

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**



**Phan Anh Dũng**

**ỨNG DỤNG MÔ HÌNH DỮ LIỆU ĐỒ THỊ  
TRONG PHÁT TRIỂN MẠNG THÔNG TIN SỨC KHỎE**

**LUẬN VĂN THẠC SĨ KỸ THUẬT**  
**(Theo định hướng ứng dụng)**

**HÀ NỘI - NĂM 2023**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

---



**Phan Anh Dũng**

**ỨNG DỤNG MÔ HÌNH DỮ LIỆU ĐỒ THỊ  
TRONG PHÁT TRIỂN MẠNG THÔNG TIN SỨC KHỎE**

**Chuyên ngành: Khoa học máy tính**

**Mã số: 8.48.01.01**

**LUẬN VĂN THẠC SĨ KỸ THUẬT  
(Theo định hướng ứng dụng)**

**NGƯỜI HƯỚNG DẪN KHOA HỌC**

**PGS.TS. HOÀNG HỮU HẠNH**

**HÀ NỘI – NĂM 2023**

## LỜI CAM ĐOAN

Tôi xin cam đoan rằng luận văn "Ứng dụng mô hình dữ liệu đồ thị trong phát triển mạng thông tin sức khỏe" là sáng kiến nghiên cứu của bản thân tôi.

Trong luận văn này, ngoài các tài liệu tham khảo đã được trích dẫn, tôi cam kết không sử dụng bất kỳ nghiên cứu hoặc tài liệu của người khác mà không được trích dẫn đầy đủ theo quy định. Toàn bộ nội dung của luận văn được tổng hợp từ nhiều nguồn tài liệu và quan điểm cá nhân của tôi. Các kết quả và số liệu được trình bày trong luận văn là chính xác và chưa từng được công bố trong bất kỳ công trình hoặc luận văn nào khác.

Luận văn này chưa từng được sử dụng để đăng ký hoặc nhận bất kỳ bằng cấp nào tại bất kỳ cơ sở giáo dục nào. Tôi cam đoan sự chính trực và trung thực trong quá trình nghiên cứu và viết luận văn của mình.

Trân trọng.

*Hà Nội, ngày    tháng    năm 2023.*

Tác giả luận văn

**Phan Anh Dũng**

## LỜI CẢM ƠN

Em xin dành lời cảm ơn chân thành nhất đến PGS.TS. Hoàng Hữu Hạnh vì đã tận tình hướng dẫn em trong quá trình thực hiện luận văn thạc sĩ về chủ đề "Ứng dụng mô hình dữ liệu đồ thị trong phát triển mạng thông tin sức khỏe" tại Học viện Công nghệ Bưu chính Viễn thông.

Thời gian qua, sự đồng hành của thầy trong công việc nghiên cứu và hướng dẫn em đã giúp cho luận văn của em được hoàn thành một cách tốt nhất. Trong quá trình làm luận văn, em đã nhận được sự chỉ bảo kiến thức, truyền dạy kinh nghiệm rất tận tâm của thầy. Những bài giảng của thầy vô cùng sâu sắc và giá trị, đã giúp em hiểu sâu hơn về mô hình dữ liệu đồ thị và ứng dụng của nó trong việc làm đồ án liên quan đến lĩnh vực y tế, sức khỏe.

Em cũng xin gửi lời cảm ơn sâu sắc tới các thầy, các cô Học viện Công nghệ Bưu chính Viễn thông vì đã hướng dẫn, giảng dạy, cung cấp cho em nhiều tài liệu, kiến thức, kinh nghiệm trong suốt quá trình học thạc sĩ tại đây. Những kết quả đạt được từ luận văn của em đã mang lại nhiều giá trị và ứng dụng thực tế trong lĩnh vực sức khỏe. Em tin rằng nghiên cứu của em sẽ đóng góp vào việc nâng cao chất lượng dịch vụ y tế và hỗ trợ cho người dùng trong việc quản lý sức khỏe của mình.

Một lần nữa, em xin chân thành cảm ơn PGS.TS. Hoàng Hữu Hạnh đã hướng dẫn em trong suốt thời gian qua. Sự giúp đỡ và động viên của Thầy đã đóng góp rất lớn để em hoàn thành luận văn của mình một cách thành công. Em cảm thấy may mắn và tự hào khi được làm việc và học tập dưới sự hướng dẫn của Thầy.

Em xin chân thành cảm ơn!

*Hà Nội, ngày    tháng    năm 2023*

Học viên

**Phan Anh Dũng**

## MỤC LỤC

<b>LỜI CAM ĐOAN .....</b>	<b>i</b>
<b>LỜI CẢM ƠN .....</b>	<b>ii</b>
<b>MỤC LỤC .....</b>	<b>iii</b>
<b>DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT .....</b>	<b>v</b>
<b>DANH MỤC HÌNH ẢNH .....</b>	<b>v</b>
<b>DANH MỤC BẢNG BIỂU .....</b>	<b>vi</b>
<b>MỞ ĐẦU .....</b>	<b>1</b>
<b>CHƯƠNG 1. CƠ SỞ LÝ THUYẾT .....</b>	<b>5</b>
<b>1.1. Mô hình và khoa học dữ liệu .....</b>	<b>5</b>
<i>1.1.1. Mô hình dữ liệu quan hệ .....</i>	<i>5</i>
<i>1.1.2. Mô hình dữ liệu hướng đối tượng .....</i>	<i>8</i>
<i>1.1.3. Mô hình dữ liệu đồ thị .....</i>	<i>13</i>
<i>1.1.4. Phân tích, đánh giá các cơ sở dữ liệu .....</i>	<i>17</i>
<i>1.1.5. Phương pháp mô hình hóa dữ liệu .....</i>	<i>17</i>
<i>1.1.6. So sánh các loại đồ thị: RDF, Knowledge Graph và Property Graph .....</i>	<i>18</i>
<i>1.1.7. Khoa học dữ liệu đồ thị .....</i>	<i>25</i>
<b>1.2. Các hệ thống quản trị dữ liệu đồ thị .....</b>	<b>26</b>
<i>1.2.1. GraphDB .....</i>	<i>26</i>
<i>1.2.2. Neo4j .....</i>	<i>29</i>
<i>1.2.3. Jena .....</i>	<i>33</i>
<i>1.2.4. So sánh và đánh giá .....</i>	<i>35</i>
<b>1.3. Kết luận chương .....</b>	<b>36</b>

<b>CHƯƠNG 2. BÀI TOÁN PHÁT TRIỂN MẠNG THÔNG TIN SỨC KHỎE VÀ THIẾT KẾ HỆ THỐNG</b> .....	38
<b>2.1. Giới thiệu bài toán mạng thông tin sức khỏe</b> .....	38
2.1.1. Bài toán phát triển mạng thông tin sức khỏe .....	38
2.1.2. Các chức năng cơ bản của mạng thông tin sức khỏe .....	39
2.1.3. Các hướng tiếp cận và giải quyết bài toán .....	42
2.1.4. Đề xuất giải pháp giải quyết.....	51
<b>2.2. Áp dụng phát triển mạng thông tin sức khỏe</b> .....	52
2.2.1. Các bước xây dựng hệ thống dữ liệu theo mô hình dữ liệu đồ thị.....	52
2.2.2. Thiết kế CSDL đồ thị trên Neo4j.....	53
<b>2.3. Xây dựng hệ thống phần mềm truy vấn và trực quan hóa</b> .....	57
2.3.1. Lập Use case. ....	57
2.3.2. Lập biểu đồ lớp. ....	62
2.3.3. Lập biểu đồ hoạt động. ....	62
2.3.4. Lập biểu đồ tuần tự. ....	63
<b>2.4. Kết luận chương</b> .....	63
<b>CHƯƠNG 3. KẾT QUẢ VÀ ĐÁNH GIÁ</b> .....	65
<b>3.1. Thu thập và chia sẻ thông tin sức khỏe</b> .....	65
<b>3.2. Kịch bản kiểm thử hệ thống</b> .....	66
<b>3.3. Triển khai cài đặt và kết quả</b> .....	67
<b>3.4. Lập trình các module</b> .....	68
<b>3.5. Thực nghiệm hệ thống</b> .....	71
<b>3.6. Đánh giá kết quả hệ thống</b> .....	73
<b>3.7. Kết luận chương</b> .....	75

<b>KẾT LUẬN</b> .....	76
<b>TÀI LIỆU THAM KHẢO</b> .....	77

## **DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT**

<b>Viết tắt</b>	<b>Tiếng Anh</b>	<b>Tiếng Việt</b>
CSDL		Cơ Sở Dữ Liệu
MHDL		Mô Hình Dữ Liệu

## **DANH MỤC HÌNH ẢNH**

Hình 1- 1: Minh họa mô hình dữ liệu quan hệ [11] .....	5
Hình 1- 2: Minh họa mô hình dữ liệu hướng đối tượng.....	9
Hình 1- 3: Minh họa mô hình dữ liệu đồ thị [12] .....	13
Hình 1- 4: Ví dụ về RDF [13] .....	19
Hình 1- 5: Ví dụ về Knowledge Graph [14] .....	20
Hình 1- 6: Ví dụ về Property Graph [15] .....	21
Hình 2- 1: Usecase Quản lý Thông tin Người dùng .....	58
Hình 2- 2: Usecase Quản lý thông tin Bệnh nhân.....	59
Hình 2- 3: Usecase Tìm kiếm thông tin bệnh án.....	60
Hình 2- 4: Usecase Phân phối công việc cho Bác sỹ .....	61
Hình 2- 5: Biểu đồ lớp.....	62
Hình 2- 6: Biểu đồ hoạt động Phân công công việc cho Bác sỹ .....	62
Hình 2- 7: Biểu đồ tuần tự quá trình Login.....	63

Hình 3- 1: Màn hình đăng nhập hệ thống .....	72
Hình 3- 2: Màn hình công việc bác sỹ .....	72
Hình 3- 3: Màn hình lịch khám chi tiết (cho Bác sỹ).....	73
Hình 3- 4: Màn hình chi tiết thông tin bệnh nhân.....	73

## **DANH MỤC BẢNG BIỂU**

Bảng 1- 1: Ưu điểm và nhược điểm của GraphDB.....	35
Bảng 1- 2: Ưu điểm và nhược điểm của Neo4j.....	35
Bảng 1- 3: Ưu điểm và nhược điểm của Jena .....	36
Bảng 2- 1: So sánh 3 mô hình dữ liệu về tính mở rộng .....	44
Bảng 2- 2: So sánh 3 mô hình dữ liệu về tính linh hoạt.....	45
Bảng 2- 3: So sánh 3 mô hình dữ liệu về tính tương thích, tính dễ sử dụng.....	46
Bảng 2- 4: So sánh 3 mô hình dữ liệu về hiệu suất.....	47
Bảng 2- 5: So sánh 3 mô hình dữ liệu về tính bảo mật .....	48
Bảng 2- 6: So sánh 3 mô hình dữ liệu về tính tương tác.....	49
Bảng 2- 7: Các đối tượng trong mạng thông tin sức khỏe .....	53



## MỞ ĐẦU

### 1. Tính cấp thiết của đề tài

Trong những năm gần đây, cùng với sự phát triển kinh tế thì nhu cầu tư vấn khám chữa bệnh của người dân gia tăng nhanh chóng. Các công tác xã hội được khuyến khích phát triển trong lĩnh vực y tế nhằm hỗ trợ các y bác sĩ giảm bớt áp lực công việc, nâng cao hiệu quả điều trị. Cùng với sự phát triển của công nghệ thông tin, các kênh truyền thông xã hội ra đời thiết lập xu hướng truyền thông mới dần thay thế các kênh truyền thông cổ điển như báo chí, phát thanh, truyền hình... trong việc kết nối, giao tiếp giữa bệnh viện - bệnh nhân, giữa các bệnh viện và giữa các bệnh nhân. Trong đó thì mạng thông tin được đón nhận và được sử dụng phổ biến. Các mạng thông tin hiện hành thiếu các nội dung, các công cụ cần thiết chuyên sâu để đáp ứng các nhu cầu giao tiếp, kết nối về y tế. Vì vậy, nhu cầu có một mạng thông tin sức khỏe chuyên biệt để làm nơi giao lưu, trao đổi thông tin là cấp thiết. [1][2] [3]

Mạng thông tin sức khỏe có ý nghĩa quan trọng trong việc phát triển nền y tế nước nhà. Nâng cao khả năng khám, chữa bệnh ban đầu của người bệnh. Tăng cường hiểu biết, thông tin về bệnh nhân từ sớm của các y bác sĩ. Mạng thông tin sức khỏe có thể lưu trữ, đăng tải nhiều thông tin về y tế và các hoạt động liên quan một cách nhanh chóng, cập nhật với chi phí tối thiểu, có sức lan tỏa cao. Điều đó giúp ích lớn cho việc phổ biến thông tin về y tế, sức khỏe đến người dân và các y bác sĩ nhằm nâng cao hiệu quả khám chữa bệnh. Người bệnh dễ dàng tiếp cận các thông tin chính thống từ ngành Y tế, tránh được các thông tin xấu, sai lệch về y tế nhằm bảo vệ sức khỏe cá nhân và gia đình. [4][5]

Mạng thông tin đã và đang được nghiên cứu, ứng dụng nên có rất nhiều các tài liệu, công cụ hỗ trợ trong việc xây dựng, bảo trì một hệ thống như vậy. Mô hình dữ liệu đồ thị và khoa học dữ liệu đồ thị (Graph Data Science) là một hướng nghiên cứu ứng dụng được sử dụng hiện nay trong việc hỗ trợ phát triển mạng thông tin với các đặc điểm điển hình như lượng dữ liệu lớn, mức độ tăng trưởng nhanh, đa dạng

các loại quan hệ dữ liệu. Qua đó thấy được tính khả thi của việc phát triển mạng thông tin sức khỏe. [6][7]

## 2. Tổng quan vấn đề nghiên cứu

Vấn đề nghiên cứu của đề tài: ứng dụng mô hình dữ liệu đồ thị và khoa học dữ liệu dựa trên đồ thị (graph data science) trong phát triển mạng thông tin sức khỏe.

Đầu tiên, cần nghiên cứu về mạng thông tin. Cần làm rõ thế nào là mạng thông tin? Các đặc trưng riêng biệt cần có của mạng thông tin trong lĩnh vực y tế hay còn gọi là mạng thông tin sức khỏe. Xác định đối tượng, mục tiêu sử dụng của mạng thông tin này trong việc chia sẻ thông tin nhằm nâng cao chất lượng chăm sóc y tế. Các vấn đề khó khăn thực tiễn đặt ra trong sự phát triển tin học y tế của các khối bệnh viện Việt Nam hiện nay. Các bệnh viện gặp khó khăn trong việc tích hợp các dữ liệu, thông tin y tế. Bệnh nhân gặp khó trong việc tiếp cận các nguồn thông tin y tế chính thống. Sau đó, luận văn cần xây dựng ra được cơ sở lý thuyết của việc sử dụng cơ sở dữ liệu đồ thị bằng cách nghiên cứu các mô hình, các hướng tiếp cận nhằm giải quyết bài toán thiết lập mạng thông tin. Các cơ sở dữ liệu nào đã triển khai trên thực tế? Các cơ sở dữ liệu quan hệ, hướng đối tượng và đồ thị có ưu điểm, nhược điểm là gì? Khả năng phát triển, mở rộng về sau của các CSDL đó có thuận lợi không, có phù hợp với nhu cầu đặt ra của mạng thông tin không? Sau đó quá trình tiến hành, thử nghiệm có thuận lợi không?

Luận văn cần đặt ra các nghiên cứu hệ quản trị dữ liệu đồ thị. Khi phát triển mạng thông tin thì mô hình dữ liệu là rất quan trọng. Nó quyết định đến việc thiết kế, phát triển hệ thống sau này. Việc tìm kiếm các nền tảng liên quan đến dữ liệu đồ thị với các công cụ, cộng đồng hỗ trợ mạnh mẽ là rất quan trọng. Luận văn cần khảo sát một số nền tảng hệ quản trị CSDL để lựa chọn ra nền tảng phù hợp với các tiêu chí đặt ra (Neo4j, Jena, GraphDB...). Hệ quản trị CSDL này cần lưu trữ các mô hình dữ liệu đồ thị trên đó và có các công cụ truy vấn, trực quan mạnh để hỗ trợ phát triển các ứng dụng. [6]

Tiếp sau đó, luận văn sẽ nghiên cứu về việc phát triển mạng thông tin sức khỏe cụ thể là xây dựng website và cơ sở dữ liệu để cung cấp các truy xuất thông tin về sức khỏe, gồm các đối tượng thông tin được liên kết với nhau: bác sĩ, phòng khám, hệ thống thông tin ngành, chuyên ngành, các cấp y tế dự phòng v.v... Làm rõ hệ thống này sử dụng mô hình và hệ thống dữ liệu nào. Những mô hình đó có điểm mạnh/hạn chế gì để có thể vận dụng tốt vào trong hệ thống và ứng dụng. Phần tiếp theo của luận văn sẽ là phân áp dụng mô hình dữ liệu đồ thị xây dựng nên hệ thống truy xuất thông tin sức khỏe này. Sau cùng là sự so sánh, đánh giá khả năng, tính phù hợp của mô hình dữ liệu đồ thị với các mô hình dữ liệu khác trong việc phát triển mạng thông tin sức khỏe.

### **3. Mục đích nghiên cứu**

Về mặt lý luận:

- Nghiên cứu các mô hình dữ liệu phổ biến: mô hình dữ liệu hướng đối tượng, mô hình dữ liệu quan hệ và mô hình dữ liệu đồ thị.
- Hiểu rõ cơ sở lý thuyết về cơ sở dữ liệu đồ thị và khoa học dữ liệu đồ thị và các ứng dụng của nó.
- Nắm dc phương pháp, kinh nghiệm xây dựng, phát triển các mô hình thông tin sức khỏe từ các nước trên thế giới.

Về thực tiễn:

- Phân tích, so sánh, đánh giá, thử nghiệm trên các mô hình dữ liệu với tập dữ liệu đầu vào ban đầu. Tiêu chí đánh giá là tốc độ phản hồi, khả năng đáp ứng, khả năng mở rộng, tính an toàn dữ liệu, ...
- Sử dụng, vận hành Neo4j, xây dựng ứng dụng linh hoạt, hoạt động với các chức năng cơ bản minh họa cho tính linh động của mô hình.
- Xây dựng một ứng dụng web và CSDL cung cấp thông tin ngành y tế, trong đó có các module chính: thu thập, chuyển đổi và lưu trữ dữ liệu dựa trên mô hình dữ liệu đồ thị; các module truy vấn và trực quan hoá dữ liệu dựa trên các

ưu điểm của mô hình dữ liệu đồ thị nhằm cung cấp thông tin cho người dùng trong hệ thống một cách hiệu quả với các thông tin được liên kết.

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

Đối tượng nghiên cứu:

- Các mô hình dữ liệu đặc biệt là mô hình dữ liệu đồ thị, khoa học dữ liệu
- Hệ quản trị CSDL đồ thị Neo4j.
- Các công cụ để lập trình ứng dụng trong phát triển mạng thông tin.
- Ứng dụng trong phát triển mạng thông tin y tế và sức khỏe.

Phạm vi nghiên cứu:

- Nghiên cứu lý thuyết mô hình dữ liệu đồ thị, khoa học dữ liệu đồ thị và một số hệ quản trị CSDL đồ thị (Neo4j, Jena, GraphDB...).
- Xây dựng mô hình, kiểm thử phần mềm trong phạm vi một cơ quan y tế địa phương tuyến cơ sở.

#### **5. Phương pháp nghiên cứu**

Phương pháp nghiên cứu lý thuyết:

- Nghiên cứu cơ sở lý thuyết về các mô hình dữ liệu.
- Đọc và phân tích các tài liệu về mô hình dữ liệu đồ thị và các nghiên cứu liên quan về khoa học dữ liệu đồ thị.
- Nghiên cứu các thuật toán, phương pháp luận trong việc phát triển mạng thông tin ứng dụng hệ thống quản trị dữ liệu đồ thị.

Phương pháp thực nghiệm:

- Xây dựng, thử nghiệm và đánh giá độ hiệu quả của các mô hình dữ liệu.
- Xây dựng hệ thống mạng thông tin sức khỏe dựa trên mô hình dữ liệu đồ thị.
- Kiểm thử tính năng, đánh giá chất lượng sản phẩm.

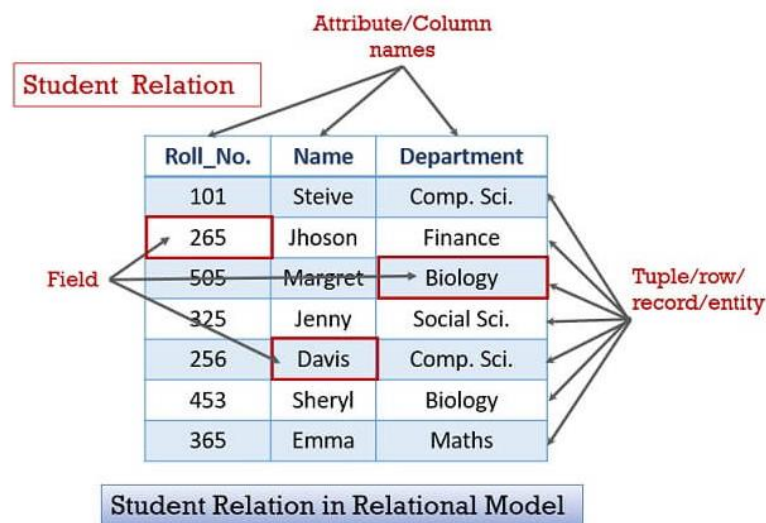
# CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

## 1.1. Mô hình và khoa học dữ liệu

### 1.1.1. Mô hình dữ liệu quan hệ.

#### a. Khái niệm mô hình dữ liệu quan hệ:

Mô hình dữ liệu (MHDL) quan hệ (RDM – Relational Data Model) được phát triển bởi Ted Codd của IBM vào những năm 1970 và được triển khai thương mại trong khoảng 10 năm sau để lưu trữ và xử lý dữ liệu trong cơ sở dữ liệu. RDM là mô hình sử dụng CSDL quan hệ. Trong đó, dữ liệu được biểu hiện dưới dạng một tập hợp các quan hệ, tương ứng với bảng giá trị trong đó mỗi quan hệ có các thuộc tính (attributes) và bộ giá trị (tuples) tương ứng với các cột và hàng. Mỗi bộ giá trị tượng trưng cho một thực thể hoặc mỗi quan hệ trong thế giới thực và tên của quan hệ cùng các thuộc tính cung cấp thông tin về ý nghĩa của từng bộ giá trị.



Hình 1- 1: Minh họa mô hình dữ liệu quan hệ [11]

CSDL quan hệ có nhiều ưu điểm, bao gồm khả năng xử lý dữ liệu lớn và khả năng truy xuất dữ liệu nhanh chóng. Nó cũng là CSDL phổ biến trong các hệ thống quản lý cơ sở dữ liệu quan hệ như MySQL, PostgreSQL và Oracle.

Tuy nhiên, CSDL quan hệ cũng có những hạn chế. Một số vấn đề thường gặp bao gồm khó khăn trong việc mô tả các mối quan hệ phức tạp và khó khăn trong việc

lưu trữ các dữ liệu phi cấu trúc. Điều này có thể dẫn đến hiệu suất chậm và khó khăn trong việc phân tích các dữ liệu phức tạp.

***b. Cấu trúc và thành phần của một cơ sở dữ liệu quan hệ:***

Một CSDL quan hệ bao gồm các thành phần sau:

**Bảng (Table):** Là đơn vị cơ bản nhất trong CSDL quan hệ, đại diện cho một tập hợp các bản ghi (records) có cùng cấu trúc. Mỗi bảng bao gồm một số cột (columns) để lưu trữ các thuộc tính (attributes) của đối tượng (entity) hoặc mối quan hệ (relationship) trong hệ thống.

**Cột (Column):** Mỗi cột trong bảng đại diện cho một thuộc tính của đối tượng hoặc mối quan hệ được lưu trữ trong bảng đó. Mỗi cột có một tên duy nhất và một kiểu dữ liệu (data type), như số nguyên (integer), chuỗi (string), ngày tháng (date) và địa chỉ email (email address) ...

**Dòng (Row):** Mỗi dòng trong bảng đại diện cho một bản ghi (record) chứa giá trị của các thuộc tính tương ứng. Mỗi dòng cũng có một khóa chính (primary key) đại diện cho giá trị duy nhất để xác định dòng đó trong bảng.

**Khóa chính (Primary key):** Là thuộc tính hoặc tập hợp các thuộc tính đại diện cho giá trị duy nhất để xác định mỗi dòng trong bảng. Khóa chính thường được sử dụng để liên kết các bảng với nhau thông qua các khóa ngoại (foreign key).

**Khóa ngoại (Foreign key):** Là thuộc tính trong một bảng đại diện cho giá trị của khóa chính trong một bảng khác. Khóa ngoại được sử dụng để liên kết các bảng với nhau, tạo ra mối quan hệ giữa các đối tượng và mối quan hệ trong hệ thống.

**Ràng buộc (Constraint):** Là các quy tắc để giới hạn giá trị được lưu trữ trong các cột của bảng. Ràng buộc bao gồm ràng buộc kiểu dữ liệu (data type constraint), ràng buộc duy nhất (unique constraint), ràng buộc không null (not null constraint) và ràng buộc khóa ngoại (foreign key constraint).

***c. Các phương thức truy vấn dữ liệu trong cơ sở dữ liệu quan hệ:***

**SELECT:** Truy vấn SELECT là phương thức truy vấn dữ liệu cơ bản nhất trong CSDL quan hệ. Truy vấn này được sử dụng để lựa chọn dữ liệu từ một hoặc nhiều bảng. Các điều kiện WHERE được sử dụng để giới hạn số lượng bản ghi được trả về.

**INSERT:** Truy vấn INSERT được sử dụng để thêm dữ liệu mới vào bảng. Nó cho phép chèn một hoặc nhiều bản ghi mới vào bảng chỉ định.

**UPDATE:** Truy vấn UPDATE được sử dụng để thay đổi dữ liệu đang tồn tại trong bảng. Nó cho phép cập nhật một hoặc nhiều bản ghi trong bảng chỉ định.

**DELETE:** Truy vấn DELETE được sử dụng để xóa dữ liệu khỏi bảng. Nó cho phép xóa một hoặc nhiều bản ghi trong bảng chỉ định.

**JOIN:** Truy vấn JOIN được sử dụng để kết hợp dữ liệu từ hai hoặc nhiều bảng vào một kết quả truy vấn duy nhất. Nó cho phép truy xuất dữ liệu từ các bảng liên quan đến nhau thông qua các khóa ngoại.

**GROUP BY:** Truy vấn GROUP BY được sử dụng để nhóm các bản ghi trong bảng thành các nhóm dựa trên một hoặc nhiều cột. Nó cho phép tính toán các hàm tổng hợp như SUM, COUNT, AVG, MAX và MIN trên các nhóm dữ liệu khác nhau.

**ORDER BY:** Truy vấn ORDER BY được sử dụng để sắp xếp các bản ghi trong kết quả truy vấn theo thứ tự tăng dần hoặc giảm dần của một hoặc nhiều cột.

**DISTINCT:** Truy vấn DISTINCT được sử dụng để loại bỏ các bản ghi trùng lặp trong kết quả truy vấn.

#### ***d. Ví dụ minh họa***

Xem xét một mối quan hệ HỌC SINH với các thuộc tính ROLL\_NO, NAME, ADDRESS, PHONE và AGE được hiển thị dưới đây.

ROLL_NO	NAME	ADDRESS	PHONE	AGE
1	RAM	DELHI	9455123451	18
2	RAMESH	GURGAON	9652431543	18
3	SUJIT	ROHTAK	9156253131	20
4	TÙNG	DELHI		18

Thuộc tính: Thuộc tính là các thuộc tính xác định một thực thể. ví dụ: ROLL\_NO, NAME, ADDRESS, PHONE và AGE

Lược đồ quan hệ: Một lược đồ quan hệ xác định cấu trúc của mỗi quan hệ và đại diện cho tên của mỗi quan hệ với các thuộc tính của nó. ví dụ: STUDENT (ROLL\_NO, NAME, ADDRESS, PHONE và AGE) là lược đồ quan hệ cho STUDENT. Nếu một lược đồ có nhiều hơn 1 quan hệ, nó được gọi là Lược đồ quan hệ.

Tuple: Mỗi hàng trong mỗi quan hệ được gọi là một tuple. Mỗi quan hệ trên chứa 4 bộ tuple, một trong số đó được hiển thị là:

1	RAM	DELHI	9455123451	18
---	-----	-------	------------	----

Cột: Cột đại diện cho tập hợp các giá trị cho một thuộc tính cụ thể. Cột ROLL\_NO được trích xuất từ mỗi quan hệ STUDENT.

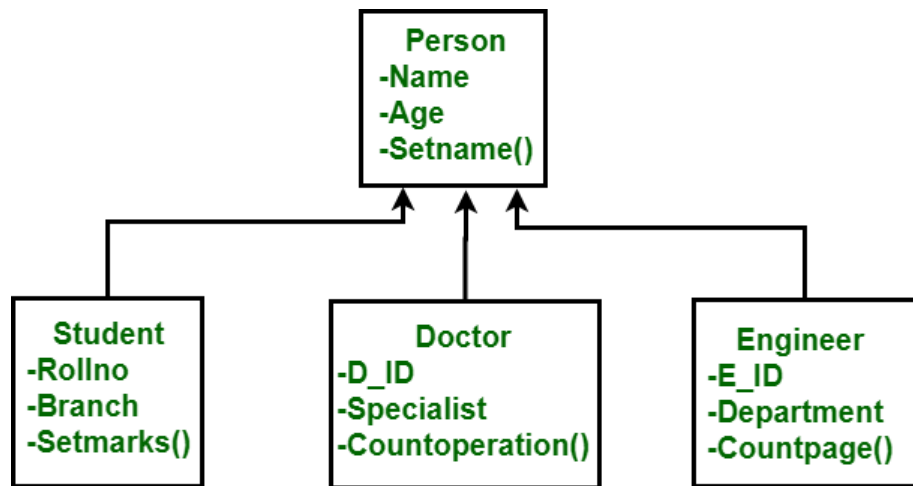
### *1.1.2. Mô hình dữ liệu hướng đối tượng*

#### ***a. Khái niệm mô hình dữ liệu hướng đối tượng***

Mô hình dữ liệu hướng đối tượng (Object-oriented data model) là một MHDL trong đó cơ sở dữ liệu là CSDL hướng đối tượng. Nó được biểu diễn dưới dạng các đối tượng (object), mỗi đối tượng là một thực thể của một lớp (class) cụ thể trong hệ thống. MHDL hướng đối tượng đưa ra một cách tiếp cận khác biệt so với MHDL quan hệ, trong đó các thực thể và mối quan hệ giữa chúng được biểu diễn bằng các đối tượng và phương thức của chúng.

CSDL hướng đối tượng được sử dụng rộng rãi trong các ứng dụng phần mềm, đặc biệt là trong các ứng dụng lớn và phức tạp. Nó giúp cho việc phát triển phần mềm trở nên dễ dàng hơn bởi vì nó cho phép phát triển theo kiểu đối tượng và cũng cho phép sử dụng lại các đối tượng đã được xây dựng.





Hình 1- 2: Minh họa mô hình dữ liệu hướng đối tượng.

Một số đặc điểm của CSDL hướng đối tượng bao gồm:

- Đối tượng (Object): Là thực thể trong hệ thống và có các thuộc tính (attribute) và phương thức (method) tương ứng.
- Lớp (Class): Là một mô tả chung về một tập hợp các đối tượng có thuộc tính và phương thức chung. Các đối tượng được tạo ra từ các lớp này.
- Kế thừa (Inheritance): Cho phép một lớp con (subclass) được tạo ra từ một lớp cha (superclass), kế thừa các thuộc tính và phương thức của lớp cha và có thể có thêm thuộc tính và phương thức của riêng nó.
- Đa hình (Polymorphism): Cho phép các đối tượng của các lớp khác nhau được sử dụng trong cùng một context và có thể có hành vi khác nhau dựa trên lớp của chúng.
- Trừu tượng (Abstraction): Cho phép che giấu chi tiết bên trong của một đối tượng, chỉ hiển thị các thuộc tính và phương thức cần thiết cho việc sử dụng đối tượng.

#### ***b. Cấu trúc và thành phần của một cơ sở dữ liệu hướng đối tượng***

CSDL hướng đối tượng bao gồm các thành phần sau:

Lớp (Class): là một khái niệm trừu tượng để đại diện cho một tập hợp các đối tượng có thuộc tính và hành vi tương tự. Mỗi lớp có một tên duy nhất và các thuộc tính, phương thức của lớp được định nghĩa trong phần khai báo lớp.

Đối tượng (Object): là một thực thể cụ thể của một lớp, có thể được tạo ra và sử dụng trong chương trình. Mỗi đối tượng có một tên duy nhất và có thể có các thuộc tính và phương thức của lớp mà nó thuộc về.

Thuộc tính (Attribute/Property): là các đặc điểm của đối tượng, được định nghĩa trong khai báo lớp và có thể được truy xuất bằng cách sử dụng tên của thuộc tính.

Phương thức (Method): là các hành động mà một đối tượng có thể thực hiện, được định nghĩa trong khai báo lớp và có thể được gọi bằng cách sử dụng tên của phương thức.

Kế thừa (Inheritance): là một khái niệm trong mô hình hướng đối tượng cho phép một lớp được kế thừa các thuộc tính và phương thức của một lớp khác.

Đa hình (Polymorphism): là một tính năng trong mô hình hướng đối tượng cho phép một phương thức có thể có nhiều hình thái khác nhau.

Đóng gói (Encapsulation): là một tính năng trong mô hình hướng đối tượng cho phép che giấu các chi tiết của lớp và chỉ cho phép truy cập thông qua các phương thức được định nghĩa.

Các thành phần này cùng nhau tạo thành một CSDL hướng đối tượng hoàn chỉnh, cho phép lập trình viên dễ dàng xây dựng và quản lý các đối tượng và tương tác giữa chúng trong ứng dụng của mình.

### ***c. Mỗi quan hệ giữa các đối tượng trong cơ sở dữ liệu hướng đối tượng***

Trong CSDL hướng đối tượng, các đối tượng có thể tương tác với nhau thông qua các mối quan hệ. Mối quan hệ giữa các đối tượng được biểu diễn bằng các khai

báo trong các lớp. Các mối quan hệ này có thể là mối quan hệ một-nhiều, nhiều-nhiều hoặc một-một.

- Mối quan hệ một - nhiều (one-to-many) là mối quan hệ giữa hai đối tượng, trong đó một đối tượng có thể có nhiều đối tượng liên quan. Ví dụ: một bộ phận trong công ty có thể có nhiều nhân viên.
- Mối quan hệ nhiều - nhiều (many-to-many) là mối quan hệ giữa hai đối tượng, trong đó mỗi đối tượng có thể liên quan đến nhiều đối tượng khác. Ví dụ: mỗi đơn hàng có thể có nhiều sản phẩm và mỗi sản phẩm có thể được đặt hàng nhiều lần trong các đơn hàng khác nhau.
- Mối quan hệ một - một (one-to-one) là mối quan hệ giữa hai đối tượng, trong đó một đối tượng chỉ liên quan đến một đối tượng khác. Ví dụ: một hộ dân chỉ có một địa chỉ.

#### ***d. Các phương thức truy vấn dữ liệu trong cơ sở dữ liệu hướng đối tượng***

Trong CSDL hướng đối tượng, để truy vấn và thao tác dữ liệu, cần sử dụng các phương thức truy vấn dữ liệu như sau:

Ngôn ngữ truy vấn đối tượng (Object Query Language - OQL): là một ngôn ngữ truy vấn dữ liệu đối tượng được thiết kế để truy vấn dữ liệu từ cơ sở dữ liệu đối tượng. OQL sử dụng cú pháp tương tự như SQL, tuy nhiên, nó cho phép truy vấn dữ liệu trực tiếp từ các đối tượng thay vì từ các bảng như trong SQL.

Ngôn ngữ truy vấn đối tượng dựa trên phương pháp (Method-based Object Query Language - MOQL): là một ngôn ngữ truy vấn đối tượng dựa trên phương thức. MOQL cho phép truy vấn dữ liệu bằng cách sử dụng các phương thức được định nghĩa trên các lớp đối tượng. Điều này giúp cho MOQL linh hoạt hơn OQL trong việc truy vấn dữ liệu, đặc biệt là khi xử lý dữ liệu phức tạp.

Ngôn ngữ truy vấn đối tượng phi cấu trúc (Object Query Language - Unstructured - OQLUS): là một ngôn ngữ truy vấn đối tượng được thiết kế để truy vấn dữ liệu phi cấu trúc. OQLUS cho phép truy vấn dữ liệu trên các đối tượng không

có cấu trúc nhất định, đồng thời cũng hỗ trợ truy vấn dữ liệu từ các loại đối tượng khác nhau như văn bản, hình ảnh, âm thanh, video...

Ngôn ngữ truy vấn đối tượng lồng nhau (Nested Object Query Language - NOQL): là một ngôn ngữ truy vấn đối tượng dựa trên các đối tượng lồng nhau. NOQL cho phép truy vấn dữ liệu bằng cách sử dụng các đối tượng lồng nhau, từ đó giúp cho việc truy vấn dữ liệu dễ dàng hơn.

Ngôn ngữ truy vấn đối tượng chủ động (Active Object Query Language - AOQL): là một ngôn ngữ truy vấn đối tượng được thiết kế để tương tác với các đối tượng động trong hệ thống. AOQL cho phép truy vấn dữ liệu bằng cách sử dụng các hoạt động của đối tượng, từ đó giúp cho việc truy vấn dữ liệu

#### ***e. Ví dụ minh họa***

Sử dụng hình minh họa Hình 1-2 để làm ví dụ

**Các đối tượng:** Một đối tượng là một sự trừu tượng của một thực thể trong thế giới thực hoặc có thể nói nó là một thể hiện của lớp. Các đối tượng đóng gói dữ liệu và mã vào một đơn vị duy nhất cung cấp khả năng trừu tượng hóa dữ liệu bằng cách ẩn các chi tiết triển khai khỏi người dùng. Ví dụ: Các trường hợp Student, Doctor, Engineer ở hình.

**Thuộc tính:** Một thuộc tính mô tả các thuộc tính của đối tượng. Ví dụ: Đối tượng là STUDENT và thuộc tính của nó là Roll no, Branch, Setmarks() trong lớp Student.

**Phương thức:** Phương thức đại diện cho hành vi của một đối tượng. Về cơ bản, nó đại diện cho hành động trong thế giới thực. Ví dụ: Tìm một STUDENT đánh dấu trong hình trên là Setmarks().

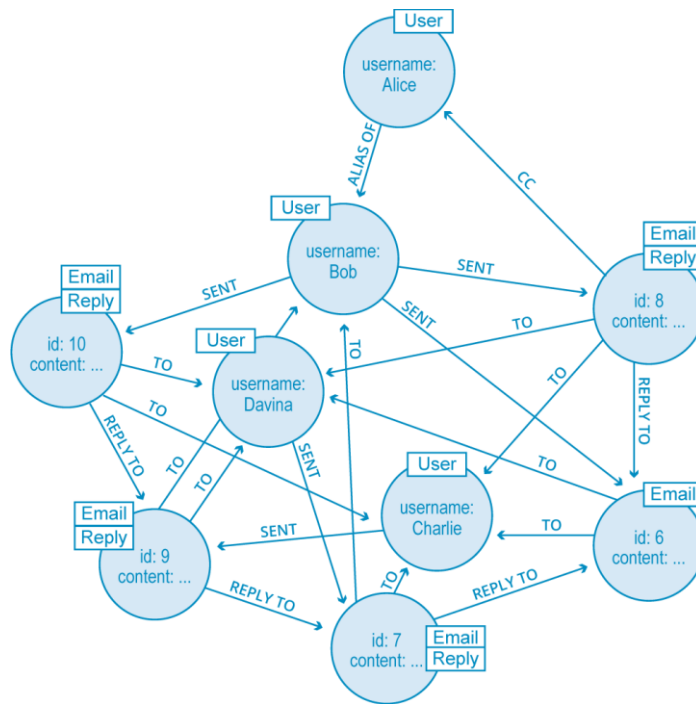
**Class:** Một lớp là một tập hợp các đối tượng tương tự có cấu trúc được chia sẻ, tức là các thuộc tính và hành vi, tức là các phương thức. Một đối tượng là một thể hiện của lớp. Ví dụ: Person, Student, Doctor, Engineer trong hình 1-2.

### 1.1.3. Mô hình dữ liệu đồ thị.

#### a. Khái niệm mô hình dữ liệu đồ thị

Mô hình dữ liệu đồ thị là một kiểu MHDL sử dụng CSDL đồ thị để biểu diễn và mô tả các mối quan hệ giữa các đối tượng thông qua các đỉnh (vertex) và cạnh (edge) trên đồ thị. Nó được sử dụng rộng rãi trong các ứng dụng liên quan đến mạng xã hội, công nghệ web, khoa học dữ liệu và truy vấn ngôn ngữ tự nhiên.

Một đồ thị là một tập hợp các đỉnh và cạnh được sắp xếp theo một số quy tắc. Mỗi đỉnh đại diện cho một đối tượng và các cạnh đại diện cho các mối quan hệ giữa các đối tượng đó. Các đối tượng và mối quan hệ có thể được đặc trưng bằng các thuộc tính (property) và trọng số (weight), tùy thuộc vào ứng dụng cụ thể. [20]



Hình 1- 3: Minh họa mô hình dữ liệu đồ thị [12]

MHDL đồ thị có thể được biểu diễn dưới dạng đồ thị hướng hoặc đồ thị vô hướng. Trong đồ thị hướng, các cạnh có hướng đi từ đỉnh này đến đỉnh khác. Trong đồ thị vô hướng, các cạnh không có hướng đi cụ thể, mà thường được sử dụng để biểu diễn các mối quan hệ đối xứng. [20]

Các đồ thị có thể được lưu trữ trong các cơ sở dữ liệu đồ thị, được thiết kế để hỗ trợ các truy vấn phức tạp dựa trên mối quan hệ giữa các đối tượng trên đồ thị.

### ***b. Cấu trúc và thành phần của một cơ sở dữ liệu đồ thị***

CSDL đồ thị bao gồm các thành phần sau:

- **Đỉnh (Vertex):** là các đối tượng được biểu diễn bằng các nút trong đồ thị. Mỗi đỉnh có một tên và một số thuộc tính (attribute) nhất định.
- **Cạnh (Edge):** là các mối quan hệ giữa các đỉnh trong đồ thị. Mỗi cạnh kết nối hai đỉnh và có một số thuộc tính nhất định.
- **Đồ thị (Graph):** là tổng thể của các đỉnh và các cạnh trong CSDL đồ thị.
- **Thuộc tính (Attribute):** là thông tin được lưu trữ cho mỗi đỉnh hoặc cạnh trong đồ thị. Thuộc tính có thể được sử dụng để cung cấp thông tin chi tiết về các đối tượng hoặc mối quan hệ trong đồ thị.
- **Đường đi (Path):** là một chuỗi các cạnh kết nối các đỉnh trong đồ thị. Đường đi được sử dụng để truy vấn các thông tin liên quan đến các mối quan hệ giữa các đỉnh trong đồ thị.
- **Đồ thị con (Subgraph):** là một phần của đồ thị gốc được chọn ra bằng cách lựa chọn một tập hợp con các đỉnh và các cạnh của đồ thị gốc.
- **Độ dài đường đi (Path length):** là số lượng các cạnh trong đường đi. Độ dài đường đi được sử dụng để tính toán độ phức tạp của các thuật toán truy vấn dữ liệu đồ thị.

Các thành phần trong CSDL đồ thị phức tạp hơn so với CSDL quan hệ và CSDL hướng đối tượng, và được sử dụng chủ yếu trong các ứng dụng liên quan đến phân tích mạng, tìm kiếm thông tin và xử lý ngôn ngữ tự nhiên.

### ***c. Các loại đồ thị***

Các loại đồ thị:

Đồ thị vô hướng (Undirected graph): là đồ thị mà các đỉnh được kết nối với nhau theo các cạnh không hướng. Tức là khi có một cạnh nối giữa đỉnh A và đỉnh B thì việc đi từ A đến B hoặc từ B đến A đều được cho phép.

Đồ thị có hướng (Directed graph): là đồ thị mà các đỉnh được kết nối với nhau theo các cạnh có hướng. Tức là khi có một cạnh nối từ đỉnh A đến đỉnh B thì việc đi từ A đến B và đi từ B đến A là hai hành động khác nhau và chỉ có đi từ A đến B được cho phép.

Đồ thị trọng số (Weighted graph): là đồ thị mà các cạnh được gán trọng số để biểu thị cho một thuộc tính nào đó của mối quan hệ giữa các đỉnh. Ví dụ, trong một đồ thị biểu diễn mạng lưới đường phố, trọng số có thể biểu thị cho chi phí của việc di chuyển giữa hai điểm trên đường.

Đồ thị vô hướng có trọng số (Undirected weighted graph): là đồ thị mà các đỉnh được kết nối với nhau theo các cạnh không hướng và mỗi cạnh có một trọng số.

Đồ thị có hướng có trọng số (Directed weighted graph): là đồ thị mà các đỉnh được kết nối với nhau theo các cạnh có hướng và mỗi cạnh có một trọng số.

#### ***d. Các phương thức truy vấn dữ liệu trong cơ sở dữ liệu đồ thị***

Trong CSDL đồ thị, có các phương thức truy vấn dữ liệu sau:

DFS (Depth First Search): DFS là phương thức tìm kiếm dữ liệu sử dụng một ngăn xếp (stack) để theo dõi các đỉnh đang được thăm. Phương thức này bắt đầu bằng cách chọn một đỉnh bất kỳ trong đồ thị, đánh dấu nó đã được thăm, và duyệt tất cả các đỉnh kề của đỉnh này, thực hiện điều này đệ quy. Nếu đỉnh kề chưa được thăm, thì cần đánh dấu nó đã được thăm và đẩy nó vào ngăn xếp.

BFS (Breadth First Search): BFS là phương pháp tìm kiếm đồ thị bằng cách theo chiều rộng, sử dụng một hàng đợi (queue) để theo dõi các đỉnh đang được duyệt. Phương pháp này bắt đầu từ đỉnh gốc, đánh dấu nó đã được thăm và đẩy nó vào hàng đợi. Tiếp theo, lấy một đỉnh khác từ đầu hàng đợi và duyệt tất cả các đỉnh kề của nó,

đánh dấu chúng đã được thăm và đẩy chúng vào cuối hàng đợi. Phương pháp này tiếp tục lặp lại việc này cho tất cả các đỉnh kề khác cho đến khi tất cả các đỉnh đều được thăm.

**Shortest Path Algorithms:** Có nhiều thuật toán tìm đường đi ngắn nhất trong đồ thị như Dijkstra, Bellman-Ford và Floyd-Warshall. Các thuật toán này được sử dụng để tìm đường đi ngắn nhất từ một đỉnh đến tất cả các đỉnh còn lại trong đồ thị hoặc tìm đường đi ngắn nhất giữa hai đỉnh cụ thể.

**Traversal Algorithms:** Có nhiều thuật toán duyệt đồ thị khác nhau, bao gồm DFS và BFS đã đề cập ở trên. Tuy nhiên, các thuật toán duyệt đồ thị khác cũng có thể được sử dụng, bao gồm thuật toán Topological Sort, thuật toán Strongly Connected Components, và thuật toán Minimum Spanning Tree.

**Matching Algorithms:** Matching Algorithms được sử dụng để tìm kiếm các cặp đỉnh trong đồ thị mà có một liên kết giữa chúng. Một số thuật toán phổ biến bao gồm thuật toán Max Flow-Min Cut.

#### **e. Ví dụ minh họa**

Sử dụng hình minh họa trên: Hình 1-3

Mô hình dữ liệu mẫu: Phát hiện gian lận trong liên lạc qua email. Trong ví dụ này, kiểm tra một ứng dụng phát hiện gian lận thông qua phân tích thông tin liên lạc qua email của User. Ứng dụng cụ thể này đang tìm kiếm hành vi lừa đảo và các mẫu gửi email đáng ngờ có thể cho thấy hành vi bất hợp pháp hoặc phi đạo đức.

Dữ liệu đồ thị có thể là bất cứ thứ gì. Trong ví dụ hình 1-3, Node nội dung email được đưa vào trung tâm để có thể theo dõi hiệu quả mối quan hệ của email đó với các User tương tác. Hệ thống sẽ theo dõi được ai đã viết email, gửi cho ai, CC, BCC cho ai. Không chỉ là 1 email mà nhiều email có thể được theo dõi và đánh giá. Theo thời gian, máy chủ ghi lại các thông tin đa dạng, phong phú giúp đánh giá chi tiết và sát nhất hoạt động của các user trong hệ thống. Từ đó phục vụ mục tiêu phát hiện gian lận thông qua email.



#### *1.1.4. Phân tích, đánh giá các cơ sở dữ liệu*

Các CSDL cần được phân tích, đánh giá đầy đủ và chi tiết. Đây là quá trình xem xét các yếu tố liên quan đến khả năng hoạt động, bảo trì và phát triển của các CSDL. Để phân tích và đánh giá một CSDL, luận văn xem xét các yếu tố dưới đây:

- Tính nhất quán (Consistency): Một CSDL được coi là nhất quán nếu các quan hệ giữa các đối tượng trong mô hình được định nghĩa rõ ràng và đúng đắn, không có sự mâu thuẫn hoặc trùng lặp trong dữ liệu.
- Hiệu suất (Performance): Một CSDL hiệu quả khi có khả năng xử lý dữ liệu một cách nhanh chóng và hiệu quả, đáp ứng được yêu cầu về thời gian và tài nguyên.
- Khả năng mở rộng (Scalability): Một CSDL được coi là có khả năng mở rộng nếu nó có thể mở rộng để đáp ứng nhu cầu của người dùng và số lượng dữ liệu lớn hơn.
- Khả năng bảo mật (Security): Một CSDL được coi là bảo mật nếu nó có khả năng bảo vệ dữ liệu khỏi các mối đe dọa bên ngoài.
- Tính linh hoạt (Flexibility): Một CSDL linh hoạt khi có khả năng thích nghi và điều chỉnh để đáp ứng các yêu cầu của người dùng.

Đối với CSDL quan hệ, thường tập trung vào các yếu tố như cấu trúc bảng, tính nhất quán, hiệu suất và mối quan hệ giữa các bảng. Đối với CSDL hướng đối tượng, thường tập trung vào cấu trúc đối tượng, tính nhất quán của các đối tượng và mối quan hệ giữa các đối tượng. Đối với CSDL đồ thị, thường tập trung vào cấu trúc của đồ thị, tính nhất quán và hiệu suất của các phương thức truy vấn dữ liệu.

#### *1.1.5. Phương pháp mô hình hóa dữ liệu*

Mô hình hóa dữ liệu là quá trình biến đổi dữ liệu từ dạng phi cấu trúc hoặc dạng cấu trúc chưa phù hợp thành các cấu trúc dữ liệu phù hợp với CSDL được chọn. Quá trình này giúp tạo ra một CSDL đầy đủ, dễ hiểu và có thể sử dụng để thực hiện các tác vụ khác nhau như truy vấn dữ liệu, phân tích và đánh giá.

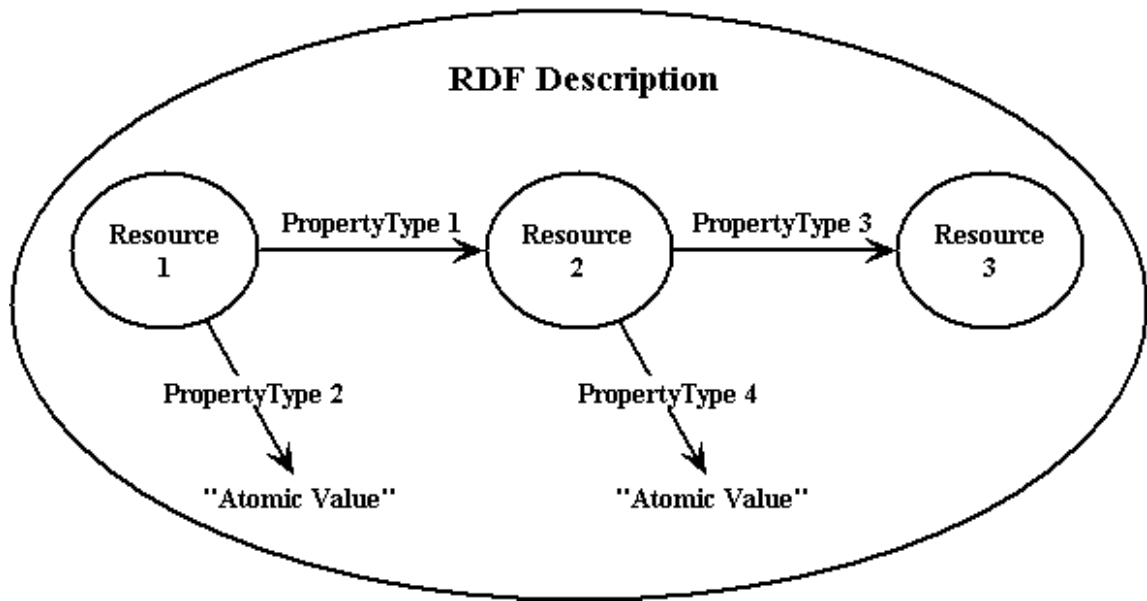
Để thực hiện mô hình hóa dữ liệu, có thể thực hiện theo các bước sau:

- Bước 1 - Thu thập và khảo sát dữ liệu: thu thập các nguồn dữ liệu và đánh giá tính hợp lệ, độ chính xác, độ hoàn thiện của dữ liệu.
- Bước 2 - Chuẩn bị dữ liệu: tiến hành rút trích và xử lý dữ liệu bằng các công cụ và phương pháp phù hợp, bao gồm tiền xử lý, chọn lọc và biến đổi dữ liệu.
- Bước 3 - Thiết kế CSDL: dựa trên mục đích và yêu cầu của dự án, thiết kế CSDL phù hợp bao gồm các đối tượng, thuộc tính, quan hệ và các ràng buộc giữa chúng.
- Bước 4 - Triển khai CSDL: triển khai CSDL vào hệ thống thông tin sử dụng các công cụ và phương pháp phù hợp.
- Bước 5 - Kiểm tra và đánh giá CSDL: kiểm tra và đánh giá tính đúng đắn, độ tin cậy và khả năng mở rộng của CSDL. Nếu cần, tiến hành điều chỉnh và cải thiện CSDL.
- Bước 6 - Duy trì và cập nhật CSDL: duy trì và cập nhật CSDL để đảm bảo tính đúng đắn và phù hợp với mục đích sử dụng.

#### *1.1.6. So sánh các loại đồ thị: RDF, Knowledge Graph và Property Graph*

##### *a. Trình bày về các loại đồ thị phổ biến:*

**Resource Description Framework (RDF):** là một loại đồ thị để mô tả thông tin về các tài nguyên trên web, chẳng hạn như các trang web, đối tượng, người dùng, sản phẩm hoặc sự kiện. RDF đặt trọng tâm vào các mối quan hệ giữa các tài nguyên và cho phép biểu diễn thông tin dưới dạng các ba mảng chính: tài nguyên, thuộc tính và giá trị. [15]

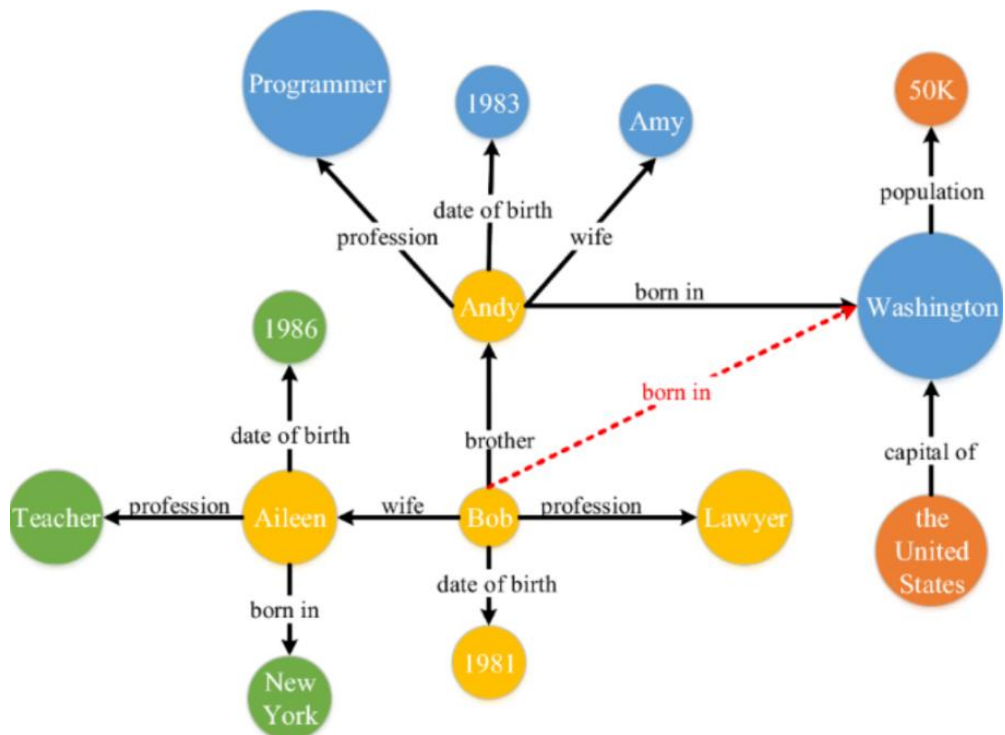


Hình 1- 4: Ví dụ về RDF [13]

Mỗi tài nguyên được đại diện bằng một URI (Uniform Resource Identifier), và thuộc tính được mô tả bằng các cặp URI và giá trị. Các giá trị có thể là các giá trị nguyên thủy như chuỗi hoặc số, hoặc cũng có thể là các URI của tài nguyên khác. RDF sử dụng các tri thức mô tả để cho phép các tài nguyên có thể được liên kết với nhau bằng cách sử dụng các mối quan hệ được mô tả trong các tri thức này.

RDF thường được sử dụng để biểu diễn dữ liệu trên web, trong các ngành như thư viện, khoa học và các lĩnh vực liên quan đến dữ liệu phức tạp. Nó cũng được sử dụng trong các ứng dụng như các công cụ tìm kiếm và các ứng dụng phân tích dữ liệu.

**Knowledge Graph:** là một dạng đồ thị, tập trung vào mô hình hóa tri thức của thế giới thực bằng cách xây dựng các mối quan hệ giữa các thực thể khác nhau. Điều này cho phép mô hình hóa và tương tác với tri thức dưới dạng đồ thị có cấu trúc, giúp cho việc tìm kiếm và truy vấn thông tin trở nên dễ dàng và nhanh chóng hơn.

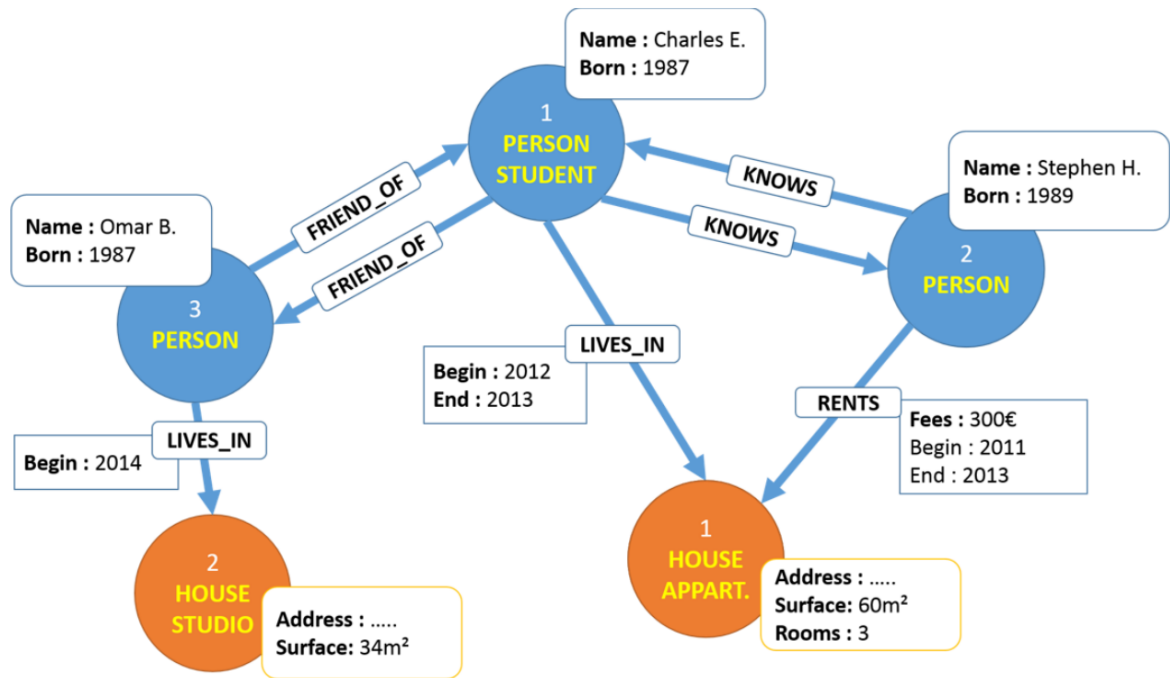


Hình 1- 5: Ví dụ về Knowledge Graph [14]

Knowledge Graph được xây dựng dựa trên các nguyên tắc của web semantic và Linked Data, cho phép các tri thức được biểu diễn dưới dạng các bộ ba RDF (Resource Description Framework). Các thực thể được mô tả bằng các URI (Uniform Resource Identifier) và các mối quan hệ giữa chúng được biểu diễn bằng các cặp thuộc tính-giá trị. Các tri thức này có thể được kết hợp từ nhiều nguồn khác nhau, tạo thành một Knowledge Graph phong phú và toàn diện.

Knowledge Graph được sử dụng trong nhiều lĩnh vực như tìm kiếm thông tin, trí tuệ nhân tạo, truy vấn thông tin y tế, và nhiều lĩnh vực khác. Ví dụ, Google sử dụng Knowledge Graph để cung cấp kết quả tìm kiếm có cấu trúc và thông tin liên quan hơn cho người dùng, trong khi các tổ chức y tế sử dụng Knowledge Graph để phân tích dữ liệu y tế và đưa ra quyết định chăm sóc sức khỏe tốt hơn.

**Property Graph:** là một dạng đồ thị sử dụng để mô hình hóa các mối quan hệ giữa các đối tượng. Trong Property Graph, các đối tượng được biểu diễn bởi các nút (node), các mối quan hệ giữa các đối tượng được biểu diễn bởi các cạnh (edge), và các thuộc tính được gán cho các đối tượng và các mối quan hệ.



Hình 1- 6: Ví dụ về Property Graph [15]

Mỗi nút trong Property Graph có một loại đối tượng và một tập hợp các thuộc tính, ví dụ như một nút "person" có thể có các thuộc tính như "name", "age" và "gender". Các cạnh cũng có thể có các thuộc tính, ví dụ như một cạnh "friend" giữa hai nút "person" có thể có các thuộc tính như "date met" và "frequency of contact".

Property Graph cho phép định nghĩa các quan hệ đa chiều giữa các đối tượng, cho phép biểu diễn các mối quan hệ phức tạp giữa các đối tượng. Property Graph cũng hỗ trợ truy vấn dữ liệu phức tạp bằng cách sử dụng ngôn ngữ truy vấn trực quan và dễ hiểu như Cypher.

Tương tự như các dạng đồ thị khác, Property Graph cũng có thể được sử dụng trong nhiều lĩnh vực khác nhau như mạng xã hội, kho dữ liệu lớn, hệ thống khuyến nghị, v.v.

### ***b. So sánh các cơ sở dữ liệu***

**Tính nhất quán (Consistency):** là khả năng giữ cho dữ liệu được đồng bộ và không bị xung đột trong các bảng, đối tượng hoặc đồ thị.

- CSDL quan hệ, tính nhất quán được đảm bảo bởi các ràng buộc toàn vẹn và các quan hệ khóa ngoại giữa các bảng. Nếu các quan hệ này được thiết lập đúng cách, thì dữ liệu trong các bảng sẽ luôn được đồng bộ và không bị xung đột.
- CSDL hướng đối tượng, tính nhất quán được đảm bảo bởi các phương thức hành động trên đối tượng. Khi các phương thức này được thực thi, dữ liệu được kiểm tra và đảm bảo rằng không có xung đột xảy ra.
- CSDL đồ thị, tính nhất quán được đảm bảo bởi các ràng buộc trên các đỉnh và cạnh trong đồ thị. Các ràng buộc này đảm bảo rằng các đỉnh và cạnh được kết nối đúng cách và không có xung đột xảy ra.

**Hiệu suất (Performance):** hiệu suất của các CSDL phụ thuộc vào nhiều yếu tố, bao gồm kích thước của dữ liệu, tần suất truy vấn và tính năng của phần mềm hỗ trợ.

- CSDL quan hệ có hiệu suất tốt nhất trong việc xử lý dữ liệu cấu trúc và phức tạp, đặc biệt là khi có các mối quan hệ phức tạp giữa các bảng dữ liệu.
- CSDL hướng đối tượng có thể có hiệu suất tốt trong việc xử lý các đối tượng và quan hệ phức tạp, nhưng có thể khó khăn trong việc xử lý dữ liệu cấu trúc.
- CSDL đồ thị thường được sử dụng để lưu trữ và truy vấn các dữ liệu có tính chất liên kết và phức tạp, nhưng hiệu suất có thể giảm nếu có nhiều cấu trúc phức tạp trong đồ thị.

Tuy nhiên, đây chỉ là những thước đo đơn giản và chung chung. Hiệu suất của mỗi CSDL cũng phụ thuộc vào nhiều yếu tố khác nhau, và có thể khác nhau tùy thuộc vào ứng dụng cụ thể. Vì vậy, việc lựa chọn CSDL phù hợp phải dựa trên các yêu cầu cụ thể của ứng dụng và sự đánh giá kỹ lưỡng của từng CSDL.

Với mạng thông tin/mạng xã hội, CSDL đồ thị sẽ là lựa chọn tốt nhất vì tính chất của mạng thông tin phù hợp với mô hình đồ thị. Trong mạng thông tin, các thành viên và mối quan hệ giữa chúng có thể được biểu diễn bằng các đỉnh và cạnh trong

đồ thị. Điều này cho phép các thuật toán đồ thị được sử dụng để tìm kiếm, phân tích và truy vấn dữ liệu mạng xã hội nhanh chóng và hiệu quả hơn so với các CSDL khác.

Tuy nhiên, nếu các tính năng khác như tính nhất quán, dễ hiểu và dễ bảo trì của mô hình quan hệ hay tính linh hoạt, tái sử dụng của mô hình hướng đối tượng có ưu thế hơn trong một số trường hợp. Do đó, việc lựa chọn CSDL phù hợp nhất phụ thuộc vào yêu cầu cụ thể của ứng dụng và các yếu tố khác như kích thước dữ liệu, tính chất của dữ liệu và mục tiêu của người sử dụng.

**Khả năng mở rộng (Scalability):** là một trong những yếu tố quan trọng trong việc thiết kế và triển khai các hệ thống dữ liệu. Đối với các ứng dụng có khối lượng dữ liệu lớn hoặc nhu cầu mở rộng cao trong tương lai, khả năng mở rộng là yếu tố quan trọng để đảm bảo hiệu suất và sức chứa của hệ thống.

- CSDL quan hệ, khả năng mở rộng có thể bị hạn chế bởi giới hạn về số lượng bảng và quan hệ giữa chúng, đặc biệt là khi các bảng phải tham chiếu đến nhau thông qua khóa ngoại. Tuy nhiên, với sự phát triển của công nghệ cơ sở dữ liệu và việc sử dụng các kỹ thuật như sharding hoặc partitioning, các hệ quản trị cơ sở dữ liệu quan hệ có thể mở rộng được đến hàng trăm hoặc hàng nghìn máy chủ.
- CSDL hướng đối tượng, khả năng mở rộng phụ thuộc vào cách mà các đối tượng được tạo và lưu trữ trong cơ sở dữ liệu. Tuy nhiên, với các công nghệ hiện đại như cơ sở dữ liệu đối tượng mã hóa, hệ thống phân tán, và NoSQL, các CSDL hướng đối tượng cũng có thể được mở rộng tương đối dễ dàng.
- CSDL đồ thị, khả năng mở rộng được xem là một trong những ưu điểm của mô hình này. Do đặc trưng của dữ liệu đồ thị, các cạnh và đỉnh có thể được lưu trữ trên các máy chủ khác nhau và được kết nối thông qua các liên kết mạng. Điều này cho phép CSDL đồ thị mở rộng tới hàng triệu hoặc thậm chí hàng tỉ đỉnh và cạnh.

**Khả năng bảo mật (Security):** Khả năng bảo mật trong các CSDL quan hệ, hướng đối tượng và đồ thị phụ thuộc vào cách triển khai của chúng. Tuy nhiên, có một số điểm khác nhau giữa các CSDL khi nói đến khả năng bảo mật.

- CSDL quan hệ có thể được bảo mật thông qua quản lý quyền truy cập vào cơ sở dữ liệu, mật khẩu, mã hóa dữ liệu, tường lửa và các biện pháp an ninh khác. Các hệ quản trị cơ sở dữ liệu thường có các tính năng bảo mật tích hợp sẵn như quản lý quyền truy cập, giám sát và ghi nhật ký.
- CSDL hướng đối tượng cũng có thể được bảo mật thông qua quản lý quyền truy cập, mã hóa và các biện pháp an ninh khác. Tuy nhiên, CSDL này có thể khó khăn trong việc bảo mật truy cập vào các đối tượng con của một đối tượng cha, đặc biệt là khi các đối tượng con này có quyền truy cập vào các thuộc tính riêng tư.
- CSDL đồ thị cũng có thể được bảo mật bằng các biện pháp an ninh tương tự như các mô hình khác, nhưng có thể khó khăn hơn trong việc quản lý quyền truy cập khi có hàng nghìn hoặc hàng triệu đỉnh và cạnh trong đồ thị. Ngoài ra, một điểm đáng chú ý là CSDL đồ thị có thể phải đối mặt với các vấn đề về bảo mật liên quan đến tính toàn vẹn và sự tin cậy của các nút và cạnh trong đồ thị.

**Tính linh hoạt (Flexibility):** là một khía cạnh quan trọng khi đánh giá sự phù hợp của các CSDL cho một ứng dụng cụ thể. Trong môi trường phát triển phần mềm, tính linh hoạt có thể đề cập đến khả năng thích nghi với thay đổi của yêu cầu của khách hàng, khả năng mở rộng để xử lý tải lớn và sự linh hoạt trong thiết kế để thay đổi cấu trúc dữ liệu một cách dễ dàng.

- CSDL đồ thị thường được xem là có tính linh hoạt cao hơn so với CSDL quan hệ và hướng đối tượng. Điều này bởi vì CSDL đồ thị cho phép thêm bớt các đỉnh và cạnh một cách dễ dàng và có thể xử lý được các mối quan hệ phức tạp giữa các đối tượng. Nó cũng cho phép tìm kiếm, phân tích và xử lý dữ liệu một cách hiệu quả trong các hệ thống lớn và phức tạp.



- CSDL quan hệ có khả năng linh hoạt tương đối trong việc mở rộng, tuy nhiên, nó có thể gặp khó khăn khi xử lý các truy vấn phức tạp trong các hệ thống dữ liệu lớn.
- CSDL hướng đối tượng có tính linh hoạt cao trong việc thay đổi cấu trúc dữ liệu và có thể được sử dụng cho các ứng dụng phức tạp. Tuy nhiên, nó có thể không đủ linh hoạt để xử lý các mối quan hệ phức tạp giữa các đối tượng và không phải lúc nào cũng thích hợp cho các ứng dụng lớn với lưu lượng dữ liệu cao.

#### *1.1.7. Khoa học dữ liệu đồ thị*

##### ***a. Khái niệm về khoa học dữ liệu đồ thị***

Khoa học dữ liệu đồ thị (Graph Data Science) là một lĩnh vực của khoa học dữ liệu tập trung vào nghiên cứu và phân tích dữ liệu đồ thị. Dữ liệu đồ thị là một loại dữ liệu bao gồm các đỉnh (nodes) và các cạnh (edges) nối giữa các đỉnh, đại diện cho mối quan hệ giữa các đối tượng.

Mục tiêu của khoa học dữ liệu đồ thị là tìm hiểu các quy luật và mô hình hóa dữ liệu đồ thị để đưa ra các phân tích và dự đoán chính xác hơn về các mối quan hệ giữa các đối tượng trong dữ liệu. Khoa học dữ liệu đồ thị có thể áp dụng cho nhiều lĩnh vực khác nhau như mạng xã hội, giao thông vận tải, tài chính và y tế.

##### ***b. Ứng dụng của khoa học dữ liệu đồ thị***

Khoa học dữ liệu đồ thị đang được áp dụng rộng rãi trong nhiều lĩnh vực, đặc biệt là trong các lĩnh vực có liên quan đến mạng xã hội, mạng thông tin, y tế, tài chính, v.v. Các ứng dụng của khoa học dữ liệu đồ thị bao gồm:

- Mạng xã hội: Khoa học dữ liệu đồ thị đã được sử dụng rộng rãi để phân tích và hiểu hành vi của người dùng trên mạng xã hội, giúp các doanh nghiệp có thể tăng hiệu quả quảng cáo và tương tác với khách hàng. Nó cũng có thể được sử dụng để phát hiện sự lan truyền của thông tin sai lệch và tin đồn trên mạng xã hội.

- **Mạng thông tin:** Khoa học dữ liệu đồ thị có thể giúp phát hiện các kết nối ẩn giữa các thông tin trên mạng thông tin, giúp cải thiện hiệu quả tìm kiếm và phân tích thông tin.
- **Y tế:** Khoa học dữ liệu đồ thị có thể được sử dụng để tìm ra các mối liên hệ giữa các bệnh, các tác nhân gây bệnh và những người mắc bệnh, giúp giảm thiểu nguy cơ lây nhiễm và cải thiện chất lượng chăm sóc sức khỏe.
- **Tài chính:** Khoa học dữ liệu đồ thị có thể giúp tăng cường phân tích rủi ro và quản lý tài sản bằng cách phát hiện các mối liên hệ giữa các yếu tố tài chính khác nhau.

## 1.2. Các hệ thống quản trị dữ liệu đồ thị

### 1.2.1. *GraphDB*

#### **a. Giới thiệu *GraphDB***

GraphDB là một hệ thống quản trị cơ sở dữ liệu đồ thị được phát triển bởi công ty Ontotext, hỗ trợ lưu trữ, truy vấn và xử lý dữ liệu đồ thị lớn. Được phát triển dựa trên cơ sở dữ liệu RDF (Resource Description Framework), GraphDB cung cấp các tính năng và công cụ phức tạp cho việc phân tích, tìm kiếm và truy vấn các mối quan hệ giữa các thực thể trong một đồ thị. [20]

Các tính năng của GraphDB bao gồm:

- **Lưu trữ dữ liệu đồ thị:** GraphDB hỗ trợ lưu trữ dữ liệu đồ thị với các định dạng dữ liệu như RDF/XML, N-Triples, Turtle, N3, RDF/JSON, JSON-LD và RDFa.
- **Các loại câu truy vấn đa dạng:** GraphDB hỗ trợ các loại câu truy vấn đa dạng như RDF/SPARQL, RDFS++, OWL Horst, OWL 2 QL, OWL 2 EL và OWL 2 RL.
- **Hỗ trợ các tính năng truy vấn thông minh:** GraphDB cung cấp các tính năng truy vấn thông minh như truy vấn định tuyến (routing query), truy vấn hội tụ (converging query) và truy vấn hợp nhất (merging query).

- Hỗ trợ các tính năng phân tích dữ liệu đồ thị: GraphDB cung cấp các tính năng phân tích dữ liệu đồ thị như phân tích mạng (network analysis), phân tích lân cận (proximity analysis) và phân tích cộng đồng (community analysis).
- Hỗ trợ khả năng mở rộng: GraphDB có khả năng mở rộng để xử lý các tập dữ liệu đồ thị lớn.
- Hỗ trợ giao diện đồ họa: GraphDB cung cấp một giao diện đồ họa cho phép người dùng trực quan hóa dữ liệu đồ thị và thực hiện các truy vấn đồ họa.

Tóm lại, GraphDB là một hệ thống quản trị cơ sở dữ liệu đồ thị mạnh mẽ và đa tính năng, cung cấp các tính năng và công cụ phức tạp cho việc lưu trữ, truy vấn và xử lý dữ liệu đồ thị lớn.

### ***b. Cách sử dụng GraphDB để lưu trữ và truy xuất dữ liệu đồ thị***

Để sử dụng GraphDB để lưu trữ và truy xuất dữ liệu đồ thị, các bước được tiến hành như sau:

- Bước 1: Cài đặt và khởi chạy GraphDB. Sau khi cài đặt xong, có thể khởi động GraphDB bằng cách chạy tệp tin **graphdb** trong thư mục **bin** của GraphDB.
- Bước 2: Tạo cơ sở dữ liệu đồ thị: Trong GraphDB, một cơ sở dữ liệu đồ thị được gọi là một repository. Có thể tạo một repository mới bằng cách truy cập vào giao diện quản lý của GraphDB và chọn **Create new repository**. Sau đó, cần chọn loại repository là **GraphDB-SE** (Standard Edition) và cấu hình các thông số như tên repository, địa chỉ và cổng của server.
- Bước 3: Nhập dữ liệu vào repository: Dữ liệu có thể nhập vào repository bằng cách sử dụng các công cụ và API của GraphDB. Các công cụ này cho phép nhập dữ liệu từ các nguồn khác nhau, như tệp tin RDF, SPARQL Endpoint, hoặc từ các hệ thống khác thông qua các giao thức như HTTP, REST API.
- Bước 4: Truy xuất dữ liệu đồ thị: Khi đã nhập dữ liệu vào repository, có thể truy xuất dữ liệu đồ thị bằng cách sử dụng các công cụ và API của GraphDB.

Các công cụ này cho phép thực hiện các truy vấn SPARQL để lấy ra các thông tin từ repository và hiển thị dữ liệu đồ thị dưới dạng đồ thị hoặc bảng.

- Bước 5: Quản lý repository: GraphDB cung cấp nhiều tính năng để quản lý các repository, bao gồm sao lưu và phục hồi dữ liệu, tối ưu hóa câu truy vấn, và đảm bảo tính bảo mật của dữ liệu.

### ***c. Một số ứng dụng của GraphDB trong thực tế***

Tìm kiếm thông tin khoa học: GraphDB được sử dụng để tìm kiếm thông tin khoa học với các bộ dữ liệu như PubMed và Web of Science. Các thông tin khoa học được lưu trữ dưới dạng đồ thị, giúp cho việc truy xuất thông tin trở nên dễ dàng hơn. [20][24]

Quản lý tri thức: GraphDB cũng được sử dụng để quản lý tri thức trong các hệ thống như phân loại sản phẩm và hệ thống khuyến nghị. Thông tin về các mục tiêu sản phẩm, đối tượng khách hàng, các thuộc tính sản phẩm và các thuộc tính khác được lưu trữ dưới dạng đồ thị, giúp cho việc phân tích và khai thác thông tin trở nên dễ dàng hơn.

Hệ thống thông tin y tế: GraphDB được sử dụng để lưu trữ và quản lý thông tin y tế, bao gồm thông tin về bệnh nhân, lịch sử bệnh án, thuốc và chế độ ăn uống. Thông tin y tế được lưu trữ dưới dạng đồ thị, giúp cho việc tìm kiếm thông tin, khai thác thông tin và đưa ra quyết định trở nên dễ dàng hơn.

Hệ thống phân tích dữ liệu: GraphDB cũng được sử dụng trong các hệ thống phân tích dữ liệu. Các bộ dữ liệu được lưu trữ dưới dạng đồ thị, giúp cho việc truy xuất và phân tích thông tin trở nên dễ dàng hơn. Ví dụ như trong phân tích mạng xã hội, GraphDB được sử dụng để lưu trữ và quản lý thông tin về các quan hệ giữa người dùng, giúp cho việc phân tích các quan hệ này trở nên dễ dàng hơn.

Hệ thống gợi ý: GraphDB cũng được sử dụng để xây dựng các hệ thống gợi ý. Thông tin về sở thích của người dùng, các sản phẩm đã mua và các sản phẩm tương tự được lưu trữ dưới dạng đồ thị, giúp cho việc xây dựng các hệ thống gợi ý trở

Ngoài ra, GraphDB cũng được sử dụng rộng rãi trong các ứng dụng liên quan đến trí tuệ nhân tạo (AI), bao gồm xử lý ngôn ngữ tự nhiên (NLP) và khai thác dữ liệu (data mining), giúp tối ưu hóa các công cụ tìm kiếm và phân tích dữ liệu đồ thị. Một ví dụ cụ thể là ứng dụng của GraphDB trong phân tích các mối quan hệ trong dữ liệu y tế để tìm ra các biểu hiện lâm sàng mới hoặc tối ưu hóa quy trình chẩn đoán bệnh.

Ngoài ra, GraphDB cũng được sử dụng trong các ứng dụng về mạng xã hội, truy xuất thông tin khách hàng và quản lý nguồn lực, đặc biệt là trong các công ty lớn với lượng dữ liệu lớn. GraphDB có khả năng tính toán phức tạp và phân tích các mối quan hệ giữa các đối tượng, giúp các doanh nghiệp tối ưu hóa việc quản lý dữ liệu và tối đa hóa giá trị của chúng.

Tóm lại, GraphDB là một hệ thống quản trị dữ liệu đồ thị mạnh mẽ, được sử dụng rộng rãi trong nhiều lĩnh vực và có nhiều tính năng ưu việt. Việc sử dụng GraphDB sẽ giúp tối ưu hóa quy trình lưu trữ và truy xuất dữ liệu đồ thị, giúp tăng hiệu quả và độ chính xác của các ứng dụng phân tích dữ liệu.

### *1.2.2. Neo4j*

#### ***a. Giới thiệu về Neo4j***

Neo4j là một hệ quản trị cơ sở dữ liệu đồ thị (Graph Database) mã nguồn mở được xây dựng trên nền tảng Java, có khả năng lưu trữ, xử lý và truy xuất dữ liệu đồ thị với tốc độ nhanh và hiệu quả cao. Điểm nổi bật của Neo4j là khả năng lưu trữ và quản lý các mối quan hệ phức tạp giữa các đối tượng, đồng thời hỗ trợ nhiều tính năng phong phú để truy xuất và xử lý dữ liệu đồ thị. [24][25]

Các tính năng của Neo4j bao gồm:

- Lưu trữ và quản lý dữ liệu đồ thị với khả năng mở rộng lên tới hàng tỉ đỉnh và cạnh.
- Hỗ trợ truy vấn dữ liệu đồ thị bằng ngôn ngữ truy vấn Cypher, với cú pháp đơn giản và dễ hiểu.

- Cung cấp nhiều tính năng độc đáo để truy xuất và xử lý dữ liệu đồ thị, bao gồm: tìm kiếm đường đi ngắn nhất, tìm kiếm nội dung liên quan, phân tích cộng đồng (community detection), phân tích trực quan dữ liệu (visualization), và nhiều tính năng khác.
- Hỗ trợ các công nghệ mới như blockchain và machine learning, giúp cải thiện tính bảo mật và khả năng phân tích dữ liệu.
- Cung cấp các giao diện lập trình ứng dụng (API) đa dạng và dễ sử dụng, bao gồm các API dành cho Java, .NET, Python, JavaScript, và nhiều ngôn ngữ khác.

Với những tính năng đặc biệt của mình, Neo4j đã được sử dụng rộng rãi trong các lĩnh vực như mạng xã hội, mạng thông tin, y tế, tài chính, và các ứng dụng khác liên quan đến dữ liệu đồ thị.

#### ***b. Cách sử dụng Neo4J để lưu trữ và truy xuất dữ liệu đồ thị***

Neo4J là một hệ quản trị cơ sở dữ liệu đồ thị (Graph Database) với khả năng lưu trữ dữ liệu đồ thị rất mạnh mẽ và truy vấn dữ liệu rất hiệu quả. Trong Neo4J, dữ liệu được tổ chức thành các nút (node) và các mối quan hệ (relationship) giữa các nút. [25]

Để lưu trữ dữ liệu đồ thị trong Neo4J, có thể sử dụng ngôn ngữ truy vấn Cypher. Để thêm một nút mới vào cơ sở dữ liệu, có thể sử dụng cú pháp:

```
CREATE (n:Label {property1: value1, property2: value2, ...})
```

Trong đó, **Label** là nhãn của nút, **property1**, **property2**, ... là các thuộc tính của nút và **value1**, **value2**, ... là giá trị của các thuộc tính đó. Ví dụ, để thêm một nút đại diện cho một người có tên là John và tuổi là 30, có thể sử dụng câu lệnh sau:

```
CREATE (:Person {name: 'John', age: 30})
```

Để thêm một mối quan hệ giữa hai nút, có thể sử dụng cú pháp:

```
MATCH (node1:Label1 {property1: value1}), (node2:Label2 {property2: value2})
CREATE (node1)-[relationship:Label {property1: value1, property2: value2, ...}]-
>(node2)
```

Trong đó, **Label** là nhãn của mỗi quan hệ, **property1**, **property2**, ... là các thuộc tính của mỗi quan hệ và **value1**, **value2**, ... là giá trị của các thuộc tính đó. Ví dụ, để thêm một mối quan hệ "friend" giữa John và Mary, có thể sử dụng câu lệnh sau:

```
MATCH (john:Person {name: 'John'}), (mary:Person {name: 'Mary'}) CREATE (john)-
[:friend]->(mary)
```

Để truy xuất dữ liệu trong Neo4J, có thể sử dụng câu lệnh truy vấn Cypher. Ví dụ, để truy xuất tất cả các nút có nhãn "Person", có thể sử dụng câu lệnh sau:

```
MATCH (p:Person) RETURN p
```

Để truy xuất các nút và mối quan hệ theo các tiêu chí cụ thể, có thể sử dụng các câu lệnh truy vấn Cypher phức tạp hơn.

Để lưu trữ dữ liệu đồ thị, Neo4J sử dụng cấu trúc lưu trữ gọi là "property graph". Cấu trúc này bao gồm các nút, quan hệ và thuộc tính. Mỗi nút trong Neo4J được định danh bởi một ID và có thể có một hoặc nhiều thuộc tính. Quan hệ giữa các nút được định nghĩa bởi các thuộc tính quan hệ, cũng có thể có một hoặc nhiều thuộc tính. Với cấu trúc lưu trữ này, Neo4J cho phép truy vấn dữ liệu đồ thị bằng ngôn ngữ truy vấn Cypher. Cypher cho phép người dùng thực hiện các truy vấn phức tạp trên dữ liệu đồ thị, bao gồm cả truy vấn theo chiều sâu và truy vấn theo chiều rộng.

Để truy xuất dữ liệu đồ thị trong Neo4J, người dùng có thể sử dụng giao diện người dùng đồ họa (Graphical User Interface - GUI) hoặc giao diện dòng lệnh. Giao diện đồ họa cung cấp cho người dùng một giao diện trực quan để truy vấn và hiển thị dữ liệu đồ thị, trong khi giao diện dòng lệnh cho phép người dùng sử dụng Cypher để truy vấn dữ liệu đồ thị một cách linh hoạt và tiện lợi. Neo4J cũng hỗ trợ các API để

tích hợp với các ứng dụng bên ngoài, giúp người dùng tương tác với dữ liệu đồ thị thông qua các ứng dụng khác.

### ***c. Một số ứng dụng của Neo4J trong thực tế***

Neo4j là một hệ thống quản lý cơ sở dữ liệu đồ thị phổ biến và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau. Dưới đây là một số ứng dụng của Neo4j trong thực tế: [25]

- **Mạng xã hội:** Neo4j đã được sử dụng để phát triển một số mạng xã hội lớn như LinkedIn và Facebook. Việc sử dụng Neo4j giúp tăng hiệu suất truy vấn và tính khả dụng của các ứng dụng mạng xã hội.
- **Tìm kiếm:** Neo4j có thể được sử dụng để xây dựng các hệ thống tìm kiếm dựa trên dữ liệu đồ thị. Ví dụ, các công ty tìm kiếm việc làm như Glassdoor đã sử dụng Neo4j để tìm kiếm các cơ hội việc làm và kết nối các ứng viên với nhà tuyển dụng.
- **Tài chính:** Neo4j đã được sử dụng trong nhiều ứng dụng tài chính, bao gồm phát hiện gian lận và xác định các mối quan hệ giữa các công ty và ngân hàng.
- **Y tế:** Neo4j có thể được sử dụng để phát triển các hệ thống quản lý thông tin y tế và phân tích dữ liệu y tế, bao gồm quản lý bệnh nhân, theo dõi tình trạng bệnh và phát hiện các mối quan hệ giữa các bệnh.
- **Logistics:** Neo4j đã được sử dụng để phát triển các hệ thống quản lý vận chuyển và phân tích dữ liệu logistics, bao gồm quản lý đường dẫn vận chuyển, tối ưu hóa tuyến đường và quản lý kho.
- **Bảo mật:** Neo4j có thể được sử dụng để xây dựng các hệ thống bảo mật dữ liệu, bao gồm quản lý danh sách kiểm soát truy cập, phát hiện xâm nhập và phân tích các mối quan hệ giữa các đối tượng bảo mật.

Trên đây là một số ứng dụng của Neo4j trong thực tế, và đây cũng là lý do tại sao Neo4j được coi là một trong những hệ thống quản lý cơ sở dữ liệu đồ thị phổ biến và được sử dụng rộng rãi nhất hiện nay.



### 1.2.3. Jena

#### a. Giới thiệu về Jena và các tính năng của nó

Apache Jena là một framework mã nguồn mở được sử dụng để phát triển các ứng dụng semantic web, đặc biệt là trong lĩnh vực Linked Data. Jena hỗ trợ việc xử lý RDF (Resource Description Framework), OWL (Web Ontology Language) và các tri thức liên quan. Nó cung cấp các công cụ cho việc truy vấn và phân tích dữ liệu đồ thị, tạo và truy vấn các triple store, và xây dựng các ứng dụng liên quan đến web semantic.

Các tính năng chính của Jena bao gồm:

- Hỗ trợ các định dạng RDF và OWL, bao gồm các định dạng RDF/XML, N-Triples, Turtle, RDF/JSON, và RDFa.
- Các API cho việc tạo, thêm, xóa và truy vấn các triple store.
- Các công cụ truy vấn SPARQL (SPARQL Protocol and RDF Query Language) cho phép truy vấn dữ liệu RDF.
- Các API cho việc xây dựng các ứng dụng semantic web như việc phát hiện và trích xuất tri thức từ các tài liệu RDF.
- Hỗ trợ các công cụ để xây dựng các ứng dụng web dựa trên dữ liệu đồ thị.
- Các tính năng nâng cao cho phân tích dữ liệu như reasoner và các công cụ hỗ trợ phát hiện tri thức bất đồng nhất.

Với những tính năng mạnh mẽ này, Jena đã được sử dụng rộng rãi trong các dự án semantic web, Linked Data và các ứng dụng quản lý tri thức.

#### b. Cách sử dụng Jena để lưu trữ và truy xuất dữ liệu đồ thị

Apache Jena là một công cụ phần mềm mã nguồn mở để lưu trữ và truy xuất dữ liệu đồ thị RDF. Jena cung cấp một thư viện đa nền tảng cho các ứng dụng RDF và SPARQL, cùng với các công cụ và tiện ích hỗ trợ các chức năng quản lý dữ liệu đồ thị như lập chỉ mục, truy vấn và thống kê.

Các bước sử dụng Jena để lưu trữ và truy xuất dữ liệu đồ thị như sau:

- Bước 1: Khai báo đối tượng Dataset và thiết lập các tùy chọn kết nối.
- Bước 2: Tạo một đối tượng Model để lưu trữ dữ liệu đồ thị RDF.
- Bước 3: Thêm dữ liệu vào Model bằng cách sử dụng các phương thức của lớp Model, như add() và read().
- Bước 4: Truy xuất dữ liệu bằng cách sử dụng SPARQL hoặc các phương thức của lớp Model, như listStatements() hoặc getProperty().

Jena cung cấp các công cụ và tiện ích để hỗ trợ các chức năng quản lý dữ liệu đồ thị, bao gồm cả chỉ mục hóa dữ liệu, tạo các kết nối đến các nguồn dữ liệu khác nhau và cung cấp các giao diện người dùng để truy vấn và quản lý dữ liệu đồ thị.

Ngoài ra, Jena cũng hỗ trợ các tiêu chuẩn và các công nghệ liên quan đến dữ liệu đồ thị như RDF, RDFS, OWL và SPARQL, giúp cho việc lưu trữ và truy xuất dữ liệu đồ thị trở nên tiện lợi và hiệu quả hơn.

### **c. Một số ứng dụng của Jena trong thực tế**

**Linked Data:** Jena hỗ trợ việc phát triển các ứng dụng Linked Data thông qua việc hỗ trợ các tiêu chuẩn và giao thức đang được sử dụng trong lĩnh vực này như RDF, RDFS, OWL, SPARQL, ... Nó cũng cung cấp các công cụ để tạo và quản lý các tập dữ liệu Linked Data.

**Phân tích văn bản:** Jena có thể được sử dụng để phân tích các tài liệu văn bản và tạo ra các đồ thị dữ liệu từ các thông tin được trích xuất. Nó cũng cung cấp các công cụ để phân tích các dữ liệu semi-structured và unstructured.

**Bảo mật và quản lý định danh:** Jena hỗ trợ các tính năng bảo mật và quản lý định danh của dữ liệu, cho phép người dùng đăng nhập, phân quyền truy cập và quản lý các quyền truy cập đến dữ liệu.

Quản lý tri thức: Jena có thể được sử dụng để quản lý tri thức và chuyển đổi dữ liệu giữa các định dạng khác nhau. Nó cung cấp các công cụ để tạo và quản lý các mô hình tri thức, từ đó giúp cho việc tổ chức và truy xuất dữ liệu trở nên dễ dàng hơn.

Mạng xã hội: Jena có thể được sử dụng để phân tích và quản lý các mạng xã hội. Nó cung cấp các công cụ để xác định các mối quan hệ giữa các thành viên và phân tích các hoạt động trên mạng xã hội. Nó cũng có thể được sử dụng để xây dựng các ứng dụng mạng xã hội.

#### 1.2.4. So sánh và đánh giá

##### a. GraphDB:

*Bảng 1- 1: Ưu điểm và nhược điểm của GraphDB*

Ưu điểm	Nhược điểm
<ul style="list-style-type: none"> <li>- Cung cấp tính năng lập chỉ mục dữ liệu và truy vấn đồ thị nhanh chóng.</li> <li>- Có khả năng mở rộng linh hoạt bằng cách thêm các node và cluster mới.</li> <li>- Hỗ trợ RDF và SPARQL, đây là 2 tiêu chuẩn quan trọng trong việc lưu trữ và truy vấn dữ liệu đồ thị.</li> <li>- Có cộng đồng phát triển lớn và hỗ trợ chính thức từ công ty Ontotext.</li> </ul>	<ul style="list-style-type: none"> <li>- Có giá cả đắt đỏ hơn so với các hệ thống khác.</li> <li>- Hạn chế về tính linh hoạt và tùy biến trong việc lập trình ứng dụng.</li> </ul>

##### b. Neo4j:

*Bảng 1- 2: Ưu điểm và nhược điểm của Neo4j*

Ưu điểm	Nhược điểm
<ul style="list-style-type: none"> <li>- Có tính năng lập chỉ mục dữ liệu và truy vấn đồ thị nhanh chóng.</li> </ul>	<ul style="list-style-type: none"> <li>- Hạn chế về các tiêu chuẩn RDF, SPARQL</li> </ul>

<ul style="list-style-type: none"> <li>- Hỗ trợ truy vấn đồ thị thông qua ngôn ngữ Cypher, cho phép người dùng thao tác với dữ liệu đồ thị một cách dễ dàng.</li> <li>- Có cộng đồng phát triển lớn và hỗ trợ chính thức từ công ty Neo4j.</li> </ul>	
---	--

**c. Jena:**

*Bảng 1- 3: Ưu điểm và nhược điểm của Jena*

Ưu điểm	Nhược điểm
<ul style="list-style-type: none"> <li>- Là một trong những hệ thống quản trị dữ liệu đồ thị miễn phí và mở mã nguồn.</li> <li>- Hỗ trợ RDF và SPARQL, đây là 2 tiêu chuẩn quan trọng trong việc lưu trữ và truy vấn dữ liệu đồ thị.</li> <li>- Có tính năng linh hoạt trong việc tạo các câu truy vấn SPARQL.</li> </ul>	<ul style="list-style-type: none"> <li>- Có tính năng lập chỉ mục và truy vấn đồ thị chậm hơn so với các hệ thống khác.</li> <li>- Khả năng mở rộng chưa tốt.</li> </ul>

### 1.3. Kết luận chương

Trong chương này, luận văn đã trình bày các cơ sở lý thuyết quan trọng liên quan đến nghiên cứu về mạng thông tin sức khỏe. Đầu tiên, luận văn đã nêu các mô hình dữ liệu quan hệ, hướng đối tượng và đồ thị, và nhận thức được ưu điểm và hạn chế của các CSDL ứng với từng mô hình. Luận văn đã tiến hành so sánh các dạng đồ thị phổ biến và chỉ ra được ưu điểm, nhược điểm của từng dạng. Từ đó có cơ sở để lựa chọn loại cơ sở dữ liệu phù hợp trong bài toán về mạng thông tin sức khỏe.

Chương này đã đặt nền tảng lý thuyết và kiến thức cần thiết để hiểu và nghiên cứu về mạng thông tin sức khỏe và hệ thống quản trị cơ sở dữ liệu đồ thị. luận văn đã tìm hiểu về các mô hình dữ liệu quan trọng, đánh giá và so sánh các hệ quản trị dữ

liệu quan hệ, hướng đối tượng, đồ thị và xác định rằng Neo4j là một giải pháp tối ưu cho mạng thông tin sức khỏe của luận văn.

Chương tiếp theo sẽ tập trung vào bài toán phát triển mạng thông tin sức khỏe và thiết kế hệ thống. Luận văn sẽ trình bày về các chức năng của mạng thông tin sức khỏe, quy trình nghiệp vụ, và các yêu cầu cần đạt của hệ thống. Luận văn cũng sẽ xem xét cách xây dựng hệ thống phần mềm truy vấn và trực quan hóa để hỗ trợ việc quản lý và tìm kiếm thông tin sức khỏe hiệu quả.

## CHƯƠNG 2. BÀI TOÁN PHÁT TRIỂN MẠNG THÔNG TIN SỨC KHỎE VÀ THIẾT KẾ HỆ THỐNG

### 2.1. Giới thiệu bài toán mạng thông tin sức khỏe

#### 2.1.1. Bài toán phát triển mạng thông tin sức khỏe

Bài toán phát triển mạng thông tin sức khỏe trong luận văn có thể được trình bày như sau:

**Phạm vi hệ thống:** Mạng thông tin sức khỏe nhằm tạo ra một hệ thống quản lý thông tin liên quan đến sức khỏe của các cá nhân và các dịch vụ y tế. Hệ thống này sẽ cho phép lưu trữ, truy xuất, và chia sẻ thông tin sức khỏe một cách an toàn và hiệu quả.

**Đối tượng tham gia hệ thống:** Hệ thống mạng thông tin sức khỏe sẽ liên quan đến các đối tượng sau:

- Bệnh nhân: Người sử dụng dịch vụ y tế và cần quản lý thông tin sức khỏe của mình.
- Bác sĩ và nhân viên y tế: Người cung cấp dịch vụ y tế và có quyền truy cập và cập nhật thông tin sức khỏe của bệnh nhân.
- Quản trị hệ thống: Người quản lý và duy trì hệ thống mạng thông tin sức khỏe.

**Quy trình nghiệp vụ của hệ thống:** Hệ thống mạng thông tin sức khỏe sẽ bao gồm các quy trình nghiệp vụ như sau:

- Đăng ký và xác thực người dùng: Bệnh nhân, bác sĩ và nhân viên y tế cần đăng ký và xác thực tài khoản để truy cập vào hệ thống.
- Quản lý thông tin bệnh nhân: Bác sĩ và nhân viên y tế có thể cập nhật thông tin sức khỏe của bệnh nhân, bao gồm lịch sử bệnh án, kết quả xét nghiệm, đơn thuốc và các thông tin khác liên quan.
- Tra cứu thông tin bệnh nhân: Bác sĩ và nhân viên y tế có thể tra cứu thông tin sức khỏe của bệnh nhân để đưa ra các quyết định chẩn đoán và điều trị.

- Phân phối công việc: Bác sĩ và nhân viên y tế có thể nhận và phân công công việc trong hệ thống, bao gồm lịch trình khám bệnh, quản lý dịch vụ y tế và các nhiệm vụ khác.

**Yêu cầu cần đạt của hệ thống:** Hệ thống mạng thông tin sức khỏe cần đáp ứng các yêu cầu sau:

- Bảo mật: Hệ thống cần đảm bảo an toàn và bảo mật thông tin sức khỏe của bệnh nhân.
- Truy cập dễ dàng: Bác sĩ và nhân viên y tế cần có khả năng truy cập và cập nhật thông tin sức khỏe của bệnh nhân một cách dễ dàng và nhanh chóng.
- Truy xuất thông tin: Hệ thống cần hỗ trợ tra cứu và truy xuất thông tin sức khỏe của bệnh nhân để phục vụ cho quá trình chẩn đoán và điều trị.
- Tích hợp hệ thống: Hệ thống cần tích hợp với các hệ thống khác trong ngành y tế, bao gồm hệ thống quản lý bệnh viện, hệ thống xét nghiệm, và các ứng dụng y tế khác.

Với những yêu cầu và quy trình nghiệp vụ trên, mục tiêu của luận văn là xây dựng một hệ thống mạng thông tin sức khỏe hiệu quả và tiện ích cho việc quản lý thông tin sức khỏe và cung cấp dịch vụ y tế.

### *2.1.2. Các chức năng cơ bản của mạng thông tin sức khỏe*

Mạng thông tin sức khỏe là hệ thống thông tin tập trung được thiết kế để quản lý thông tin sức khỏe của cá nhân hoặc của một cộng đồng. Mục đích của mạng thông tin sức khỏe là cung cấp các thông tin liên quan đến sức khỏe và bệnh tật cho những người quan tâm như bệnh nhân, các nhà cung cấp dịch vụ y tế và các chuyên gia trong lĩnh vực y tế. [16]

Các chức năng cơ bản của mạng thông tin sức khỏe bao gồm:

- Thu thập thông tin sức khỏe của người dùng: Mạng thông tin sức khỏe cho phép người dùng cập nhật thông tin sức khỏe của họ, bao gồm lịch sử bệnh án, kết quả xét nghiệm và chẩn đoán của bác sĩ.

- Chia sẻ thông tin sức khỏe: Mạng thông tin sức khỏe cho phép người dùng chia sẻ thông tin sức khỏe của mình với các nhà cung cấp dịch vụ y tế và các chuyên gia trong lĩnh vực y tế cũng như phân phối công việc giữa các bệnh viện, các bác sĩ trong cùng một bệnh viện.
- Cập nhật thông tin sức khỏe: Mạng thông tin sức khỏe cho phép người dùng cập nhật thông tin sức khỏe của họ để các nhà cung cấp dịch vụ y tế có thể cập nhật thông tin này trong hồ sơ của bệnh nhân.
- Phân tích thông tin sức khỏe: Mạng thông tin sức khỏe cho phép các chuyên gia trong lĩnh vực y tế phân tích thông tin sức khỏe để đưa ra các khuyến nghị về chăm sóc sức khỏe.
- Tra cứu thông tin sức khỏe: Mạng thông tin sức khỏe cung cấp các công cụ để người dùng tra cứu thông tin liên quan đến sức khỏe và bệnh tật.

Chức năng phân phối công việc của bác sĩ dựa trên hồ sơ bệnh án là một tính năng quan trọng của mạng thông tin sức khỏe. Khi bệnh nhân đi khám bệnh, thông tin bệnh án của họ được lưu trữ trong hệ thống và các bác sĩ có thể truy cập vào thông tin này để thực hiện việc phân tích và đưa ra chẩn đoán.

Tính năng phân phối công việc sẽ giúp phân phối các trường hợp bệnh án cho các bác sĩ dựa trên chuyên môn, kinh nghiệm, thời gian rảnh của họ và nhiều yếu tố khác. Hệ thống sẽ tự động tìm kiếm bác sĩ phù hợp nhất để giải quyết các trường hợp bệnh án, giúp tối ưu hóa công việc của bác sĩ và đảm bảo chất lượng chăm sóc cho bệnh nhân.

Ngoài ra, tính năng này cũng giúp quản lý và giám sát công việc của các bác sĩ, từ đó đưa ra các cải tiến trong quản lý tài nguyên và dự báo nhu cầu tuyển dụng, phát triển hệ thống chăm sóc sức khỏe tốt hơn.

**Phân phối công việc (cho bác sĩ)** dựa trên hồ sơ bệnh án là một ví dụ về bài toán liên quan đến mạng thông tin sức khỏe và đồ thị dữ liệu. Trong bài toán này, cần phải sử dụng các thuật toán đồ thị để xác định mối quan hệ giữa các bác sĩ và các



bệnh nhân, từ đó phân chia công việc cho các bác sĩ phù hợp với chuyên môn và thời gian rảnh của họ.

So với cách tiếp cận truyền thống với SQL, sử dụng Neo4j có nhiều ưu điểm khi giải quyết bài toán này. Đầu tiên, Neo4j được thiết kế để lưu trữ và truy vấn dữ liệu đồ thị, điều này giúp cho việc xác định mối quan hệ giữa các đối tượng trở nên dễ dàng và nhanh chóng hơn. Thứ hai, các thuật toán đồ thị được tích hợp sẵn trong Neo4j, giúp cho việc thực hiện các tính toán đồ thị như tìm kiếm đường đi ngắn nhất, tìm kiếm theo độ tương đồng giữa các đối tượng, tìm kiếm các cụm đồ thị liên quan, v.v. trở nên đơn giản hơn. Cuối cùng, Neo4j còn hỗ trợ các tính năng đồ thị như các thuộc tính và quan hệ có hướng và vô hướng, các thuộc tính phức tạp và các quan hệ đa cấp, giúp cho việc mô hình hóa dữ liệu đồ thị trở nên linh hoạt và dễ dàng hơn.

Tóm lại, sử dụng Neo4j để giải quyết bài toán phân phối công việc cho bác sĩ dựa trên hồ sơ bệnh án có nhiều ưu điểm hơn so với SQL, như tích hợp sẵn các thuật toán đồ thị, hỗ trợ các tính năng đồ thị và làm cho việc mô hình hóa dữ liệu đồ thị trở nên linh hoạt và dễ dàng hơn.

### **Chia sẻ thông tin**

Trong mạng thông tin sức khỏe, chia sẻ thông tin là một chức năng cơ bản và quan trọng để đảm bảo cho việc chăm sóc sức khỏe tốt hơn. Chia sẻ thông tin giúp cho các nhân viên y tế và bệnh nhân có thể truy cập được thông tin y tế của mình từ mọi nơi và mọi thời điểm, từ đó hỗ trợ cho quá trình chăm sóc sức khỏe được tiến hành tốt hơn.

Việc sử dụng mạng thông tin sức khỏe cho phép các bệnh nhân có thể quản lý thông tin y tế của mình một cách dễ dàng hơn, giúp cho việc điều trị và chăm sóc sức khỏe được tối ưu hóa. Các nhân viên y tế cũng có thể truy cập được thông tin y tế của bệnh nhân từ xa, từ đó giúp cho quá trình điều trị và chăm sóc được đẩy nhanh hơn.

Ngoài ra, việc chia sẻ thông tin còn giúp cho các nhà nghiên cứu y tế thu thập dữ liệu một cách dễ dàng hơn để phân tích và đưa ra các quyết định trong việc phòng

ngừa và điều trị các bệnh. Chia sẻ thông tin cũng giúp cho các nhà sản xuất dược phẩm có thể tiếp cận được thông tin về các bệnh nhân và nhu cầu về thuốc, từ đó hỗ trợ cho quá trình nghiên cứu và phát triển sản phẩm mới.

Tuy nhiên, việc chia sẻ thông tin cũng đặt ra những vấn đề về bảo mật và riêng tư thông tin, đặc biệt là trong lĩnh vực y tế. Do đó, cần phải có các quy định về bảo mật và quyền riêng tư, cùng với các hệ thống kiểm soát và giám sát đảm bảo rằng thông tin được chia sẻ một cách an toàn và đảm bảo tính bảo mật của người dùng.

### *2.1.3. Các hướng tiếp cận và giải quyết bài toán*

Tập trung vào các hướng tiếp cận và giải quyết bài toán trong phát triển mạng thông tin sức khỏe sử dụng CSDL đồ thị. Trong đó, luận văn sẽ tìm hiểu về các phương pháp xây dựng và quản lý đồ thị trong mạng thông tin sức khỏe, các thuật toán đồ thị áp dụng trong việc phân tích dữ liệu và đưa ra quyết định, cũng như những thách thức trong việc áp dụng CSDL đồ thị trong mạng thông tin sức khỏe và các giải pháp để giải quyết những thách thức đó. [16][17]

Các hướng tiếp cận và giải quyết bài toán sử dụng các mô hình khác nhau như:

- Mô hình quan hệ: sử dụng CSDL quan hệ để lưu trữ dữ liệu y tế và áp dụng các kỹ thuật truy vấn để sắp xếp công việc cho bác sĩ.
- Mô hình đồ thị: sử dụng CSDL đồ thị để lưu trữ dữ liệu y tế và áp dụng các thuật toán đồ thị để tối ưu hóa việc phân phối công việc cho bác sĩ.
- Mô hình hướng đối tượng: sử dụng CSDL hướng đối tượng để lưu trữ dữ liệu y tế và áp dụng các kỹ thuật truy vấn và phân tích đối tượng để sắp xếp công việc cho bác sĩ.

**Các giải pháp đang được triển khai thực tế trong việc lập mạng thông tin sức khỏe:**

Hệ thống quản lý thông tin bệnh án điện tử: Đây là một giải pháp đang được triển khai rộng rãi trong các bệnh viện và cơ sở y tế trên toàn thế giới. Hệ thống này cho phép lưu trữ và quản lý thông tin bệnh án của bệnh nhân bằng cách sử dụng các

phần mềm quản lý bệnh án điện tử. Việc áp dụng hệ thống này giúp cho việc tra cứu thông tin bệnh án của bệnh nhân trở nên dễ dàng và thuận tiện hơn.

**Mạng xã hội y tế:** Đây là một mô hình mạng xã hội chuyên dành cho lĩnh vực y tế, nơi mà các chuyên gia y tế, bác sĩ và bệnh nhân có thể chia sẻ thông tin và kinh nghiệm về các bệnh lý và điều trị. Mạng xã hội y tế cũng cung cấp cho người dùng các công cụ để theo dõi sức khỏe và cập nhật các thông tin y tế mới nhất.

**Các ứng dụng di động liên quan đến sức khỏe:** Hiện nay có rất nhiều ứng dụng di động được phát triển để hỗ trợ việc quản lý sức khỏe của người dùng, bao gồm các ứng dụng theo dõi lượng calo, các ứng dụng đo nhịp tim, ứng dụng đo đường huyết, ứng dụng tư vấn chăm sóc sức khỏe và nhiều ứng dụng khác.

**Công nghệ truyền thông và thông tin y tế:** Các công nghệ truyền thông như video hội thảo, truyền hình Internet và các công nghệ khác được sử dụng để truyền tải thông tin y tế và giúp bệnh nhân tiếp cận dịch vụ y tế từ xa. Các công nghệ này cũng giúp cho các chuyên gia y tế có thể tư vấn và hỗ trợ bệnh nhân từ xa, đặc biệt là trong những trường hợp khẩn cấp.

Các giải pháp trên đều liên quan đến các CSDL quan hệ, hướng đối tượng và đồ thị. Ví dụ, trong giải pháp sử dụng hệ thống quản lý cơ sở dữ liệu (CSDL) để lưu trữ thông tin sức khỏe, có thể sử dụng CSDL quan hệ để thiết kế CSDL và các câu truy vấn dựa trên SQL để truy xuất dữ liệu. Trong khi đó, trong giải pháp sử dụng mạng thông tin sức khỏe, có thể sử dụng các mô hình đồ thị để mô tả các mối quan hệ giữa các yếu tố sức khỏe, từ đó xây dựng các thuật toán phân tích dữ liệu để phát hiện ra các mối liên hệ này. Cuối cùng, trong giải pháp sử dụng hướng tiếp cận dữ liệu hướng đối tượng, có thể sử dụng các lớp đối tượng để đại diện cho các thực thể trong hệ thống thông tin sức khỏe và quản lý các mối quan hệ giữa chúng.

### **Các tiêu chí để so sánh, đánh giá các mô hình**

**Tính mở rộng:** Khả năng mở rộng của mô hình để có thể xử lý được lượng dữ liệu lớn và đáp ứng nhu cầu sử dụng của mạng thông tin sức khỏe.

**Tính linh hoạt:** Khả năng mô hình thích ứng với các yêu cầu thay đổi của mạng thông tin sức khỏe, ví dụ như thêm các tính năng mới, thay đổi cấu trúc dữ liệu, hoặc sửa đổi các yêu cầu về hiệu năng.

**Tính tương thích:** Khả năng tương thích của mô hình với các công nghệ và hệ thống sẵn có trong mạng thông tin sức khỏe, như các hệ thống quản lý cơ sở dữ liệu, các công nghệ lưu trữ, các giao thức truyền thông, và các hệ thống an ninh.

**Tính dễ sử dụng:** Mức độ dễ sử dụng của mô hình cho các lập trình viên và những người phát triển hệ thống, bao gồm khả năng lập trình, tương tác với các công cụ và các API, và khả năng hỗ trợ của cộng đồng lập trình viên.

**Hiệu năng:** Khả năng xử lý và truy vấn dữ liệu nhanh và hiệu quả của mô hình, đặc biệt là với các dữ liệu lớn và phức tạp.

**Tính bảo mật:** Mức độ bảo mật của mô hình đối với các thông tin y tế và dữ liệu nhạy cảm, bao gồm khả năng xác thực người dùng, mã hóa dữ liệu và phân quyền truy cập.

**Tính tương tác:** Khả năng tương tác của mô hình với người dùng cuối và các ứng dụng khác, bao gồm khả năng tích hợp với các ứng dụng bên thứ ba và cung cấp các API cho các lập trình viên.

### So sánh 3 mô hình về tính mở rộng

*Bảng 2- 1: So sánh 3 mô hình dữ liệu về tính mở rộng*

CSDL	Nội dung
Quan hệ	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình này có tính mở rộng tốt khi xử lý các truy vấn phức tạp. Việc thêm hoặc sửa đổi các bảng trong CSDL quan hệ cũng khá đơn giản.</li> <li>- Nhược điểm: Tuy nhiên, mô hình này có hạn chế về khả năng mở rộng theo chiều ngang (horizontal scaling), tức là việc tăng khả năng chịu tải</li> </ul>

	của hệ thống bằng cách thêm các node (cụm máy chủ) mới vào hệ thống.
Hướng đối tượng	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình này có tính mở rộng tốt khi xử lý các truy vấn liên quan đến các đối tượng (object-oriented) phức tạp. Nó cũng cho phép thêm các đối tượng mới vào hệ thống một cách linh hoạt và dễ dàng.</li> <li>- Nhược điểm: Tuy nhiên, mô hình hướng đối tượng cũng có hạn chế về khả năng mở rộng theo chiều ngang. Ngoài ra, việc thêm mới các thuộc tính cho một đối tượng có thể ảnh hưởng đến hiệu suất của các truy vấn.</li> </ul>
Đồ thị	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình này có tính mở rộng tốt khi xử lý các truy vấn phức tạp liên quan đến quan hệ giữa các đối tượng, ví dụ như tìm kiếm đường đi ngắn nhất giữa các đối tượng hoặc phân tích mối quan hệ giữa chúng. Nó cho phép thêm các đỉnh (node) mới vào hệ thống một cách linh hoạt và dễ dàng.</li> <li>- Nhược điểm: Tuy nhiên, CSDL đồ thị cũng có hạn chế về khả năng mở rộng theo chiều ngang, đặc biệt khi số lượng quan hệ giữa các đối tượng tăng lên</li> </ul>

### So sánh 3 mô hình về tính linh hoạt

*Bảng 2- 2: So sánh 3 mô hình dữ liệu về tính linh hoạt*

CSDL	Nội dung
Quan hệ	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình quan hệ có khả năng linh hoạt trong việc thay đổi cấu trúc bảng hoặc thêm mới bảng một cách dễ dàng, đặc biệt phù hợp cho các ứng dụng có quy mô lớn và dữ liệu thay đổi thường xuyên.</li> <li>- Nhược điểm: Tuy nhiên, mô hình quan hệ không linh hoạt trong việc mô tả các mối quan hệ phức tạp giữa các đối tượng, đặc biệt là khi có nhiều mối quan hệ phức tạp.</li> </ul>

Hướng đối tượng	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình hướng đối tượng có khả năng mô tả các đối tượng và mối quan hệ giữa chúng một cách rõ ràng và dễ hiểu, phù hợp cho các ứng dụng có tính chất phức tạp và thay đổi liên tục.</li> <li>- Nhược điểm: Tuy nhiên, mô hình hướng đối tượng không linh hoạt trong việc thay đổi cấu trúc của đối tượng một cách dễ dàng.</li> </ul>
Đồ thị	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình đồ thị có khả năng linh hoạt trong việc mô tả các mối quan hệ phức tạp giữa các đối tượng, đặc biệt phù hợp cho các ứng dụng có tính chất phức tạp và thay đổi liên tục.</li> <li>- Nhược điểm: Tuy nhiên, mô hình đồ thị không linh hoạt trong việc thêm mới hoặc xóa bỏ các đối tượng và quan hệ giữa chúng một cách dễ dàng. Ngoài ra, việc truy vấn dữ liệu trên đồ thị có thể phức tạp và tốn nhiều thời gian.</li> </ul>

### So sánh 3 mô hình về tính tương thích, tính dễ sử dụng

*Bảng 2- 3: So sánh 3 mô hình dữ liệu về tính tương thích, tính dễ sử dụng*

CSDL	Nội dung
Quan hệ	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình quan hệ được sử dụng phổ biến trong hệ thống quản lý cơ sở dữ liệu, các công cụ và phần mềm hỗ trợ rất phong phú và đa dạng. Người sử dụng dễ dàng tiếp cận và hiểu được mô hình này, đồng thời nó cũng đảm bảo tính tương thích với các hệ thống khác.</li> <li>- Nhược điểm: Mô hình quan hệ hạn chế trong việc mô tả các quan hệ phức tạp và các liên kết giữa các đối tượng.</li> </ul>
Hướng đối tượng	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình hướng đối tượng phù hợp với các ứng dụng yêu cầu tính động và linh hoạt cao, đặc biệt là các ứng dụng web. Nó cho phép các đối tượng có tính kế thừa, tổ chức, đóng gói và tái sử dụng dễ dàng.</li> <li>- Nhược điểm: Mô hình hướng đối tượng không thể đại diện cho các quan hệ phức tạp như các quan hệ nhiều-nhiều.</li> </ul>

Đồ thị	<ul style="list-style-type: none"> <li>- Ưu điểm: Mô hình đồ thị có khả năng mô tả các quan hệ phức tạp và liên kết giữa các đối tượng một cách rõ ràng. Nó cũng cho phép mô tả các mối quan hệ nhiều-nhiều giữa các đối tượng.</li> <li>- Nhược điểm: Việc triển khai mô hình đồ thị thường cần các công cụ và kỹ thuật riêng biệt, và không phổ biến như các mô hình khác. Điều này có thể làm tăng độ phức tạp của hệ thống và độ khó khăn trong việc sử dụng và bảo trì.</li> </ul>
--------	---

### So sánh 3 mô hình về hiệu suất

*Bảng 2- 4: So sánh 3 mô hình dữ liệu về hiệu suất*

CSDL	Nội dung
Quan hệ	<ul style="list-style-type: none"> <li>- Ưu điểm: Hiệu suất cao khi thực hiện các truy vấn phức tạp trong CSDL lớn. Có thể tối ưu hóa hiệu suất bằng cách sử dụng các chỉ mục, khóa ngoại, trigger, store procedure, ...</li> <li>- Nhược điểm: Khó khăn trong việc quản lý quan hệ giữa các đối tượng khi dữ liệu phức tạp, đặc biệt khi có nhiều mối quan hệ.</li> </ul>
Hướng đối tượng	<ul style="list-style-type: none"> <li>- Ưu điểm: Cung cấp khả năng mô tả dữ liệu phức tạp bằng các đối tượng và mối quan hệ giữa chúng. Có thể tối ưu hóa hiệu suất bằng cách sử dụng caching và lazy loading.</li> <li>- Nhược điểm: Hiệu suất truy vấn có thể giảm nếu dữ liệu phức tạp và có quá nhiều đối tượng, cũng như khó khăn trong việc quản lý các quan hệ giữa các đối tượng.</li> </ul>
Đồ thị	<ul style="list-style-type: none"> <li>- Ưu điểm: Truy vấn đồ thị nhanh, tập trung vào việc truy vấn quan hệ giữa các đỉnh trong đồ thị, cho phép lưu trữ các thông tin phức tạp. Đặc biệt, các tính năng phân tích đồ thị, như PageRank hay Betweenness Centrality, giúp tìm ra các đỉnh quan trọng trong đồ thị.</li> </ul>

	- Nhược điểm: Đòi hỏi lưu trữ dữ liệu phức tạp trong các đỉnh và cạnh của đồ thị. Hiệu suất truy vấn có thể bị giảm nếu đồ thị quá lớn và phức tạp.
--	---

### So sánh 3 mô hình về tính bảo mật

*Bảng 2- 5: So sánh 3 mô hình dữ liệu về tính bảo mật*

CSDL	Nội dung
Quan hệ	<p>- Ưu điểm:</p> <p>Có khả năng kiểm soát truy cập và phân quyền trên dữ liệu.</p> <p>Cung cấp các công cụ phân tích bảo mật mạnh mẽ để phát hiện lỗ hổng bảo mật và các hành vi tấn công.</p> <p>Có thể thiết kế các ràng buộc duy nhất để đảm bảo tính toàn vẹn của dữ liệu.</p> <p>- Nhược điểm:</p> <p>Có thể khó khăn trong việc thiết lập và quản lý các quy tắc bảo mật phức tạp.</p> <p>Không thể đáp ứng các yêu cầu bảo mật linh hoạt và động.</p> <p>Việc phân quyền và quản lý truy cập trở nên phức tạp khi cơ sở dữ liệu quá lớn.</p>
Hướng đổi tượng	<p>- Ưu điểm:</p> <p>Có khả năng kiểm soát truy cập và phân quyền trên dữ liệu.</p> <p>Cung cấp các công cụ phân tích bảo mật mạnh mẽ để phát hiện lỗ hổng bảo mật và các hành vi tấn công.</p> <p>Có thể thiết kế các ràng buộc duy nhất để đảm bảo tính toàn vẹn của dữ liệu.</p> <p>- Nhược điểm:</p> <p>Có thể khó khăn trong việc thiết lập và quản lý các quy tắc bảo mật phức tạp.</p>



	<p>Không thể đáp ứng các yêu cầu bảo mật linh hoạt và động.</p> <p>Việc phân quyền và quản lý truy cập trở nên phức tạp khi cơ sở dữ liệu quá lớn.</p>
Đề thi	<p>- Ưu điểm:</p> <p>Có tính linh hoạt cao trong việc định nghĩa quyền truy cập và phân quyền.</p> <p>Cung cấp các tính năng bảo mật phức tạp hơn, bao gồm cả kiểm soát truy cập và kiểm soát trình tự.</p> <p>Có thể đáp ứng các yêu cầu bảo mật linh hoạt và động.</p> <p>- Nhược điểm:</p> <p>Yêu cầu khối lượng lớn bộ nhớ và tài nguyên máy tính để hoạt động.</p> <p>Khó khăn trong việc đảm bảo tính toàn vẹn của dữ liệu khi có quá nhiều liên kết giữa các nút.</p>

### So sánh 3 mô hình dữ liệu về tính tương tác

*Bảng 2- 6: So sánh 3 mô hình dữ liệu về tính tương tác*

CSDL	Nội dung
Quan hệ	<p>- Ưu điểm:</p> <p>Thao tác dữ liệu đơn giản với các câu lệnh SQL, dễ dàng truy vấn và thêm sửa xóa dữ liệu.</p> <p>Có thể tạo quan hệ giữa các bảng thông qua khóa ngoại, hỗ trợ việc tối ưu hóa các truy vấn phức tạp.</p> <p>- Nhược điểm:</p> <p>Khó khăn trong việc mô tả các mối quan hệ phức tạp giữa các đối tượng, đặc biệt là khi có nhiều mối quan hệ giữa chúng.</p> <p>Khó khăn trong việc đảm bảo tính toàn vẹn của dữ liệu khi thao tác cập nhật và xóa dữ liệu.</p>

	<p>Khó khăn trong việc mở rộng hệ thống khi cần thêm bảng mới hoặc thay đổi cấu trúc bảng hiện có.</p>
Hướng đối tượng	<p>- Ưu điểm:</p> <p>Dữ liệu được mô tả dưới dạng các đối tượng, đơn giản và dễ hiểu.</p> <p>Dễ dàng thêm, sửa, xoá các đối tượng và thuộc tính của chúng.</p> <p>Hỗ trợ tính đóng gói, kế thừa, đa hình, đảm bảo tính toàn vẹn dữ liệu.</p> <p>- Nhược điểm:</p> <p>Không hỗ trợ việc tối ưu hóa truy vấn phức tạp.</p> <p>Khó khăn trong việc định nghĩa các quan hệ giữa các đối tượng phức tạp.</p> <p>Khó khăn trong việc mở rộng hệ thống khi cần thêm đối tượng mới hoặc thay đổi cấu trúc đối tượng hiện có.</p>
Đồ thị	<p>- Ưu điểm:</p> <p>Dữ liệu được mô tả dưới dạng các đỉnh và cạnh, đơn giản và dễ hiểu.</p> <p>Dễ dàng tạo, thêm, sửa, xoá đỉnh và cạnh giữa chúng.</p> <p>Hỗ trợ việc tối ưu hóa truy vấn phức tạp, đặc biệt là các truy vấn liên quan đến mối quan hệ giữa các đỉnh.</p> <p>Dễ dàng mở rộng hệ thống khi cần thêm đỉnh mới hoặc thay đổi quan hệ giữa các đỉnh.</p> <p>- Nhược điểm:</p> <p>Khó khăn trong việc truy vấn theo hướng ngược lại: CSDL đồ thị không phải lúc nào cũng hiệu quả khi cần truy vấn theo hướng ngược lại.</p> <p>Khó khăn trong việc thao tác với dữ liệu có cấu trúc phức tạp: CSDL đồ thị không phù hợp cho việc lưu trữ và truy vấn các dữ liệu có cấu trúc phức tạp như dữ liệu thừa kế hoặc dữ liệu mô hình thực thể-đặc tính.</p> <p>Đòi hỏi sự quản lý và duy trì dữ liệu tốt: Vì CSDL đồ thị phụ thuộc vào các quan hệ giữa các đối tượng, việc duy trì các đồ thị phức tạp có thể là một thách thức lớn.</p>

#### 2.1.4. Đề xuất giải pháp giải quyết

Với cách tiếp cận này, luận văn sẽ sử dụng mô hình đồ thị để biểu diễn các quan hệ giữa các đối tượng trong hệ thống sức khỏe, từ đó xây dựng một mạng thông tin sức khỏe hoàn chỉnh và hiệu quả.

Để đưa ra giải pháp tối ưu, cần xem xét các yếu tố sau:

- Phù hợp với yêu cầu của bài toán: Giải pháp đề xuất phải đáp ứng được các yêu cầu cơ bản của bài toán lập mạng thông tin sức khỏe. Nó cần phải có khả năng đáp ứng cho các yêu cầu về tính linh hoạt, tính mở rộng, tính tương thích, tính bảo mật và hiệu suất của hệ thống.
- Sử dụng mô hình đồ thị: Mô hình đồ thị là một phương pháp mạnh mẽ để biểu diễn các quan hệ giữa các đối tượng trong hệ thống sức khỏe. Vì vậy, giải pháp đề xuất nên sử dụng mô hình đồ thị để biểu diễn và quản lý thông tin sức khỏe.
- Sử dụng công nghệ mới nhất: Để đảm bảo tính hiệu quả và độ chính xác của hệ thống, giải pháp đề xuất cần sử dụng các công nghệ mới nhất và tiên tiến nhất để xử lý và quản lý dữ liệu.
- Thiết kế hệ thống tối ưu: Giải pháp đề xuất cần tập trung vào việc thiết kế hệ thống tối ưu, đảm bảo khả năng mở rộng và tương thích với các hệ thống khác, giúp tối ưu hóa hiệu suất và tăng tính bảo mật của hệ thống.
- Tối ưu hóa quá trình truy vấn: Với mạng thông tin sức khỏe, quá trình truy vấn thông tin là rất quan trọng. Giải pháp đề xuất cần tối ưu hóa quá trình này để đảm bảo tính hiệu quả và độ chính xác của hệ thống.

Dựa trên ưu và nhược điểm của các mô hình quan hệ, hướng đối tượng và đồ thị, cùng với yêu cầu của bài toán lập mạng thông tin sức khỏe, mô hình đồ thị được đưa ra là mô hình tối ưu.

Mô hình đồ thị cho phép mô tả rõ ràng các quan hệ giữa các thực thể trong hệ thống thông tin sức khỏe, đồng thời giúp cho việc tìm kiếm, truy vấn và phân tích thông tin trở nên dễ dàng hơn. Mô hình này cũng có tính linh hoạt và mở rộng tốt,

cho phép thêm mới và chỉnh sửa các thực thể, quan hệ một cách dễ dàng. Ngoài ra, mô hình đồ thị cũng có khả năng tương tác cao, cho phép người dùng tương tác với các thực thể, quan hệ và thu được thông tin cần thiết.

## 2.2. Áp dụng phát triển mạng thông tin sức khỏe

### 2.2.1. Các bước xây dựng hệ thống dữ liệu theo mô hình dữ liệu đồ thị

Để xây dựng hệ thống dữ liệu theo MHDL đồ thị cho mạng thông tin sức khỏe, các bước chính bao gồm:

- **Thiết kế schema:** Thiết kế schema là quá trình xác định các loại đối tượng và mối quan hệ giữa chúng. Đối tượng có thể là các thực thể như bệnh nhân, bác sĩ, bệnh viện, thuốc và mối quan hệ giữa chúng như là một bệnh nhân được điều trị bởi một bác sĩ, bệnh nhân được chẩn đoán mắc bệnh bởi một bệnh viện hay một thuốc được sử dụng để điều trị một bệnh. Schema cũng xác định các thuộc tính và giá trị của các đối tượng.
- **Triển khai cơ sở dữ liệu:** Sau khi thiết kế schema, cần triển khai cơ sở dữ liệu bằng cách sử dụng hệ quản trị cơ sở dữ liệu đồ thị Neo4j.
- **Nhập dữ liệu:** Dữ liệu có thể được nhập vào từ nhiều nguồn khác nhau như hệ thống thông tin y tế, hồ sơ bệnh án, các thiết bị y tế hoặc các hệ thống mạng thông tin sức khỏe khác. Dữ liệu cần được chuyển đổi và tối ưu hóa để phù hợp với schema và cơ sở dữ liệu.
- **Tối ưu hóa và tinh chỉnh:** Cần tối ưu hóa và tinh chỉnh hệ thống dữ liệu để đảm bảo tính nhất quán và đáp ứng nhu cầu của người dùng. Điều này có thể bao gồm cải tiến hiệu suất, tối ưu hóa kết cấu dữ liệu và cải thiện bảo mật.
- **Phát triển ứng dụng web:** Cuối cùng, cần phát triển ứng dụng để truy xuất và hiển thị dữ liệu từ hệ thống đồ thị. Các ứng dụng này có thể bao gồm giao diện người dùng, các công cụ tìm kiếm và phân tích dữ liệu và các ứng dụng di động.

### 2.2.2. Thiết kế CSDL đồ thị trên Neo4j

#### a. Xác định các đối tượng và mối quan hệ giữa chúng trong mạng thông tin sức khỏe

Mạng thông tin sức khỏe có các chức năng cơ bản đã đặt ra ở mục 2.1.1. Bước đầu mạng thông tin sẽ có chức năng “Thu thập thông tin sức khỏe”, chức năng “Chia sẻ thông tin sức khỏe”. Vì vậy luận văn đặt ra thiết kế các đối tượng trong mạng như sau:

Bảng 2- 7: Các đối tượng trong mạng thông tin sức khỏe

STT	Đối tượng	Mô tả
1	AdminHeThong	AdminHeThong có quyền điều khiển toàn bộ hệ thống và quản lý các tài khoản đăng nhập vào hệ thống.
2	AdminBenhVien	AdminBenhVien quản lý hoạt động của các bác sỹ và phòng khám trong bệnh viện.
3	BacSy	BacSy là người chăm sóc sức khỏe cho bệnh nhân, có liên quan đến KhoaKhamBenh và DichVuKhamBenh.
4	BenhNhan	BenhNhan là người bệnh được khám và chữa trị tại bệnh viện, có thông tin HoSoBenhAn được tạo ra khi bệnh nhân đăng ký khám bệnh.
5	BenhVien	BenhVien là nơi cung cấp dịch vụ y tế, có mối quan hệ với các đối tượng khác trong hệ thống.
6	ChuyenMon	ChuyenMon là các chuyên môn y khoa khác nhau có liên quan đến các KhoaKhamBenh.
7	DichVuKhamBenh	DichVuKhamBenh là các dịch vụ khám bệnh mà bệnh viện cung cấp cho bệnh nhân, có liên quan đến KhoaKhamBenh và BacSy.

8	HoSoBenhAn	HoSoBenhAn là tài liệu lưu trữ thông tin về sức khỏe của bệnh nhân.
9	KhoaKhamBenh	KhoaKhamBenh là các khoa chuyên môn trong bệnh viện, có liên quan đến các ChuyenMon, DichVuKhamBenh và BacSy.
10	LichSuCongViec	LichSuCongViec là lịch sử làm việc của các nhân viên trong bệnh viện, bao gồm cả BacSy và AdminBenhVien.
11	TaiKhoan	TaiKhoan là thông tin đăng nhập của các đối tượng trong hệ thống, được quản lý bởi AdminHeThong.
12	TinhHuyenXa	TinhHuyenXa là các đơn vị hành chính trong địa phương, có thể được liên kết đến địa điểm của BenhVien.
13	BaseModel	Đối tượng dùng chung chứa các thuộc tính cơ bản
14	ThongTinNguoiDung	ThongTinNguoiDung là thông tin cá nhân của các đối tượng trong hệ thống.

***b. Thiết kế schema cho CSDL đồ thị.***

Dựa trên các đối tượng và mối quan hệ giữa chúng, luận văn có thể thiết kế schema cho CSDL đồ thị như sau:

Node:

- AdminHeThong
- AdminBenhVien
- BacSy
- BenhNhan
- BenhVien
- ChuyenMon

- DichVuKhamBenh
- HoSoBenhAn
- KhoaKhamBenh
- LichSuCongViec
- TaiKhoan
- TinhHuyenXa

Relationship:

- CO\_THONG\_TIN: quan hệ giữa các node "AdminHeThong", "AdminBenhVien", "BacSy", "BenhNhan", "TaiKhoan" với node "ThongTinNguoiDung"
- CO\_BASE: quan hệ giữa các node "AdminHeThong", "AdminBenhVien", "BacSy", "BenhNhan", "BenhVien" và "TaiKhoan" với node "BaseModel"
- LAM\_VIEC\_TAI: quan hệ giữa các node "BacSy", "DichVuKhamBenh" và "KhoaKhamBenh" với node "BenhVien"
- CO\_CHUYEN\_MON: quan hệ giữa các node "BacSy" với node "ChuyenMon"
- CO\_DIA\_CHI: quan hệ giữa các node "BenhVien" và "TinhHuyenXa"
- DUOC\_KHAM\_BENH: quan hệ giữa các node "BenhNhan", "DichVuKhamBenh" và "BacSy"
- CO\_HO\_SO: quan hệ giữa các node "BenhNhan" và "HoSoBenhAn"
- CO\_CONG\_VIEC: quan hệ giữa các node "BacSy" và "LichSuCongViec"

Các quan hệ này sẽ giúp mô tả chính xác các mối quan hệ giữa các đối tượng trong hệ thống, từ đó giúp cho việc truy vấn và tìm kiếm dữ liệu dễ dàng hơn.

Schema được thể hiện trong Neo4j như sau:

```
(:AdminHeThong)-[:CO_THONG_TIN]->(:ThongTinNguoiDung)
(:AdminBenhVien)-[:CO_THONG_TIN]->(:ThongTinNguoiDung)
(:BacSy)-[:CO_THONG_TIN]->(:ThongTinNguoiDung)
(:BenhNhan)-[:CO_THONG_TIN]->(:ThongTinNguoiDung)
```

```

(:TaiKhoan)-[:CO_THONG_TIN]->(:ThongTinNguoiDung)

(:AdminHeThong)-[:CO_BASE]->(:BaseModel)
(:AdminBenhVien)-[:CO_BASE]->(:BaseModel)
(:BacSy)-[:CO_BASE]->(:BaseModel)
(:BenhNhan)-[:CO_BASE]->(:BaseModel)
(:BenhVien)-[:CO_BASE]->(:BaseModel)
(:TaiKhoan)-[:CO_BASE]->(:BaseModel)

(:BacSy)-[:LAM_VIEC_TAI]->(:BenhVien)-[:LAM_VIEC_TAI]->(:DichVuKhamBenh)
(:BacSy)-[:LAM_VIEC_TAI]->(:BenhVien)-[:LAM_VIEC_TAI]->(:KhoaKhamBenh)

(:BacSy)-[:CO_CHUYEN_MON]->(:ChuyenMon)

(:BenhVien)-[:CO_DIA_CHI]->(:TinhHuyenXa)

(:BenhNhan)-[:DUOC_KHAM_BENH]->(:DichVuKhamBenh)<-
[:DUOC_KHAM_BENH]-(:BacSy)

(:BenhNhan)-[:CO_HO_SO]->(:HoSoBenhAn)

(:BacSy)-[:CO_CONG_VIEC]->(:LichSuCongViec)

```

### ***c. Tạo các nodes và relationships trong CSDL đồ thị.***

Dữ liệu đầu vào sẽ được giả lập trong các file .csv. Tiến hành import dữ liệu các file .csv đó vào trong CSDL Neo4j, từ đó sẽ có các node và dữ liệu ban đầu.

Sau đó tiến thiết lập các relationships giữa các node có trong CSDL đó.

### ***d. Tối ưu hóa CSDL đồ thị để đạt hiệu suất cao nhất.***

Một số phương án được đưa ra để tối ưu hóa CSDL đồ thị:

- Update thiết kế schema đồ thị phù hợp với nhu cầu sử dụng và tính linh hoạt của hệ thống.
- Sử dụng các thuật toán và công nghệ mới nhất để tối ưu hoá việc lưu trữ và truy vấn dữ liệu trong CSDL đồ thị.



- Áp dụng các kỹ thuật indexing và caching để tăng tốc độ truy vấn dữ liệu và giảm thời gian truy cập vào CSDL đồ thị.
- Tối ưu hóa các câu truy vấn để giảm thiểu số lượng node và relationship được trả về, đồng thời giảm thiểu thời gian xử lý truy vấn.
- Sử dụng các công cụ và phần mềm quản lý CSDL đồ thị chuyên nghiệp để đảm bảo tính ổn định và độ tin cậy của hệ thống.
- Thực hiện các hoạt động định kỳ để kiểm tra và đánh giá hiệu suất của CSDL đồ thị, từ đó đưa ra các cải tiến và điều chỉnh phù hợp.

## 2.3. Xây dựng hệ thống phần mềm truy vấn và trực quan hóa

### 2.3.1. Lập Use case.

Để xây dựng hệ thống phần mềm truy vấn và trực quan hóa cho mạng thông tin sức khỏe, luận văn đưa ra một số use case sau:

#### ***a. Usecase: Quản lý thông tin người dùng:***

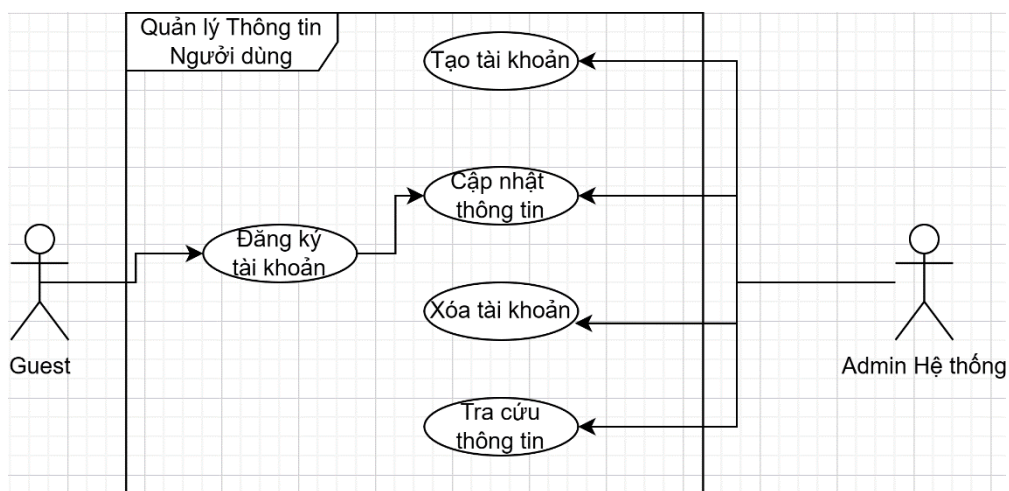
Mô tả: Use case này mô tả việc quản lý thông tin người dùng trong hệ thống, bao gồm tạo mới người dùng, cập nhật thông tin người dùng, xóa người dùng khỏi hệ thống và xem danh sách người dùng trong hệ thống.

Người sử dụng chính: Admin hệ thống, Guest

Luồng sự kiện chính

- Tạo mới người dùng: Admin chọn chức năng "Tạo mới người dùng" hoặc Guest chọn chức năng "Đăng ký tài khoản" (Guest chỉ đăng ký tài khoản "Bệnh nhân"). Hệ thống hiển thị form để nhập thông tin người dùng, kiểm tra tính hợp lệ của thông tin nhập vào và lưu thông tin người dùng vào CSDL đồ thị.
- Cập nhật thông tin người dùng: Hệ thống hiển thị danh sách người dùng trong hệ thống và cho phép Admin tìm kiếm người dùng cần cập nhật thông tin. Admin chọn người dùng cần cập nhật và thực hiện việc sửa thông tin người dùng. Hệ thống cập nhật thông tin người dùng vào CSDL đồ thị.

- Xóa người dùng khỏi hệ thống: Hệ thống hiển thị danh sách người dùng trong hệ thống và cho phép Admin tìm kiếm người dùng cần xóa. Admin chọn người dùng cần xóa và thực hiện việc xóa người dùng. Hệ thống xóa thông tin người dùng khỏi CSDL đồ thị.
- Xem danh sách người dùng trong hệ thống: Hệ thống hiển thị danh sách người dùng trong hệ thống và cho phép Admin tìm kiếm người dùng theo tên đăng nhập hoặc vai trò.



Hình 2- 1: Usecase Quản lý Thông tin Người dùng

**b. Usecase: Quản lý thông tin bệnh nhân:**

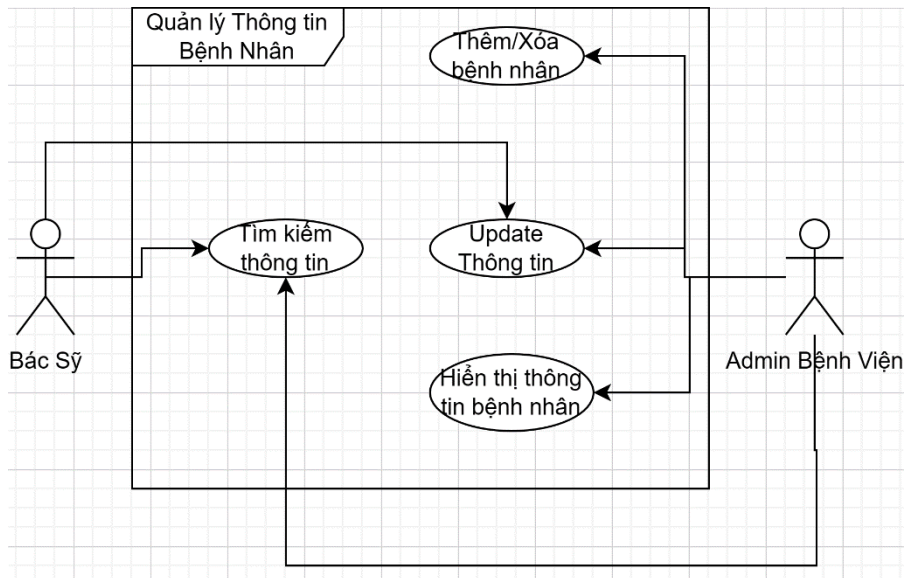
Mô tả: Use case này cho phép admin bệnh viện, bác sỹ có thể quản lý thông tin của các bệnh nhân trong hệ thống của 1 bệnh viện.

Người tương tác: admin bệnh viện, bác sỹ

- Admin bệnh viện có thể thêm/xóa bệnh nhân trong hệ thống
- Người dùng có thể tìm kiếm thông tin bệnh nhân bằng cách nhập các thông tin liên quan như tên, mã số bệnh nhân, số điện thoại, địa chỉ,...
- Hệ thống hiển thị thông tin chi tiết của bệnh nhân, bao gồm tên, ngày sinh, giới tính, số điện thoại, địa chỉ, tiền sử bệnh lý, kết quả khám bệnh,...

- Người dùng có thể cập nhật hoặc thêm mới thông tin của bệnh nhân bằng cách chọn chức năng tương ứng và nhập thông tin mới.

Hệ thống cập nhật thông tin mới vào hệ thống và thông báo cho người dùng biết.



Hình 2- 2: Usecase Quản lý thông tin Bệnh nhân

### c. Usecase: Tìm kiếm thông tin bệnh án của bệnh nhân

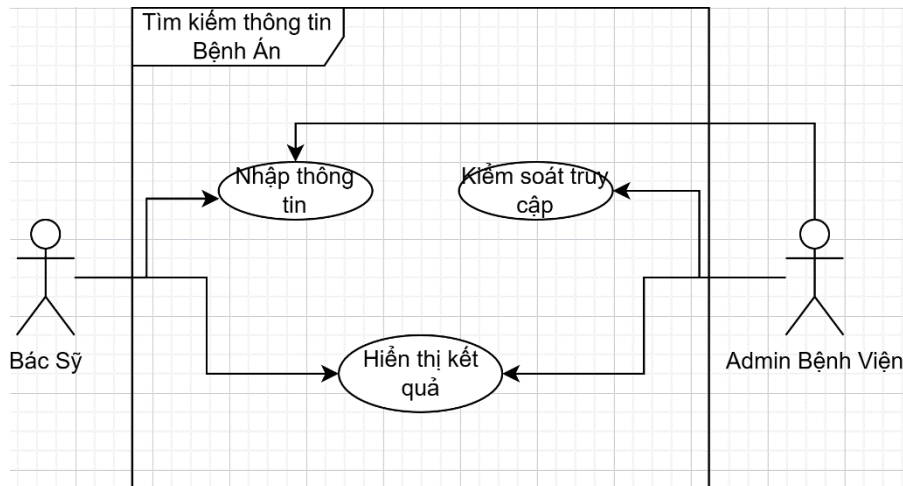
Mô tả: Use case này cho phép người dùng tìm kiếm thông tin về bệnh án của một bệnh nhân trong hệ thống.

Người dùng: Bác sĩ, Admin bệnh viện.

Luồng sự kiện chính:

- Người dùng truy cập vào chức năng tìm kiếm thông tin bệnh án.
- Admin bệnh viện có thể kiểm soát truy cập việc tìm kiếm thông tin
- Người dùng nhập thông tin cần tìm kiếm, bao gồm: mã số bệnh nhân hoặc tên bệnh nhân hoặc các thông tin khác liên quan đến bệnh nhân.
- Hệ thống tiến hành tìm kiếm thông tin trong CSDL đồ thị và trả về kết quả tìm kiếm bao gồm các thông tin chi tiết về bệnh án của bệnh nhân.

- Người dùng có thể tiếp tục tìm kiếm thông tin khác hoặc thoát khỏi chức năng tìm kiếm.



Hình 2- 3: Usecase Tìm kiếm thông tin bệnh án

#### d. Usecase: Phân phối công việc (cho Bác sĩ)

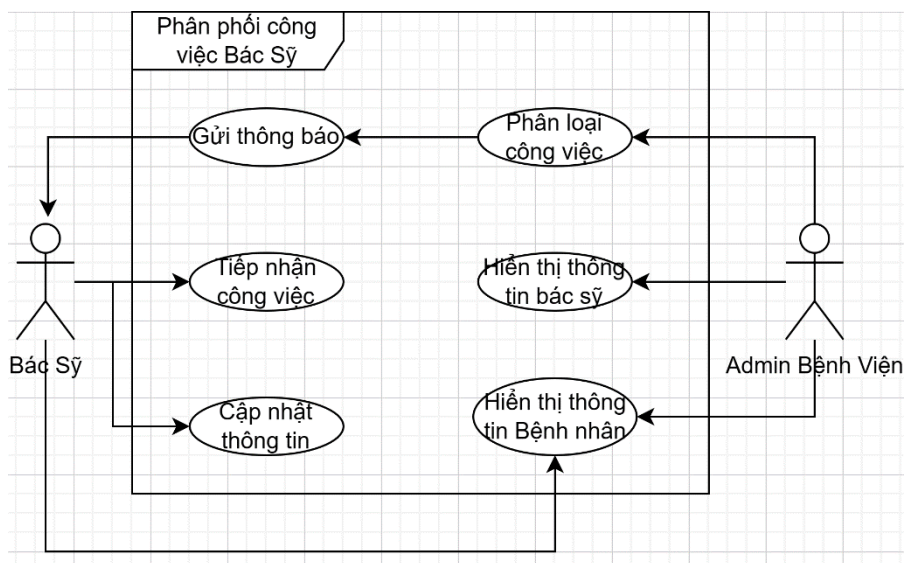
Mô tả: Use case này mô tả quá trình phân phối công việc cho các bác sĩ trong bệnh viện.

Người tương tác: Admin bệnh viện, bác sĩ

Kịch bản chính:

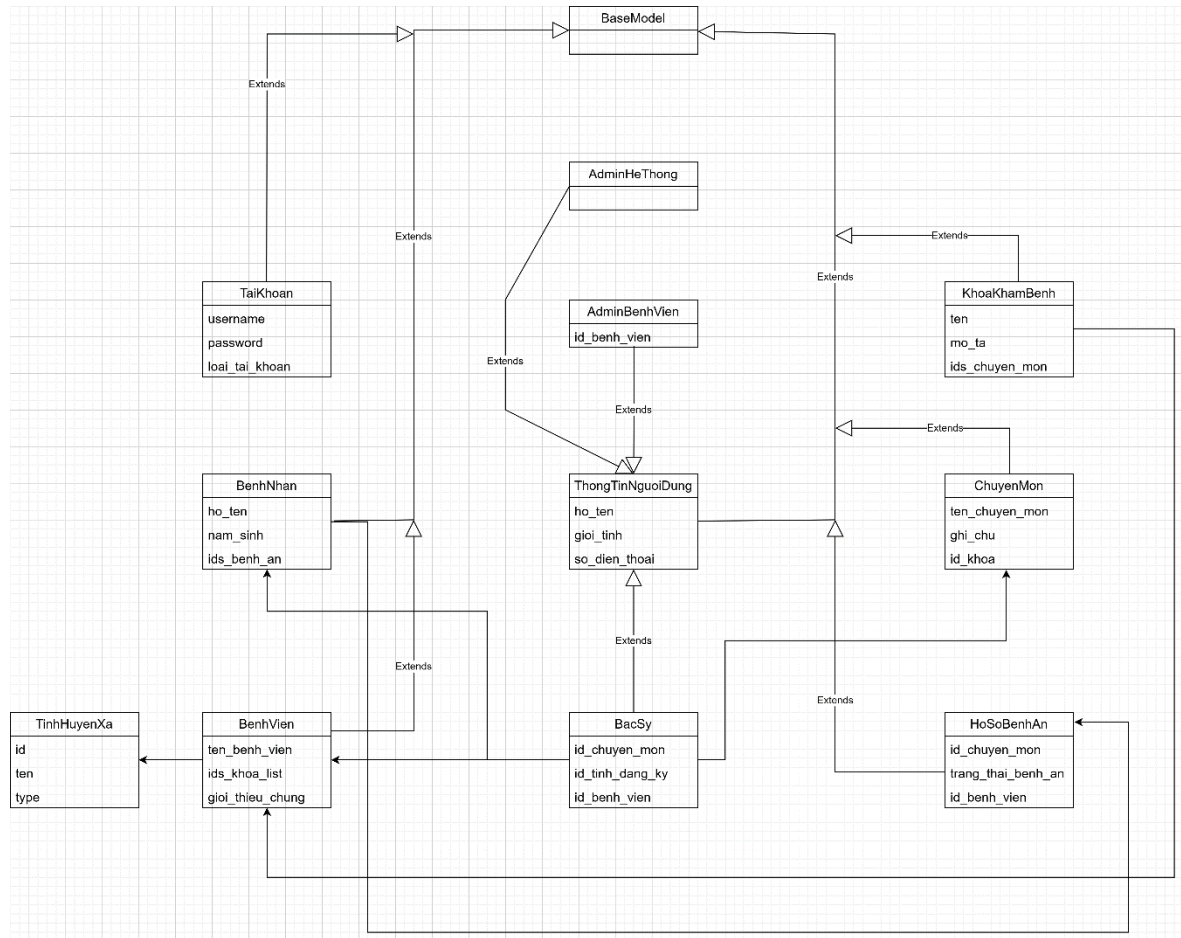
- Admin bệnh viện mở chức năng phân công công việc cho bác sĩ. Hệ thống hiển thị danh sách các bác sĩ đang hoạt động trong bệnh viện.
- Admin bệnh viện chọn bác sĩ cần phân công công việc và chọn chức năng phân công công việc. Hệ thống hiển thị các thông tin liên quan đến bác sĩ như chuyên môn, kinh nghiệm, số lượng bệnh nhân đang điều trị.
- Admin bệnh viện chọn loại công việc cần phân công cho bác sĩ, ví dụ như khám bệnh, chỉ định xét nghiệm, lập kế hoạch điều trị,...
- Hệ thống hiển thị thông tin về bệnh nhân đang chờ được phân công công việc, gồm tên bệnh nhân, thông tin bệnh án, trạng thái hiện tại của bệnh nhân.
- Admin bệnh viện chọn bệnh nhân cần phân công công việc cho bác sĩ.

- Hệ thống gửi thông tin về công việc được phân công đến bác sỹ tương ứng.
- Bác sỹ đăng nhập vào hệ thống và xem thông tin về công việc được phân công.
- Bác sỹ thực hiện công việc được phân công và cập nhật trạng thái của bệnh nhân.
- Hệ thống cập nhật trạng thái của bệnh nhân và thông tin về công việc đã thực hiện vào cơ sở dữ liệu.



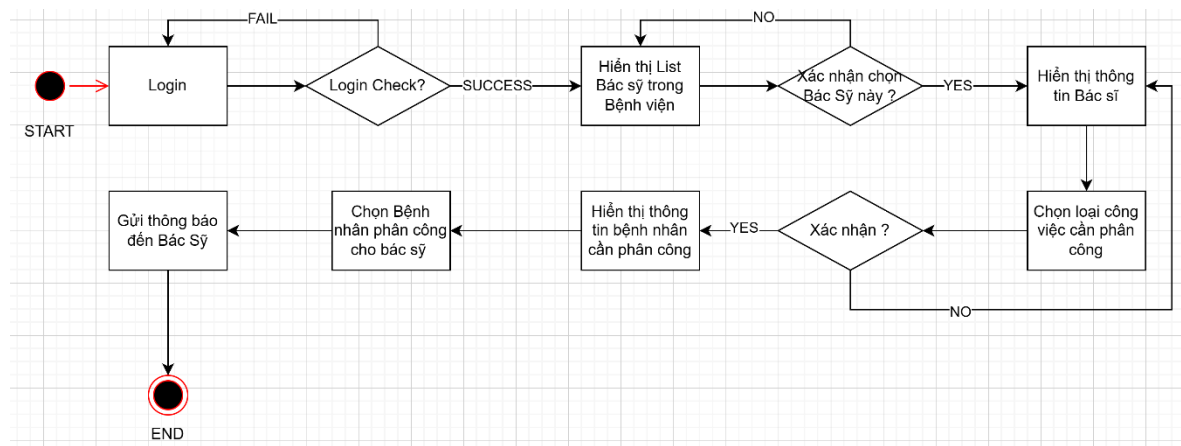
Hình 2- 4: Usecase Phân phối công việc cho Bác sỹ

### 2.3.2. Lập biểu đồ lớp.



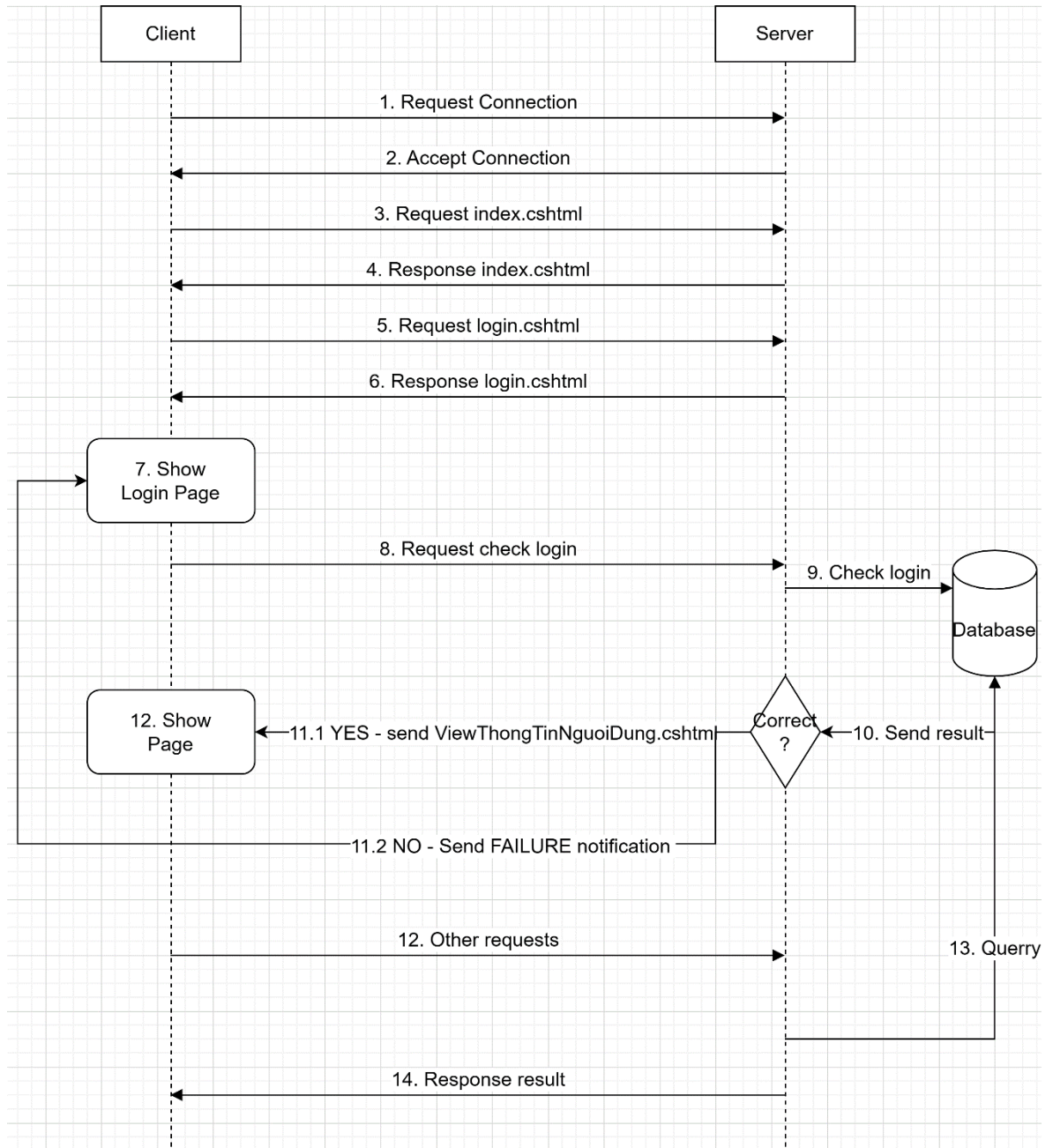
Hình 2- 5: Biểu đồ lớp

### 2.3.3. Lập biểu đồ hoạt động.



Hình 2- 6: Biểu đồ hoạt động Phân công công việc cho Bác sỹ

### 2.3.4. Lập biểu đồ tuần tự.



Hình 2- 7: Biểu đồ tuần tự quá trình Login

## 2.4. Kết luận chương

Chương 2 tập trung vào bài toán phát triển mạng thông tin sức khỏe và thiết kế hệ thống cho bài toán này. Luận văn đã trình bày về bài toán phát triển mạng thông tin sức khỏe sẽ giải quyết trong luận văn, các chức năng của nó, và quy trình nghiệp



vụ của hệ thống. Luận văn cũng đã tìm hiểu về việc xây dựng hệ thống phần mềm truy vấn và trực quan hóa để hỗ trợ việc quản lý và tìm kiếm thông tin sức khỏe hiệu quả.

Trong quá trình xây dựng hệ thống phần mềm truy vấn và trực quan hóa, luận văn đã lập use case để xác định các chức năng cần có và quy trình sử dụng của hệ thống. Luận văn đã sử dụng biểu đồ lớp để mô hình hóa các đối tượng và quan hệ trong hệ thống, biểu đồ hoạt động để mô tả luồng hoạt động của các chức năng, và biểu đồ tuần tự để minh họa quá trình tương tác giữa các thành phần của hệ thống.

Các kết quả của chương này giúp luận văn trình bày cách nhìn tổng quan về bài toán phát triển mạng thông tin sức khỏe và thiết kế hệ thống. Từ đó đã xác định được các yêu cầu cần đạt của hệ thống, và thiết kế một hệ thống phần mềm truy vấn và trực quan hóa để hỗ trợ việc quản lý và tìm kiếm thông tin sức khỏe một cách hiệu quả.

Chương tiếp theo sẽ tập trung vào việc thu thập thông tin ban đầu, kết quả và đánh giá của hệ thống đã xây dựng. Từ đó sẽ đánh giá mức độ đạt được của các yêu cầu đặt ra và phân tích hiệu suất, tính nhất quán, tính mở rộng, tính bảo mật và tính linh hoạt của hệ thống.



## CHƯƠNG 3. KẾT QUẢ VÀ ĐÁNH GIÁ

### 3.1. Thu thập và chia sẻ thông tin sức khỏe














#### Thu thập thông tin sức khỏe:

Trong phần này, luận văn trình bày về công việc thu thập thông tin sức khỏe trong mạng thông tin sức khỏe. Đây là một phần quan trọng trong quá trình xây dựng hệ thống, vì thông tin sức khỏe chính là cơ sở dữ liệu quan trọng để hỗ trợ quản lý và cung cấp dịch vụ chăm sóc sức khỏe.

Các nguồn thông tin sức khỏe thu thập: các hồ sơ bệnh án công khai trên các nguồn dữ liệu mở ở bệnh viện, phòng khám, các cơ sở y tế. Từ đó tiến hành chỉnh sửa dữ liệu sao cho phù hợp với mục đích của mạng thông tin sức khỏe

Các thông tin sức khỏe được thu thập bao gồm thông tin cá nhân, tiền sử bệnh, kết quả xét nghiệm, đơn thuốc, và các tài liệu y tế khác.

Các thông tin được lưu trữ dưới dạng file .csv với các trường dữ liệu tương ứng với các thông tin mà luận văn quan tâm được liệt kê dưới đây:

AdminBenhVien.csv	 AdminBenhVien.csv
AdminHeThong.csv	 AdminHeThong.csv
BacSy.csv	 BacSy.csv
BenhNhan.csv	 BenhNhan.csv
BenhVien.csv	 BenhVien.csv
CongViec.csv	 CongViec.csv
DichVuKhamBenh.csv	 DichVuKhamBenh.csv
HoSoBenhAn.csv	 HoSoBenhAn.csv
KhoaKhamBenh.csv	 HoSoBenhAn_4.csv
LichSuCongViec.csv	 KhoaKhamBenh.csv
TaiKhoan.csv	 LichSuCongViec.csv
TinhHuyenXa.csv	 TaiKhoan.csv
	 TinhHuyenXa.csv

### **Chia sẻ thông tin**

Luận văn sử dụng nguồn dữ liệu mở và đồng thời tiến hành thêm dữ liệu dạng ngẫu nhiên theo một mẫu cho trước để gia tăng số lượng bản ghi dữ liệu nhằm tăng khả năng đánh giá hiệu quả của hệ thống đã xây dựng. Các dữ liệu được thiết lập quyền riêng tư và đặt chế độ chỉ xem để đảm bảo tính bảo mật và quyền riêng tư trong quá trình thu thập thông tin sức khỏe. Hệ thống mạng thông tin sức khỏe được xây dựng trên hệ thống máy tính “local” do vậy đảm bảo tính bảo mật cũng như tính an toàn dữ liệu. Dữ liệu đều thiết đặt quyền truy cập, chỉ có người có mật khẩu mới có thể truy cập và xem thông tin hệ thống.

### **3.2. Kịch bản kiểm thử hệ thống**

Trong phần này, luận văn trình bày chi tiết về kịch bản kiểm thử được thiết kế để đánh giá hệ thống mạng thông tin sức khỏe theo các yêu cầu quan trọng như bảo mật, truy cập dễ dàng, truy xuất thông tin và tích hợp hệ thống. Dưới đây là một mô tả chi tiết về các bước kiểm thử và hoạt động tương ứng của từng yêu cầu:

#### **Kiểm thử bảo mật:**

- Thử đột nhập: luận văn thực hiện các kỹ thuật đột nhập vào hệ thống để kiểm tra tính bảo mật. Điều này bao gồm việc kiểm tra các lỗ hổng bảo mật và khả năng chống tấn công của hệ thống.
- Kiểm tra quyền truy cập: luận văn xác minh và kiểm tra quyền truy cập của người dùng vào hệ thống. Điều này đảm bảo rằng chỉ những người dùng được ủy quyền mới có thể truy cập vào thông tin sức khỏe.

#### **Kiểm thử truy cập dễ dàng:**

- Kiểm tra giao diện người dùng: luận văn đánh giá tính dễ sử dụng và trực quan của giao diện người dùng. Điều này đảm bảo rằng người dùng có thể dễ dàng tương tác và thực hiện các tác vụ trên hệ thống một cách thuận tiện.

- Kiểm tra thời gian phản hồi: luận văn đo và đánh giá thời gian phản hồi của hệ thống đối với các yêu cầu từ người dùng. Điều này giúp đảm bảo rằng hệ thống có khả năng phản hồi nhanh chóng và đáp ứng yêu cầu truy cập dễ dàng.

#### **Kiểm thử truy xuất thông tin:**

- Kiểm tra tốc độ truy xuất: luận văn đo và đánh giá tốc độ truy xuất thông tin từ cơ sở dữ liệu. Điều này giúp đảm bảo rằng hệ thống có khả năng truy xuất thông tin nhanh chóng và hiệu quả.
- Kiểm tra tính chính xác: luận văn kiểm tra tính chính xác của thông tin được truy xuất từ cơ sở dữ liệu. Điều này đảm bảo rằng thông tin được hiển thị cho người dùng là chính xác và đáng tin cậy.

#### **Kiểm thử tích hợp hệ thống:**

- Kiểm tra giao tiếp hệ thống: luận văn kiểm tra khả năng giao tiếp và tương tác giữa hệ thống mạng thông tin sức khỏe và các hệ thống khác. Điều này đảm bảo rằng hệ thống có khả năng tích hợp và làm việc một cách liền mạch với các hệ thống khác.
- Kiểm tra tính đúng đắn: luận văn kiểm tra tính đúng đắn của dữ liệu và thông tin được truyền qua lại giữa các hệ thống. Điều này đảm bảo rằng dữ liệu được chia sẻ và truyền tải một cách chính xác và không bị mất mát.

Qua các kịch bản kiểm thử này, luận văn sẽ có một cái nhìn toàn diện về hiệu năng và tính đúng đắn của hệ thống mạng thông tin sức khỏe. Kết quả từ quá trình kiểm thử này sẽ cung cấp thông tin quan trọng để đánh giá và cải thiện hệ thống, đảm bảo rằng nó đáp ứng tốt các yêu cầu quan trọng và đem lại trải nghiệm tốt cho người dùng.

### **3.3. Triển khai cài đặt và kết quả**

Để triển khai dự án mạng thông tin sức khỏe, dự án sử dụng Neo4j làm cơ sở dữ liệu đồ thị và C# để phát triển website quản lý.

Trước khi triển khai, dự án đã thiết kế và xây dựng schema cho cơ sở dữ liệu đồ thị Neo4j, bao gồm các nodes và relationships cần thiết để quản lý thông tin người dùng và phân công công việc cho bác sỹ.

Sau đó, dự án đã triển khai cài đặt Neo4j trên máy tính và tạo các nodes và relationships trong cơ sở dữ liệu. dự án sử dụng ngôn ngữ lập trình C# để xây dựng website mạng thông tin sức khỏe và kết nối với Neo4j sử dụng thư viện Neo4j.Driver. Website của dự án cung cấp cho người dùng các chức năng như quản lý thông tin người dùng, phân công công việc cho bác sỹ, tra cứu thông tin bệnh án và tìm kiếm bệnh nhân. Người dùng có thể đăng nhập vào hệ thống bằng tài khoản của mình để sử dụng các chức năng này.

Sau khi triển khai, dự án đã kiểm tra và đánh giá hiệu suất của hệ thống. Kết quả cho thấy, hệ thống hoạt động ổn định và đáp ứng được yêu cầu của người dùng.

Tổng kết, triển khai thành công dự án mạng thông tin sức khỏe dựa trên Neo4j và C# đã giúp dự án quản lý được thông tin người dùng và phân công công việc cho bác sỹ một cách hiệu quả và tiện lợi. Đồng thời, nó cũng cung cấp cho người dùng một giao diện trực quan và dễ sử dụng để tương tác với hệ thống.

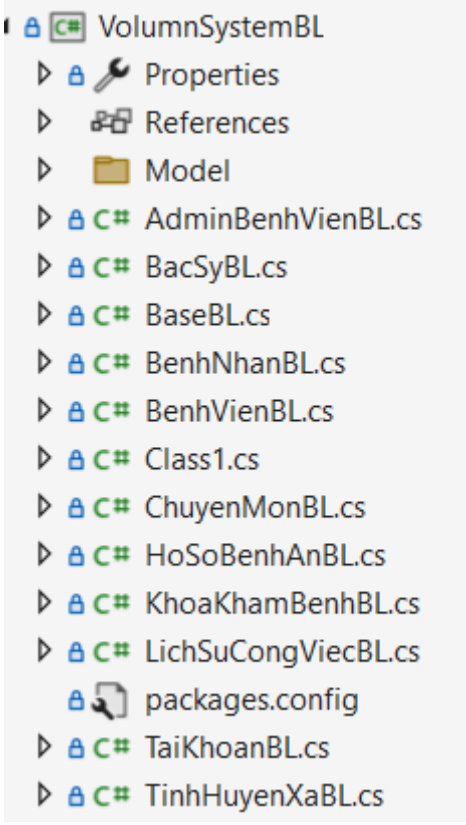
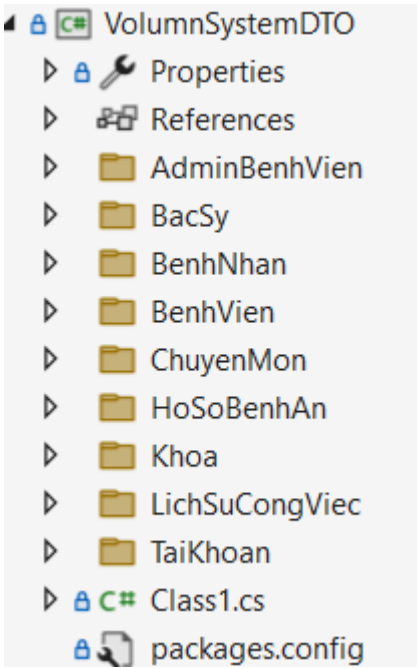
### **3.4. Lập trình các module**

Sau khi đã triển khai cài đặt CSDL đồ thị trên Neo4j, tiếp theo là lập trình các module để cung cấp các chức năng cho website mạng thông tin sức khỏe.

Dưới đây là các module:

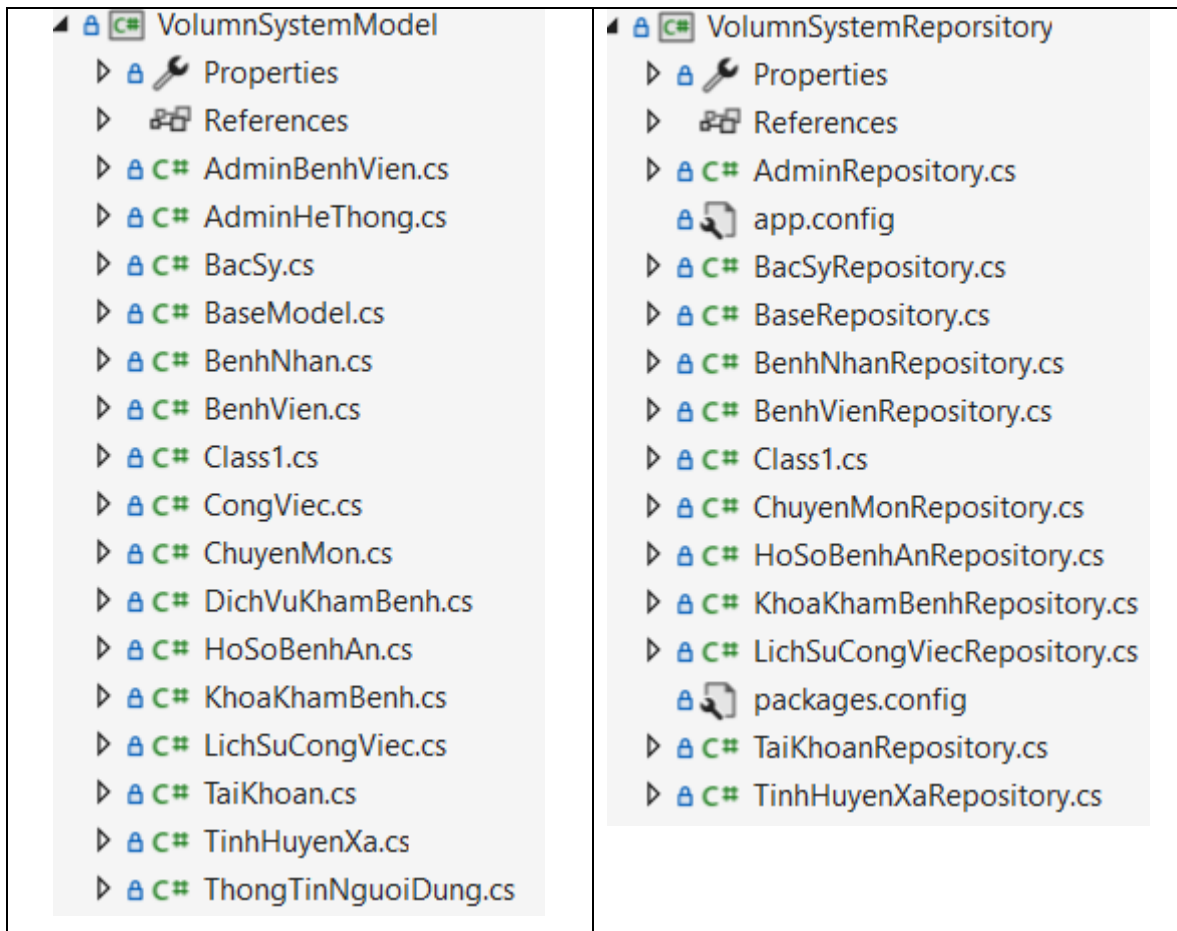
**VolumnSystemBL (Business Logic):** Module này chứa các lớp và phương thức để xử lý logic kinh doanh của ứng dụng. Điều này bao gồm các tính toán, quy trình và kiểm soát kinh doanh liên quan đến việc quản lý khối lượng (volume) trong hệ thống.

**VolumnSystemDTO (Data Transfer Object):** Module này chứa các đối tượng truyền tải dữ liệu giữa các thành phần của hệ thống. Điều này bao gồm đối tượng để đóng gói và truyền dữ liệu giữa các lớp.

 <p><b>VolumnSystemBL</b></p> <ul style="list-style-type: none"> <li>Properties</li> <li>References</li> <li>Model</li> <li>AdminBenhVienBL.cs</li> <li>BacSyBL.cs</li> <li>BaseBL.cs</li> <li>BenhNhanBL.cs</li> <li>BenhVienBL.cs</li> <li>Class1.cs</li> <li>ChuyenMonBL.cs</li> <li>HoSoBenhAnBL.cs</li> <li>KhoaKhamBenhBL.cs</li> <li>LichSuCongViecBL.cs</li> <li>packages.config</li> <li>TaiKhoanBL.cs</li> <li>TinhHuyenXaBL.cs</li> </ul>	 <p><b>VolumnSystemDTO</b></p> <ul style="list-style-type: none"> <li>Properties</li> <li>References</li> <li>AdminBenhVien</li> <li>BacSy</li> <li>BenhNhan</li> <li>BenhVien</li> <li>ChuyenMon</li> <li>HoSoBenhAn</li> <li>Khoa</li> <li>LichSuCongViec</li> <li>TaiKhoan</li> <li>Class1.cs</li> <li>packages.config</li> </ul>
--	---

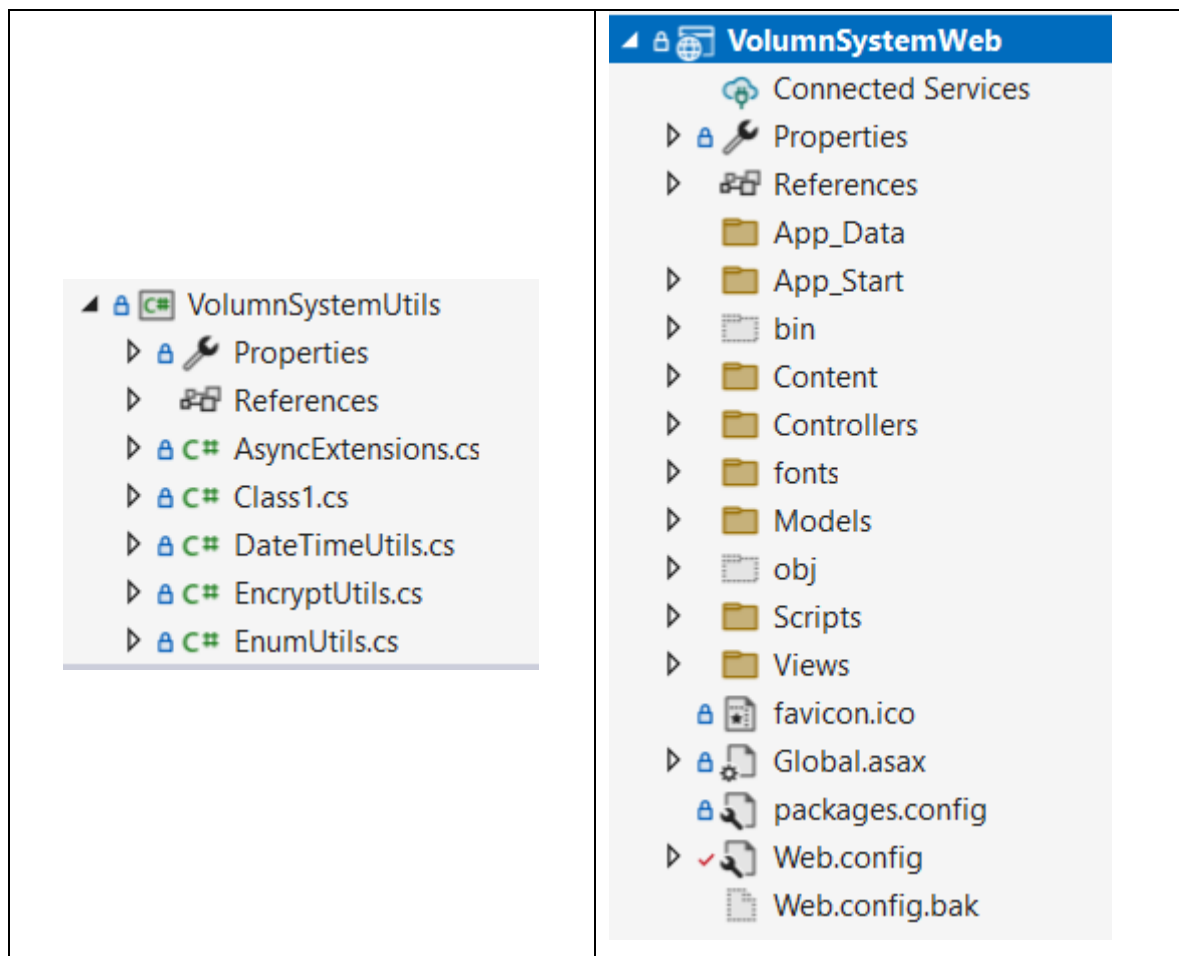
**VolumnSystemModel:** Module này chứa các lớp đại diện cho các thực thể (entities) trong hệ thống, bao gồm các lớp để biểu diễn các đối tượng như khối lượng, phiên, tài khoản người dùng, vv.

**VolumnSystemRepository:** Module này chứa các lớp và phương thức để truy cập và lưu trữ dữ liệu trong cơ sở dữ liệu. Điều này bao gồm các phương thức để lấy dữ liệu, thêm, sửa và xóa các thực thể từ cơ sở dữ liệu.



**VolumnSystemUtils (Utilities):** Module này chứa các phương thức tiện ích (utility) để hỗ trợ việc xử lý dữ liệu và thao tác với hệ thống. Điều này bao gồm các phương thức để mã hóa mật khẩu, kiểm tra đăng nhập, tính toán khối lượng, vv.

**VolumnSystemWeb:** Module này chứa các lớp và tài nguyên để hiển thị giao diện người dùng của ứng dụng web. Điều này bao gồm các trang web, các tài nguyên như CSS, JavaScript và hình ảnh, cũng như các phương thức để xử lý các yêu cầu HTTP và trả về dữ liệu cho người dùng.



### 3.5. Thực nghiệm hệ thống

Kết quả chạy thử nghiệm

#### *a. Đăng nhập hệ thống*

**BenhVienTot.com**

**Đăng nhập vào hệ thống**

Tên đăng nhập

★ abcd...

Mật khẩu

**Đăng nhập**

[Hoặc tạo tên đăng nhập mới](#)

Powered by [Dũng Phan](#)

Hình 3- 1: Màn hình đăng nhập hệ thống

### ***b. Công việc bác sỹ***

**BenhVienTot** ☰

🔔 4 🗨️ 3 👤 **Đỗ Nguyên Nam** ▾

📊 Dashboard

📅 **Lịch khám** ^

○ Lịch khám chi tiết

© Copyright **BenhVienTot**. All Rights Reserved  
Designed by [Dũng Phan](#)

Hình 3- 2: Màn hình công việc bác sỹ



### c. Lịch khám chi tiết

**BenhVienTot** Search

Dashboard  
Lịch khám  
Lịch khám chi tiết

**Công việc trong ngày**  
Home / Công việc / Công việc trong ngày

**Tìm kiếm**  
Ngày tìm kiếm: 05/02/2023 - 05/02/2023 **Tìm kiếm** **Đặt lại**

**Danh sách công việc trong ngày**

#	Tên bệnh án	Tên bệnh nhân	Ngày bắt đầu (dự tính)	Ngày kết thúc (dự tính)	Ngày bắt đầu (thực tế)	Trạng thái hoàn thành	
1	Nội khoa	Huỳnh Hữu Trung	28/04/2023	02/05/2023	02/05/2023	Chưa hoàn thành	Hoàn thành
2	Nội khoa	Đặng Hồng Nhật	28/04/2023	02/05/2023	02/05/2023	Chưa hoàn thành	Hoàn thành
3	Nội khoa	Hà Thành Dũng	28/04/2023	02/05/2023	02/05/2023	Chưa hoàn thành	Hoàn thành

Hình 3- 3: Màn hình lịch khám chi tiết (cho Bác sỹ)

### d. Chi tiết thông tin bệnh nhân

**BenhVienTot** Search

Dashboard  
Lịch khám  
Lịch khám chi tiết

**Chi tiết bệnh nhân**  
Home / Bệnh nhân / Chi tiết bệnh nhân

**Thông tin bệnh nhân**

Họ tên: Huỳnh Hữu Trung  
Giới tính: Nam  
Năm sinh: 1985  
Số điện thoại: 979538492  
Email: huynh.huu.t  
Địa chỉ: Tứ Liêm

**Danh sách bệnh án**

#	Tên bệnh án	Tên chuyên môn	Ngày nhập viện	Ngày xuất viện	Trạng thái hoàn thành
1	Nội khoa	Nội khoa	28/04/2023	01/05/2023	Chưa đóng bệnh án
2	Nội khoa	Nội khoa	28/04/2023	01/05/2023	Chưa đóng bệnh án
3	Nội khoa	Nội khoa	28/04/2023	01/05/2023	Chưa đóng bệnh án

Hình 3- 4: Màn hình chi tiết thông tin bệnh nhân

## 3.6. Đánh giá kết quả hệ thống

Kết quả hệ thống ứng với các kịch bản kiểm thử

**Đánh giá bảo mật:**

- Hệ thống sử dụng CSDL Neo4j version 5.8 cùng với library .NET version 4.4 là những phiên bản cập nhật nên có đầy đủ tính năng bảo vệ an toàn bảo mật cho hệ thống. Việc thử đột nhập hệ thống để xem các dữ liệu không thực hiện được.
- Các user đăng ký với hệ thống được phân ra thành các Type khác nhau với các quyền khác nhau. Việc kiểm tra ủy quyền của user trong hệ thống đã thực hiện thành công. User với Type nào thì có quyền tương ứng, không xem được dữ liệu của user Type khác

#### **Đánh giá giao diện người dùng:**

- Người dùng có thể dễ dàng tương tác và thực hiện các tác vụ trên hệ thống một cách thuận tiện. Bác sỹ có thể vào hệ thống và xem các công việc hàng ngày, các bệnh án cần phải được giải quyết. Bệnh nhân có thể kiểm tra dễ dàng tình trạng sức khỏe của bản thân cũng như thông tin các loại thuốc và cách sử dụng của chúng.
- Hệ thống có thời gian tương tác với phản hồi của người dùng rất nhanh chóng (mili-giây) và cho kết quả chính xác.

#### **Đánh giá truy xuất thông tin:**

- Hệ thống có tốc độ truy xuất thông tin đến cơ sở dữ liệu rất nhanh (40ms). Việc import dữ liệu từ các file hệ thống vào được thực hiện nhanh chóng với thời gian ngắn (120ms)
- Hệ thống truy xuất dữ liệu vào CSDL và lấy ra được thông tin chính xác với câu truy vấn đó. Đảm bảo kết quả vận hành hệ thống đạt yêu cầu

**Đánh giá tích hợp hệ thống:** Hệ thống được xây dựng với quy mô nhỏ và thực hiện trên các máy tính nội bộ. Do vậy chưa đánh giá được khả năng tích hợp, giao tiếp thông tin của hệ thống mạng thông tin sức khỏe với các mạng khác. Cần đưa hệ thống lên internet và thử nghiệm quy mô nhỏ để đánh giá được sát hơn khả năng tích hợp hệ thống.

### 3.7. Kết luận chương

Trong chương 3, luận văn đã tiến hành trình bày các cách thức thu thập và chia sẻ thông tin sức khỏe. Thông tin này được sử dụng trong luận văn đảm bảo các yêu cầu ban đầu về dữ liệu. Luận văn đã đưa ra kịch bản kiểm thử hệ thống để đánh giá sơ bộ hiệu quả, khả năng vận hành của hệ thống có đáp ứng với các yêu cầu đề ra ban đầu hay không. Sau đó luận văn đã tiến hành lập trình các module và thực nghiệm hệ thống.

Kết quả và đánh giá hệ thống mạng thông tin sức khỏe dựa trên các kịch bản kiểm thử đã được thiết kế. Các kịch bản kiểm thử này giúp luận văn kiểm tra các khía cạnh quan trọng của hệ thống và đưa ra đánh giá chi tiết về hiệu suất, tính bảo mật, truy cập dễ dàng, truy xuất thông tin và tích hợp hệ thống.

Hệ thống mạng thông tin sức khỏe đã được triển khai và hoạt động tốt với các chức năng chính như quản lý người dùng, phân công công việc cho bác sỹ và tìm kiếm thông tin bệnh án của bệnh nhân. Hệ thống sử dụng CSDL đồ thị Neo4j để lưu trữ và quản lý các thông tin liên quan đến sức khỏe, giúp cho việc tìm kiếm và truy xuất thông tin được thực hiện nhanh chóng và chính xác.

Các chức năng đã được triển khai đáp ứng được nhu cầu quản lý và cung cấp thông tin liên quan đến sức khỏe cho người dùng, đặc biệt là cho bác sỹ và các cơ sở y tế. Hệ thống cho phép quản lý các thông tin liên quan đến bệnh nhân, bác sỹ, cơ sở y tế, kết quả khám bệnh, chẩn đoán, hồ sơ bệnh án và các thông tin khác, giúp cho việc theo dõi tình trạng sức khỏe của bệnh nhân và đưa ra các phương án điều trị phù hợp.

Tuy nhiên, vẫn còn một số hạn chế trong quá trình triển khai và sử dụng hệ thống, như khả năng tương thích với các hệ thống khác, độ tin cậy của dữ liệu và hiệu suất khi có số lượng lớn người dùng truy cập cùng lúc. Để cải thiện và nâng cao hiệu quả của hệ thống, cần tiếp tục nghiên cứu và phát triển các tính năng mới, cải tiến các khâu quản lý và bảo mật dữ liệu, đồng thời tối ưu hóa hệ thống để đáp ứng nhu cầu của người dùng trong thời gian ngắn nhất.

## KẾT LUẬN

Trong đề tài này, luận văn đã thực hiện xây dựng một hệ thống mạng thông tin sức khỏe dựa trên CSDL đồ thị Neo4j và ngôn ngữ lập trình C#. Hệ thống này được phát triển để giải quyết các vấn đề trong việc quản lý thông tin sức khỏe và phân phối công việc trong lĩnh vực y tế. Luận văn đã thiết kế một cấu trúc CSDL đồ thị phù hợp cho việc lưu trữ thông tin sức khỏe, bao gồm các đối tượng như bệnh nhân, bác sỹ, thuốc và bệnh tật. Đồng thời, luận văn đã phát triển các chức năng để thêm, sửa đổi, xóa thông tin trên CSDL đồ thị.

Hệ thống cũng được thiết kế với các tính năng quản lý người dùng và phân phối công việc cho bác sỹ. Các chức năng này bao gồm quản lý tài khoản người dùng, phân công công việc và theo dõi tiến độ công việc.

Để triển khai hệ thống, luận văn sử dụng Neo4j làm CSDL đồ thị và lập trình bằng ngôn ngữ C#. Kết quả đạt được là hệ thống có thể quản lý được thông tin sức khỏe và phân phối công việc cho bác sỹ.

Tuy nhiên, hệ thống vẫn còn một số hạn chế như chưa hỗ trợ tính năng đa ngôn ngữ, chưa có tính năng tìm kiếm thông tin bệnh nhân theo nhiều tiêu chí khác nhau. Để cải thiện hệ thống, luận văn sẽ tiếp tục phát triển các tính năng mới và tối ưu hóa hiệu suất hệ thống.

Tổng kết lại, đề tài này đã đạt được mục tiêu ban đầu trong việc xây dựng hệ thống mạng thông tin sức khỏe để quản lý thông tin sức khỏe và phân phối công việc trong lĩnh vực y tế. Hệ thống này có tiềm năng để phát triển và ứng dụng rộng rãi trong thực tế.

## TÀI LIỆU THAM KHẢO

*Tài liệu tham khảo Tiếng Việt:*

- [1] Bộ Y tế (2011), *Kỷ yếu Hội thảo quốc tế về chuẩn hóa hệ thống thông tin y tế*, Quảng Nam.
- [2] Bộ Y tế (2008), *Báo cáo về “Số lượng người điều trị ngoại chấn 1 năm”*.
- [3] Cục Quản lý Khám, chữa bệnh - Bộ Y tế (2008), *Số liệu kiểm tra 932 bệnh viện năm 2008*.
- [4] Trần Xuân Chức, Trần Văn Tuyên, Hoàng Văn Tiến, Nguyễn Sơn Hải, Trần Thị Diệu Trinh (2014), “Giải pháp ứng dụng thu thập và cung cấp thông tin y tế chủ động tới cộng đồng”, Hội thi Tin học khối cán bộ, công chức trẻ toàn quốc lần II- 2014.
- [5] Lê Hồng Hà, Trần Xuân Chức, Kiều Mai (2015), “Y tế di động và triển vọng phát triển tại Việt Nam”, *Kỷ yếu Hội nghị ứng dụng Công nghệ thông tin trong ngành y tế lần thứ 7*, Bộ Y tế, tr. 114 - 116.
- [6] Nguyễn Huy Khánh, (2012). *Các công nghệ lập trình hiện đại*, Đại học Khoa học Tự nhiên TP. Hồ Chí Minh.
- [7] Nguyễn Hoàng Phương, Phí Văn Thâm, Nguyễn Tuấn Khoa (2008), *Kỷ yếu hội thảo khoa học: Ứng dụng Công nghệ thông tin trong quản lý bệnh viện*, Trung tâm tin học, Bộ Y tế.
- [8] Nguyễn Đức Thuận, Vũ Duy Hải, Trần Anh Vũ (2006), *Hệ thống thông tin y tế*, Nhà xuất bản Bách khoa Hà Nội.
- [9] WB, WHO, UNICEF, JICA (2013), *Bao phủ chăm sóc sức khỏe toàn dân: Việt Nam – Các sáng kiến quốc gia, thách thức và vai trò của hoạt động hợp tác quốc tế*.
- [10] WB (2014), *Phát triển kỹ năng: Xây dựng lực lượng lao động cho một nền kinh tế thị trường hiện đại ở Việt Nam*.

*Tài liệu tham khảo Tiếng Anh:*

- [11] Neha T, (2019 November), *Relational Data Model* [Online]. Available: <https://binaryterms.com/relational-data-model.html>
- [12] Bryce Merkl Sasaki, (2018 Jul), Graph Databases for Beginners: Data Modeling Pitfalls to Avoid [Online], Available: <https://neo4j.com/blog/data-modeling-pitfalls/>
- [13] Atakan Güney, (2019 Oct), Introduction to Resource Description Framework and SPARQL (RDF 101) [Online]. Available: <https://medium.com/@atakanguney94/introduction-to-resource-description-framework-and-sparql-rdf-101-5857f4a6a8a6>
- [14] Ajitesh Kumar, (2022 Augst), Knowledge Graph Concepts & Machine Learning: Examples [Online]. Available: <https://vitalflux.com/knowledge-graph-concepts-machine-learning-examples/>
- [15] Raouf Bouhali, Anne Laurent (2015), Exploiting RDF Open Data Using NoSQL Graph Databases, University of Montpellier
- [16] Alberz Akrawi (2010), Social Network System, SE-100 44 Stockholm, Sweden.
- [17] Alexander Richter, Michael Koch (2014), Functions of Social Networking Services.
- [18] Gunther Eysenbach, MD, MPH (2008), Medicine 2.0 Proceedings, Toronto, Canada.
- [19] Honigman (2010), Friending, Tweeting, Blogging & LinkingIn: Legal Considerations for Health Care Providers.

- [20] Ian Robinson, Jim Webber, and Emil Eifrem (2013), *Graph Databases*, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.
- [21] Ian Robinson (2010), *RESTful Application Protocols*.
- [22] IDG Communication, Inc., (2018), *State of Digital Business Transformation*.
- [23] Marcos Boyington, Po Chen, Grace Kum, Van Le-Pham, Eric Morales, Jake Warmerdam, Cheuk (Anna) Yu, Jingren Zhou (2004), *Design Specifications for Social Networking System*.
- [24] Mark Needham, Amy E. Hodler, (2019), *Graph Algorithms Practical Examples in Apache Spark & Neo4j*, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.
- [25] The Neo4j Team neo4j.org (2013), *The Neo4j Manual*.