

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

VIỆN TRÍ TUỆ NHÂN TẠO

\*\*\*



BÁO CÁO MÔN KỸ THUẬT VÀ CÔNG NGHỆ DỮ LIỆU LỚN

ĐỀ TÀI

GRAPH ANALYTICS WITH CHAT DATA USING NEO4J

Nhóm sinh viên thực hiện:

- Nguyễn Tuấn Thành
- Nguyễn Mạnh Hùng
- Nguyễn Công Thành
- Đinh Văn Sinh
- Vũ Đình Thọ

Giảng viên hướng dẫn:

TS. Trần Hồng Việt  
Lương Sơn Bá

HÀ NỘI, 12/2024

## MỞ ĐẦU

Công nghệ Big data đã đạt đến đỉnh cao trong việc thực hiện các chức năng của nó. Trong tháng 8/2015 Big data đã vượt ra khỏi bảng xếp hạng những công nghệ mới nổi Cycle Hype của Gartner và tạo một tiếng vang lớn cho xu hướng công nghệ của thế giới. Big data chứa trong mình rất nhiều thông tin quý giá mà trích xuất thành công, nó sẽ giúp rất nhiều trong lĩnh vực như y tế, giao thông, giáo dục, xã hội, tài chính và rất nhiều lĩnh vực khác.

Chính vì thế các framework, DBSM ngày càng càng được xử lý, phát triển và áp dụng mạnh. Một trong những hệ quản trị cơ sở dữ liệu đồ thị tiên phong và sử dụng rộng rãi nhất hiện nay là Neo4j.

Do thông tin và tốc độ phát triển mạng xã hội ngày càng cao lên một cách chóng mặt nên chúng em đã chọn đề tài “ **Graph analytics with chat data using Neo4j**” để làm báo cáo kết thúc môn học của mình.

Báo cáo gồm 4 chương:

Chương 1: Tổng quan về dữ liệu lớn và NoSQL và Neo4j.

Chương 2: Thuật toán dùng cho phân lớp dữ liệu.

Chương 3: Ứng dụng Neo4j để phân tích và trực quan hóa dữ liệu.

Chương 4: Kết luận và hướng phát triển.

## **MỤC LỤC**

### **CHƯƠNG 1: TỔNG QUAN VỀ DỮ LIỆU LỚN, NOSQL VÀ NEO4J**

- 1.1 Định nghĩa.
- 1.2 Đặc trưng cơ bản của dữ liệu lớn.
- 1.3 Tổng quan về NoSQL.
- 1.4 Tổng quan về Neo4j.

### **CHƯƠNG 2 : CÁC THUẬT TOÁN DÙNG CHO PHÂN LỚP DỮ LIỆU**

- 2.1 Thuật toán PageRank.
- 2.2 Thuật toán Label Propagation Algorithm (LPA).

### **CHƯƠNG 3: ỨNG DỤNG NEO4J ĐỂ PHÂN TÍCH VÀ TRỰC QUAN HÓA DỮ LIỆU**

- 3.1 Tổng quan về dữ liệu chat.
- 3.2 Phân tích mối quan hệ.
- 3.3 Phân tích nâng cao.
- 3.4 Trực quan hóa.

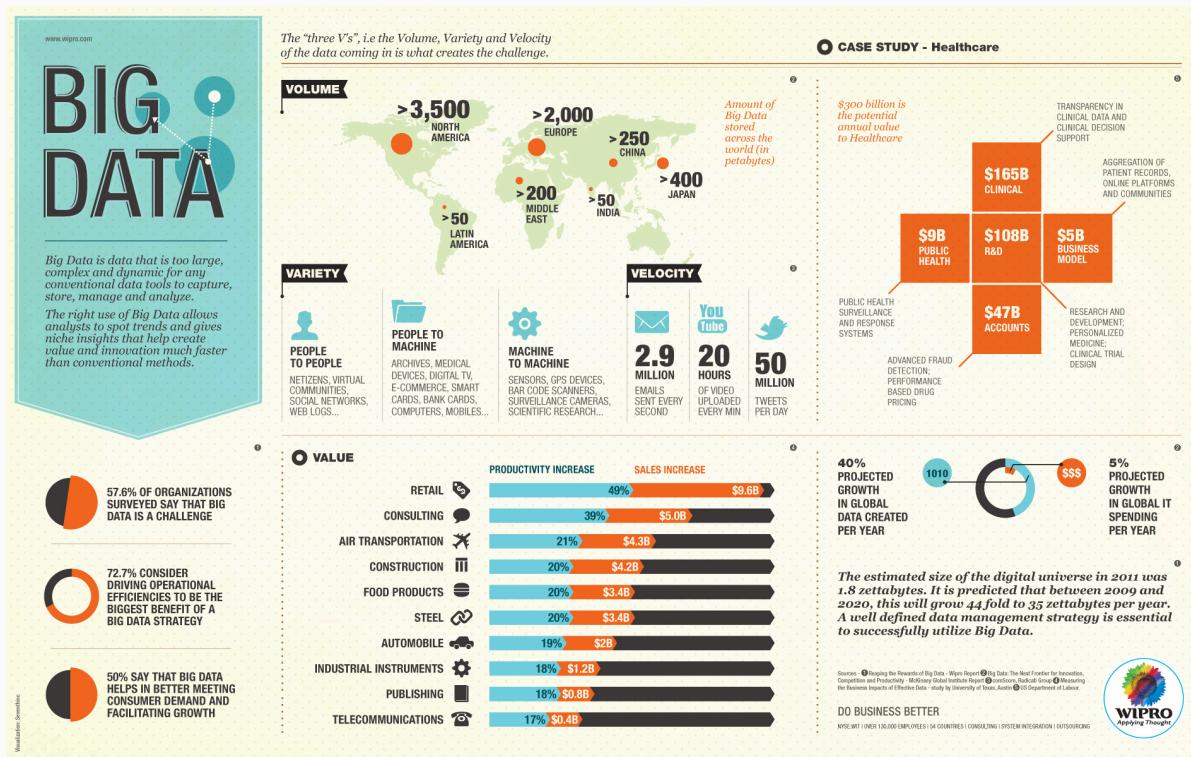
### **CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

- 4.1 Kết luận.
- 4.2 Hướng phát triển.

# CHƯƠNG 1: TỔNG QUAN VỀ DỮ LIỆU LỚN, NOSQL VÀ NEO4J

## 1.1 Định nghĩa.

➤ Theo wikipedia: Dữ liệu lớn là một thuật ngữ chỉ bộ dữ liệu lớn hoặc phức tạp mà các phương pháp truyền thống không đủ các ứng dụng để xử lý dữ liệu này. Theo Gartner : Dữ liệu lớn là những nguồn thông tin có đặc điểm chung khởi lượng lớn, tốc độ nhanh và dữ liệu định dạng dưới nhiều hình thức khác nhau, do đó muốn khai thác được phải đòi hỏi phải có hình thức mới để đưa ra quyết định khám phá và tối ưu hóa quy trình. Dữ liệu đến từ rất nhiều nguồn khác nhau:

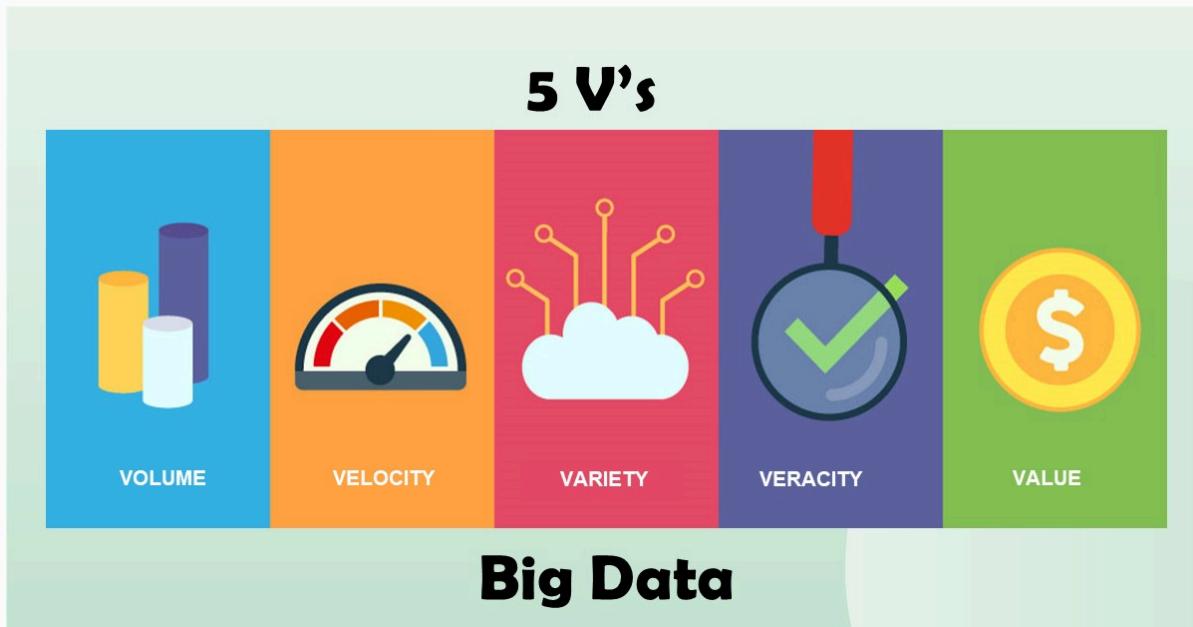


Hình 1. Minh họa nguồn gốc của dữ liệu.

➤ Một số lợi ích có thể mang lại như: Cắt giảm chi phí, tiết kiệm thời gian và giúp tối ưu hóa sản phẩm, hỗ trợ con người đưa ra những quyết định đúng và hợp lý hơn.

## 1.2 Đặc trưng cơ bản của dữ liệu lớn.

- (1) Khối lượng lớn (Volume): Khối lượng dữ liệu rất lớn và đang ngày càng tăng lên, tính đến 2014 thì có thể trong khoảng vài trăm terabyte.
- (2) Tốc độ (Velocity): Khối lượng dữ liệu gia tăng rất nhanh.
- (3) Đa dạng (Variety): Ngày nay hơn 80% dữ liệu được sinh ra là phi cấu trúc( tài liệu, blog, hình ảnh,...)
- (4) Độ tin cậy/chính xác(Veracity): Bài toán phân tích và loại bỏ dữ liệu thiếu chính xác và nhiễu đang là tính chất quan trọng của bigdata.
- (5) Giá trị(Value): Giá trị thông tin mang lại.



Hình 2.5V of Big Data

## 1.3 Tổng quan về NoSQL.

- NoSQL là một loại hệ thống quản lý cơ sở dữ liệu (DBMS) được thiết kế để xử lý và lưu trữ khối lượng lớn dữ liệu phi cấu trúc và bán cấu trúc. NoSQL sử dụng các mô hình dữ liệu linh hoạt có thể thích ứng với những thay đổi trong cấu trúc dữ liệu và có khả năng mở rộng theo chiều ngang để xử lý lượng dữ liệu ngày càng tăng.



Hình 3.Biểu tượng của NoSQL

- Cơ sở dữ liệu NoSQL thường được phân thành bốn loại chính: Document databases (mongoDB), Key-value stores (Cassandra, Apache HBASE), Wide Column stores (redis), Graph databases (Neo4j).

Document Database	Graph Databases
<p>Couchbase MarkLogic™ mongoDB</p>	<p>Neo4j InfiniteGraph The Distributed Graph Database</p>
Wide Column Stores	Key-Value Databases
<p>redis Amazon DynamoDB riak AEROSPIKE</p>	<p>HYPERTABLE INC ACCUMULO™ Cassandra APACHE HBASE Amazon SimpleDB</p>

=> Kết luận: là một loại hệ thống quản lý cơ sở dữ liệu (DBMS) xử lý và lưu trữ khối lượng lớn dữ liệu trong đó có Cơ sở dữ liệu đồ thị (graph database).

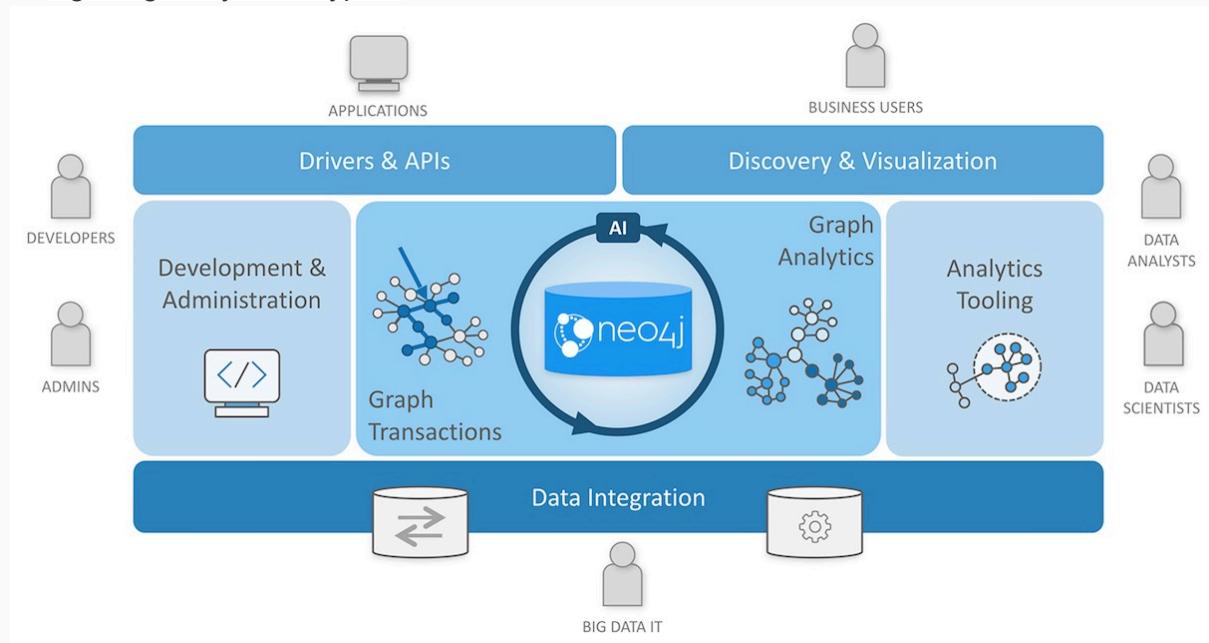
## 1.4 Tổng quan về Neo4j.

➤ Định nghĩa: là một cơ sở dữ liệu đồ thị (graph database) mạnh mẽ, được thiết kế để lưu trữ, quản lý và truy vấn dữ liệu dựa trên cấu trúc đồ thị, tập trung vào các nút (nodes), cạnh (relationships), và thuộc tính (properties).

### ➤ Lưu trữ và quản lý dữ liệu

- Neo4j lưu trữ dữ liệu dưới dạng đồ thị lồng nhau thay vì dạng bảng như cơ sở dữ liệu quan hệ.
- Tối ưu hóa các mối quan hệ, cho phép truy vấn nhanh chóng nhờ vào cách tổ chức và duyệt đồ thị hiệu quả.

### ➤ Ngôn ngữ truy vấn: Cypher



# CHƯƠNG 2 : CÁC THUẬT TOÁN DÙNG CHO PHÂN LỚP DỮ LIỆU

## 2.1 Thuật toán PageRank.

### a) Tổng quan

➤ Giới thiệu : PageRank là một thuật toán phân tích đồ thị nổi tiếng, được sử dụng ban đầu bởi Google để xếp hạng các trang web dựa trên liên kết giữa chúng. Ý tưởng chính là mỗi

node (đỉnh) trong đồ thị nhận được "điểm xếp hạng" dựa trên điểm của các node liên kết đến nó. Các node quan trọng hơn (được nhiều node khác liên kết) sẽ có điểm PageRank cao hơn.

➤ Ý tưởng :

- Xác định người dùng quan trọng nhất trong một mạng hội thoại.
- Xác định các tin nhắn hoặc chủ đề có ảnh hưởng lớn trong hội thoại.

b) Ý tưởng cơ bản

PageRank gán điểm PR cho mỗi node dựa trên công thức:

$$PR(i) = \frac{1-d}{N} + d \cdot \sum_{j \in L(i)} \frac{PR(j)}{C(j)}$$

- $PR(i)$  : Điểm PageRank của node  $i$ .
- $N$ : Số lượng node trong đồ thị.
- $d$ : Hệ số giảm( $0 < d < 1$ ), thường là 0.85, đại diện cho xác suất tiếp tục theo liên kết.
- $L(i)$ : Tập các node liên kết đến node  $i$ .
- $C(j)$ : Số lượng liên kết ra từ node  $j$ .
- $\frac{1-d}{N}$ : Thành phần "nhảy ngẫu nhiên" (random jump), đảm bảo điểm PageRank không bị mắc kẹt.

c) Triển khai

- Khởi tạo:

- Gán giá trị PageRank ban đầu cho mỗi node  $PR(i) = \frac{1}{N}$ .

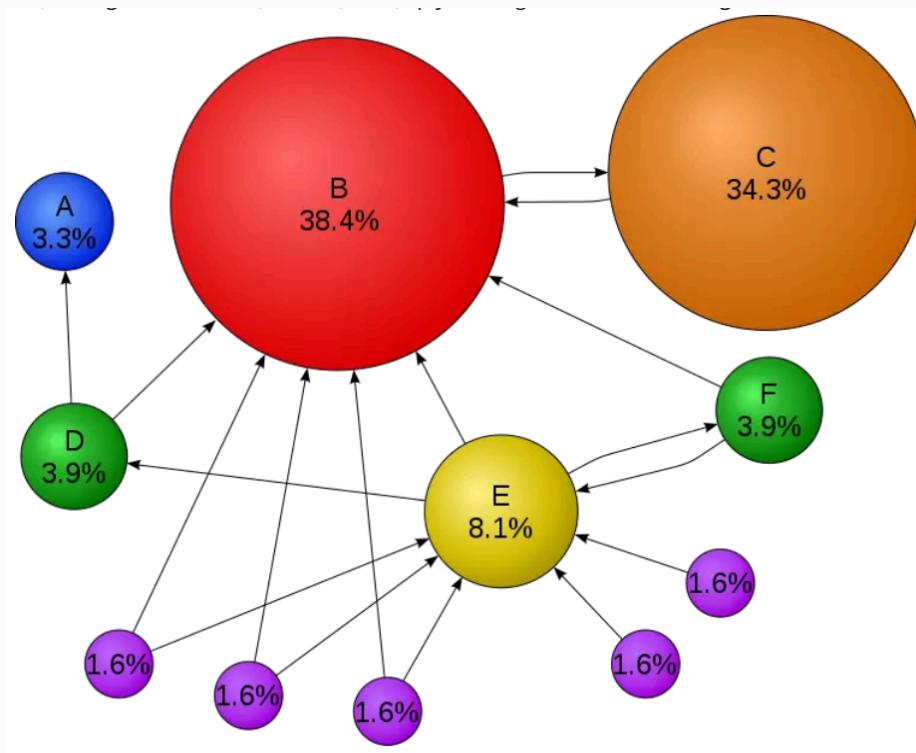
- Tính toán lặp:

- Tính giá trị  $PR_{\text{mới}}$  cho mỗi node dựa trên giá trị từ bước trước đó.
- Tiếp tục lặp cho đến khi giá trị hội tụ (khoảng chênh lệch giữa các lần lặp dưới một ngưỡng nhỏ, ví dụ  $10^{-6}$ ).

- Kết quả:

- Các node có giá trị PageRank cao nhất được xem là quan trọng nhất trong đồ thị.

d) Ví dụ minh họa



Một minh họa đơn giản về thuật toán PageRank. Phần trăm thể hiện tầm quan trọng của các trang và các mũi tên biểu thị siêu liên kết. Rank của các trang liên kết lại dựa trên rank của các trang khác liên kết tới nó. Do đó, *PageRank* của một trang web luôn được xác định dựa trên *PageRank* của các trang web khác.

## 2.2 Thuật toán Label Propagation Algorithm (LPA).

### a) Tổng quan

➤ Giới thiệu : LPA là một thuật toán không giám sát được sử dụng để phát hiện cộng đồng trong đồ thị. Ý tưởng chính là các node trong cùng một cộng đồng có xu hướng chia sẻ nhãn giống nhau.

➤ Ý tưởng :

- Mỗi node trong đồ thị bắt đầu với một nhãn duy nhất.
- Qua các bước lặp (iterations), node cập nhật nhãn của mình dựa trên nhãn phổ biến nhất từ các hàng xóm.
- Quá trình kết thúc khi không còn sự thay đổi nhãn hoặc đạt số lần lặp tối đa.

### b) Triển khai

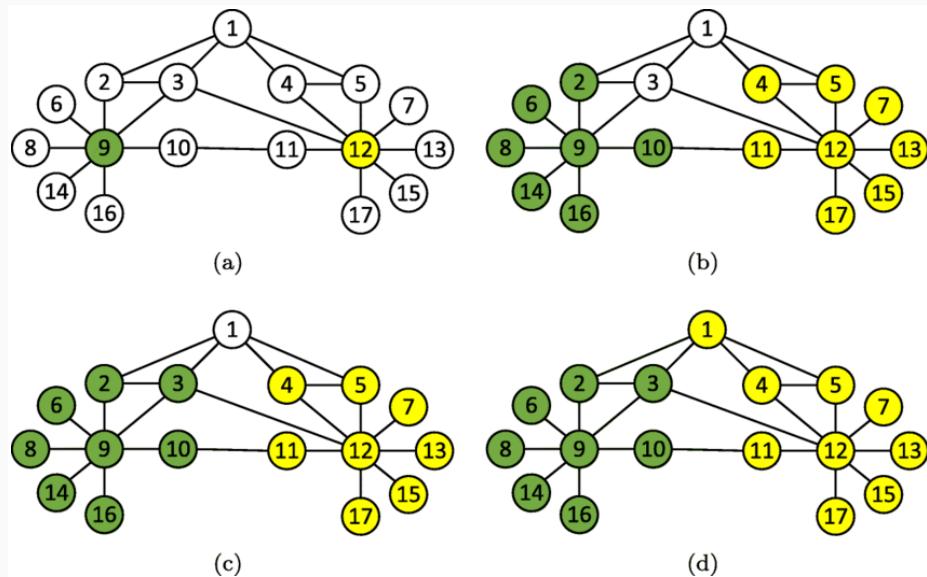
- Khởi tạo: Gán một nhãn duy nhất cho mỗi node.

- Lan truyền nhãn:

- Với mỗi node, kiểm tra nhãn của các hàng xóm.
- Gán nhãn phổ biến nhất trong hàng xóm đó cho node hiện tại.

- Lặp lại: Tiếp tục quá trình cho đến khi nhãn ổn định hoặc đạt số lần lặp tối đa.

c) Ví dụ minh họa



## CHƯƠNG 3: ỨNG DỤNG NEO4J ĐỂ PHÂN TÍCH VÀ TRỰC QUAN HÓA DỮ LIỆU

### 3.1 Tổng quan về dữ liệu chat.

- Các đỉnh bao gồm các label sau: User, Team, ChatItem, TeamChatSession
- Các cạnh bao gồm các quan hệ sau: CreateChat, CreateSession, InteractsWith, Join, Leave, Mentioned, OwnedBy, PartOf, ResponseTo
- Các thuộc tính bao gồm: id, timeStamp

File Name	Fields	Description
chat_create_team_chat.csv	userid	the user id assigned to the user
	teamid	the id of the team

	teamChatSessionID	a unique id for the chat session
	timestamp	a timestamp denoting when the chat session created
chat_item_team_chat.csv	userid	the user id assigned to the user
	teamchatsessionid	a unique id for the chat session
	chatitemid	a unique id for the chat item
	timestamp	a timestamp denoting when the chat item created
chat_join_team_chat.csv	userid	the user id assigned to the user
	teamChatSessionID	a unique id for the chat session
	timestamp	a timestamp denoting when the user join in a chat session
chat_leave_team_chat.csv	userid	the user id assigned to the user
	teamchatsessionid	a unique id for the chat session
	timestamp	a timestamp denoting when the user leave a chat session
chat_mention_team_chat.csv	ChatItemId	the id of the ChatItem

	userid	the user id assigned to the user
	timeStamp	a timestamp denoting when the user mentioned by a chat item
chat_respond_team_chat.csv	chatid1	the id of the chat post 1
	chatid2	the id of the chat post 2
	timestamp	a timestamp denoting when the chat post 1 responds to the chat post 2

## 3.2 Phân tích mối quan hệ.

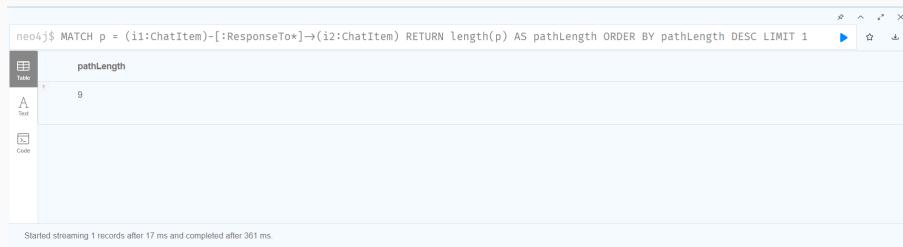
### 3.2.1. Phân tích chuỗi hội thoại dài nhất

(1) Tạo quan hệ ResponseTo giữa các ChatItem:

```

1 LOAD CSV FROM "file:///chat_respond_team_chat.csv" AS row
2 MATCH (c1:ChatItem {id: toInteger(row[0])})
3 MATCH (c2:ChatItem {id: toInteger(row[1])})
4 MERGE (c1)-[:ResponseTo {timeStamp: row[2]}]->(c2)
    
```

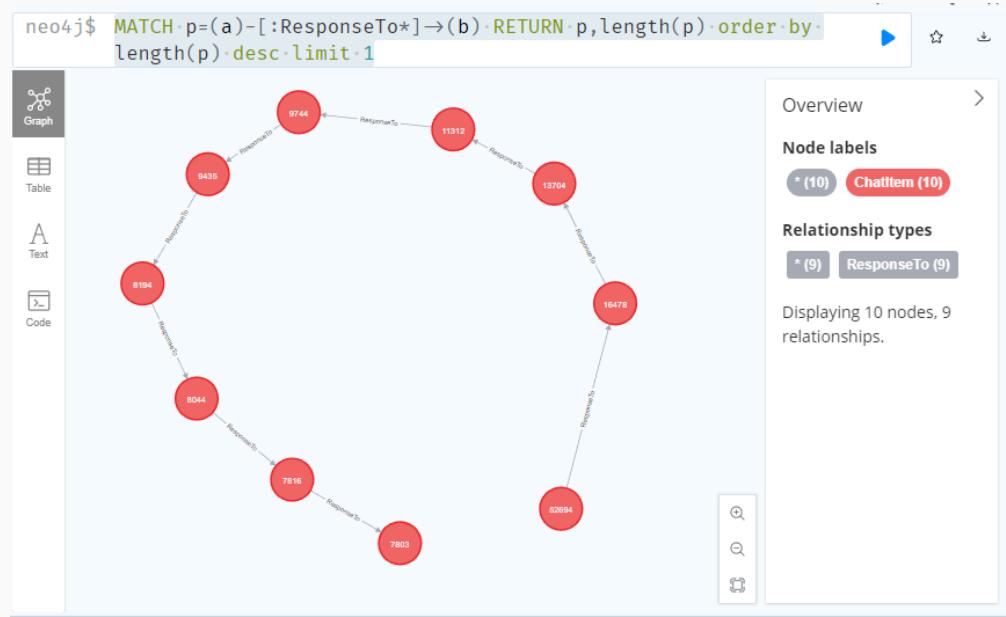
(2) Độ dài của đoạn chat dài nhất:



The screenshot shows the Neo4j browser interface with a results table. The table has one column labeled 'pathLength'. There is one row with the value '9'. Below the table, the status bar indicates: 'Started streaming 1 records after 17 ms and completed after 361 ms'.

pathLength
9

(3) Trực quan số tin nhắn trong đoạn chat dài nhất:



(4) Số người tham gia đoạn chat dài nhất:

```

1   match p = (ii)-[:ResponseTo*]-(i2)
2   where length(p) = 9
3   with p
4   match (u)-[:CreateChat]-(i)
5   where i in nodes(p)
6   return count(distinct u)
7
8   count(distinct u)
9   5

```

(5) Đánh giá:

=> Đoạn chat dài nhất chỉ có 9 tin nhắn của 5 người dùng cho thấy tin nhắn trong cuộc trò chuyện khá ngắn gọn khi trung bình mỗi người chỉ có 2 tin nhắn .

### 3.2.2. Phân tích sự tương quan giữa người dùng hoạt động nhiều nhất (chattiest users) và nhóm hoạt động nhiều nhất (chattiest teams)

#### 1. Top 10 người dùng hoạt động nhiều nhất:

	u.id	count(i)
1	394	115
2	2067	111
3	1087	109
4	209	109
5	554	107
6	1627	105
7	516	105
8	999	105
9	668	104

Lấy top 3 người dùng có số đoạn chat dài nhất :

Users	Number of Chats
394	115
2067	111
209	109
1087	109

2. Top 10 nhóm hoạt động nhiều nhất :



The screenshot shows the Neo4j browser interface with a query results table. The query is:neo4j\$ match (i)-[:PartOf\*]→(c)-[:OwnedBy\*]→(t) return t.id, count(c) order by count(c) desc limit 10

```
Table
```

	t.id	count(c)
1	82	1324
2	185	1036
3	112	957
4	18	844
5	194	836
6	129	814
7	52	788
8	136	783
9	146	746

Lấy top 3 team có số đoạn chat nhiều nhất :

<b>Teams</b>	<b>Number of Chats</b>
82	1324
185	1036
112	957

### 3. Đánh giá :

```

1 match (u)-[:CreateChat*]→(i)-[:PartOf*]→(c)-[:OwnedBy*]→(t)
2 return u.id, t.id, count(c)
3 order by count(c) desc limit 10
4

```

u.id	t.id	count(c)
394	63	115
2067	7	111
209	7	109
1087	77	109
554	181	107
999	52	105
1627	7	105
516	7	105
668	89	104

=> Ta thấy rằng người dùng có id là 999 thuộc team 52 là người dùng hoạt động nhiều trong nhóm cũng hoạt động nhiều, còn lại các người dùng nhiều khác đều nằm ngoài team hoạt động nhiều. Điều này cho thấy người dùng nhiều đa số là hoạt động trong team hoạt động ít .

#### 3.2.3. Phân tích hoạt động của các nhóm người dùng

##### 1. Tạo quan hệ InteractsWith :

```

MATCH (u1:User)-[:CreateChat]->(i1:ChatItem)-[:Mentioned]->(u2:User)
MERGE (u1)-[:InteractsWith]->(u2)
WITH u1, u2
MATCH
(u1:User)-[:CreateChat]->(i1:ChatItem)-[:ResponseTo]->(i2:ChatItem)<-[:CreateChat]->(u2:User)
MERGE (u1)-[:InteractsWith]->(u2)
WITH u1
MATCH (u1)-[r:InteractsWith]->(u1)
DELETE r

```

2. Kiểm tra neighbors của một vài người dùng hoạt động nhiều thuộc top 10:

```
neo4j$ match (u1:User{id:394})-[r:InteractsWith]-(u2:User) with u1,collect(distinct u2.id) as neighbors return u1.id,neighbors
```

u1.id	neighbors
394	[1012, 2011, 1782, 1997]

Started streaming 1 records after 13 ms and completed after 16 ms.

```
neo4j$ match (u1:User{id:2011})-[r:InteractsWith]-(u2:User) with u1,collect(distinct u2.id) as neighbors return u1.id,neighbors
```

u1.id	neighbors
2011	[394, 1782, 1012, 1997]

Started streaming 1 records after 13 ms and completed after 14 ms.

3. Bảng tính hệ số tương tác ( đánh giá tương tác của người dùng và neighbors của họ):

Hệ số tương tác = Tổng số tương tác / Tổng số tương tác có thể xảy ra

Trong đó: Tổng số tương tác có thể xảy ra =  $K*(K-1)$  , K là số lượng người dùng trong neighbors + người dùng đó.

Top 3 người dùng có hệ số tương tác cao nhất là:

User ID	Coefficient
209	0.95
394	1,00
2067	0,93

### 3.3 Phân tích nâng cao.

#### 3.3.1. Phát hiện cộng đồng (Communication Detection)

(1) Tạo 1 graph projection để phân tích đồ thị bằng các thuật toán trong Graph Data Science

```
1 CALL gds.graph.project(
2   'newChatGraph',
3   ['User', 'TeamChatSession'],
4   {Mentioned: {orientation: 'NATURAL'}, CreatesChat:
5     {orientation: 'NATURAL'}, Joins: {orientation: 'NATURAL'}})
6 
```

nodeProjection	relationshipProjectio
{User: {label: "User", properties: {}}, TeamChatSession: {label: "TeamChatSession", properties: {}}}	{CreatesChat: {aggregat
	erse: false, propert
	ies: {}, type: "Men
	tion: "NATURAL", indexI

MAX COLUMN WIDTH:

(2) Phát hiện ra các cộng đồng bằng thuật toán Label Propagation

```
1 CALL gds.labelPropagation.stream('newChatGraph')
2 YIELD nodeId, communityId
3 WITH gds.util.asNode(nodeId) AS node, communityId
4 RETURN node.id AS userId, communityId
5 ORDER BY communityId, userId
6 
```

	userId	communityId
1	9	254
2	437	254
3	568	254
4	1204	254
5	1429	254
6	1454	254
7		

Started streaming 9825 records after 16 ms and completed after 254 ms, displaying first 1000 rows.

(3) Đánh giá:

```

1 CALL gds.labelPropagation.stream('newChatGraph')
2 YIELD nodeId, communityId
3 WITH communityId, COUNT(nodeId) AS communitySize
4 RETURN
5   MAX(communitySize) AS maxCommunitySize,
6   MIN(communitySize) AS minCommunitySize,
7   AVG(communitySize) AS avgCommunitySize;
8

```

	maxCommunitySize	minCommunitySize	avgCommunitySize
1	17	1	1.0869375273044923

```

1 CALL gds.labelPropagation.stream('newChatGraph')
2 YIELD nodeId, communityId
3 WITH communityId, COUNT(nodeId) AS communitySize
4 RETURN COUNT(DISTINCT communityId) AS totalCommunities, SUM(communitySize) AS totalNodes;
5

```

	totalCommunities	totalNodes
1	9156	9952

=> Cộng đồng lớn nhất là 17 người, bé nhất là 1 người. Tuy nhiên, trung bình cộng đồng là 1.08 cho thấy phần lớn các cộng đồng rất ít người (tổng số người là 9952 , tổng số cộng đồng là 9156)

### 3.3.2. Phân tích mức độ ảnh hưởng (PageRank)

(1) Dự án hóa đồ thị:

```

neo4j$ CALL gds.graph.project('userGraph', ['User'], {InteractsWith: []})

```

	nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
1	<pre>         {           "User": {             "label": "User",             "properties": {               ...             }           }         }       </pre>	<pre>         {           "InteractsWith": {             "aggregation": "DEFAULT",             "orientation": "NATURAL",             "indexInverse": false,             "properties": {               ...             },             "type": "InteractsWith"           }         }       </pre>	"userGraph"	9698	4578	127

Started streaming 1 records after 8 ms and completed after 142 ms.

Tạo ra đồ thị gồm các User và quan hệ chính được sử dụng để đánh giá là InteractsWith, tất cả các thuộc tính được để ở mặc định.

(2) Chạy pageRank:

```

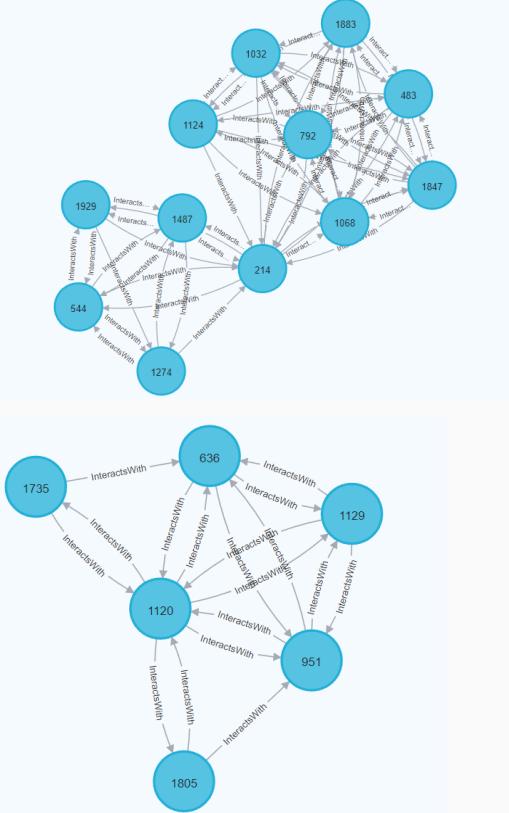
1 CALL gds.pageRank.stream('userGraph', {
2   maxIterations: 100,
3   dampingFactor: 0.85
4 })
5 YIELD nodeId, score
6 RETURN gds.util.asNode(nodeId).id AS userID, score
7 ORDER BY score DESC LIMIT 3

```

	userID	score
1	1120	1.8896714606986198
2	214	1.8165448568139224
3	735	1.7870382002932863

Started streaming 3 records after 11 ms and completed after 1434 ms.

- maxIterations: số lần lặp của thuật toán
- dampingFactor: tham số d trong công thức của thuật toán



Trực quan hóa tương tác của những node có điểm số cao nhất

(3) Kiểm tra điểm số của những người dùng tương tác với những người dùng có điểm trong top:

```

1 CALL gds.pageRank.stream('userGraph', {
2   |   maxIterations: 100
3 })
4 YIELD nodeId, score
5 WITH gds.util.asNode(nodeId) AS topUser, score
6 ORDER BY score DESC LIMIT 1
7 WITH topUser
8 MATCH (topUser)-[:InteractsWith]-(otherUser: User)
9 WITH DISTINCT otherUser, topUser
10 CALL gds.pageRank.stream('userGraph', {
11   |   maxIterations: 100
12 })

```

	topUser.id	otherUser.id	score
1	1120	636	1.7488867176137413
2	1120	951	1.5667301212667228
3	1120	1129	1.4106688525377586
4	1120	1735	0.47124413149340044
5	1120	1805	0.47124413149340044

Started streaming 5 records after 20 ms and completed after 1985 ms.

```

1 CALL gds.pageRank.stream('userGraph', {
2   maxIterations: 100
3 })
4 YIELD nodeId, score
5 WITH gds.util.asNode(nodeId) AS topUser, score
6 ORDER BY score DESC
7 SKIP 1
8 LIMIT 1
9 WITH topUser
10 MATCH (topUser)-[:InteractsWith]-(otherUser: User)
11 WITH DISTINCT otherUser, topUser
12 CALL gds.pageRank.stream('userGraph', {

```

Table	topUser.id	otherUser.id	score
1	214	1487	1.3517230271875131
2	214	792	1.2239398840244125
3	214	1068	1.2022433617683148
4	214	1847	1.2022433617683141
5	214	483	1.2022433617683141
6	214	1883	1.0934207372794935

(4) Tính trung bình điểm số của những người tương tác với người có điểm thuộc top:

```

1 CALL gds.pageRank.stream('userGraph', {
2   maxIterations: 100
3 })
4 YIELD nodeId, score
5 WITH gds.util.asNode(nodeId) AS topUser, score
6 ORDER BY score DESC
7 LIMIT 2
8 WITH topUser
9 MATCH (topUser)-[r:InteractsWith]-(otherUser: User)
10 WITH DISTINCT otherUser, topUser, r
11 CALL gds.pageRank.stream('userGraph', {
12   maxIterations: 100

```

Table	userId	avgPageRankScore	numInteractsWith
1	1120	1.1337547908810048	10
2	214	1.0734911329553853	18

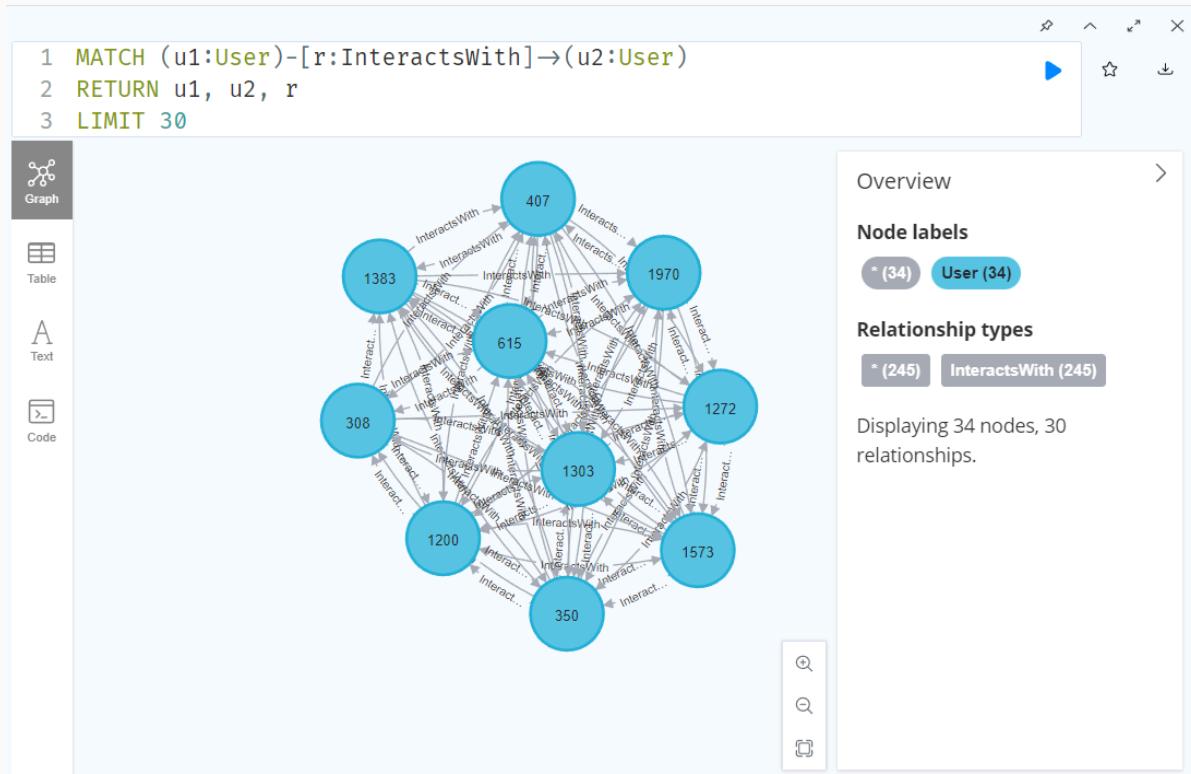
Started streaming 2 records after 25 ms and completed after 14410 ms.

(5) Đánh giá

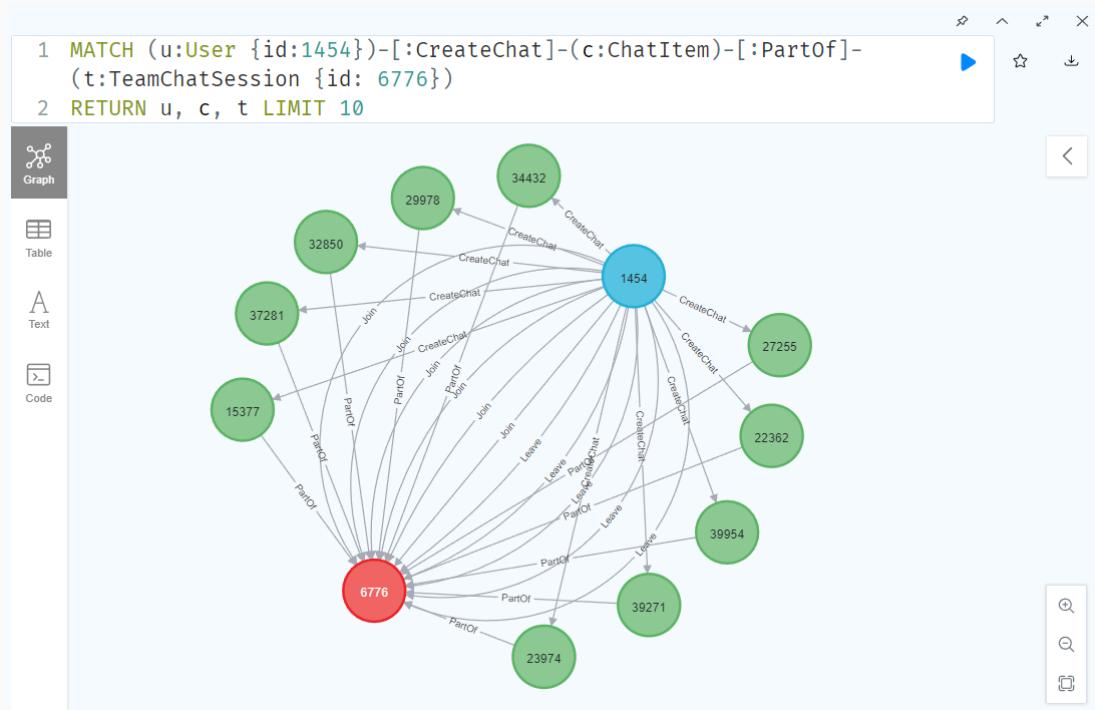
Những người dùng có điểm số cao thường là những người được tương tác (được nhắc đến, phản hồi) nhiều ở trong các phiên chat hoặc là người dùng tương tác với những người cũng có điểm số pageRank cao

## 3.4 Trực quan hóa.

### 3.4.1. Trực quan hóa mạng lưới người dùng và tương tác



### 3.4.2. Trực quan hóa tương tác trong nhóm



Trực quan hóa việc 1 User tương tác trong 1 phiên nhóm chat

# CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 4.1 Kết luận.

Sử dụng Neo4j để phân tích dữ liệu chat đã đạt được những kết quả khả quan, chứng minh được tiềm năng của cơ sở dữ liệu đồ thị trong việc khám phá mối quan hệ phức tạp giữa các cuộc trò chuyện. Và khi triển khai nhóm chúng em đã đạt được một số kết quả sau:

- Hiểu về các dạng cơ sở dữ liệu SQL, NoSQL và hiểu về Neo4j của NoSQL.
- Hiểu chi tiết về các thuật toán như: PageRank, Label Propagation Algorithm (LPA).
- Xây dựng thành công mô hình đồ thị cho dữ liệu chat.
- Triển khai truy vấn phân tích phân tích mối quan hệ và xu hướng giao tiếp.
- Trực quan hóa mạng lưới người dùng và tương tác nhóm.
- Chứng minh khả năng ứng dụng Neo4j trong phân tích dữ liệu phi cấu trúc.
- Đánh giá chương trình.

Tuy nhiên song song với kết quả đạt được vẫn còn một số hạn chế như:

- Việc phân tích dữ liệu vẫn còn ở mức nhở.
- Chương trình demo mới chỉ tập trung vào phân tích và trực quan hóa cơ bản chứ nhóm em chưa quét sâu vào từng nghiệp vụ phân tích chuyên sâu.

## 4.2 Hướng phát triển.

Dùng Neo4j để phân tích dữ liệu chat là một giải pháp đầy hứa hẹn mở ra nhiều khả năng nghiên cứu và ứng dụng. Bằng cách liên tục cải tiến và mở rộng, chúng ta có thể xây dựng một hệ thống phân tích chuyên sâu, cung cấp insights có giá trị từ dữ liệu giao tiếp. Đánh giá sơ bộ thì Neo4j rất tiềm năng và để có thể có khả năng phát triển các hướng sau:

- Nâng cao khả năng phân tích:
  - Phân tích tình cảm và ngôn ngữ: Tích hợp thuật toán NLP để đánh giá cảm xúc trong từng tin nhắn, thêm nút và quan hệ phản ánh sắc thái cảm xúc.
  - Học máy trên đồ thị: Sử dụng thuật toán học máy như GNN(Graph Neural Networks) dự đoán xu hướng giao tiếp, nhóm người dùng.
- Mở rộng nguồn dữ liệu:
  - Đa nền tảng: Thu thập từ các nguồn như Facebook, Zalo... rồi chuẩn hóa cấu trúc để dễ dàng tích hợp.
  - Tích hợp dữ liệu phi văn bản: Thêm nút cho quan hệ hình ảnh, video, file được chia sẻ và phân tích mối quan hệ dựa trên đa phương tiện.
- Giao diện và trực quan hóa:
  - Dashboard tương tác: Phát triển giao diện trực quan hóa đồ thị, hỗ trợ lọc, tìm kiếm và khám phá mối quan hệ.
  - Báo cáo tự động: Sinh báo cáo định kỳ để cảnh báo xu hướng và mối quan hệ quan trọng.
- Cải tiến hiệu xuất và bảo mật:
  - Tối ưu hóa truy vấn: Áp dụng các kỹ thuật như index và partition.
  - Bảo mật dữ liệu: Triển khai cơ chế mã hóa và kiểm soát truy cập.

➤ Ứng dụng thực tiễn:

- Phân tích kinh doanh: Ứng dụng marketing và chăm sóc khách hàng để tìm nhóm khách hàng tiềm năng cũng như các kế hoạch cho team marketing lên kế hoạch chăm sóc và tìm kiếm khách hàng.
- Nghiên cứu xã hội: Phân tích mạng lưới quan hệ xã hội. Nghiên cứu các mẫu giao tiếp và lan truyền thông tin.

Trong quá trình làm bài tập lớn, nhóm em đã rất cố gắng tìm hiểu và tham khảo nhiều nguồn tài liệu liên quan. Tuy nhiên không tránh khỏi những thiếu sót, rất mong được sự đóng góp ý kiến của thầy và các bạn để báo cáo và kĩ năng của chúng em được hoàn thiện hơn.

#### Tài liệu tham khảo

- [1] <https://neo4j.com/docs/>
- [2] <https://vi.wikipedia.org/wiki/Neo4J>
- [3] <https://www.geeksforgeeks.org/neo4j-introduction/>
- [4] <https://appmaster.io/vi/blog/co-so-du-lieu-do-thi-neo4j>