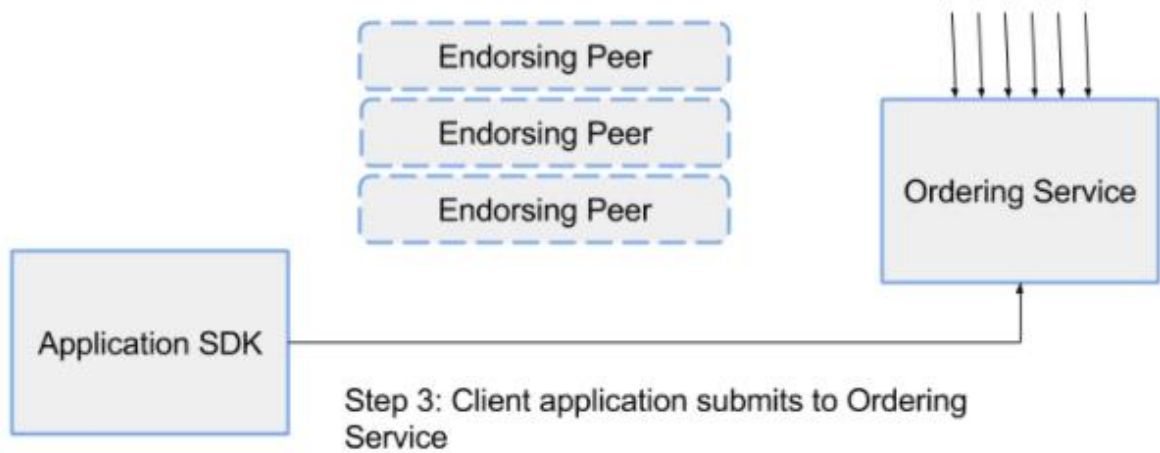


*Hình 2.21. Các Endorsers nắm bắt tập hợp các dữ liệu Read và Written, tạo RW Sets, và trả về cho ứng dụng khách sau khi được ký bởi Endorsing Peer*

Endorsing Peer phải giữ các hợp đồng thông minh (Chaincode) để mô phỏng các đề xuất giao dịch.

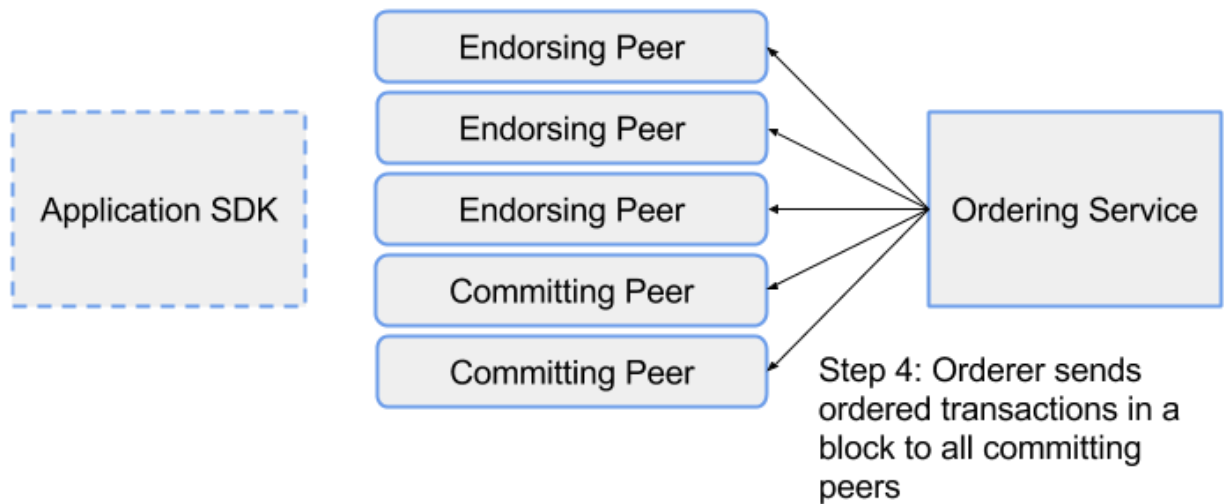
Chúng thực giao dịch: là việc xác nhận bằng cách đã ký đối với kết quả của giao dịch được mô phỏng. Phương thức xác nhận giao dịch phụ thuộc vào chính sách chứng thực được chỉ định khi chaincode được triển khai. Một ví dụ về chính sách chứng thực là: “Phần lớn các Endorsing Peer phải chứng thực các giao dịch”. Do chính sách chứng thực được chỉ định cho một chaincode cụ thể, nên các kênh (Channel) khác nhau có thể có các chính sách chứng thực khác nhau.

**Bước 3:** Tiếp theo, ứng dụng sẽ gửi giao dịch được chứng thực và RW Sets đến Ordering Service. Quá trình Ordering xảy ra trên mạng, song song với các giao dịch được chứng thực và các RW Sets được gửi bởi các ứng dụng khác.



Hình 2.22. Ứng dụng gửi giao dịch được chứng thực và RW Sets đến Ordering Service

**Bước 4:** Ordering Service nhận các giao dịch được chứng thực và các RW Sets, sắp xếp thông tin này thành một khối và phân phối khối đó cho tất cả các Committing Peer.

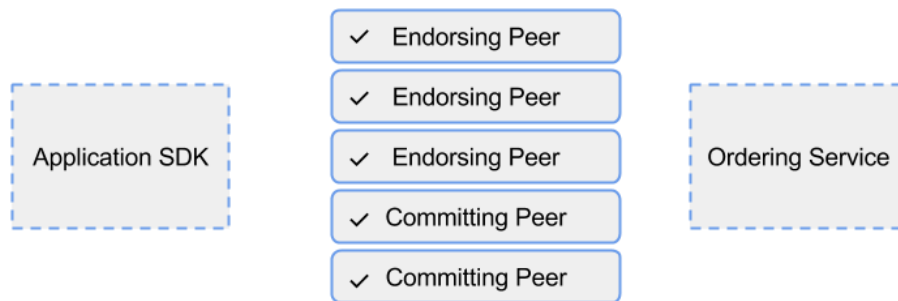


Hình 2.23. Orderer gửi các giao dịch đã được sắp xếp thông tin thành một khối đến tất cả các Committing Peer

Ordering Service được tạo thành từ một nhóm các Orderer, các nút này không xử lý các giao dịch, hợp đồng thông minh hay duy trì sổ cái. Ordering Service chấp nhận các giao dịch được chứng thực và chỉ định thứ tự mà các giao dịch đó sẽ được đưa lên Blockchain. Mặc định trong Hyperledger Fabric phiên bản 1.0 thì Ordering Service là Kafka.

Trong mạng Blockchain, các giao dịch phải được ghi vào sổ cái theo thứ tự thống nhất. Thứ tự của các giao dịch phải được thiết lập để đảm bảo rằng các cập nhật cho Blockchain là hợp lệ khi chúng được xác nhận trong mạng. Không giống như Bitcoin Blockchain, nơi mà việc Ordering xảy ra thông qua giải các câu đố phức tạp hoặc thông qua việc đào, Hyperledger Fabric cho phép các tổ chức lựa chọn cơ chế Ordering phù hợp với mạng của mình. Tính mô-đun và tính linh hoạt này làm cho Hyperledger Fabric có lợi thế vô cùng to lớn đối với các ứng dụng của doanh nghiệp. Hyperledger Fabric cung cấp 3 cơ chế Ordering: SOLO, Kafka và Simplified Byzantine Fault Tolerance.

**Bước 5:** Committing Peer xác nhận giao dịch bằng cách kiểm tra để đảm bảo rằng các RW Sets vẫn khớp với trạng thái hiện tại của Blockchain. Cụ thể, dữ liệu Read tồn tại khi Endorser mô phỏng giao dịch giống hệt với trạng thái hiện tại của Blockchain. Khi Committing Peer xác nhận giao dịch, giao dịch được ghi vào trong sổ cái và trạng thái của Blockchain sẽ được cập nhật với dữ liệu Write từ RW Sets



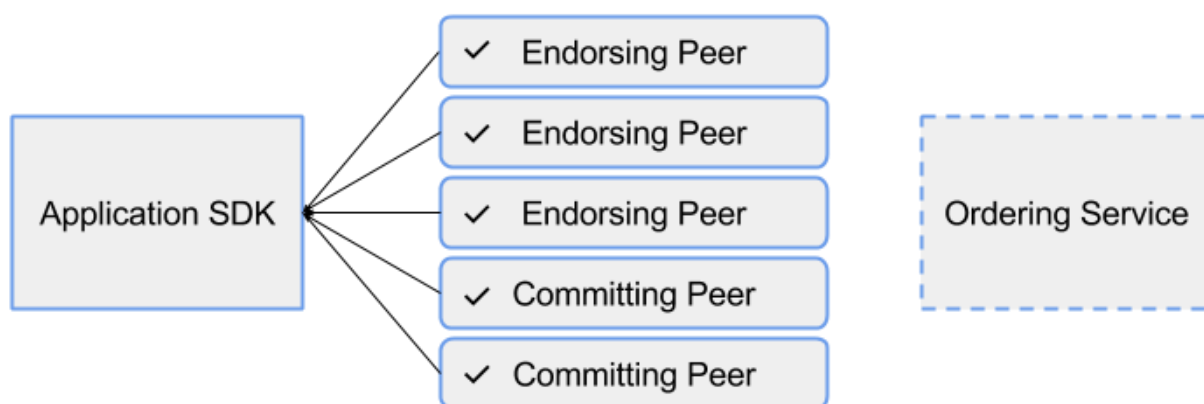
Step 5: Committing peers validate each transaction in the block

*Hình 2.24. Committing Peers xác nhận mỗi Transaction trong khối có khớp với Blockchain không*

Nếu giao dịch thất bại, nghĩa là nếu Committing Peer thấy rằng RW Sets không khớp với trạng thái hiện tại của Blockchain, giao dịch được đặt trong một khối vẫn sẽ được bao gồm trong gói đó, nhưng nó sẽ được đánh dấu là không hợp lệ và trạng thái của Blockchain sẽ không được cập nhật.

Các Committing Peer chịu trách nhiệm thêm các khối chứa các giao dịch vào sổ cái và cập nhật trạng thái cho toàn bộ mạng.

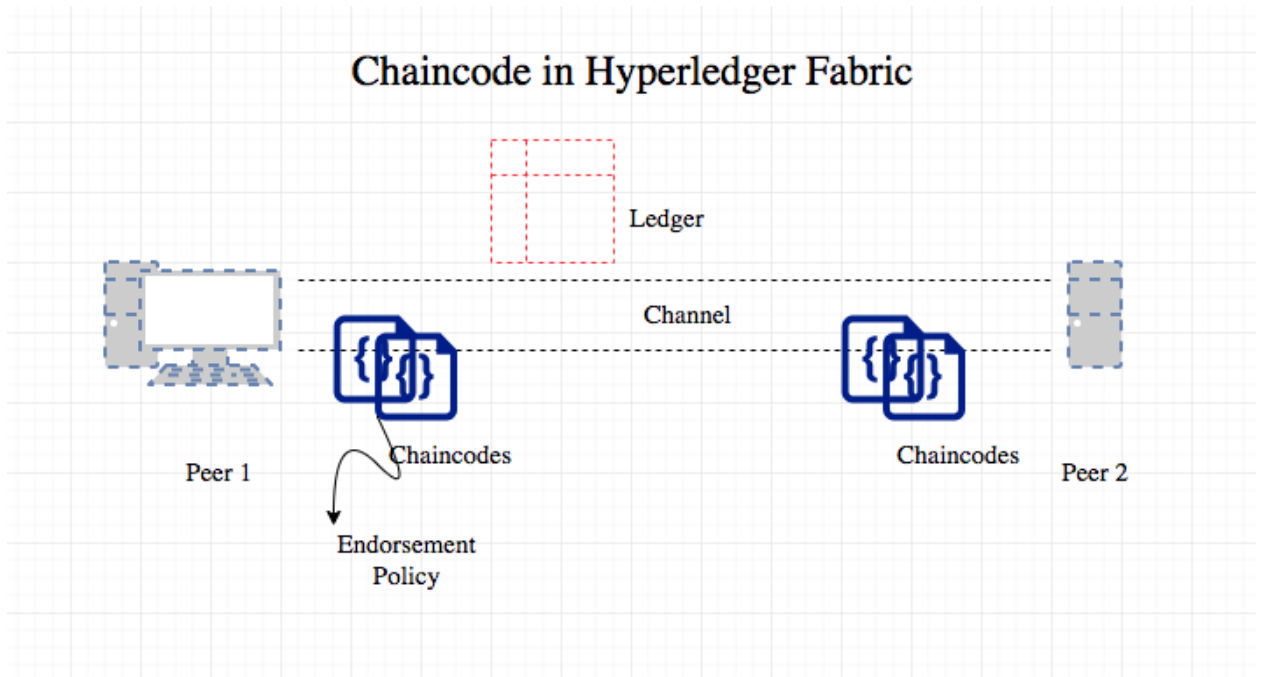
**Bước 6:** Cuối cùng, các Committing Peer thông báo không đồng bộ cho ứng dụng khách về sự thành công hay thất bại của giao dịch. Các ứng dụng sẽ được thông báo bởi mỗi Committing Peer.



Step 6: Committing peers asynchronously notify the Application of the results of the transaction.

*Hình 2.25. Committing Peer thông báo không đồng bộ cho ứng dụng khách về sự thành công hay thất bại của giao dịch.*

*f. Hyperledger Fabric Chaincode*



*Hình 2.26. Chaincode trong Hyperledger Fabric*

Trong Hyperledger Fabric, Chaincode là Smart Contract, chạy trên các Peer và tạo ra các Transaction. Hơn nữa, nó cho phép người dùng tạo các giao dịch trong sổ cái chia sẻ mạng Hyperledger Fabric và cập nhật trạng thái của tài sản trên Network. Hiện tại, Chaincode được viết bằng Golang hoặc JavaScript

Các ứng dụng tương tác với sổ cái Blockchain thông qua Chaincode. Do đó, Chaincode cần phải được cài đặt trên mọi thiết bị ngang hàng sẽ xác nhận giao dịch và được khởi tạo trên Channel.

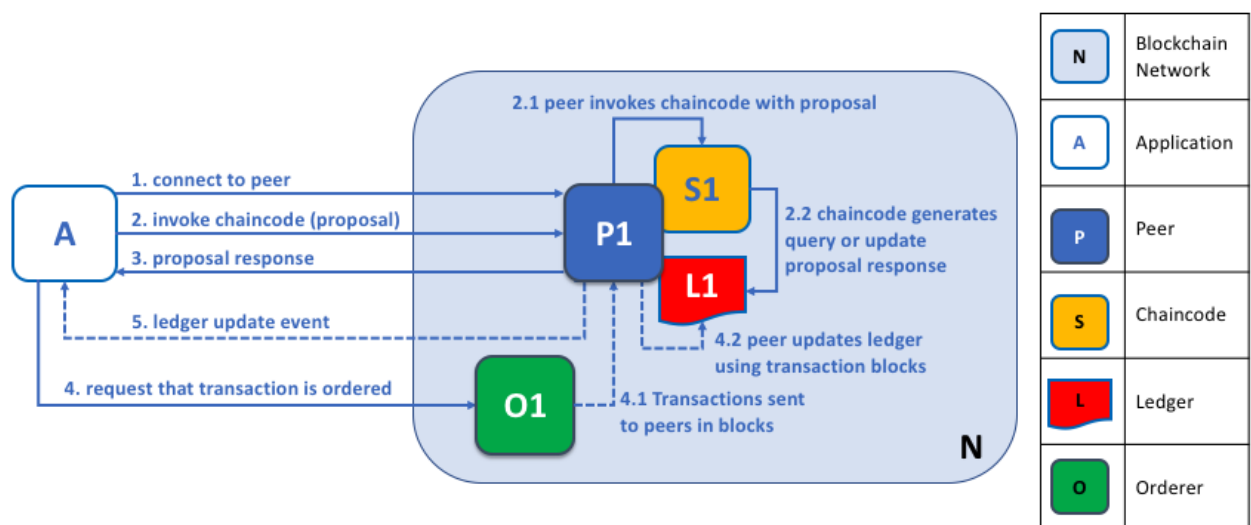
Khi tạo ra chaincode, có hai phương thức mà người dùng cần phải thực hiện:

- **Init:** được gọi khi chaincode nhận được giao dịch khởi tạo (instantiate) hoặc nâng cấp (upgrade). Đây là nơi bạn sẽ khởi tạo bất kỳ trạng thái ứng dụng nào.

- **Invoke:** được gọi khi nhận được giao dịch gọi (invoke) để xử lý bất kỳ đề xuất giao dịch nào.

Chaincode phải được cài đặt bằng lệnh cài đặt mã chuỗi ngang hàng và được khởi tạo bằng lệnh khởi tạo Chaincode ngang hàng trước khi mã chuỗi có thể được gọi. Sau đó, các Transaction có thể được tạo bằng cách sử dụng lệnh gọi mã chuỗi ngang hàng hoặc lệnh truy vấn chuỗi mã ngang hàng.

#### g. Các thành phần trong mạng Hyperledger Fabric



Hình 2.27. Các thành phần trong mạng Hyperledger Fabric

**Network:** là tập hợp các nút ngang hàng (Peer) xử lý dữ liệu tạo thành mạng Blockchain. Mạng này chịu trách nhiệm duy trì một sổ cái được nhân rộng trong mạng, mang tính nhất quán

**Channel:** cho phép tập hợp các Peer và ứng dụng giao tiếp với nhau trong mạng Blockchain

**Ledger:** chứa trạng thái hiện tại của mạng Blockchain và chuỗi các yêu cầu giao dịch. Các thông tin trong sổ cái Ledger chỉ được phép thêm, không thể xóa hay sửa được.

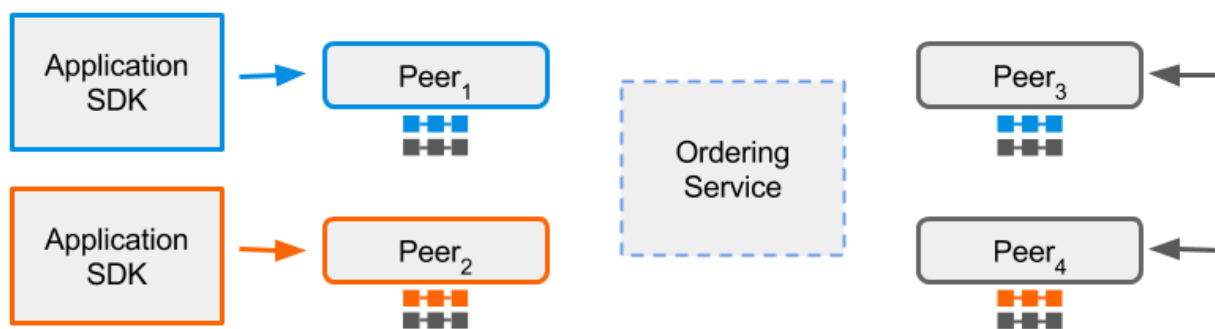
**Chaincode (Smart Contract):** bao gồm các định nghĩa về tài sản (asset) và các business logic để sửa đổi thông tin của tài sản (asset) đó.

**Orderer:** là tập hợp các nút yêu cầu các giao dịch thành một khối.

#### *h. Channel trong Hyperledger Fabric*

Các kênh (Channel) cho phép các tổ chức sử dụng cùng một mạng, trong khi vẫn duy trì sự riêng biệt giữa nhiều Blockchain. Chỉ các thành viên trong cùng một kênh mà giao dịch được thực hiện mới có thể thấy cụ thể chi tiết của giao dịch đó.

Cơ chế này hoạt động bằng cách ủy thác các giao dịch cho các sổ cái khác nhau. Chỉ các thành viên của kênh mới được tham gia vào quá trình đồng thuận, trong khi các thành viên khác của mạng không thấy các giao dịch trên kênh đó.



*Hình 2.28. Mô tả Channel trong Hyperledger Fabric*

Hình trên cho thấy có ba kênh riêng biệt là xanh dương, cam và xám. Mỗi kênh đều có ứng dụng, sổ cái và các Peer riêng.

Các Peer này có thể thuộc về nhiều mạng hoặc nhiều kênh. Các Peer tham gia vào nhiều kênh mô phỏng và xác nhận các giao dịch với các sổ cái khác nhau. Ordering Service là giống nhau trên bất kỳ mạng hoặc kênh nào.

#### *i. State Database*

Trạng thái hiện tại của dữ liệu trong Hyperledger, Blockchain biểu thị các giá trị mới nhất cho tất cả các tài sản trong sổ cái. Do đó, trạng thái hiện tại đại diện cho tất cả các giao dịch đã được xác nhận trên Channel, nên đôi khi nó được gọi là trạng thái của cả mạng Blockchain.

Các chaincode sẽ thực hiện các giao dịch đối với trạng thái hiện tại của dữ liệu. Để làm cho các tương tác của chaincode trở nên cực kỳ hiệu quả, các cặp key/value mới nhất cho mỗi tài sản sẽ được lưu trữ trong State Database.

Các tùy chọn cơ sở dữ liệu trạng thái bao gồm LevelDB và CouchDB. LevelDB là cơ sở dữ liệu trạng thái khóa-giá trị mặc định được nhúng trong quy trình ngang hàng. CouchDB là một cơ sở dữ liệu trạng thái bên ngoài thay thế tùy chọn. Giống như kho lưu trữ khóa-giá trị LevelDB, CouchDB có thể lưu trữ bất kỳ dữ liệu nhị phân nào được mô hình hóa trong mã chuỗi (chức năng đính kèm CouchDB được sử dụng nội bộ cho dữ liệu nhị phân không phải JSON). Nhưng với tư cách là một kho lưu trữ tài liệu JSON, CouchDB còn cho phép truy vấn phong phú đối với dữ liệu mã chuỗi, khi các giá trị mã chuỗi (ví dụ: tài sản) được mô hình hóa dưới dạng dữ liệu JSON.

Cả LevelDB và CouchDB đều hỗ trợ các hoạt động mã chuỗi lõi như nhận và đặt khóa (tài sản) và truy vấn dựa trên các khóa. Các khóa có thể được truy vấn theo phạm vi và các khóa tổng hợp có thể được mô hình hóa để cho phép các truy vấn tương đương đối với nhiều tham số. Ví dụ: khóa tổng hợp của chủ sở hữu, `property_id` có thể được sử dụng để truy vấn tất cả các tài sản thuộc sở hữu của một thực thể nhất định. Các truy vấn dựa trên khóa này có thể được sử dụng cho các truy vấn chỉ đọc đối với sổ cái, cũng như trong các giao dịch cập nhật sổ cái.

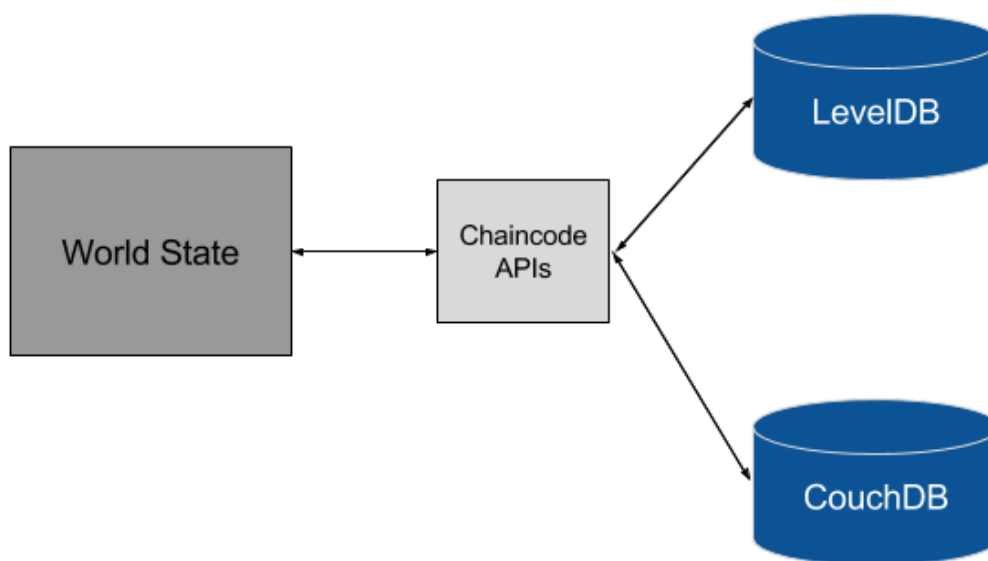


State Database chỉ đơn giản là chế độ xem được lập chỉ mục trong chuỗi các giao dịch đã được xác nhận. Do đó, nó có thể được tái tạo từ chuỗi bất cứ lúc nào. State Database sẽ tự động được phục hồi (hoặc được tạo ra nếu cần thiết) khi các Peer khởi động, trước khi các giao dịch mới được chấp nhận. State Database mặc định là LevelDB, ngoài ra có thể thay thế bằng CouchDB.

LevelDB là state database mặc định chứa các cặp key/value trong Hyperledger Fabric và chỉ lưu trữ các cặp key/value này

CouchDB là một thay thế cho LevelDB. Không giống như LevelDB, CouchDB chứa các JSON objects. CouchDB là duy nhất ở chỗ nó hỗ trợ khóa, phạm vi khóa và các truy vấn đầy đủ dữ liệu.

LevelDB và CouchDB trong Hyperledger Fabric là rất giống nhau về cấu trúc và chức năng. Cả hai đều hỗ trợ các hoạt động của core chaincode, chẳng hạn như nhận và thiết lập các khóa tài sản chính và truy vấn dựa trên các khóa này. Với cả LevelDB và CouchDB, các khóa có thể được truy vấn theo phạm vi và các khóa tổng hợp có thể được mô hình hóa để cho phép truy vấn tương ứng với nhiều tham số. Nhưng với tư cách là một nơi lưu trữ các tài liệu JSON, CouchDB cũng cho phép truy vấn đa dạng đối với các dữ liệu chaincode, khi các giá trị trong chaincode (ví dụ như các tài sản) được mô hình hóa dưới dạng dữ liệu JSON.



*Hình 2.29. State Database trong Hyperledger Fabric*

## **2.2. Lý do sử dụng Hyperledger**

Hyperledger Fabric đang cung cấp một nền tảng blockchain được thiết kế để cho phép trao đổi một tài sản hoặc trạng thái của một tài sản được đồng ý, duy trì và xem bởi tất cả các bên trong một nhóm được phép. Một đặc điểm chính của Hyperledger Fabric là tài sản được xác định bằng kỹ thuật số, với tất cả những người tham gia chỉ đơn giản là đồng ý về đại diện/ đặc tính của nó. Do đó, Hyperledger Fabric có thể hỗ trợ nhiều loại tài sản; từ hữu hình (bất động sản và phân cứng) đến vô hình (hợp đồng và sở hữu trí tuệ).

Khi một tổ chức được cấp quyền truy cập vào mạng blockchain, thì tổ chức đó có khả năng tạo và duy trì kênh riêng với các thành viên được chỉ định khác. Ví dụ, hãy giả sử có bốn tổ chức kinh doanh đồ trang sức. Họ có thể quyết định sử dụng Hyperledger Fabric vì họ tin tưởng lẫn nhau, nhưng không đến mức vô điều kiện. Tất cả họ có thể đồng ý về logic kinh doanh để giao dịch đồ trang sức và tất cả có thể duy trì một sổ cái toàn cầu để xem tình trạng hiện tại của thị trường trang sức của họ (gọi đây là kênh liên

minh). Ngoài ra, hai hoặc nhiều tổ chức trong số này có thể quyết định hình thành một blockchain riêng thay thế cho một trao đổi nhất định mà họ muốn giữ bí mật (ví dụ: giá X cho số lượng Y của tài sản Z). Họ có thể thực hiện giao dịch này mà không ảnh hưởng đến kênh tập đoàn rộng hơn của họ, hoặc, nếu muốn, kênh riêng này có thể phát một số mức dữ liệu tham chiếu đến kênh liên danh của họ.

	<b>Public Blockchain</b>	<b>Private Blockchain</b>	
Danh tính	Ẩn danh Truy cập tự do không hạn chế	Xác định danh tính Truy cập hạn chế	Xác định danh tính Truy cập hạn chế
Đọc ghi dữ liệu	Đọc hoặc ghi dữ liệu không giới hạn	Phân quyền đọc hoặc ghi dữ liệu	Phân quyền đọc hoặc ghi dữ liệu
Tốc độ	Chậm	Nhanh	Nhanh
Chi phí	Rẻ	Đắt	Rẻ
Giao dịch	Công khai với cả mạng lưới	Không công khai	Không công khai

*Bảng 2.2. So sánh Public Blockchain, Private Blockchain và Permissioned Blockchain*

Việc xây dựng một hệ thống mà Bác sĩ và Bệnh nhân cần danh tính rõ ràng thì ta thấy là việc sử dụng Private Blockchain là hoàn toàn hợp lý. Ngoài ra mục tiêu của luận văn là đọc ghi dữ liệu bệnh án của bệnh nhân.

Ngoài ra, do Hyperledger Fabric hỗ trợ phân quyền và đọc ghi rõ ràng, hỗ trợ SmartContract (hợp đồng thông minh) sẽ đáp ứng được mục tiêu ban đầu đã đề ra.

	<i>Bitcoin</i>	<i>Ethereum</i>	<i>Hyperledger</i>
<i>Phân quyền</i>	Không	Không	Có
<i>Giả danh</i>	Có	Không	Không
<i>Sổ cái (Ledger)</i>	Có	Có	Có
<i>Hợp đồng thông minh (Smart Contract)</i>	Không	Có	Có
<i>Chi phí cho mỗi giao dịch</i>	Có	Có	Không
<i>Cơ chế đồng thuận</i>	Sổ cái	Sổ cái	Giao dịch
<i>Đồng tiền điện tử</i>	Có	Có	Không

*Bảng 2.3. So sánh Bitcoin, Ethereum và Hyperledger*

### **2.3. Kết chương**

Trong chương này, khóa luận tập trung vào tìm hiểu về công nghệ mới Blockchain và các nền tảng công nghệ Blockchain mà tác giả định hướng ứng dụng vào đề tài của khóa luận.

Tác giả định hướng xây dựng ứng dụng dựa việc kết hợp trên hai nền tảng của công nghệ Blockchain là Hyperledger Fabric và Hyperledger Composer.

Thông qua đây, tác giả nắm một số kiến thức như sau:

- Kiến thức tổng quan về Blockchain: giới thiệu sơ lược, phân loại Blockchain, phương pháp lưu trữ trong Blockchain và vai trò, lợi ích mà Blockchain mang lại.

- Kiến thức về Hyperledger Fabric và Hyperledger Composer để sử dụng cho các mô hình cần triển khai.

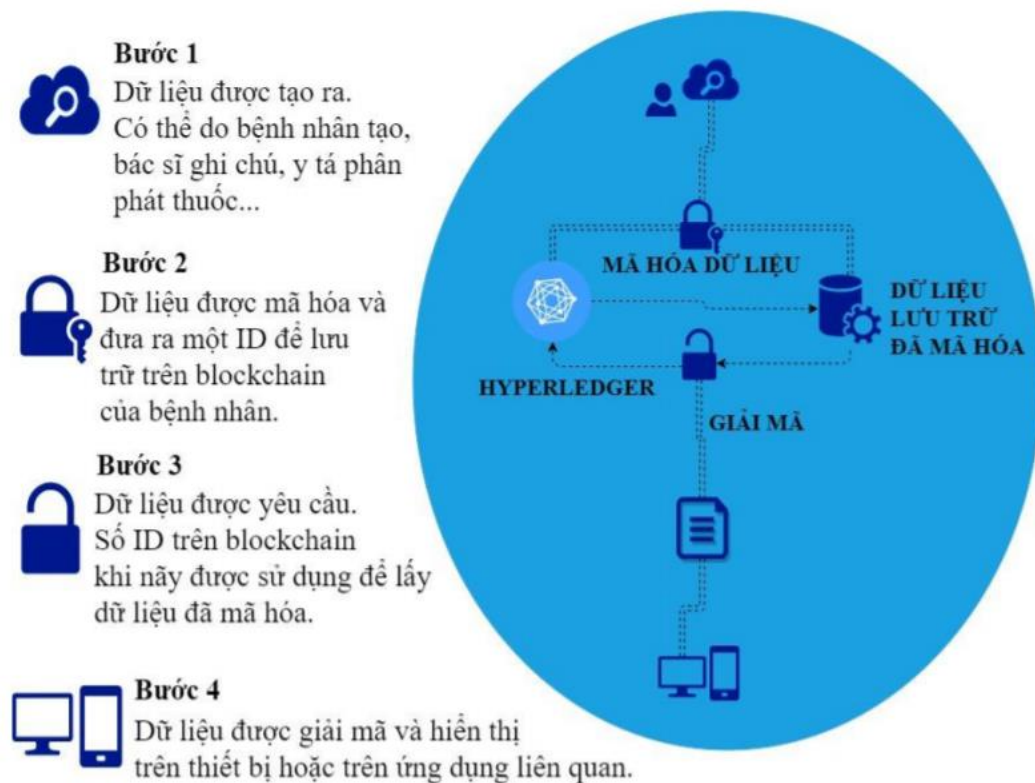


Trong luận văn này, tôi xây dựng ứng dụng quản lý bệnh án đơn giản của một tổ chức bệnh viện trong mạng lưới bệnh viện. Các chức năng gồm có:

- Đăng ký bệnh nhân mới
- Đăng nhập
- Yêu cầu và xem thông tin bệnh nhân, bác sĩ
- Yêu cầu xem và cập nhật dữ liệu bệnh án của bệnh nhân.
- Cấp quyền cho các yêu cầu trên.
- Thu hồi các quyền trên.

### 3.2. Các quy trình trong hệ thống

#### 3.2.1. Quy trình hoạt động của hệ thống và mã hóa dữ liệu



Hình 3.2. Quy trình hoạt động của hệ thống và mã hóa dữ liệu

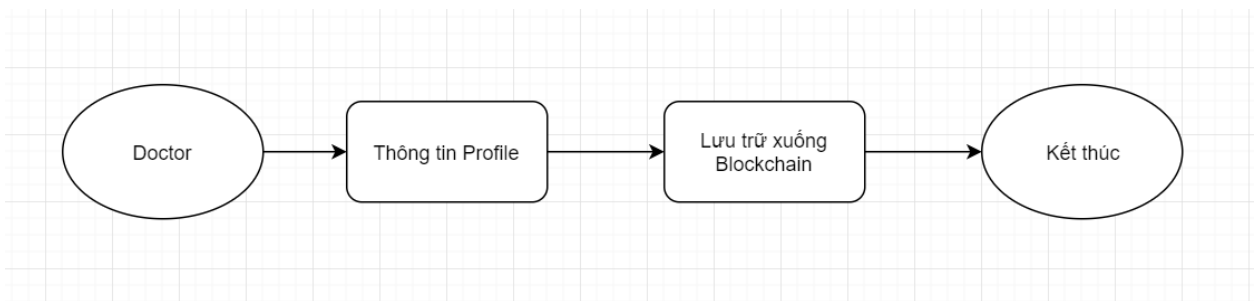
Quy trình hoạt động trong một giao dịch của hệ thống và quá trình nhập dữ liệu diễn ra như sau:

- Dữ liệu được tạo ra trên thiết bị di động của bệnh nhân, bác sĩ có thể ghi thông tin chuẩn đoán trong quá trình khám chữa bệnh hoặc quá trình được sĩ kê thuốc cho bệnh nhân lên Blockchain.
- Dữ liệu được mã hóa và số nhận dạng hay còn gọi là ID được lưu trữ trên Blockchain của bệnh nhân.
- Khi dữ liệu được yêu cầu, số ID trên Blockchain được sử dụng để lấy dữ liệu đã được mã hóa.
- Cuối cùng, dữ liệu được giải mã và sau đó hiển thị lên trên thiết bị hoặc ứng dụng có liên quan

### 3.2.2. Cập nhật thông tin Profile

Khi bắt đầu tiến hành quy trình khám chữa bệnh, Patient (bệnh nhân) và Doctor (bác sĩ) sẽ tiến hành cập nhật thông tin cá nhân của mình để đối tác của mình có thể tham khảo cũng như có tiếp nhận hoặc gửi yêu cầu trở thành đối tác của mình hay không. Đối tác có thể thêm/ chấp nhận/ từ chối yêu cầu của đối phương.

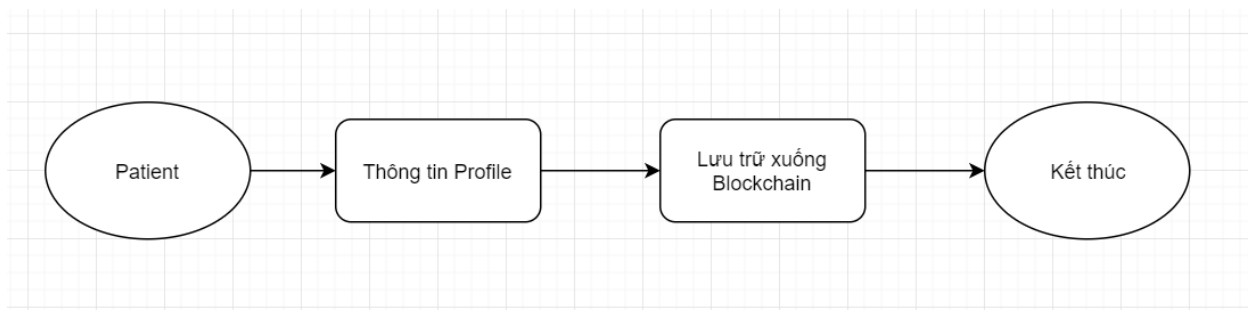
**Doctor (bác sĩ):**



*Hình 3.3. Quy trình cập nhật thông tin Doctor Profile*



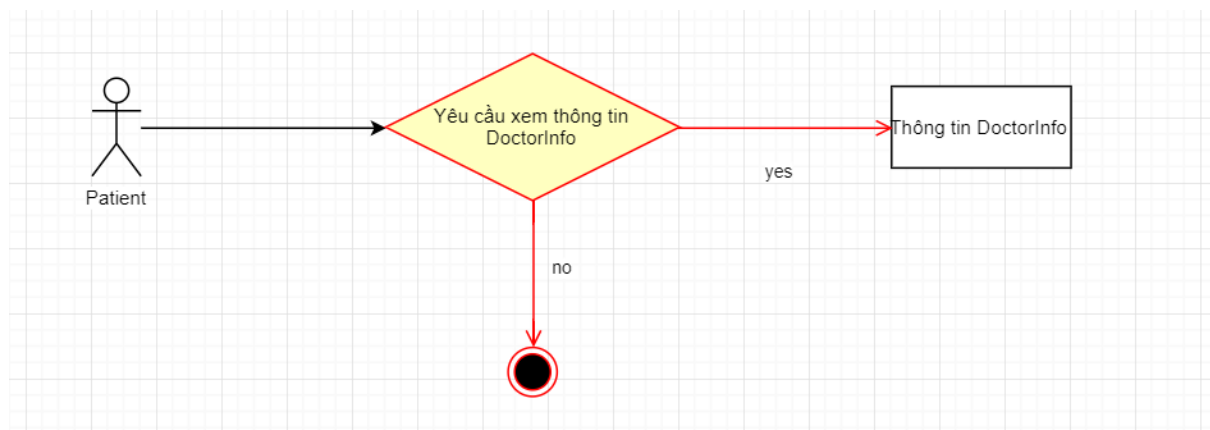
***Patient (bệnh nhân):***



*Hình 3.4. Quy trình cập nhật thông tin Patient Profile*

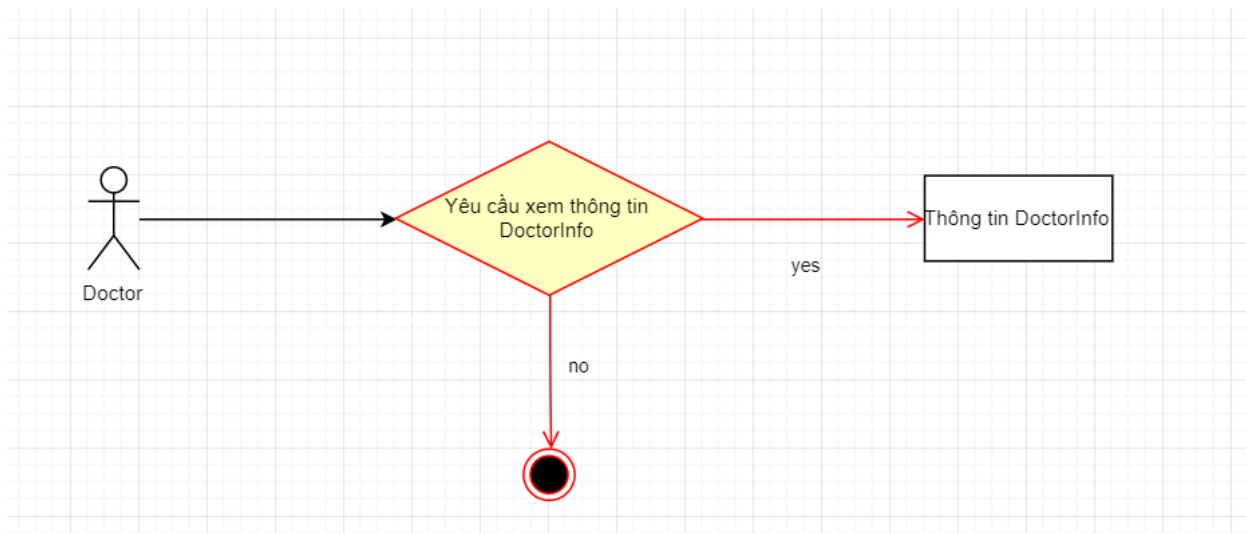
### **3.2.3. Các quy trình cấp quyền**

***a. Patient yêu cầu cấp quyền xem thông tin cá nhân bác sĩ (DoctorInfo)***



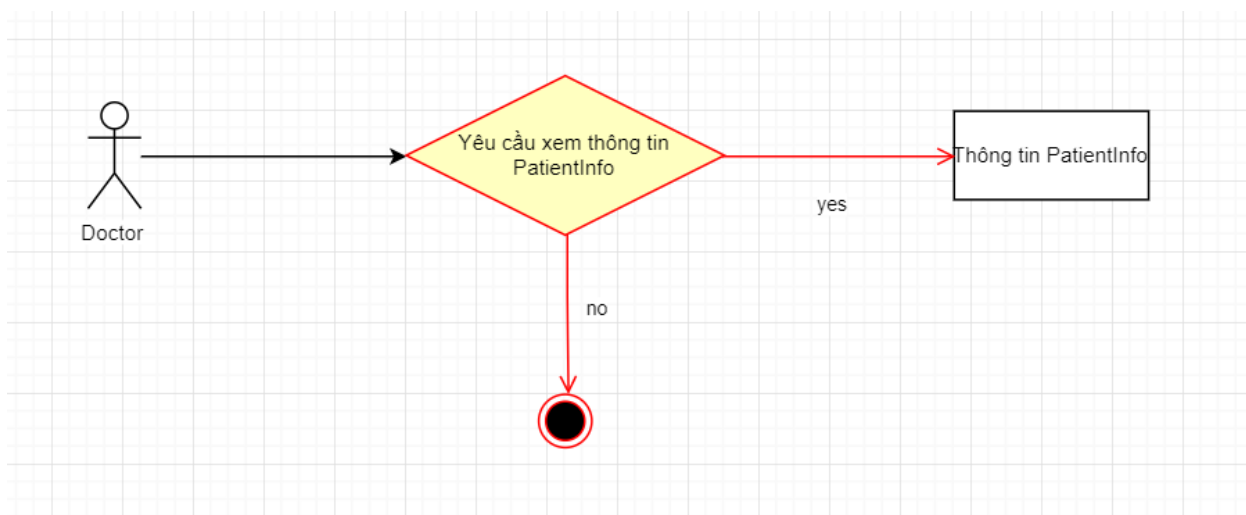
*Hình 3.5. Patient yêu cầu cấp quyền xem thông tin cá nhân của Doctor*

**b. Doctor yêu cầu cấp quyền xem thông tin cá nhân bác sĩ (DoctorInfo)**



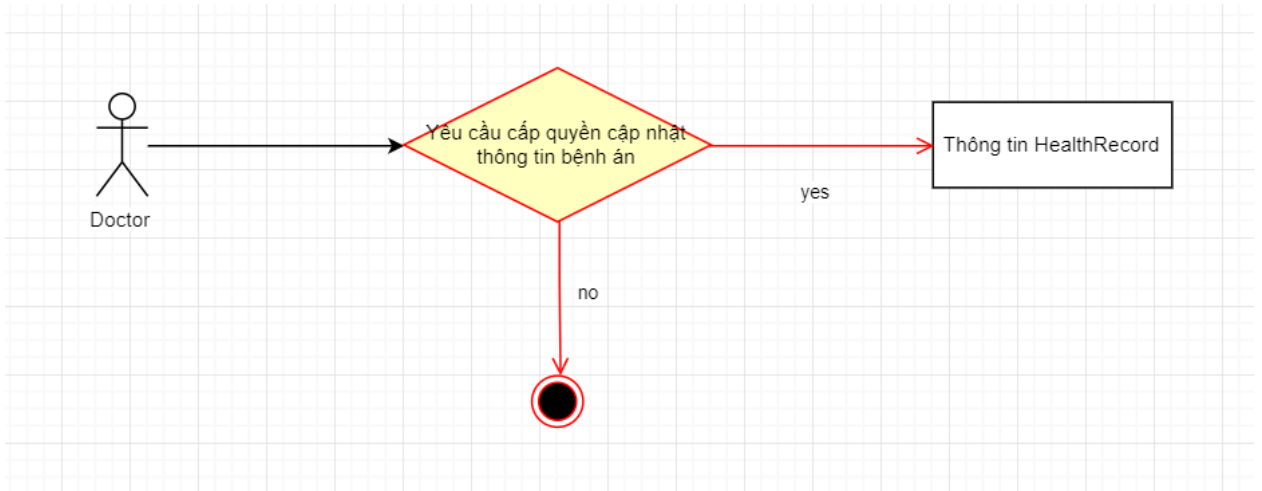
*Hình 3.6. Doctor yêu cầu xem thông tin cá nhân của Doctor khác*

**c. Doctor yêu cầu cấp quyền xem thông tin cá nhân bệnh nhân (PatientInfo)**



*Hình 3.7. Doctor yêu cầu cấp quyền xem thông tin cá nhân của Patient*

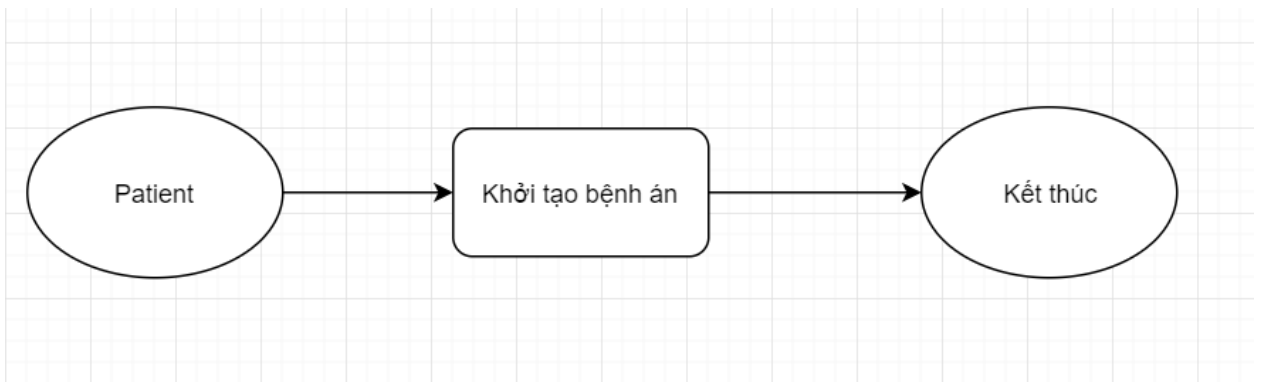
**d. Doctor yêu cầu cấp quyền cập nhật thông tin bệnh án (HealthRecord)**



Hình 3.8. Doctor yêu cầu cấp quyền cập nhật thông tin bệnh án

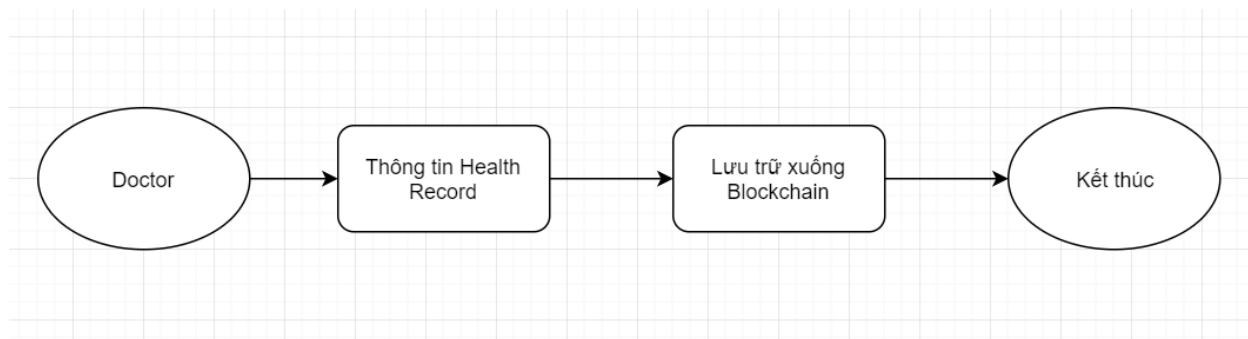
**3.2.4. Khởi tạo và cập nhật thông tin bệnh án Health Record**

**Patient (bệnh nhân): Khởi tạo thông tin bệnh án Health Record**



Hình 3.9. Patient khởi tạo bệnh án

**Doctor (bác sĩ): Cập nhật thông tin bệnh án Health Record**



*Hình 3.10. Doctor cập nhật thông tin bệnh án*

### **3.3. Kết chương**

Từ các kiến thức đã tìm hiểu được, tác giả đưa ra mô hình cho việc giải quyết bài toán ban đầu. Đồng thời tác giả cũng vẽ lại quy trình một vài thành phần của mô hình. Thông qua đó đưa ra cái nhìn từ tổng quan đến chi tiết mô phương pháp mà tác giả đưa ra để giải quyết vấn đề bài toán.

Tóm lại, chương này giúp bạn đọc có cách nhìn tổng quan và chi tiết về mô hình mà tác giả đề ra để giải quyết vấn đề bài toán.

## Chương 4. CÀI ĐẶT VÀ THỬ NGHIỆM HỆ THỐNG

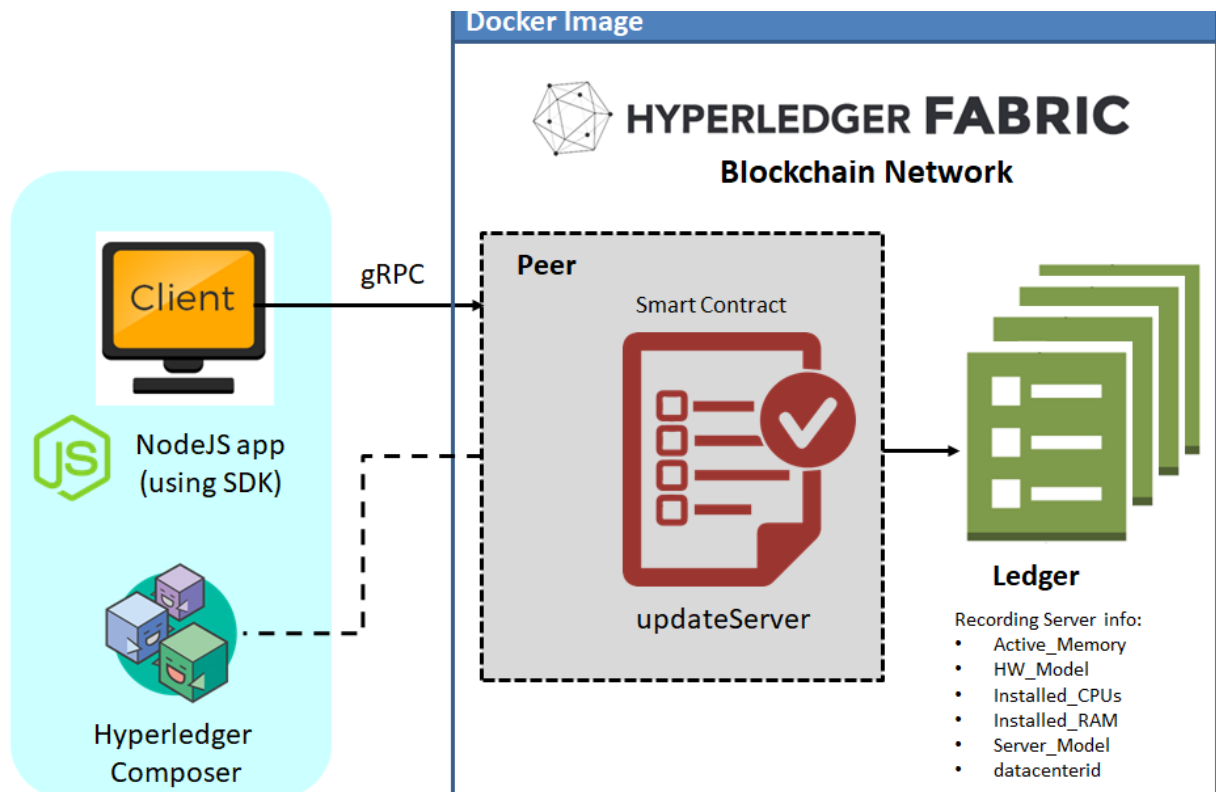
Chương này nói về cách cài đặt và thử nghiệm hệ thống đang phát triển. Tác giả nói về môi trường và công cụ xây dựng hệ thống, sơ đồ USE-CASE và cách xây dựng hệ thống. Ngoài ra, tác giả còn cung cấp hình ảnh về giao diện ứng dụng đang được tác giả phát triển.

### 4.1. Môi trường và công cụ xây dựng

Như đã được nêu ở các chương trước, tôi sử dụng kết hợp 2 bộ công cụ là Hyperledger Fabric, Hyperledger Composer để triển khai mô hình của tác giả. Mỗi bộ công cụ chịu trách nhiệm một phần trong hệ thống:

- **Hyperledger Fabric:** Đây là phần cốt lõi chạy nền bên dưới chịu trách nhiệm khởi tạo Blockchain để lưu thông tin của hệ thống.
- **Hyperledger Composer:** Công cụ này cung cấp môi trường để tạo dựng các hợp đồng thông minh, xử lý các yêu cầu từ tầng ứng dụng khi yêu cầu đến Blockchain.

Kiến trúc hệ thống:



Hình 4.1. Xây dựng hệ thống với NodeJS, Hyperledger Composer và Hyperledger Fabric

## 4.2. Sơ đồ USE-CASE

### 4.2.1. USE-CASE tổng quát



Hình 4.2. Sơ đồ USE-CASE tổng quát

**Mô tả:** Hệ thống có 2 Actor là Patient (bệnh nhân) và Doctor (bác sĩ). Để sử dụng các chức năng, cả 2 actor cần phải đăng nhập.

***Dòng sự kiện:***

Đăng nhập: Đăng nhập vào hệ thống để sử dụng các chức năng liên quan.

Quản lý thông tin cá nhân: Xem/ Cập nhật thông tin cá nhân

Quản lý bệnh án: Xem thông tin bệnh án

Cập nhật bệnh án: Cập nhật thông tin bệnh án

Xem danh sách Bác sĩ: xem thông tin danh sách có liên quan tới Bác sĩ

Xem danh sách Bệnh nhân: xem thông tin danh sách có liên quan tới Bệnh nhân

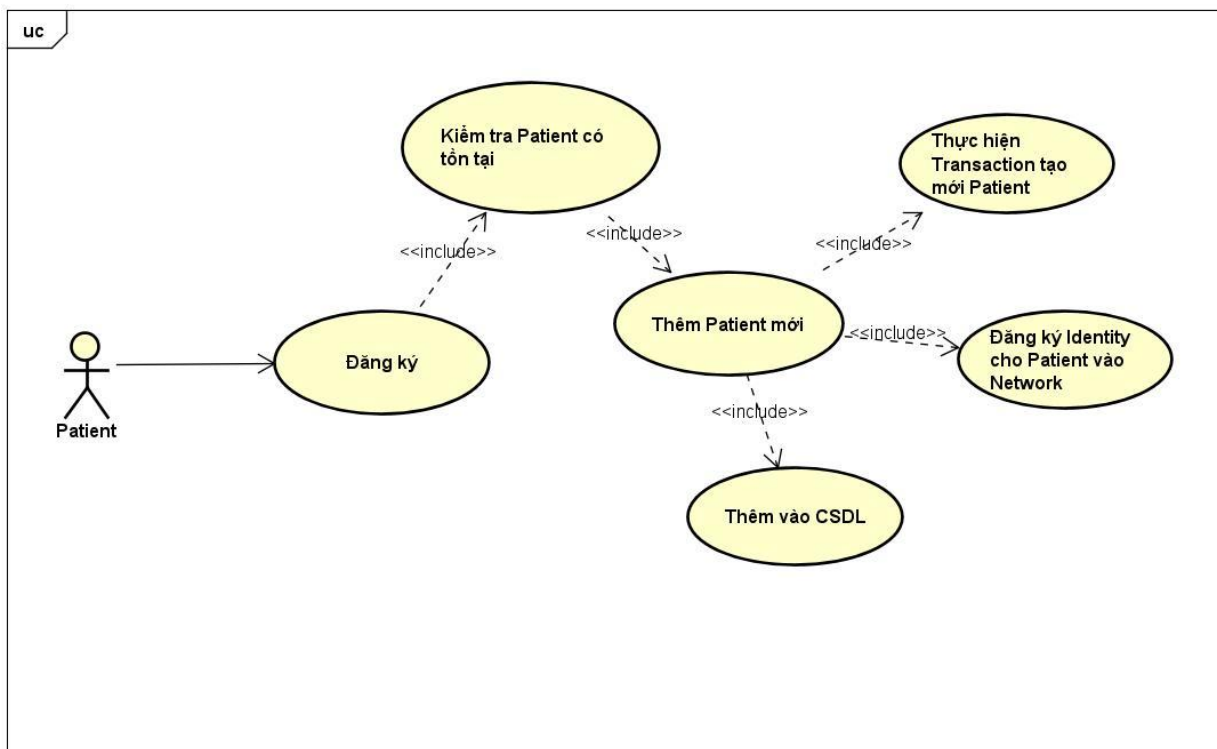
Quản lý yêu cầu: Chấp nhận/ Từ chối các Request.

Đăng xuất: Đăng xuất khỏi hệ thống



## 4.2.2. USE-CASE đăng ký

### a. USE-CASE Patient đăng ký



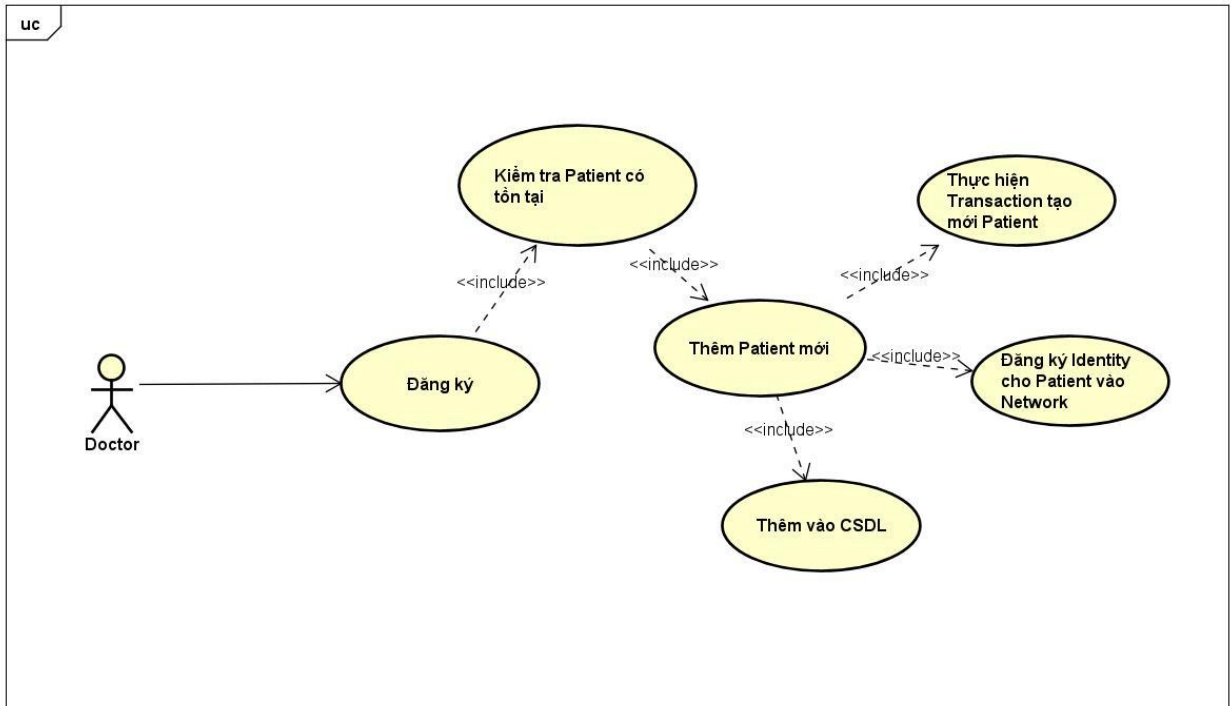
Hình 4.3. USE-CASE Patient đăng ký vào hệ thống

**Mô tả:** Chức năng đăng ký một bệnh nhân mới

**Dòng sự kiện:** Khi Patient tiến hành đăng ký thì hệ thống tiến hành kiểm tra những thông tin mà người dùng đã gửi lên. Nếu thông tin hợp lệ và chưa có thông tin trùng lặp nào trên hệ thống thì hệ thống tiến hành đăng ký Participant cho Patient này và thêm người dùng này vào Network. Ngược lại, thông tin người dùng cũng được thêm vào CSDL. Nếu đăng ký thành công sẽ trả về trang đăng nhập. Nếu đăng ký không thành công sẽ thông báo lỗi.

**Yêu cầu khác:** Không có.

**b. USE-CASE Doctor đăng ký**



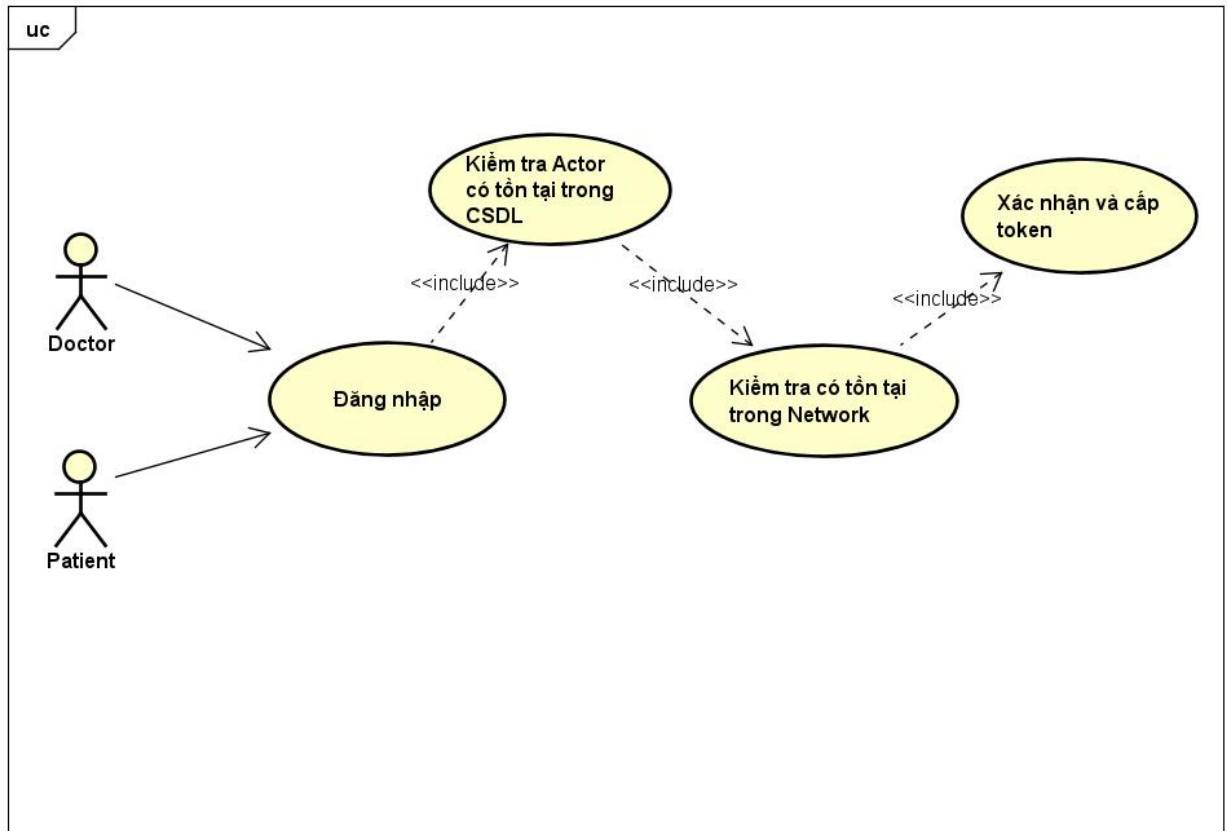
*Hình 4.4. USE-CASE Doctor đăng ký vào hệ thống*

**Mô tả:** Chức năng đăng ký một bác sĩ (Doctor) mới

**Dòng sự kiện:** Khi Doctor tiến hành đăng ký thì hệ thống tiến hành kiểm tra những thông tin mà người dùng đã gửi lên. Nếu thông tin hợp lệ và chưa có thông tin trùng lặp nào trên hệ thống thì hệ thống tiến hành đăng ký Participant cho Doctor này và thêm người dùng này vào Network. Ngược lại, thông tin người dùng cũng được thêm vào CSDL. Nếu đăng ký thành công sẽ trả về trang đăng nhập. Nếu đăng ký không thành công sẽ thông báo lỗi.

**Yêu cầu khác:** Không có.

### 4.2.3. USE-CASE đăng nhập



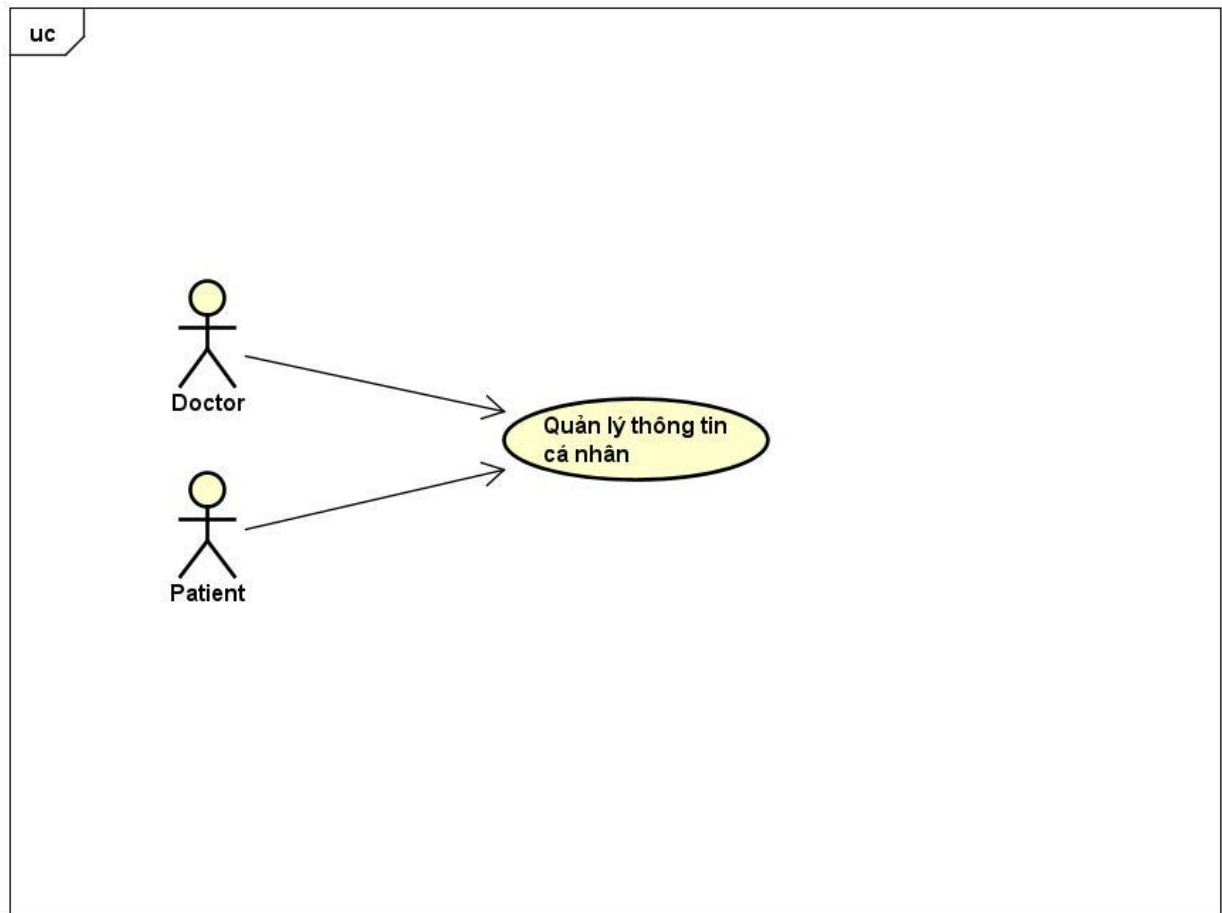
Hình 4.5. USE-CASE đăng nhập

**Mô tả:** Actor Doctor và Patient tiến hành đăng nhập vào hệ thống

**Dòng sự kiện:** Khi Actor tiến hành đăng nhập vào hệ thống, hệ thống tiến hành kiểm tra thông tin người dùng có trong CSDL hay không, nếu có, tiếp tục tiến hành kiểm tra người dùng đã có trong Network hay chưa. Nếu thỏa điều kiện, cho phép Actor đăng nhập vào hệ thống và cung cấp cho người dùng một token để làm việc. Nếu không, trả về trang Login

**Yêu cầu khác:** không có.

#### 4.2.4. USE-CASE quản lý thông tin cá nhân



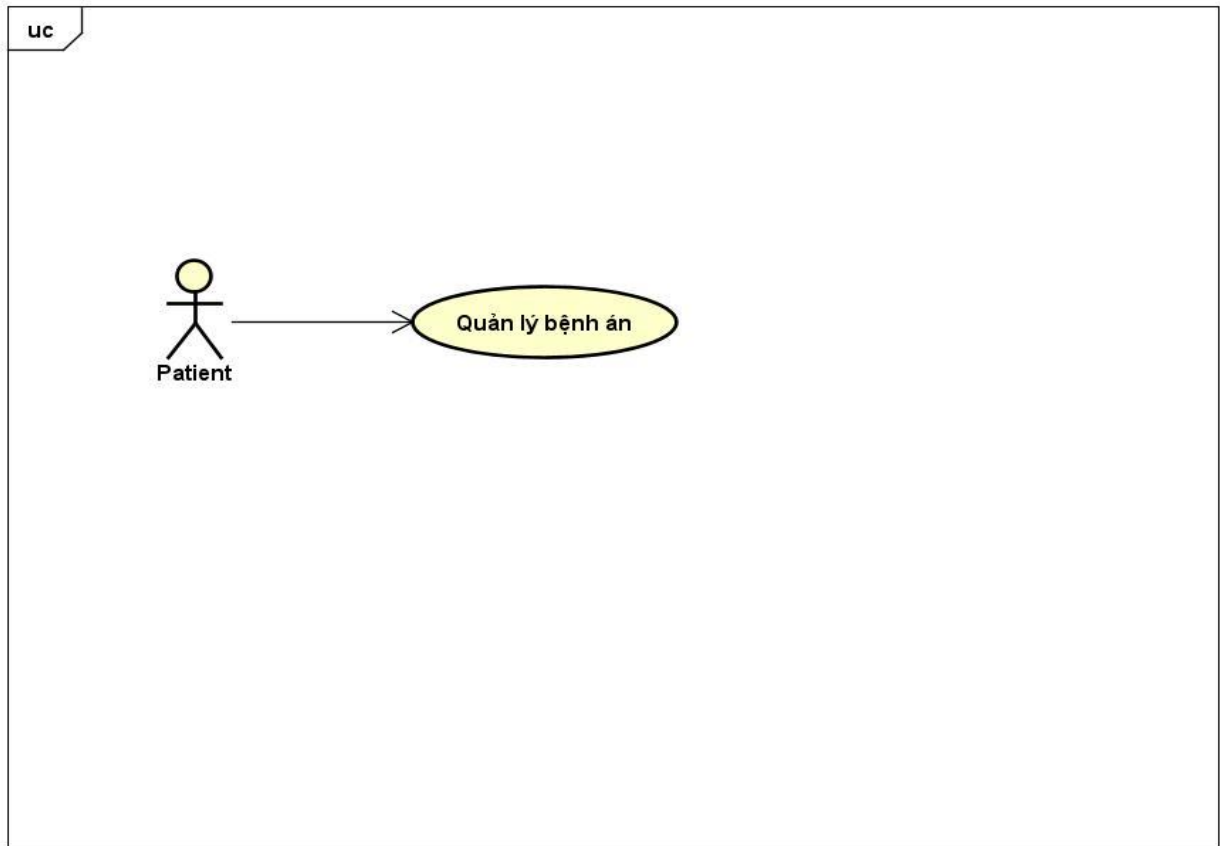
Hình 4.6. USE-CASE quản lý thông tin cá nhân

**Mô tả:** Chức năng này cho phép Actor Doctor và Patient xem/ cập nhật thông tin cá nhân của mình.

**Dòng sự kiện:** Người dùng chuyển sang trang Profile. Các thông tin bao gồm: địa chỉ, số điện thoại, chuyên khoa, giới tính, tình trạng hôn nhân, chức vụ.

**Yêu cầu khác:** phải đăng nhập thành công vào hệ thống

#### 4.2.5. USE-CASE quản lý thông tin bệnh án



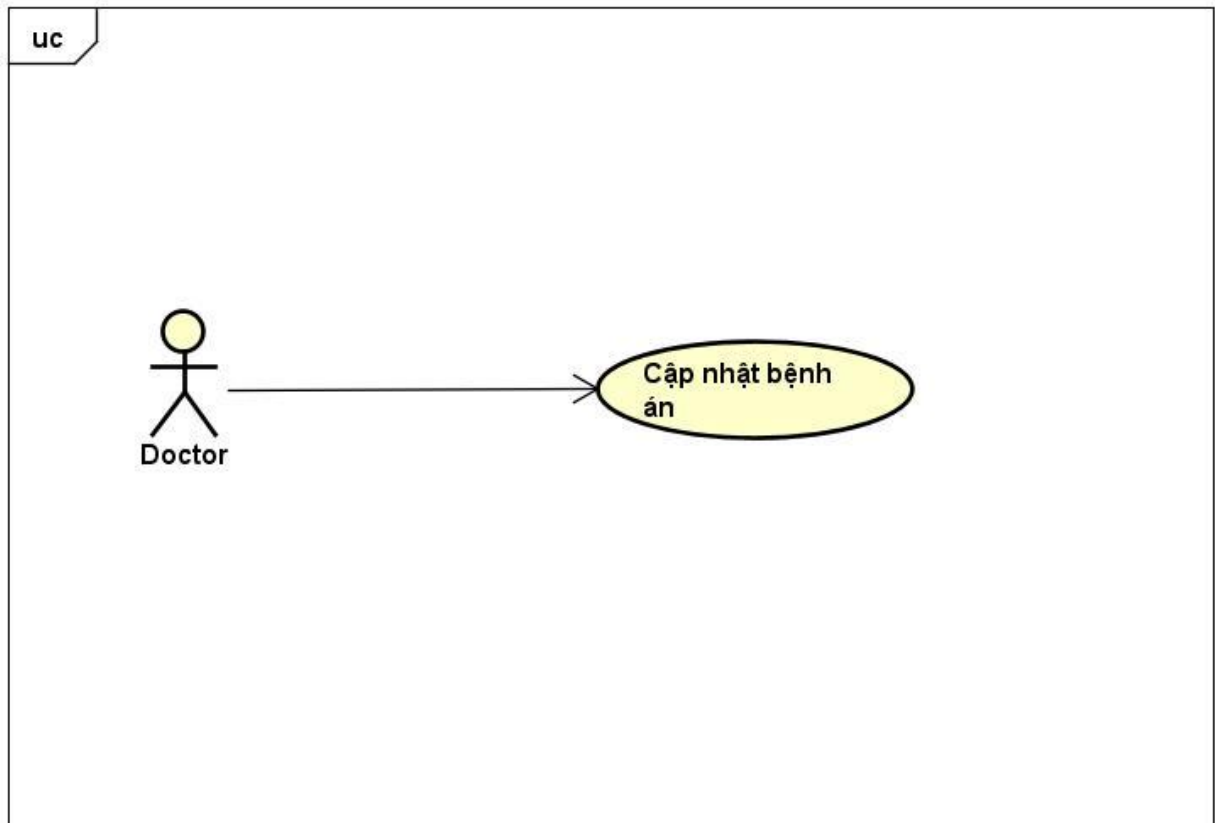
*Hình 4.7. USE-CASE Patient quản lý bệnh án*

**Mô tả:** Chức năng cho phép Patient xem/ tạo bệnh án mới

**Dòng sự kiện:** Khi người dùng sử dụng chức năng, người dùng có thể xem thông tin bệnh án của mình, nếu chưa có bệnh án nào, người dùng có thể tạo bệnh án mới và tiến hành tới gặp Doctor (Bác sĩ) khám để cập nhật thông tin bệnh án

**Yêu cầu khác:** Patient phải đăng nhập vào hệ thống.

#### 4.2.6. USE-CASE cập nhật thông tin bệnh án



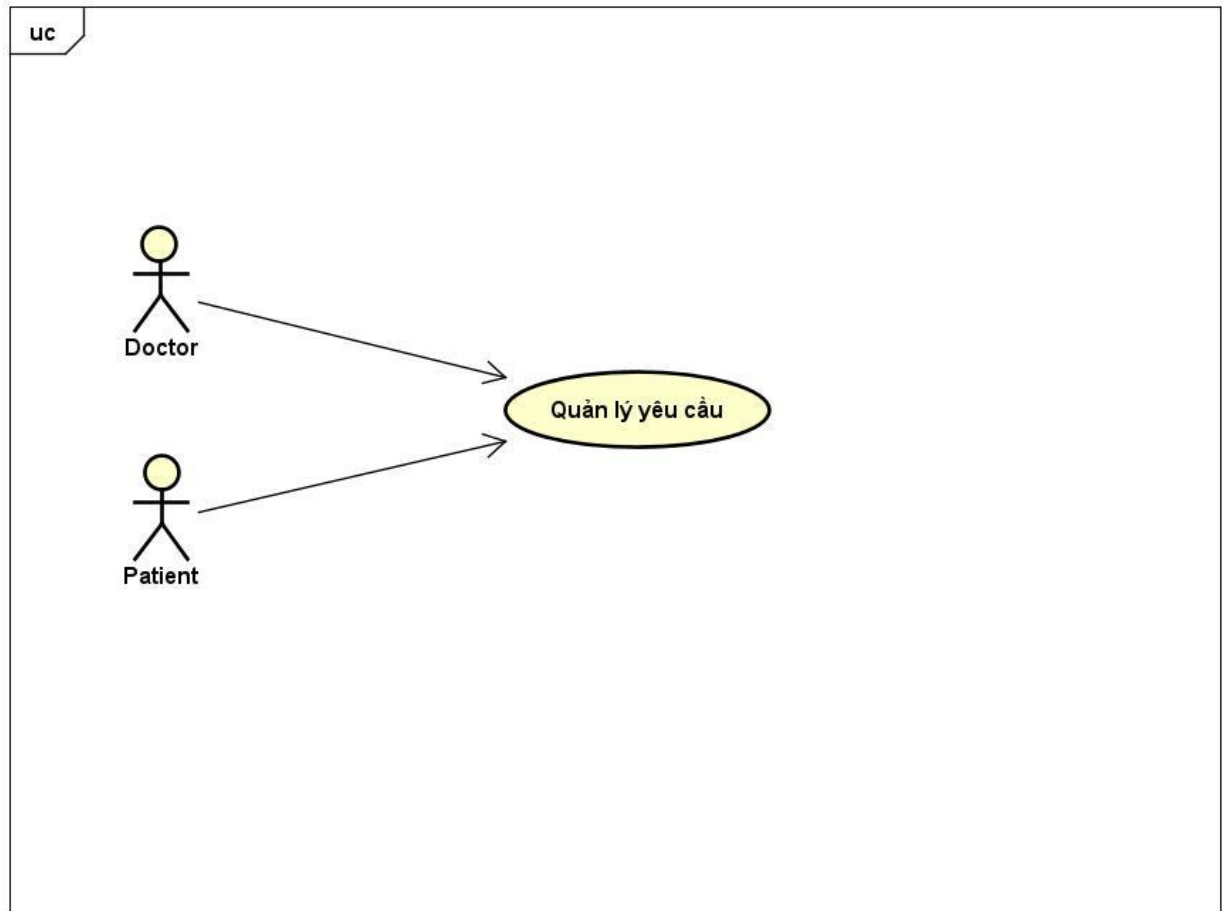
Hình 4.8. USE-CASE Doctor cập nhật thông tin bệnh án

**Mô tả:** Chức năng cho phép Doctor (Bác sĩ) cập nhật thông tin bệnh án cho bệnh nhân.

**Dòng sự kiện:** Khi Patient (Bệnh nhân) đến gặp bác sĩ để khám bệnh, bác sĩ tiến hành cập nhật các thông tin liên quan tới bệnh án cho bệnh nhân. Thông tin này không thể xóa trên hệ thống và tiến hành lưu vết lại bởi các Transaction.

**Yêu cầu khác:** chức năng cần được đăng nhập để sử dụng

#### 4.2.7. USE-CASE quản lý yêu cầu (Request)



Hình 4.9. USE-CASE quản lý yêu cầu

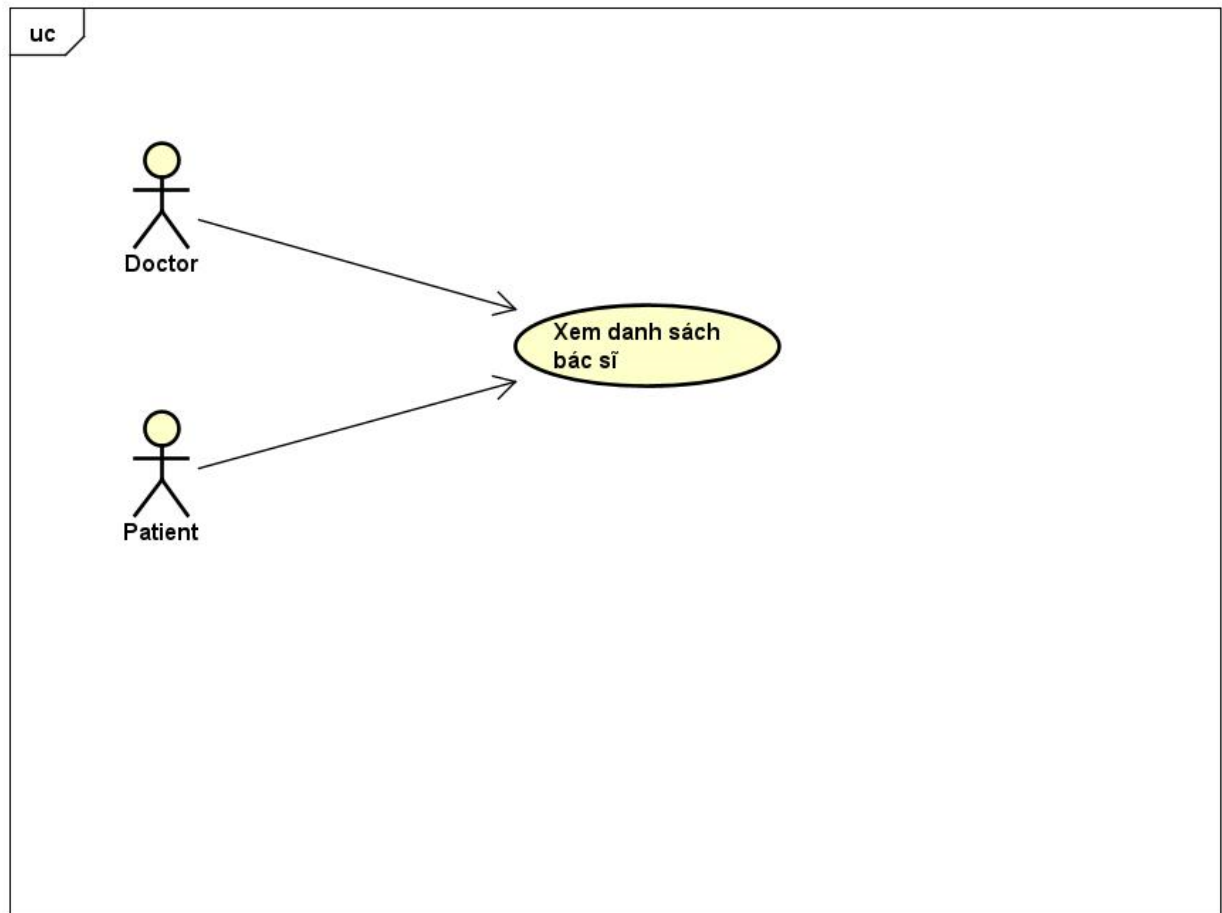
**Mô tả:** Chức năng này cho phép các Doctor (Bác sĩ) và Patient (Bệnh nhân) thực hiện thao tác add/accept/reject các Request yêu cầu cấp quyền.

**Dòng sự kiện:** USE-CASE này bắt đầu khi người dùng chuyển sang chức năng yêu cầu lên các loại tài nguyên DoctorInfo/ PatientInfo/ Health Record.

**Dòng sự kiện phụ:** Người nhận yêu cầu Request có thể accept/reject yêu cầu.

**Yêu cầu khác:** Chức năng cần được đăng nhập để sử dụng

#### 4.2.8. USE-CASE xem danh sách bác sĩ



Hình 4.10. USE-CASE xem danh sách Doctor

**Mô tả:** Chức năng cho phép Doctor (bác sĩ) và Patient (bệnh nhân) xem danh sách bác sĩ.

**Dòng sự kiện:** Khi người dùng chuyển sang trang Doctor. Các thông tin bao gồm ID, tên, chuyên khoa, chức vụ

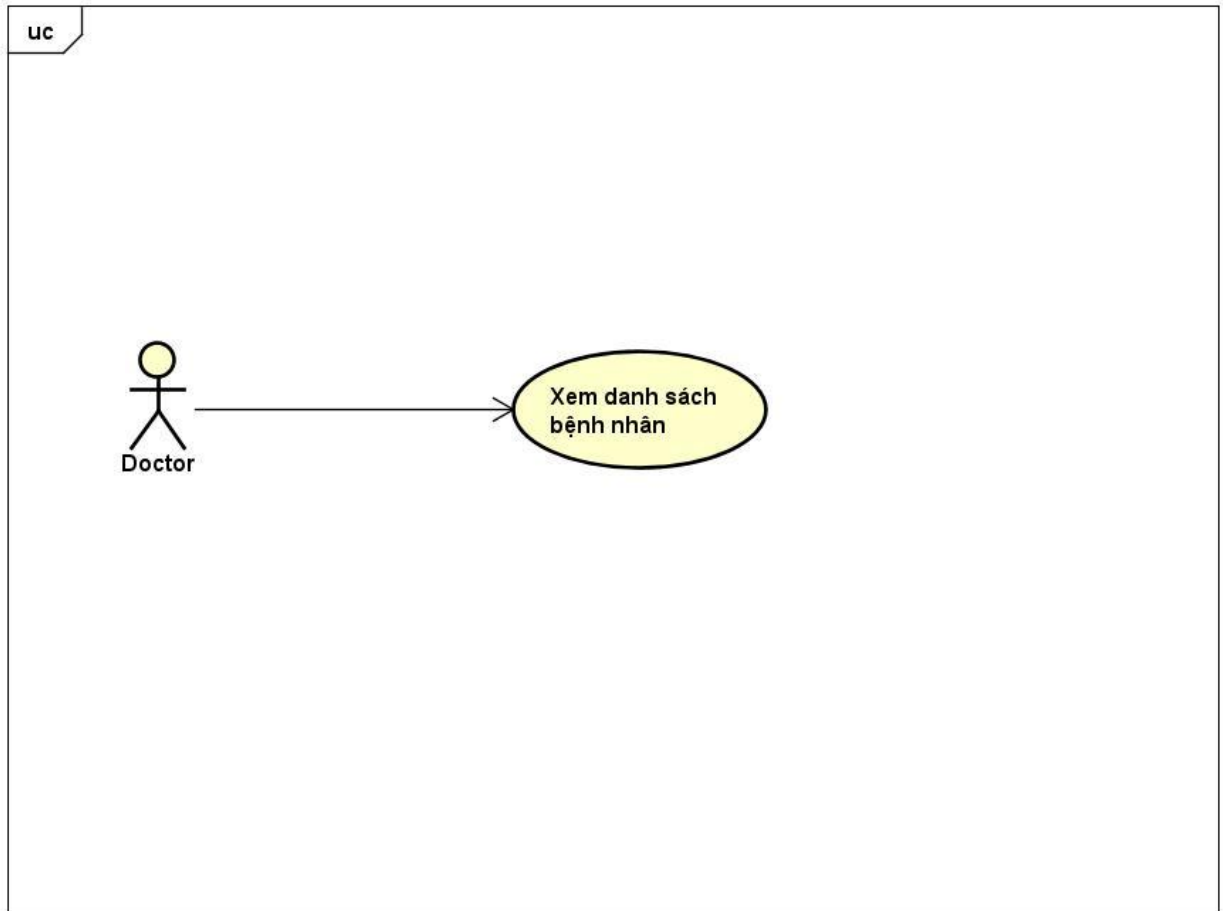
**Dòng sự kiện phụ:**

- Xem thông tin chi tiết thông tin cá nhân bác sĩ
- Cấp quyền cho từng Patient gửi yêu cầu để xem thông tin bệnh án



***Yêu cầu khác:*** Chức năng cần được đăng nhập để sử dụng

#### **4.2.9. USE-CASE xem danh sách bệnh nhân**



*Hình 4.11. USE-CASE Doctor xem danh sách Patient*

***Mô tả:*** Chức năng cho phép Doctor xem danh sách bệnh nhân

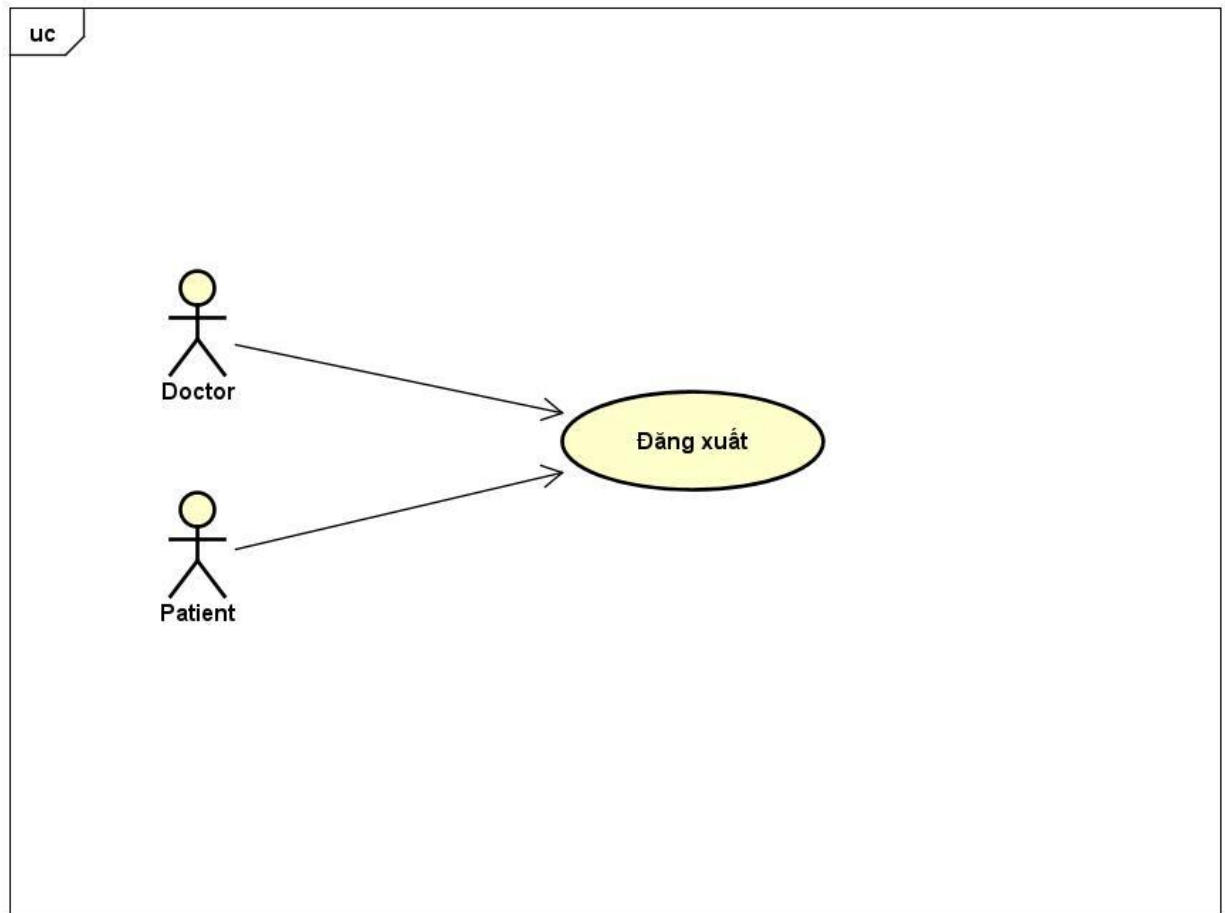
***Dòng sự kiện:*** Doctor chuyển sang trang Patient. Xem được các thông tin bao gồm ID, tên, nghề nghiệp của bệnh nhân.

***Dòng sự kiện phụ:***

- Xem thông tin chi tiết bệnh án của bệnh nhân

***Yêu cầu khác:*** chức năng cần được đăng nhập để sử dụng

#### 4.2.10. USE-CASE đăng xuất



Hình 4.12. USE-CASE đăng xuất

**Mô tả:** Chức năng cho phép người dùng đăng xuất khỏi hệ thống

**Dòng sự kiện:** Người dùng sử dụng chức năng đăng xuất và đăng xuất khỏi hệ thống

**Yêu cầu khác:** Chức năng cần được đăng nhập để sử dụng

### 4.3. Xây dựng ứng dụng

#### 4.3.1. Cài đặt môi trường

Các công cụ và môi trường tiên quyết cần có trước khi cài đặt Hyperledger gồm:

Docker Engine: Version  $\geq$  17.03

Docker-Compose: Version >= 1.8

Node: >= 8.9 (không hỗ trợ version 9)

npm: v5.x

git: >= 2.9.x

Python: 2.7.x

Code editor như VSCode...

Để cài đặt các công cụ tiên quyết, Hyperledger đã cung cấp một chương trình shell giúp tự động hóa việc cài đặt này. Ta thực hiện các bước như sau

```
cd $HOME
curl -O -k https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh
chmod u+x prereqs-ubuntu.sh
sudo apt-get install -y software-properties-common
```

### ***Cài đặt các Components:***

Cài đặt các công cụ CLI:

```
npm install -g composer-cli@0.20
npm install -g composer-rest-server@0.20
npm install -g generator-hyperledger-composer@0.20
npm install -g yo
```

### ***Cài đặt Composer-playground:***

```
npm install -g composer-playground@0.20
```

### ***Cài đặt Hyperledger Fabric:***

```
mkdir ~/fabric-dev-servers && cd ~/fabric-dev-servers  
  
curl -O https://raw.githubusercontent.com/hyperledger/composer-tools/master/packages/fabric-dev-servers/fabric-dev-servers.tar.gz  
  
tar -xvf fabric-dev-servers.tar.gz
```

Tạo một phiên bản Hyperledger Fabric và một PeerAdmin:

```
cd ~/fabric-dev-servers  
export FABRIC_VERSION=hlfv12  
./startFabric.sh  
./createPeerAdminCard.sh
```

Khởi tạo một Project mới thông qua lệnh sau và điền thông tin cần thiết:

```
yo hyperledger-composer:businessnetwork
```

```

fd@fd18:~/blockchainDocker$ yo hyperledger-composer:businessnetwork
Welcome to the business network generator
? Business network name: tutorial-network
? Description: First Network
? Author name: Ferry Djaja
? Author email: ferry.djaja@gmail.com
? License: Apache-2.0
? Namespace: org.basic.server
? Do you want to generate an empty template network? No: generate a populated sample network
  create package.json
  create README.md
  create models/org.basic.server.cto
  create permissions.acl
  create .eslintrc.yml
  create features/sample.feature
  create features/support/index.js
  create test/logic.js
  create lib/logic.js

```

Hình 4.13. Khởi tạo một Project mới thông qua Hyperledger Composer

#### 4.3.2. Định nghĩa các Participant

Participant là những người tham gia vào Business Network. Một participant có thể là một cá nhân hoặc một tổ chức và có thể tạo ra các asset, trao đổi asset với các participant khác. Họ thực hiện các thao tác với asset thông qua các Transaction. Đặc biệt mỗi participant đều có một role (vai trò) được gán và cần một identity được cấp để có thể tương tác với business network cho dù họ đã được tạo trong mạng hay chưa

Trong hệ thống đang phát triển, có hai loại Participant là Doctor (bác sĩ) và Patient (bệnh nhân)

```

abstract participant Person identified by identityCardNumber {

  o String personId

  o String name

  o String address

  o String email

  o String phone
}

```

```

o String identityCardNumber

o Gender sex

o DateTime birthday optional

o DateTime createAt optional

o DateTime updateAt optional

}

participant Doctor extends Person {

}

participant Patient extends Person{

}

```

#### 4.3.3. Định nghĩa các Enum

Hyperledger Composer cung cấp cho ta một kiểu định nghĩa dữ liệu là Enum (Enumerated) là một kiểu dữ liệu được sử dụng để xác định một trường dữ liệu có thể có 1 hoặc N giá trị định sẵn.

Enum	Mô tả
enum Gender {  o Male  o Female  o Other	Giới tính

}	
enum RequestStatus{  o Accepted  o New  o Rejected  }	Tình trạng của Request
enum ResourceType{  o HealthRecord  o PatientInfo  o DoctorInfo  o Doctor  o Patient  }	Loại Resource
enum Role{  o Patient  o Doctor  }	Role của Participant
enum Specialist{  o Cardiology	Chuyên khoa của Participant

<pre> o Obstetrics } </pre>	
<pre> enum MarriageStatus { o Single o Mariage o Divorced } </pre>	Tình trạng hôn nhân của Participant
<pre> enum HealthRecordStatus{ o New o Done } </pre>	Tình trạng của Health Record

*Bảng 4.1. Danh sách các Enum và mô tả*

#### **4.3.4. Định nghĩa các Asset của Doctor (Bác sĩ)**

Trong đề tài cũng như trong tổ chức, Doctor (Bác sĩ) có vai trò khám chữa bệnh cho các participant cho Patient (Bệnh nhân). Trong phạm vi đề tài, các bác sĩ chỉ có chức năng đơn giản là xem thông tin cá nhân PatientProfile, yêu cầu quyền truy cập đến dữ liệu bệnh án. Bác sĩ sẽ có các Asset sau:

<b>Asset</b>	<b>Mô tả</b>
<pre> asset DoctorInfo identified by doctorInfoId { o String doctorInfoId } </pre>	Thông tin cá nhân của Doctor (Bác sĩ)



<ul style="list-style-type: none"> <li>o String doctorId</li> <li>o String name</li> <li>o String address</li> <li>o String email</li> <li>o String phone</li> <li>o String identityCardNumber</li> <li>o String sex</li> <li>o DateTime birthday optional</li> <li>o Specialist specialist</li> <li>o MarriageStatus marriageStatus</li> <li>o String tittle optional</li> <li>o DateTime createAt optional</li> <li>o DateTime updateAt optional</li> <li>--&gt; Doctor[] authorizedDoctors</li> <li>--&gt; Patient[] authorizedPatients</li> <li>--&gt; Doctor owner</li> </ul> }	
asset Request identified by requestId {  o String requestId	Chi tiết Request của Doctor (Bác sĩ)

<pre> o Role requesterRole  o Role resourceOwnerRole  o ResourceType resourceType  o RequestStatus status  o String resourceId  --&gt; Person owner  --&gt; Person resourceOwner  } </pre>	
--	--

*Bảng 4.2. Danh sách các Asset của Doctor và mô tả*

#### **4.3.5. Định nghĩa các Asset của Patient (Bệnh nhân)**

Trong đề tài, participant Patient (bệnh nhân) có vai trò quản lý thông tin PatientProfile và thông tin bệnh án Health Record của mình, ủy quyền truy cập các thông tin bệnh án (Health Record) của mình cho Doctor (bác sĩ).

Bệnh nhân sẽ có các Asset sau:

<b>Asset</b>	<b>Mô tả</b>
<pre> asset PatientInfo identified by patientInfoId {  o String patientInfoId  o String patientId  o String name </pre>	Thông tin chi tiết của Patient (Bệnh nhân)

<ul style="list-style-type: none"> <li>o String address</li> <li>o String email</li> <li>o String phone</li> <li>o String identityCardNumber</li> <li>o String sex</li> <li>o DateTime birthday optional</li> <li>o String career</li> <li>o MarriageStatus marriageStatus</li> <li>o DateTime createAt optional</li> <li>o DateTime updateAt optional</li> <li>--&gt; Doctor[] authorizedDoctors</li> <li>--&gt; Patient owner</li> </ul> }	
asset Request identified by requestId { <ul style="list-style-type: none"> <li>o String requestId</li> <li>o Role requesterRole</li> <li>o Role resourceOwnerRole</li> <li>o ResourceType resourceType</li> <li>o RequestStatus status</li> </ul>	Chi tiết Request của Patient (Bệnh nhân)

<pre> o String resourceId  --&gt; Person owner  --&gt; Person resourceOwner  } </pre>	
<pre> asset HealthRecord identified by healthRecordId {    o String healthRecordId    o String hight optional    o String tuoithai optional    o String hatthu0 optional    o String hattruong0 optional    o String tieucau0 optional    o String tq0 optional    o String aptt0 optional    o String fibrinogen0 optional    o String ast0 optional    o String alt0 optional    o String creatinin0 optional    o String ure0 optional    o String auric0 optional } </pre>	<p>Thông tin Health Record của Patient (Bệnh nhân)</p>

<ul style="list-style-type: none"> <li>o String ldh0 optional</li> <li>o String damnieu0 optional</li> <li>o String damnieu24h0 optional</li> <li>o String protein0 optional</li> <li>o String albumin0 optional</li> <li>o String bilirubintp0 optional</li> <li>o String bilirubintt0 optional</li> <li>o String conclusion optional</li> <li>o HealthRecordStatus status</li> <li>--&gt; Doctor [] authorizedDoctors</li> <li>--&gt; Patient owner</li> <li>}</li> </ul>	
---	--

*Bảng 4.3. Danh sách các Asset của Patient và mô tả*

#### **4.3.6. Định nghĩa các Access Rule Control**

Access Rule Control là một phần vô cùng quan trọng trong Hyperledger Composer, cho phép kiểm soát những Resource hoặc dữ liệu mà Participant được phép truy cập.

Dữ liệu bệnh án là một loại dữ liệu vô cùng nhạy cảm, vì vậy cần đảm bảo tính riêng tư và toàn vẹn dữ liệu một cách hoàn toàn. Dữ liệu cần được phân quyền một cách vô cùng chi tiết và hợp lý.

Chỉ có admin của network có thể truy cập đến bất cứ Resource nào của Network. Sau đó, tiến hành gán Rule đến các Resource theo mong muốn.

```
rule PatientReadParticipantInfo{

    description:"Patient can read their participant info"

    participant(p):"org.basic.server.Patient"

    operation: READ

    resource(r):"org.basic.server.Patient"

    condition: (p.getIdentifier()===r.getIdentifier())

    action: ALLOW

}

rule PatientReadPatientInfo{

    description:"Patient can read Patient Info"

    participant(p):"org.basic.server.Patient"

    operation: READ,UPDATE

    resource(r):"org.basic.server.PatientInfo"

    condition: (p.getIdentifier()===r.owner.getIdentifier())

    action: ALLOW

}

rule PatientReadHealthRecord{

    description:"Patient can read Health Record"

    participant(p):"org.basic.server.Patient"
```

```

operation: READ

resource(r): "org.basic.server.HealthRecord"

condition: (p.getIdentifier()===r.owner.getIdentifier())

action: ALLOW
}

rule PatientReadDoctorInfo{

  description: "Patient can read Doctor Info"

  participant(p): "org.basic.server.Patient"

  operation: READ,UPDATE

  resource(r): "org.basic.server.DoctorInfo"

  condition: (r.authorizedPatients.some(function(patient){

    return patient.getIdentifier()===p.getIdentifier();

  })))

  action: ALLOW
}

rule PatientCreateRequest{

  description: "Patient can create read Request"

  participant: "org.basic.server.Patient"

  operation: CREATE

```

```

    resource: "org.basic.server.Request"

    action: ALLOW
}

rule PatientReadFromRequest{

    description:"Patient can see all request about Patient"

    participant(p):"org.basic.server.Patient"

    operation: READ,DELETE,UPDATE

    resource: "org.basic.server.Request"

    condition: ((p.getIdentifier() === r.owner.getIdentifier())

        || p.getIdentifier() === r.resourceOwner.getIdentifier())

    action:ALLOW
}

rule DoctorReadParticipantInfo{

    description:"Patient can read their Participant Info"

    participant(p):"org.basic.server.Doctor"

    operation: READ

    resource(r):"org.basic.server.Doctor"

    condition: (p.getIdentifier()===r.getIdentifier())

    action: ALLOW
}

```



```

}

rule DoctorReadDoctorInfo{

  description:"Doctor can read Doctor Info"

  participant(p):"org.basic.server.Doctor"

  operation: READ,UPDATE

  resource(r):"org.basic.server.DoctorInfo"

  condition: (p.getIdentifier()===r.owner.getIdentifier())

  action: ALLOW

}

rule DoctorCreateRequest{

  description: "Patient can create read Request"

  participant: "org.basic.server.Doctor"

  operation: CREATE

  resource: "org.basic.server.Request"

  action: ALLOW

}

rule DoctorReadFromRequest{

  description:"Doctor can see all request about Doctor"

  participant(p):"org.basic.server.Doctor"

```

```

operation: READ,DELETE,UPDATE

resource: "org.basic.server.Request"

condition: ((p.getIdentifier() === r.owner.getIdentifier())

  || p.getIdentifier() === r.resourceOwner.getIdentifier())

action:ALLOW
}

rule DoctorReadPatientInfo{

  description:"Patient can read Doctor Info"

  participant(p):"org.basic.server.Doctor"

  operation: READ,UPDATE

  resource(r):"org.basic.server.PatientInfo"

  condition: (r.authorizedDoctors.some(function(doctor){

    return doctor.getIdentifier()===p.getIdentifier();

  })))

  action: ALLOW

}

rule AuthorizedPatientReadDoctorInfo {

  description:"Patient can request read Doctor Info"

  participant(p):"org.basic.server.Patient"

```

```

operation: READ

resource(r):"org.basic.server.DoctorInfo"

condition: (r.authorizedPatients.some(function(patient){

    return patient.getIdentifier()===p.getIdentifier();

    })))

action: ALLOW

}

rule AuthorizedDoctorReadPatientInfo{

    description:"Doctor can request read Patient Info"

    participant(p):"org.basic.server.Doctor"

    operation: READ

    resource(r):"org.basic.server.PatientInfo"

    condition: (r.authorizedDoctors.some(function(doctor){

        return doctor.getIdentifier()===p.getIdentifier();

        })))

    action: ALLOW

}

rule AuthorizedDoctorReadHealthRecord{

    description:"Doctor can read Health Record"

```

```
participant(p):"org.basic.server.Doctor"

operation: READ,UPDATE

resource(r):"org.basic.server.HealthRecord"

condition: (r.authorizedDoctors.some(function(doctor){

    return doctor.getIdentifier()===p.getIdentifier();

    )))

action: ALLOW

}
```

#### 4.3.7. Định nghĩa các câu Query

Truy vấn trong Hyperledger Fabric tương tự như truy vấn trong cơ sở dữ liệu truyền thống.

```
/** Sample queries for business network

*/

query selectAllDoctorInfo {

    description: "Select all Doctor Info"

    statement:

        SELECT org.basic.server.DoctorInfo

}
```

```
query selectAllPatientInfo {  
  description: "Select all Patient Info"  
  statement:  
    SELECT org.basic.server.PatientInfo  
}  
query selectAllRequest {  
  description: "Select all Request"  
  statement:  
    SELECT org.basic.server.Request  
}  
query selectAllHealthRecord {  
  description: "Select all Health Record"  
  statement:  
    SELECT org.basic.server.HealthRecord  
}  
query selectHealthRecordHasIdentityCardNumber {  
  description: "Select Health Record has Id"  
  statement:  
    SELECT org.basic.server.HealthRecord
```

```

WHERE (healthRecordId == _$healthRecordId)

}

query selectRequestHasIdentityCardNumber {

  description: "Select Request have Identity Card Number"

  statement:

    SELECT org.basic.server.Request

    WHERE (owner == _$idRequester OR resourceOwner == _$idResourceOwner)

}

```

#### 4.3.8. Định nghĩa các Transaction và Chaincode

Để chạy được các Business Logic của Network thì ta cần định nghĩa các Transaction và Chaincode của mạng. Các định nghĩa được thực hiện như sau:

##### *a. Transaction và Chaincode của Doctor*

createDoctor	Tạo participant Doctor
createDoctorInfo	Tạo thông tin DoctorInfo
createDoctorIdentity	Tạo Identity cho Doctor
createRequest	Tạo thông tin Request
doctorAcceptRequestOfPatient	Doctor chấp nhận Request của Patient
doctorAcceptRequestOfDoctor	Doctor chấp nhận Request của Doctor khác

doctorRevokeRequestOfDoctor	Doctor thu hồi quyền của Doctor
doctorRevokeRequestOfPatient	Doctor thu hồi quyền của Patient
doctorUpdateHealthRecord	Doctor cập nhật thông tin Health Record

*Bảng 4.4. Transaction và Chaincode của Doctor*

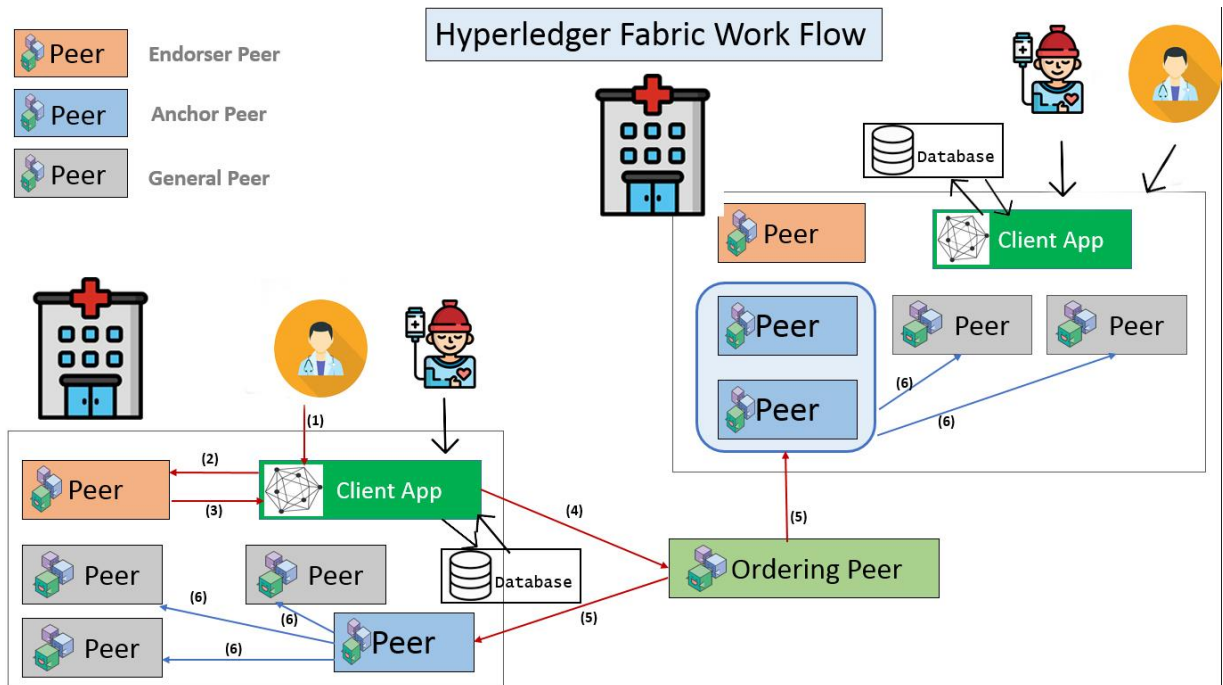
***b. Transaction và Chaincode của Patient***

<b>Transaction</b>	<b>Mô tả</b>
createPatient	Tạo thông tin participant Patient
createPatientInfo	Tạo thông tin PatientInfo
createPatientIdentity	Tạo Identity cho Patient
createRequest	Tạo thông tin Request
patientAcceptRequestOfDoctor	Patient chấp nhận Request của Doctor
patientRevokeRequestOfPatient	Patient thu hồi quyền của Doctor
createHealthRecord	Tạo thông tin Health Record

*Bảng 4.5. Transaction và Chaincode của Patient*

### 4.3.9. Xây dựng tầng Web Application

#### a. Kiến trúc hệ thống



Hình 4.14. Kiến trúc hệ thống của ứng dụng

- Tầng Web Application: Giao diện được sử dụng để tiếp nhận các hoạt động đầu vào như nhập dữ liệu, chạy các giao dịch,... Đóng vai trò là tầng API tương tác với cơ sở hạ tầng Hyperledger Fabric. Tầng này thực hiện các giao dịch được Request bởi máy client từ tầng front-end thông qua các API module đã được định nghĩa.

- Tầng Blockchain: Tầng này chính là Blockchain. Đây chính là cơ sở dữ liệu của hệ thống.

Ở giữa hai tầng này chính là các hợp đồng thông minh. Chúng chịu trách nhiệm trung gian và hoạt động hoàn toàn tự động. Tất cả các hoạt động của ứng dụng từ việc nhập dữ liệu, giao dịch hay truy xuất dữ liệu đều phải thông qua các hợp đồng thông minh này.



**Ưu điểm:** Hạn chế phơi bày tầng REST Server của Hyperledger. Tầng Web Application đóng vai trò là tầng API tương tác với Blockchain Hyperledger Fabric. Tầng Web Application thực hiện các Transaction được yêu cầu bởi người dùng cuối từ tầng front-end thông qua các chức năng đã được định nghĩa trước.

**Nhược điểm:** Tầng Web Application xử lý nhiều hơn. Cần phải viết nhiều chức năng cho tầng Web Application.

### ***b. NodeJS***

NodeJS là một mã nguồn được xây dựng dựa trên nền tảng Javascript V8 Engine, nó được sử dụng để xây dựng các ứng dụng web như các trang video clip, các forum và đặc biệt là trang mạng xã hội phạm vi hẹp. NodeJS là một mã nguồn mở được sử dụng rộng bởi hàng ngàn lập trình viên trên toàn thế giới. NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ WIndow cho tới Linux, OS X nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất.

NodeJS được sử dụng được xây dựng để xây dựng tầng Web Application bởi tính tiện dụng, dễ hiểu cũng như những ưu điểm về tốc độ, tính ổn định và khả năng xử lý bất đồng bộ vượt trội. Ngoài ra NodeJS còn nổi bật bởi khả năng xử lý Realtime, đơn luồng nhưng khả năng mở rộng cao, không cần đệm, có giấy phép...

Đồng thời, để giúp user tránh việc ghi nhớ những thông tin phức tạp, đề tài sử dụng kết hợp MongoDB để lưu những thông tin User.

NodeJS là một công cụ đa nền tảng cho phép người lập trình tạo ra các ứng dụng server side trên ngôn ngữ JavaScript.

NodeJS có hiệu suất xử lý tuyệt vời, thích hợp với các ứng dụng hướng sự kiện, thời gian thực và khả năng xử lý bất đồng bộ. Ngoài ra còn được hỗ trợ một trình quản lý npm với hàng trăm package.

Nhưng NodeJS cũng có một nhược điểm đó là ứng dụng nặng tổng tài nguyên. Nếu bạn cần xử lý các ứng dụng tốn tài nguyên CPU như encoding video, convert file, decoding encryption... hoặc các ứng dụng tương tự như vậy thì không nên dùng NodeJS. NodeJS và ngôn ngữ khác PHP, Ruby, Python.NET ...thì việc cuối cùng là phát triển các App Web.

### ***c. MongoDB***

MongoDB là một cơ sở dữ liệu NoSQL hướng đối tượng, đơn giản, linh động và có thể mở rộng.

Nó dựa trên mô hình lưu trữ NoSQL document. Các đối tượng dữ liệu được lưu trữ dưới dạng các tài liệu riêng biệt bên trong một collection – thay vì lưu trữ dữ liệu vào các cột và hàng của cơ sở dữ liệu quan hệ truyền thống.

Ngôn ngữ MongoDB là triển khai một kho lưu trữ dữ liệu cung cấp hiệu suất cao, tính sẵn sàng cao và tự động mở rộng

MongoDB sử dụng JSON hoặc BSON document để lưu trữ dữ liệu.

Các bản phân phối chung cho MongoDB hỗ trợ Windows, Linux, Mac OS X và Solaris.

Khi xây dựng hệ thống, tôi sử dụng MongoDB Cloud để triển khai cho ứng dụng này.

#### ***User Model:***

```
const mongoose = require('mongoose')

const Schema = mongoose.Schema

const userSchema=new Schema({

  // _id:mongoose.Schema.Types.ObjectId,

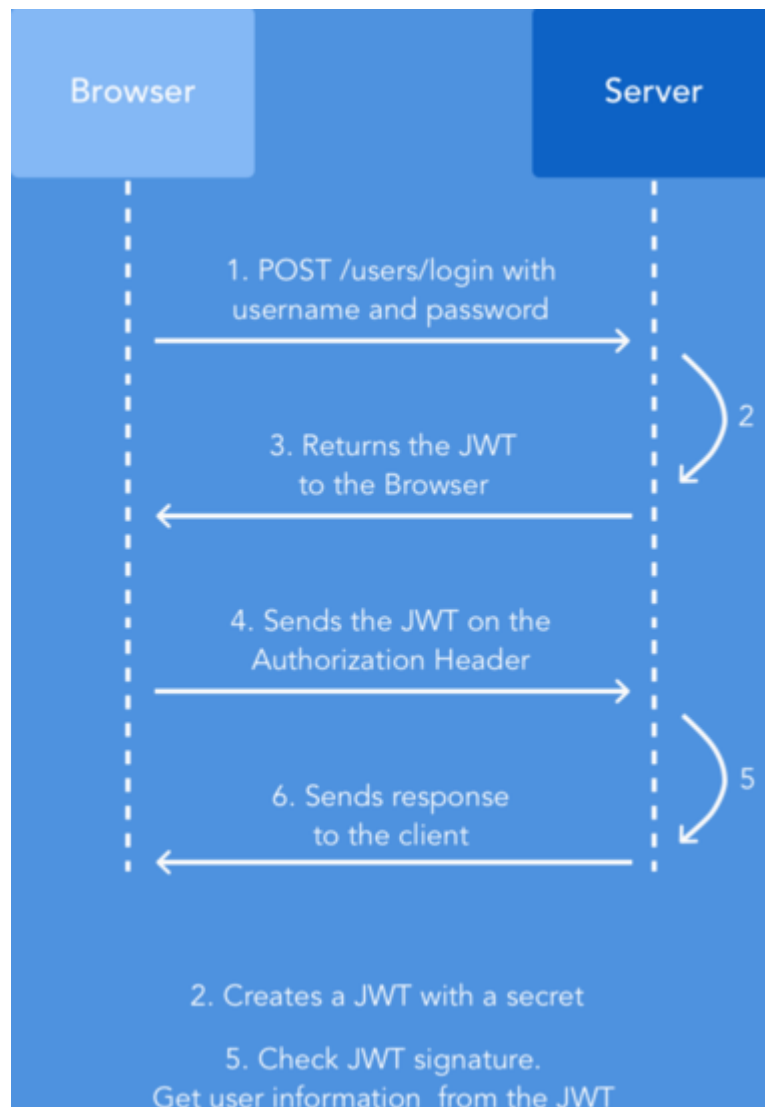
  email: {type: String, unique: true, required: true},
```

```
password: {type: String, required: true, minlength: 6},  
role: {type: String, enum: ['Doctor', 'Patient']},  
name:{type: String, required:true},  
address:{type: String, required:true},  
phone:{type: String, required:true},  
sex:{type: String, required:true,enum:['Male','Female','Other']},  
identityCardNumber:{type: String, required:true},  
createdAt: {type:Date, default:Date.now},  
deleted:{type:Boolean, default:false},  
__v:{type:Number, select:false}  
});  
  
module.exports = mongoose.model('User', userSchema)
```

#### ***d. JSON Web Token (JWT)***

Xác thực dựa trên token nổi bật khắp mọi nơi trên ứng dụng web ngày nay. Với hầu hết mọi ứng dụng web sử dụng API, token là cách tốt nhất để xử lý xác thực cho nhiều người dùng.

Xác thực dựa trên token là stateless: không lưu trữ bất kỳ thông tin nào về user trên server hoặc trong session.



*Hình 4.15. Cách thức hoạt động của JSON Web Token*

Về bản chất, JWT là một dữ liệu chữ ký dưới dạng JSON. Bởi vì nó được "kí" nên phía nhận có thể xác minh tính xác thực của nó. Dung lượng của nó rất nhỏ vì nó là JSON. JSON Web Token (JWT) là 1 tiêu chuẩn mở (RFC 7519) định nghĩa cách thức truyền tin an toàn giữa các thành viên bằng một đối tượng JSON. Thông tin này có thể được xác thực và đánh dấu tin cậy nhờ vào "chữ ký" của nó. Phần chữ ký của JWT sẽ được mã hóa lại bằng HMAC hoặc RSA.

Dữ liệu chữ kí ko có gì gọi là mới - cái hấp dẫn chúng ta ở đây là cách mà JWT được sử dụng để tạo nên 1 services RESTful đúng nghĩa - mà ko cần sessions. Dưới đây là cách giải thích đơn giản từ phía thế giới chúng ta cho cách mà nó hoạt động: Tưởng tượng rằng bạn vừa mới trở về nước sau 1 chuyến du lịch nước ngoài. Bạn làm thủ tục nhập cảnh và bạn nói rằng - "Hãy cho tôi qua, tôi có quốc tịch ở đây". Tất cả đều đúng cả nhưng làm cách nào bạn chứng minh với hải quan là bạn nói chính xác ? Tất nhiên bạn sẽ phải sử dụng hộ chiếu để xác thực danh tính của bạn. Hãy giả sử rằng tất cả các nhân viên cục xuất nhập cảnh có đủ thẩm quyền để khẳng định hộ chiếu của bạn là chính xác và được phát hành bởi văn phòng Hộ chiếu trong nước bạn. Hộ chiếu của bạn được xác thực và họ sẽ cho bạn qua. JSON Web Token bao gồm 3 phần, được ngăn cách nhau bởi dấu chấm (.):

- Header
- Payload
- Signature

Vì đóng vai trò là một REST Server nên vấn đề bảo mật ở tầng Web Application là một vấn đề cực kì quan trọng. Nên trong đề tài này, tôi sử dụng JWT, định nghĩa cách thức truyền tin an toàn giữa các bên bằng 1 đối tượng JSON.

Client gửi Request đến Server thông qua URL thông qua giao thức HTTP. Do đó nếu không có bất cứ bảo mật nào để xác thực người dùng thì mọi user đều có thể xóa user khác. Đây là một lỗ hổng bảo mật cực kì lớn và gây ra hậu quả cực kì nghiêm trọng. Vì vậy JWT thực sự là một giải pháp cực kì hiệu quả và an toàn.

Mỗi thành phần đc nêu trên (header, payload, signature) được encode base64url, sau đó được dán lại với nhau với một "chấm" để tạo nên JWT. Dưới đây là ví dụ:

```
var header = {
```

```
// Thuật toán signing.

"alg": "HS256",

// Thuộc tính type "JWT"

"typ": "JWT"

},

// Object header Base64

headerB64 = btoa(JSON.stringify(header)),

// The payload

payload = {

    "name": "John Doe",

    "admin": true

},

// Payload Object Base64

payloadB64 = btoa(JSON.stringify(payload)),

// signature

signature = signatureCreatingFunction(headerB64 + '.' + payloadB64),

// signature Base64

signatureB64 = btoa(signature),

// Đính tất cả các thành phần ở trên với dấu chấm để tạo nên JWS
```

```
jwt = headerB64 + '.' + payloadB64 + '.' + signatureB64;
```

Kết quả của một JWS sẽ nhìn như dưới đây:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bm90b2UiLCJhZG1pbWV5Ij0.OLvs36KmqB9cmsUrMpUtfhV52_iSz4bQMYJkI_TLQ
```

#### *e. Bcrypt*

Dựa vào những phương pháp trên, một biện pháp được đưa ra như là sự kết hợp của "băm", "salt", "stretching". Đó chính là bcrypt. bcrypt sử dụng một thuật toán mã hóa được gọi là "blowfish". Kết quả về từ hàm băm mật khẩu bao gồm giá trị băm, số lần salt và stretching sẽ được set lại thành một giá trị gọi là Modular Crypt Format.

**Bcrypt** trả về một chuỗi duy nhất với tất cả các tham số cần thiết thiết lập cũng như sức mạnh của nó như một chức năng băm mật khẩu. Vì vậy trên phương diện database chỉ cần một cột kiểu chuỗi có thể lưu được giá trị băm này, đây là một điểm lợi rất lớn của bcrypt. Ngoài ra, bcrypt hỗ trợ đầu vào của mỗi ký tự ở định dạng UTF-8 như là giá trị băm của chuỗi ký tiếng nhật trong ví dụ trên. Vì vậy nó có thể băm mật khẩu mà không giới hạn ký tự cho các ký tự cả chữ và số...

Vì thế, trong việc lưu trữ mật khẩu người dùng thì đây là một phương pháp cực kỳ an toàn và hiệu quả.

#### *f. Xây dựng Routes*

Method	Route	Mô tả
GET	/	Chuyển đến trang default
GET	/login	Lấy trang Đăng nhập

POST	/login	Đăng nhập
GET	/logout	Đăng xuất
GET	/register	Lấy trang Đăng ký thông tin
POST	/register	Đăng ký
GET	/index	Lấy trang Index
GET	/profile	Lấy thông tin DoctorInfo hoặc PatientInfo của người dùng
POST	/profile	Cập nhật thông tin Profile
GET	/request	Lấy thông tin Request
POST	/request	Tạo Request
GET	/doctors	Lấy danh sách Doctor
POST	/requestdoctorinfo	Request thông tin DoctorInfo của Doctor
GET	/patients	Lấy danh sách Patient
POST	/requestpatientinfo	Request thông tin PatientInfo của Patient
POST	/request/accept	Chấp nhận Request
POST	/request/reject	Từ chối Request

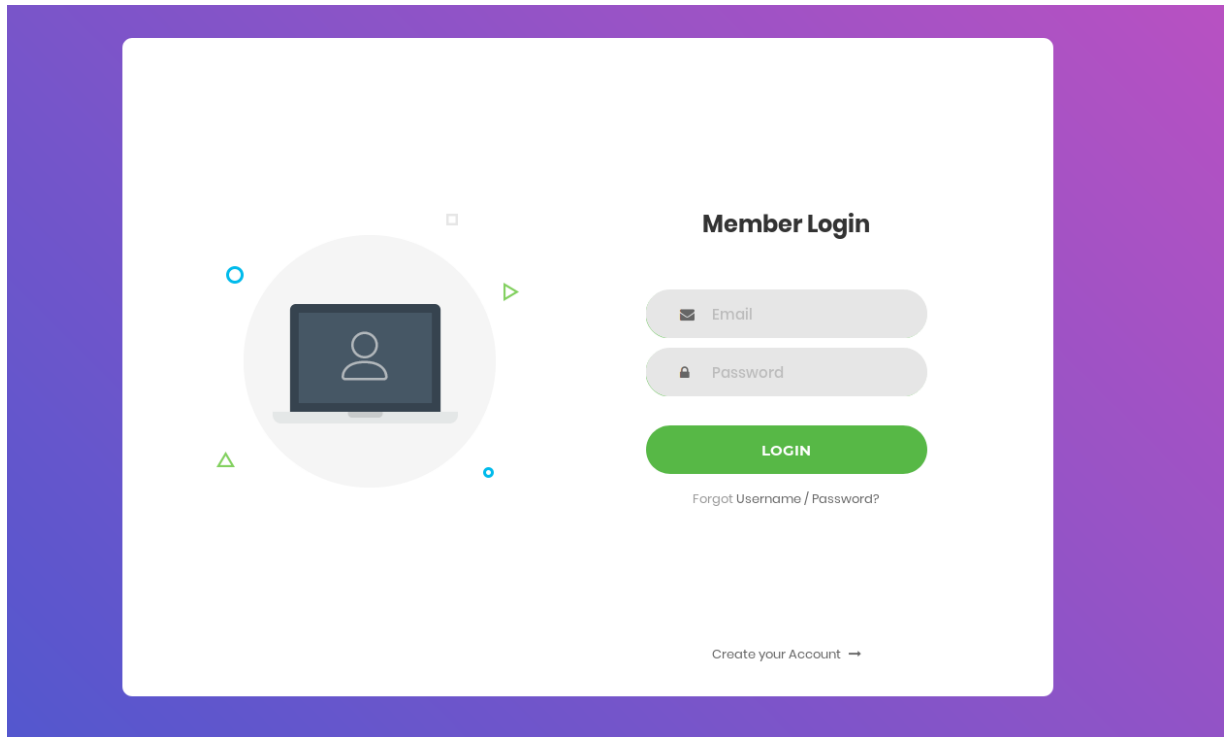


GET	/healthrecord	Lấy danh sách Health Record
POST	/healthrecord	Lấy thông tin chi tiết của Health Record
POST	/createhealthrecord	Tạo một Health Record mới
POST	/requesthealthrecord	Bác sĩ Request một Health Record
POST	/healthrecord/detail	Lấy chi tiết của một Health Record

*Bảng 4.6. Danh sách các Route và mô tả*

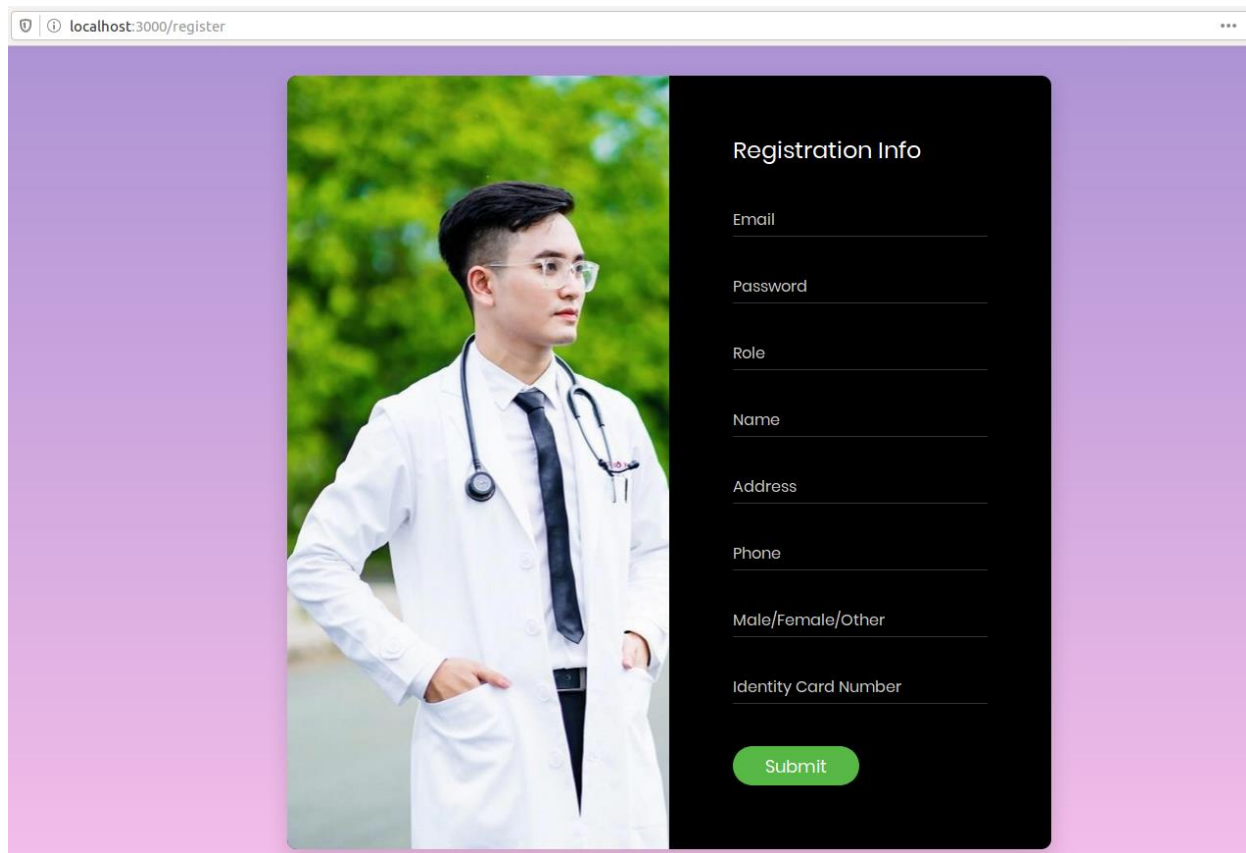
*g. Xây dựng tầng Front-end*

**Giao diện trang Login**



*Hình 4.16. Giao diện trang Login*

## Giao diện trang Đăng ký



localhost:3000/register

Registration Info

Email

Password

Role

Name

Address

Phone


Male/Female/Other

Identity Card Number


Submit

Hình 4.17. Giao diện trang Đăng ký

## Giao diện trang Profile của Doctor (Bác sĩ)



[HOME](#) [HEALTH RECORD](#) [DOCTORS](#) [PATIENTS](#) [REQUEST](#) [PROFILE](#) [LOGOUT](#)



User Info

HCMC

0932658717

Male

Cardiology

Single


Director

Edit

Save

Hình 4.18. Giao diện trang Profile của Doctor

## Giao diện trang danh sách bệnh nhân (Patient)



[HOME](#) [HEALTH RECORD](#) [DOCTORS](#) [PATIENTS](#) [REQUEST](#) [PROFILE](#) [LOGOUT](#)

Name	Career	Identity Card	Action
Nguyen Sang 02	Dev 2	123456083	<div>Request Info</div>
Nguyen Sang 03	Dev 2	123456084	<div>Request Info</div>

Hình 4.19. Giao diện trang danh sách bệnh nhân

## Giao diện trang danh sách Bác sĩ (Doctor)


 <span>HOME</span> <span>HEALTH RECORD</span> <span>DOCTORS</span> <span>PATIENTS</span> <span>REQUEST</span> <span>PROFILE</span> <span>LOGOUT</span>			
Name	Specialist	Title	Action
Nguyen Sang 01	Cardiology	Director	<button>Request Info</button>

Copyright © 2019 All rights reserved | This template is made with ♥ by [Colorlib](#)



Hình 4.20. Giao diện trang danh sách bác sĩ

## Giao diện trang danh sách bệnh án (Health Record)



MEDCARE

HEALTH SOLUTION

HOME

HEALTH RECORD

DOCTORS

PATIENTS

REQUEST

PROFILE

LOGOUT

Request


Health Record ID	Owner	Action
1	123456083	<div>Request HR</div>
2	123456084	<div>Request HR</div>

Requested

Health Record ID	Owner	Action
1	123456083	<div>View Detail</div>

Hình 4.21. Giao diện trang danh sách bệnh án

## Giao diện trang Update thông tin bệnh án (Health Record)




[HOME](#) [HEALTH RECORD](#) [DOCTORS](#) [PATIENTS](#) [REQUEST](#) [PROFILE](#) [LOGOUT](#)

Hight	<input type="text"/>	Tuoi <th>thai</th>	thai	<input type="text"/>
Hatthu0	<input type="text"/>	Hattruong0	<input type="text"/>	
Tieucau0	<input type="text"/>	Tq0	<input type="text"/>	
Aptt0	<input type="text"/>	Fibrinogen0	<input type="text"/>	
Ast0	<input type="text"/>	Alt0	<input type="text"/>	
Creatinin0	<input type="text"/>	Ure0	<input type="text"/>	
Auric0	<input type="text"/>	Ldh0	<input type="text"/>	
Damnieu0	<input type="text"/>	Damnieu24h0	<input type="text"/>	
Protein0	<input type="text"/>	Albumin0	<input type="text"/>	
Bilirubintp0	<input type="text"/>	Bilirubintt0	<input type="text"/>	
Conclusion	<input type="text"/>			

Submit

Hình 4.22. Giao diện trang cập nhật thông tin bệnh án

## Giao diện trang Request



[HOME](#) [HEALTH RECORD](#) [DOCTORS](#) [PATIENTS](#) [REQUEST](#) [PROFILE](#) [LOGOUT](#)

Be Requested


Request ID	Requester Role	Resource Owner Role	Resource Type	Resource ID	Requester ID	Resource Owner ID	Status	Action
1	Doctor	Patient	PatientInfo	123456083	123456123	123456083	Accepted	<button>Reject</button>
3	Doctor	Patient	HealthRecord	1	123456124	123456083	Accepted	<button>Reject</button>
4	Doctor	Patient	HealthRecord	1	123456123	123456083	Accepted	<button>Reject</button>

Requested


Request ID	Requester Role	Resource Owner Role	Resource Type	Resource ID	Requester ID	Resource Owner ID	Status
5	Patient	Doctor	DoctorInfo	123456123	123456083	123456123	New

Hình 4.23. Giao diện trang Request

## Giao diện trang Profile của Patient (Bệnh nhân)



[HOME](#) [HEALTH RECORD](#) [DOCTORS](#) [PATIENTS](#) [REQUEST](#) [PROFILE](#) [LOGOUT](#)



User Info

HCMC

0932658717

Male

Dev

Single

Edit

Save

Hình 4.24. Giao diện trang Profile của Patient

## Giao diện trang danh sách bác sĩ



MEDCARE

HEALTH SOLUTION

HOME

HEALTH RECORD

DOCTORS

PATIENTS

REQUEST

PROFILE

LOGOUT

Name	Specialist	Title	Action
Nguyen Sang 01	Cardiology	Director	<div>Request Info</div>
Nguyen	Cardiology	Sub Director	<div>Request Info</div>

Hình 4.25. Giao diện trang danh sách bác sĩ

## Giao diện trang Request của Patient (Bệnh nhân)



MEDCARE

HEALTH SOLUTION

HOME

HEALTH RECORD

DOCTORS

PATIENTS

REQUEST

PROFILE

LOGOUT

Be Requested

Request ID	Requester Role	Resource Owner Role	Resource Type	Resource ID	Requester ID	Resource Owner ID	Status	Action
1	Doctor	Patient	PatientInfo	123456083	123456123	123456083	Accepted	<div>Reject</div>
3	Doctor	Patient	HealthRecord	1	123456124	123456083	Accepted	<div>Reject</div>
4	Doctor	Patient	HealthRecord	1	123456123	123456083	Accepted	<div>Reject</div>


Requested

Request ID	Requester Role	Resource Owner Role	Resource Type	Resource ID	Requester ID	Resource Owner ID	Status
5	Patient	Doctor	DoctorInfo	123456123	123456083	123456123	New

Hình 4.26. Giao diện trang Request của Patient



## Giao diện danh sách các Health Record của bệnh nhân



HOMEHEALTH RECORDDOCTORSPATIENTSREQUESTPROFILELOGOUT

Create Health Record

healthRecordId	1
hight	
tuoithai	
hatthu0	
hattruong0	
tieucau0	
tq0	
aptt0	
fibrinogen0	
ast0	
alt0	
creatinin0	
ure0	
auric0	
ldh0	
damnieu0	
damnieu24h0	
protein0	
bilirubintp0	
bilirubintt0	
owner	123456083
conclusion	

Hình 4.27. Giao diện trang chi tiết Health Record của bệnh nhân

### ***h. Tính bất biến của dữ liệu***

Mỗi khi thực hiện một Transaction, dữ liệu đều được lưu lại trên Blockchain và sẽ không thể chỉnh sửa. Mỗi một Transaction đều sẽ có một lịch sử giao dịch và sẽ không thể bị làm giả bởi các thành phần trong hệ thống.

Date, Time	Entry Type	Participant	
2019-12-15, 19:30:33	AddAsset	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 18:19:20	UpdateAsset	123456083 (Patient)	<a href="#">view record</a>
2019-12-15, 18:19:17	UpdateAsset	123456083 (Patient)	<a href="#">view record</a>
2019-12-15, 18:18:47	AddAsset	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 17:55:00	UpdateAsset	123456083 (Patient)	<a href="#">view record</a>
2019-12-15, 17:54:57	UpdateAsset	123456083 (Patient)	<a href="#">view record</a>
2019-12-15, 17:51:33	AddAsset	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 17:44:41	AddAsset	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 16:46:38	ActivateCurrentIdentity	none	<a href="#">view record</a>
2019-12-15, 16:46:35	IssueIdentity	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 16:46:29	AddParticipant	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-14, 11:16:33	UpdateAsset	123456084 (Patient)	<a href="#">view record</a>

*Hình 4.28. Hình ảnh Transaction trong hệ thống*

Ví dụ về dữ liệu của một Transaction:

```
{
  "$class": "org.hyperledger.composer.system.UpdateAsset",
  "resources": [
    {
      "$class": "org.basic.server.Request",
      "requestId": "4",
      "requesterRole": "Doctor",
      "resourceOwnerRole": "Patient",
      "resourceType": "HealthRecord",
      "status": "Accepted",
    }
  ]
}
```

```

"resourceId": "1",

"owner": "resource:org.basic.server.Doctor#123456123",

"resourceOwner": "resource:org.basic.server.Patient#123456083"

}

],

"targetRegistry":

"resource:org.hyperledger.composer.system.AssetRegistry#org.basic.server.Request"

,

"transactionId":

"1548dc92abf5ab9ec617804e53d9d6a9a781a4bb0cc8ada6a2707ced80043637",

"timestamp": "2019-12-15T11:19:20.106Z"

}

```

The screenshot shows a web application interface for Hyperledger Composer. A modal window titled 'Historian Record' is open, displaying a detailed view of a transaction. The background shows a table of transactions with columns for Date, Time, Transaction, and Participant.

Date, Time	Transaction	Participant	Action
2019-12-15, 19:30:33		NetworkAdmin	<a href="#">view record</a>
2019-12-15, 18:19:20		083 (Patient)	<a href="#">view record</a>
2019-12-15, 18:19:17		083 (Patient)	<a href="#">view record</a>
2019-12-15, 18:18:47		NetworkAdmin	<a href="#">view record</a>
2019-12-15, 17:55:00		083 (Patient)	<a href="#">view record</a>
2019-12-15, 17:54:57		083 (Patient)	<a href="#">view record</a>
2019-12-15, 17:51:33	AddAsset	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 17:44:41	AddAsset	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 16:46:38	ActivateCurrentIdentity	none	<a href="#">view record</a>
2019-12-15, 16:46:35	IssueIdentity	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-15, 16:46:29	AddParticipant	admin (NetworkAdmin)	<a href="#">view record</a>
2019-12-14, 11:16:33	UpdateAsset	123456084 (Patient)	<a href="#">view record</a>

The 'Historian Record' modal shows the following transaction details:

```

1 {
2   "class": "org.hyperledger.composer.system.UpdateAsset",
3   "resources": [
4     {
5       "class": "org.basic.server.Request",
6       "requestId": "4",
7       "requesterRole": "Doctor",
8       "resourceOwnerRole": "Patient",
9       "resourceType": "HealthRecord",
10      "status": "Accepted",
11      "resourceId": "1",
12      "owner": "resource:org.basic.server.Doctor#123456123",
13      "resourceOwner": "resource:org.basic.server.Patient#123456083"
14    }
15  ],
16  "timestamp": "2019-12-15T11:19:20.106Z"
17 }

```

Hình 4.29. Hình ảnh chi tiết về một Transaction

#### **4.4. Kết chương**

Từ mô hình ứng dụng, tác giả hoàn thành việc xây dựng hệ thống quản lý bệnh án điện tử dựa trên công nghệ Blockchain. Khóa luận mô tả khá chi tiết về các đặc điểm của mô hình và thực nghiệm của khóa luận.

Đồng thời, khóa luận cũng đã đưa ra giả sử thực tế và chạy thực nghiệm với giả sử này. Nó đã đưa đến kết quả hoàn toàn thỏa mãn với giả sử thực tế được đưa ra.

Hơn thế, kết quả của phần thực nghiệm này, tác giả đưa ra kết luận như sau:

Chỉ những thành phần người tham gia trong hệ thống mới có thể thao tác. Như vậy những dữ liệu nhạy cảm sẽ không dễ dàng tiết lộ.

Với hệ thống này, dữ liệu sẽ không thể làm giả mạo nhờ công nghệ Blockchain.

## **Chương 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Chương này tác giả kết luận những gì đã làm được cũng như chưa làm được và hướng phát triển của hệ thống.

### **5.1. Đánh giá sản phẩm**

#### **5.1.1. Đã làm được**

Trong đề tài này, tôi đã làm được một hệ thống quản lý bệnh án điện tử đơn giản mà một bệnh viện cần sử dụng công nghệ Blockchain để lưu trữ dữ liệu.

Những nội dung đã làm được:

- Tìm hiểu và cài đặt thành công Công nghệ Blockchain cho ứng dụng quản lý bệnh án điện tử
- Sử dụng kết hợp hai nền tảng Blockchain là Hyperledger Fabric và Hyperledger Composer vào việc chứng thực quá trình giao dịch
- Chạy thử nghiệm trên nền tảng Hyperledger Fabric và Hyperledger Composer
- Các thao tác CRUD
- Lưu trữ dữ liệu lên Blockchain
- Phân quyền nhằm tăng tính riêng tư và phân hóa dữ liệu người dùng. Các dữ liệu nhạy cảm sẽ không bị dễ dàng chia sẻ ra ngoài nhờ việc phân cấp và phân quyền này.
- Sử dụng kết hợp MongoDB giúp cho người dùng thao tác đơn giản hơn
- Bộ dữ liệu sẽ không dễ dàng bị giả mạo khi áp dụng công nghệ Blockchain nhờ cơ chế đồng thuận giữa các Peer.

#### **5.1.2. Chưa làm được**

Thời gian thực hiện các Transaction còn hơi chậm, hiệu suất chưa cao

Chưa thực hiện được việc kết hợp đa bệnh viện

Phát triển thêm các chức năng như tầm soát thai kỳ, đặt lịch hẹn, ...

Chưa thực hiện được việc kết hợp ra đơn thuốc,...

## **5.2. Hướng phát triển**

Thông qua việc kiểm thử thành công ứng dụng công nghệ Blockchain vào mô hình quản lý bệnh án điện tử, khóa luận có thể xác định được các đối tượng cần có trong một mạng lưới Blockchain. Đồng thời, tác giả thông qua quá trình nghiên cứu của bản thân về công nghệ Blockchain giúp nhận ra và hiểu hơn về các ưu điểm và nhược điểm của công nghệ này.

Phát triển thêm các chức năng như ra đơn thuốc, kết hợp khám bảo hiểm...

Thực hiện triển khai trên nhiều hệ thống tổ chức bệnh viện

Ngoài ra, cần phát triển thêm phần chứng nhận độ uy tín của các thành phần tham gia vào hệ thống.

Cần tích hợp thêm các chức năng như đặt lịch trực tuyến, tính tiền, thanh toán...

## TÀI LIỆU THAM KHẢO

Tài liệu tiếng Việt:

1. “Building an electronic health record management system using Blockchain.” – Phong Tan Vu, Ngoc Vo Minh.
2. “Tham luận quá tải bệnh viện” - Bộ Y Tế, Cục quản lý khám chữa bệnh.

Tài liệu tiếng Anh:

1. “An Introduction to Hyperledger” 2018.
2. “Blockchain in Education”, C. a. Grech, in Joint Research Centre, 2017.
3. “Databases and Blockchain, The Difference Is In Their Purpose And Design”, Vince Tabora.
4. “Hyperledger Technologies & Permissioned Blockchains,” in Hyperledger Hanoi Meetup, Dec 21, 2017.
5. “Hyperledger Fabric Components — Technical Context” Moses Sam Paul, 2018
6. “Introduction to Hyperledger Composer”, 2018
7. “Mastering Bitcoin: unlocking digital cryptocurrencies”, M. Antonopoulos, O'Reilly Media, 2014.
8. “Medicalchain - A Blockchain for electronic health records”, Cryt Bytes Tech, Nov 17, 2017.
9. “Secure and Trustable Electronic Medical Records Sharing using Blockchain”, Alevtina Alevtina Dubovitskaya, Zhigang Xu, Samuel Ryu, Michael Schumacher, Fusheng Wang
10. “Series about Hyperledger”, Kipalog, 2018. [Online]. Available: <https://kipalog.com/posts/Series-about-Hyperledger/>.
11. “The History of Blockchain, Pt 1. First Concepts and Their Creators” – Lumi Wallet Blog, Medium.