

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ  
KHOA CÔNG NGHỆ THÔNG TIN**



**THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH  
HỌC KỲ I, NĂM HỌC 2025-2026  
THIẾT KẾ VÀ CÀI ĐẶT CƠ SỞ DỮ LIỆU NoSQL CHO HỆ THỐNG ĐÁNH  
GIÁ VÀ ĐỀ XUẤT SẢN PHẨM**

*Giảng viên hướng dẫn:*  
Th.S. GVC Phan Thị Phương Nam

*Sinh viên thực hiện:*  
Họ tên: Nguyễn Thị Như Quỳnh  
MSSV: 110123041  
Lớp: DA23TTA

***Vĩnh Long, tháng 12 năm 2025***

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ  
KHOA CÔNG NGHỆ THÔNG TIN**



**THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH  
HỌC KỲ I, NĂM HỌC 2025-2026  
THIẾT KẾ VÀ CÀI ĐẶT CƠ SỞ DỮ LIỆU NoSQL CHO HỆ THỐNG  
ĐÁNH GIÁ VÀ ĐỀ XUẤT SẢN PHẨM**

*Giảng viên hướng dẫn:*  
Th.S. GVC Phan Thị Phương Nam

*Sinh viên thực hiện:*  
Họ tên: Nguyễn Thị Như Quỳnh  
MSSV: 110123041  
Lớp: DA23TTA

*Vĩnh Long, tháng 12 năm 2025*

## NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

[illegible]

Vĩnh Long, ngày ..... tháng 01 năm 2026  
**Giảng viên hướng dẫn**

**Giảng viên hướng dẫn**

**Phan Thị Phương Nam**

[illegible]

Nguyễn Thị Như Quỳnh

## LỜI CẢM ƠN

Trước tiên, tôi xin được bày tỏ lòng tri ân sâu sắc đến cô Phan Thị Phương Nam – giảng viên hướng dẫn đồ án cơ sở ngành. Trong suốt quá trình thực hiện đề tài “Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống đánh giá và đề xuất sản phẩm”, cô đã luôn quan tâm, theo sát tiến độ, tận tình hướng dẫn và định hướng rõ ràng cho tôi trong từng giai đoạn nghiên cứu. Những nhận xét, góp ý mang tính chuyên môn của cô đã giúp tôi kịp thời điều chỉnh hướng triển khai, khắc phục khó khăn và từng bước hoàn thiện nội dung đồ án. Sự tận tâm, trách nhiệm và phong cách làm việc nghiêm túc của cô không chỉ giúp tôi hoàn thành đồ án đúng thời hạn mà còn giúp tôi rèn luyện tư duy phân tích, kỹ năng nghiên cứu và vận dụng kiến thức vào thực tiễn.

Tôi xin chân thành cảm ơn Trường Đại học Trà Vinh đã tạo điều kiện thuận lợi về cơ sở vật chất, trang thiết bị học tập cũng như môi trường học thuật để tôi có thể học tập, nghiên cứu và hoàn thành đồ án cơ sở ngành một cách hiệu quả.

Đồng thời, tôi xin gửi lời cảm ơn đến Khoa Công nghệ Thông tin cùng toàn thể quý thầy cô trong khoa đã tận tình giảng dạy, truyền đạt những kiến thức nền tảng và chuyên môn trong suốt quá trình học tập. Những kiến thức này là hành trang quan trọng giúp tôi thực hiện tốt việc phân tích yêu cầu, thiết kế và cài đặt cơ sở dữ liệu NoSQL cho đề tài.

Mặc dù đã nỗ lực hoàn thành đồ án, nhưng do thời gian thực hiện còn hạn chế và kinh nghiệm thực tế chưa nhiều, nội dung đồ án khó tránh khỏi những thiếu sót. Tôi rất mong nhận được sự đóng góp ý kiến và chỉ dẫn thêm từ quý thầy cô để đồ án được hoàn thiện hơn và có thể tiếp tục phát triển trong thời gian tới.

Cuối cùng, tôi xin kính chúc cô Phan Thị Phương Nam, quý thầy cô cùng toàn thể cán bộ, giảng viên Trường Đại học Trà Vinh luôn mạnh khỏe, gặt hái nhiều thành công trong công tác giảng dạy và nghiên cứu khoa học.

Tôi xin chân thành cảm ơn!

## MỤC LỤC

<b>TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ .....</b>	<b>1</b>
<b>KHOA CÔNG NGHỆ THÔNG TIN .....</b>	<b>1</b>
<b>THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH .....</b>	<b>1</b>
<b>TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ .....</b>	<b>1</b>
<b>KHOA CÔNG NGHỆ THÔNG TIN .....</b>	<b>1</b>
<b>THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH .....</b>	<b>1</b>
<b>TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH .....</b>	<b>10</b>
<b>MỞ ĐẦU .....</b>	<b>12</b>
<b>2. Mục đích nghiên cứu .....</b>	<b>12</b>
<b>CHƯƠNG 1: TỔNG QUAN .....</b>	<b>14</b>
<b>1.1. Tổng quan về đề tài .....</b>	<b>14</b>
<b>1.2. Hướng giải quyết .....</b>	<b>15</b>
<b>CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT .....</b>	<b>16</b>
<b>2.1. Giới thiệu về NoSQL .....</b>	<b>16</b>
2.1.1. Khái niệm .....	16
2.1.2. Đặc điểm của NoSQL .....	16
2.1.3. Các mô hình dữ liệu trong cơ sở dữ liệu NoSQL .....	16
2.1.4. Các khái niệm trong cơ sở dữ liệu NoSQL .....	18
2.1.5. Một số ưu điểm và nhược điểm của cơ sở dữ liệu NoSql .....	19
<b>2.2. Giới thiệu về MongoDB .....</b>	<b>20</b>
2.2.1. Tổng quan về MongoDB .....	20
2.2.2. Một số khái niệm cơ bản trong MongoDB .....	21
2.2.3. Các kiểu dữ liệu trong MongoDB [4], [5] .....	22
2.2.4. Các thao tác dữ liệu trên MongoDB .....	23
2.2.4.1. Tạo cơ sở dữ liệu [4] .....	23
2.2.4.2. Kiểm tra cơ sở dữ liệu hiện hành [4] .....	23

---

2.2.4.3. Kiểm tra danh sách các cơ sở dữ liệu .....	24
2.2.4.4. Xóa cơ sở dữ liệu [4] .....	24
2.2.4.5. Tạo Collection [4] .....	25
2.2.4.6. Xóa Collection [4] .....	26
2.2.4.7. Thêm mới Document [4] .....	26
2.2.4.8. Truy vấn Document [4] .....	27
2.2.4.9. Cập nhật Document [4] .....	27
2.2.4.10. Xóa Document trong MongoDB [4] .....	27
2.2.4.11. Phục hồi dữ liệu trong MongoDB [5] .....	27
2.2.5. Cơ chế Index và Aggregation .....	28
<b>CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU .....</b>	<b>29</b>
<b>3.1. Cài đặt và chuẩn bị môi trường .....</b>	<b>29</b>
3.1.1. Hạ tầng phần mềm .....	29
3.1.2. Công cụ quản trị và thao tác .....	29
<b>3.2. Khởi tạo và cấu hình cơ sở dữ liệu .....</b>	<b>30</b>
<b>3.3. Hiện thực hóa cấu trúc dữ liệu .....</b>	<b>33</b>
3.3.1. Chiến lược thiết kế .....	33
3.3.2. Chi tiết cấu trúc Document của các Collection .....	33
<b>CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU .....</b>	<b>37</b>
<b>4.1. Hiện thực hóa các nghiệp vụ .....</b>	<b>37</b>
4.1.1. Nghiệp vụ cập nhật dữ liệu .....	37
4.1.1.1. Cập nhật hồ sơ cá nhân .....	37
4.1.1.2. Cập nhật hàng loạt .....	37
4.1.2. Nghiệp vụ tra cứu và tìm kiếm .....	38
4.1.2.1. Tra cứu chi tiết .....	38
4.1.2.2. Tra cứu theo tiêu chí .....	38
4.1.2.3. Tra cứu nâng cao .....	39
4.1.3. Nghiệp vụ thống kê và báo cáo .....	40
4.1.4. Nghiệp vụ: xóa hồ sơ .....	42
<b>4.2. Kết quả đạt được về mặt chức năng .....</b>	<b>43</b>

---

---

<b>4.3. Đánh giá hiệu năng hệ thống .....</b>	<b>43</b>
<b>4.4. Kết quả các giao diện chức năng trên MongoDB Compass .....</b>	<b>44</b>
<b>4.5. Đánh giá và Phân tích kết quả .....</b>	<b>44</b>
4.5.1. Về hiệu năng và tốc độ truy xuất .....	44
4.5.2. Về tính linh hoạt và khả năng mở rộng .....	45
4.5.3. Trải nghiệm trong quá trình thực hiện .....	45
<b>CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....</b>	<b>46</b>
<b>5.1. Kết quả đạt được .....</b>	<b>46</b>
<b>5.2. Hướng phát triển .....</b>	<b>46</b>
<b>DANH MỤC TÀI LIỆU THAM KHẢO .....</b>	<b>47</b>



---

## **DANH MỤC BẢNG BIỂU**

Bảng 2.1: Tham số options của Collection .....	25
Bảng 2.2. Danh sách các tùy chọn options của Collection .....	25

---

## DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Tên đầy đủ
ACID	Atomicity, Consistency, Isolation, Durability
API	Application Programming Interface
BASE	Basically Available, Soft state, Eventual consistency
JSON	JavaScript Object Notation
JWT	JSON Web Token
MQL	MongoDB Query Language
NoSQL	Not only SQL
RDBMS	Relational Database Management System
SQL	Structured Query Language
UTF-8	Unicode Transformation Format - 8-bit
XML	eXtensible Markup Language

## TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

### Vấn đề nghiên cứu

Đề tài “Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống đánh giá và đề xuất sản phẩm” tập trung nghiên cứu bài toán lưu trữ, quản lý và khai thác dữ liệu đánh giá của người dùng trong các hệ thống thương mại điện tử. Với đặc điểm dữ liệu lớn, đa dạng và thay đổi liên tục như thông tin sản phẩm, đánh giá sao, nhận xét, lịch sử tương tác của người dùng, các hệ quản trị cơ sở dữ liệu quan hệ truyền thống gặp nhiều hạn chế về khả năng mở rộng và tính linh hoạt trong thiết kế dữ liệu.

### Hướng tiếp cận

Đề tài tiếp cận theo hướng sử dụng cơ sở dữ liệu NoSQL, cụ thể là mô hình Document. Quá trình nghiên cứu bao gồm:

- Xác định các thực thể chính trong hệ thống đánh giá và đề xuất sản phẩm như: Người dùng (Users), Sản phẩm (Products), Đánh giá (Reviews), Tương tác (Interactions) và Đề xuất (Recommendations).
- Phân tích mối quan hệ giữa các thực thể: người dùng đánh giá sản phẩm, sản phẩm nhận phản hồi từ nhiều người, và người dùng có các hành vi tương tác khác nhau (xem, mua, thích, bình luận).
- Thiết kế mô hình dữ liệu NoSQL trên MongoDB, định nghĩa cấu trúc của từng collection và document tương ứng sao cho đảm bảo tính linh hoạt, hiệu quả và dễ mở rộng.
- Đề xuất chiến lược xây dựng chỉ mục cho các trường quan trọng và tối ưu các truy vấn thường xuyên sử dụng như lọc sản phẩm theo danh mục, tìm kiếm sản phẩm tương tự, hoặc hiển thị danh sách đánh giá mới nhất.

### Cách giải quyết

Để giải quyết vấn đề này, Giải pháp được đề xuất tập trung vào việc xây dựng các collection và document với cấu trúc linh hoạt, áp dụng kỹ thuật nhúng

(embedding) và tham chiếu (referencing) để tối ưu hiệu năng lưu trữ và truy vấn. Trên cơ sở đó, hệ cơ sở dữ liệu được cài đặt bằng MongoDB và một số truy vấn cơ bản được xây dựng nhằm phục vụ chức năng quản lý đánh giá và hỗ trợ đề xuất sản phẩm.

#### Kết quả đạt được

- Mô hình cơ sở dữ liệu của đề tài đã được thiết kế và cài đặt trên MongoDB.
- Đáp ứng được các yêu cầu chính về lưu trữ, truy xuất và quản lý dữ liệu trong hệ thống đánh giá và đề xuất sản phẩm.
- Dữ liệu mẫu thử mô phỏng hoạt động thực tế của người dùng và sản phẩm. Bộ dữ liệu bao gồm thông tin sản phẩm, người dùng, các đánh giá và hành vi tương tác.

## MỞ ĐẦU

### 1. Lý do chọn đề tài

Trong bối cảnh thương mại điện tử phát triển mạnh mẽ, các hệ thống bán hàng trực tuyến ngày càng phải xử lý khối lượng dữ liệu lớn, đa dạng và liên tục thay đổi, đặc biệt là dữ liệu liên quan đến đánh giá, nhận xét và hành vi người dùng. Các hệ thống cơ sở dữ liệu quan hệ truyền thống (SQL) bộc lộ nhiều hạn chế trong việc mở rộng, linh hoạt cấu trúc dữ liệu và xử lý dữ liệu phi cấu trúc.

Cơ sở dữ liệu NoSQL ra đời như một giải pháp phù hợp cho các hệ thống hiện đại nhờ khả năng mở rộng linh hoạt, hiệu năng cao và hỗ trợ dữ liệu bán cấu trúc. Đặc biệt, trong hệ thống đánh giá và đề xuất sản phẩm, dữ liệu thường có cấu trúc phức tạp như đánh giá sao, bình luận, lịch sử tương tác, hành vi người dùng,... rất phù hợp với mô hình NoSQL.

Vì vậy, việc nghiên cứu và thực hiện thiết kế, cài đặt cơ sở dữ liệu NoSQL cho hệ thống đánh giá và đề xuất sản phẩm là cần thiết, mang tính thực tiễn cao, giúp sinh viên tiếp cận công nghệ mới và ứng dụng vào các bài toán thực tế.

### 2. Mục đích nghiên cứu

- Nghiên cứu tổng quan về cơ sở dữ liệu NoSQL và các mô hình dữ liệu phổ biến.
- Phân tích yêu cầu dữ liệu của hệ thống đánh giá và đề xuất sản phẩm.
- Thiết kế mô hình cơ sở dữ liệu NoSQL phù hợp với bài toán thực tế.
- Cài đặt và xây dựng các collection/document để lưu trữ dữ liệu người dùng, sản phẩm, đánh giá và đề xuất.
- Thực hiện các thao tác truy vấn cơ bản và nâng cao phục vụ chức năng đánh giá và gợi ý sản phẩm.

- Đánh giá ưu điểm của việc sử dụng NoSQL trong hệ thống đề xuất so với mô hình dữ liệu truyền thống.

### **3. Đối tượng nghiên cứu**

- Cơ sở dữ liệu NoSQL.
- Hệ quản trị cơ sở dữ liệu MongoDB.
- Mô hình đánh giá và đề xuất sản phẩm

### **4. Phạm vi đề tài**

- Phạm vi nội dung: Thiết kế lược đồ cơ sở dữ liệu nhập dữ liệu mẫu thử và truy vấn thử nghiệm
- Phạm vi không gian: Hệ thống được cài đặt và chạy thử trên môi trường máy tính cá nhân.

## CHƯƠNG 1: TỔNG QUAN

### 1.1. Tổng quan về đề tài

Đề tài “Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống đánh giá và đề xuất sản phẩm” nhằm nghiên cứu và phát triển một mô hình cơ sở dữ liệu có khả năng lưu trữ và xử lý hiệu quả lượng dữ liệu khổng lồ phát sinh từ các hoạt động đánh giá và tương tác của người dùng trên các nền tảng thương mại điện tử. Trong bối cảnh người tiêu dùng ngày càng dựa vào các hệ thống đánh giá và gợi ý để ra quyết định mua hàng, dữ liệu người dùng và sản phẩm thay đổi liên tục với tốc độ lớn, bao gồm đánh giá, nhận xét, hành vi tương tác và lịch sử mua hàng. Do đó, việc sử dụng cơ sở dữ liệu NoSQL sẽ giúp hệ thống đảm bảo hiệu năng cao trong truy vấn và cập nhật dữ liệu thời gian thực, đồng thời vẫn duy trì khả năng lưu trữ dữ liệu đa dạng và phi cấu trúc. Việc xây dựng một cơ sở dữ liệu linh hoạt, tốc độ cao và dễ mở rộng lược đồ cơ sở dữ liệu trở nên vô cùng cần thiết.

Đề tài lựa chọn hướng tiếp cận dựa trên NoSQL, cụ thể là thực hiện cài đặt dữ liệu trên hệ quản trị cơ sở dữ liệu MongoDB, với cấu trúc dữ liệu dạng tài liệu (document-oriented), MongoDB cho phép mô hình hóa trực quan các đối tượng như người dùng, sản phẩm và đánh giá, giúp việc thao tác và mở rộng hệ thống trở nên dễ dàng hơn. Thay vì bị giới hạn bởi các bảng và ràng buộc cố định như trong mô hình quan hệ, NoSQL cung cấp mô hình dữ liệu linh hoạt có khả năng mở rộng cấu trúc lược đồ mà không ảnh hưởng đến hiệu năng toàn hệ thống.

Trong quá trình nghiên cứu, đề tài sẽ tiến hành phân tích đặc trưng của hệ thống đánh giá và đề xuất sản phẩm, xác định các collection chính như người dùng, sản phẩm, đánh giá, bình luận và các hành vi tương tác. Từ đó, tiến hành thiết kế mô hình dữ liệu NoSQL phù hợp, đảm bảo khả năng lưu trữ linh hoạt, xử lý nhanh và dễ dàng mở rộng lược đồ cơ sở dữ liệu khi quy mô dữ liệu tăng.

Đề tài sẽ xây dựng bộ dữ liệu mẫu thử và thực hiện các truy vấn minh họa nhằm kiểm tra tính chính xác và hiệu quả của hệ thống, chẳng hạn như hiển thị danh sách đánh giá của một sản phẩm, tính điểm đánh giá trung bình, tìm kiếm sản phẩm theo từ khóa hoặc đề xuất danh sách sản phẩm tương tự dựa trên hành vi người

dùng. Qua đó, đánh giá hiệu năng, độ ổn định và khả năng mở rộng của cơ sở dữ liệu NoSQL khi ứng dụng vào các hệ thống đánh giá và gợi ý sản phẩm thực tế.

## 1.2. Hướng giải quyết

Trước hết, tiến hành phân tích yêu cầu hệ thống nhằm xác định các loại dữ liệu cần lưu trữ, bao gồm thông tin người dùng, thông tin sản phẩm, dữ liệu đánh giá (số sao, nhận xét), lịch sử tương tác và dữ liệu phục vụ đề xuất sản phẩm. Việc phân tích yêu cầu giúp làm rõ mối quan hệ giữa các dữ liệu và các chức năng cần hỗ trợ.

Tiếp theo, đề tài lựa chọn cơ sở dữ liệu NoSQL, cụ thể là mô hình Document-oriented, để thiết kế hệ thống dữ liệu. Trên cơ sở đó, xây dựng mô hình dữ liệu với các collection và document phù hợp, đảm bảo tính linh hoạt trong cấu trúc và khả năng mở rộng khi dữ liệu tăng lên.

Trong quá trình thiết kế, áp dụng các kỹ thuật nhúng (embedding) và tham chiếu (referencing) một cách hợp lý nhằm tối ưu hiệu năng lưu trữ và truy vấn dữ liệu, đặc biệt đối với các dữ liệu thường xuyên được truy xuất như đánh giá sản phẩm và thông tin người dùng.

Sau khi hoàn thiện mô hình thiết kế, tiến hành cài đặt cơ sở dữ liệu bằng MongoDB, tạo các collection, định nghĩa cấu trúc document và xây dựng dữ liệu mẫu phục vụ cho quá trình kiểm thử. Đồng thời, xây dựng một số truy vấn cơ bản nhằm hỗ trợ chức năng quản lý đánh giá và đề xuất sản phẩm.

Cuối cùng, thực hiện đánh giá kết quả thông qua việc kiểm tra khả năng lưu trữ, truy xuất và của hệ thống cơ sở dữ liệu, từ đó rút ra nhận xét về tính phù hợp của NoSQL đối với bài toán đánh giá và đề xuất sản phẩm.



## CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

### 2.1. Giới thiệu về NoSQL

#### 2.1.1. Khái niệm

NoSQL (Not Only SQL) là loại cơ sở dữ liệu phi quan hệ, được thiết kế nhằm giải quyết các hạn chế của cơ sở dữ liệu quan hệ truyền thống trong các hệ thống có quy mô lớn, dữ liệu phi cấu trúc hoặc bán cấu trúc, yêu cầu hiệu năng cao và khả năng mở rộng linh hoạt.

NoSQL cho phép lưu trữ dữ liệu dưới nhiều dạng khác nhau như Document, Key-Value, Column-Family, Graph, phù hợp với các ứng dụng web hiện đại như thương mại điện tử, mạng xã hội, hệ thống gợi ý.

#### 2.1.2. Đặc điểm của NoSQL

- Không yêu cầu lược đồ cố định;
- Khả năng mở rộng;
- Hiệu năng truy vấn cao với dữ liệu lớn;
- Dễ dàng lưu trữ dữ liệu;
- Hỗ trợ phân tán và sao lưu dữ liệu.

#### 2.1.3. Các mô hình dữ liệu trong cơ sở dữ liệu NoSQL

##### 1) Cơ sở dữ liệu Key – Value

- Cơ sở dữ liệu Key-Value Stores có các đặc tính như sau: có một tập các đối tượng dữ liệu nhận dạng, các khóa; đối với mỗi khóa, có chính xác một đối tượng dữ liệu mô tả có liên quan, giá trị cho khóa đó; chỉ định một khóa cho phép truy vấn giá trị liên quan trong cơ sở dữ liệu.

Để truy vấn dữ liệu trong CSDL, ta dựa vào key để lấy dữ liệu ra. Các CSDL dạng này có tốc độ truy vấn rất nhanh.

- Database tiêu biểu: Riak, Redis, MemCache, Project Voldemort, CouchBase
- Ứng dụng: Do tốc độ truy xuất nhanh, key-value stores thường được dùng để làm cache cho ứng dụng. Ngoài ra, nó còn được dùng để lưu thông tin trong sessions, profiles/preferences của user.

## 2) Document Store

- Cơ sở dữ liệu Document Store lưu trữ dữ liệu dưới dạng tài liệu (document), thường được lưu trữ dưới dạng JSON hoặc BSON.

Mỗi document bao gồm các cặp *key-value* và có cấu trúc linh hoạt, không yêu cầu schema cố định. Các document có thể chứa dữ liệu lồng nhau (nested) và mảng (array).

Việc truy vấn dữ liệu trong Document Store được thực hiện thông qua các thuộc tính của document, cho phép tìm kiếm, lọc và sắp xếp dữ liệu linh hoạt hơn so với Key-Value Store.

Cơ sở dữ liệu dạng này phù hợp với các ứng dụng có dữ liệu thường xuyên thay đổi.

Database tiêu biểu: MongoDB, CouchDB, Firebase Firestore

### ◆ Ứng dụng

- Document Store thường được sử dụng trong các ứng dụng web, thương mại điện tử, hệ thống quản lý nội dung, đặc biệt phù hợp với các hệ thống đánh giá và đề xuất sản phẩm do khả năng lưu trữ đánh giá, bình luận dưới dạng mảng trong một document.

## 3) Column-Family Store

- Cơ sở dữ liệu Column-Family Store lưu trữ dữ liệu theo cột thay vì theo hàng như trong cơ sở dữ liệu quan hệ.

Các cột được nhóm thành các Column Family, cho phép mỗi bản ghi có thể có số lượng cột khác nhau.

Truy vấn dữ liệu được tối ưu theo cột, giúp tăng hiệu năng khi xử lý dữ liệu lớn và các phép phân tích.

CSDL dạng này có khả năng mở rộng và phân tán cao.

Database tiêu biểu: Apache Cassandra, HBase, ScyllaDB

◆ Ứng dụng

- Column-Family Store thường được sử dụng trong các hệ thống phân tích dữ liệu lớn, hệ thống ghi log, dữ liệu thời gian thực và các ứng dụng cần xử lý khối lượng dữ liệu rất lớn.

4) Graph Database

- Cơ sở dữ liệu Graph lưu trữ dữ liệu dưới dạng đồ thị, bao gồm:

Đỉnh: đại diện cho thực thể

Cạnh: đại diện cho mối quan hệ

Thuộc tính: thông tin mô tả

Truy vấn trong Graph Database tập trung vào các mối quan hệ giữa các đối tượng, cho phép xử lý các truy vấn liên kết phức tạp với hiệu năng cao.

CSDL dạng này đặc biệt hiệu quả khi dữ liệu có nhiều mối quan hệ chặt chẽ.

Database tiêu biểu: Amazon Neptune, ArangoDB

◆ Ứng dụng:

- Graph Database thường được sử dụng trong mạng xã hội, hệ thống gợi ý, phân tích quan hệ người dùng – sản phẩm, và phát hiện gian lận.

**2.1.4. Các khái niệm trong cơ sở dữ liệu NoSQL**

- NoSQL: Là cơ sở dữ liệu phi quan hệ, dùng để lưu trữ và xử lý dữ liệu lớn, dữ liệu phi cấu trúc hoặc bán cấu trúc.

- Schema-less: Không yêu cầu cấu trúc dữ liệu cố định, cho phép thay đổi dữ liệu linh hoạt.

- Document: Đơn vị lưu trữ dữ liệu, thường ở dạng JSON/BSON, gồm các cặp key–value.

- Collection: Tập hợp các document, tương tự bảng trong CSDL quan hệ.
- Key–Value: Mô hình lưu trữ dữ liệu theo cặp khóa – giá trị, truy xuất rất nhanh.
- Replication: Sao chép dữ liệu để đảm bảo an toàn và tính sẵn sàng.
- Sharding: Chia nhỏ dữ liệu và phân tán trên nhiều máy chủ để tăng hiệu năng.
- Định lý CAP: Trong hệ phân tán, chỉ đảm bảo tối đa 2 trong 3 yếu tố: nhất quán, sẵn sàng và chịu phân vùng.
- BASE: Mô hình cho phép dữ liệu nhất quán sau một khoảng thời gian, phù hợp với NoSQL.

#### **2.1.5. Một số ưu điểm và nhược điểm của cơ sở dữ liệu NoSql**

Ưu điểm:

- Tính linh hoạt cao: NoSQL không yêu cầu schema cố định, cho phép thay đổi cấu trúc dữ liệu dễ dàng khi hệ thống phát triển.
- Khả năng mở rộng tốt: Hỗ trợ mở rộng bằng cách thêm máy chủ mới, phù hợp với hệ thống có lượng người dùng lớn.
- Hiệu năng truy vấn cao: Tối ưu cho các truy vấn đơn giản và truy xuất dữ liệu theo key, giúp hệ thống phản hồi nhanh
- Xử lý dữ liệu lớn hiệu quả: Phù hợp với Big Data và dữ liệu phi cấu trúc như văn bản, hình ảnh, log.
- Hỗ trợ phân tán: Cho phép lưu trữ dữ liệu trên nhiều máy chủ, tăng tính sẵn sàng và độ tin cậy.
- Phù hợp với ứng dụng hiện đại: Thích hợp cho web, mobile, mạng xã hội và thương mại điện tử.

Nhược điểm:

- Không đảm bảo ACID đầy đủ: NoSQL thường không hỗ trợ đầy đủ các tính chất ACID như cơ sở dữ liệu quan hệ.

- Tính nhất quán dữ liệu không cao: Dữ liệu có thể không đồng bộ ngay lập tức giữa các nút trong hệ thống.

- Hạn chế truy vấn phức tạp: Không hỗ trợ hoặc hỗ trợ hạn chế các phép JOIN và truy vấn nhiều bảng.

- Khó thiết kế ban đầu: Cần thiết kế cấu trúc dữ liệu cẩn thận để tránh dư thừa và khó mở rộng về sau.

- Khó chuẩn hóa dữ liệu: Dữ liệu dễ bị trùng lặp do cơ chế nhúng dữ liệu.

- Phụ thuộc vào từng hệ quản trị: Mỗi hệ NoSQL có cú pháp và cách quản lý khác nhau, gây khó khăn khi chuyển đổi.

## **2.2. Giới thiệu về MongoDB**

### **2.2.1. Tổng quan về MongoDB**

MongoDB là một hệ quản trị cơ sở dữ liệu NoSQL dạng Document, được phát triển bởi MongoDB Inc. Dữ liệu trong MongoDB được lưu trữ dưới dạng document theo định dạng BSON, cho phép biểu diễn dữ liệu linh hoạt và có cấu trúc lồng nhau.

Khác với cơ sở dữ liệu quan hệ truyền thống, MongoDB không yêu cầu lược đồ cố định, giúp việc thay đổi và mở rộng cấu trúc dữ liệu trở nên dễ dàng. Các document được tổ chức thành collection, và nhiều collection tạo thành một database.

MongoDB hỗ trợ các cơ chế quan trọng như indexing để tăng tốc độ truy vấn, replication nhằm đảm bảo an toàn dữ liệu và sharding để phân tán dữ liệu, giúp hệ thống hoạt động ổn định với khối lượng dữ liệu lớn.

Nhờ hiệu năng cao, khả năng mở rộng tốt và tính linh hoạt, MongoDB được sử dụng rộng rãi trong các ứng dụng web hiện đại, thương mại điện tử, mạng xã hội và đặc biệt phù hợp với các hệ thống đánh giá và đề xuất sản phẩm.

### **2.2.2. Một số khái niệm cơ bản trong MongoDB**

- Database: Là tập hợp các collection, dùng để tổ chức và quản lý dữ liệu.
- Collection: Là tập hợp các document có cùng mục đích lưu trữ, tương tự bảng trong cơ sở dữ liệu quan hệ nhưng không có schema cố định.
- Document: Là đơn vị dữ liệu cơ bản trong MongoDB, lưu trữ dưới dạng BSON. Một document gồm các cặp key–value và có thể chứa dữ liệu lồng nhau hoặc mảng.
- Field: Là các thuộc tính trong document, dùng để mô tả thông tin của đối tượng.
- \_id: Là trường bắt buộc, đóng vai trò khóa chính, dùng để định danh duy nhất mỗi document trong collection.
- BSON: Là định dạng lưu trữ nhị phân của JSON, giúp MongoDB xử lý dữ liệu nhanh và hỗ trợ nhiều kiểu dữ liệu hơn.
- Schema-less: MongoDB không yêu cầu cấu trúc dữ liệu cố định, cho phép các document trong cùng một collection có cấu trúc khác nhau.
- Index: Là cơ chế đánh chỉ mục giúp tăng tốc độ truy vấn dữ liệu, đặc biệt với các trường được truy vấn thường xuyên.
- Query: Là thao tác truy vấn dữ liệu, cho phép tìm kiếm, lọc, sắp xếp và giới hạn kết quả.
- Embedding: Là cách lưu trữ dữ liệu liên quan trực tiếp trong cùng một document, giúp giảm số lần truy vấn.
- Reference: Là cách liên kết các document thông qua khóa, phù hợp với dữ liệu lớn hoặc dùng chung nhiều nơi.

- Aggregation: Là cơ chế xử lý và tổng hợp dữ liệu, hỗ trợ các phép toán như nhóm, thống kê và tính toán.

- Replication: Là cơ chế sao chép dữ liệu giữa các máy chủ, đảm bảo an toàn và tính sẵn sàng của hệ thống.

- Sharding: Là kỹ thuật phân mảnh dữ liệu và phân tán trên nhiều máy chủ để mở rộng hệ thống.

### **2.2.3. Các kiểu dữ liệu trong MongoDB [4], [5]**

- String: Dùng để lưu trữ văn bản. Đây là kiểu dữ liệu được sử dụng nhiều nhất trong MongoDB.

- Number: Int32: Số nguyên nhỏ, Int64 Số nguyên lớn, Double Số thực, Decimal128 Số thập phân có độ chính xác cao (thường dùng cho tiền)

- Boolean: Dùng để biểu diễn trạng thái logic.

- Date: Lưu trữ thời gian theo chuẩn ISODate.

- Object: Cho phép lưu dữ liệu dạng lồng nhau (document trong document).

- Array: Cho phép lưu nhiều giá trị hoặc nhiều document trong cùng một trường.

- ObjectId: Là kiểu dữ liệu đặc biệt dùng làm khóa chính \_id. ObjectId đảm bảo mỗi document là duy nhất và không trùng lặp.

- Null: Biểu thị trường dữ liệu chưa có giá trị hoặc không áp dụng.

- Binary Data: Dùng để lưu trữ dữ liệu dạng nhị phân như hình ảnh, file, âm thanh.

- Timestamp: Dùng để ghi lại thời điểm thay đổi dữ liệu, thường phục vụ cho replication.

---

- Embedded Document: Cho phép nhúng document con vào document cha, giúp giảm số lần truy vấn.

## 2.2.4. Các thao tác dữ liệu trên MongoDB

### 2.2.4.1. Tạo cơ sở dữ liệu [4]

Trong MongoDB, việc tạo cơ sở dữ liệu được thực hiện thông qua lệnh `use`. Cơ sở dữ liệu được xác định bằng tên do người dùng đặt. Khi thực hiện lệnh `use`, nếu cơ sở dữ liệu chưa tồn tại thì MongoDB sẽ tự động tạo mới; ngược lại, nếu cơ sở dữ liệu đã tồn tại thì hệ thống sẽ chuyển sang sử dụng cơ sở dữ liệu đó.

Cú pháp:

```
use <DATABASE_NAME>
```

Trong đó, <DATABASE\_NAME> phải tuân thủ các quy tắc đặt tên sau:

- + Chỉ bao gồm chữ cái và chữ số;

- + Có phân biệt chữ hoa và chữ thường;

Không được để trống;

- + Không được chứa các ký tự đặc biệt như: /, \, ., ", \*, ,, :, |, ?, \$, khoảng trắng hoặc ký tự null (\0);

- + Độ dài tên cơ sở dữ liệu không vượt quá 64 bytes.

### 2.2.4.2. Kiểm tra cơ sở dữ liệu hiện hành [4]

Để kiểm tra cơ sở dữ liệu đang được sử dụng trong MongoDB, người dùng có thể sử dụng lệnh `db`. Lệnh này cho phép hiển thị tên của cơ sở dữ liệu hiện tại mà hệ thống đang làm việc.

Cú pháp: `Db`



---

### 2.2.4.3. Kiểm tra danh sách các cơ sở dữ liệu

Để xem danh sách các cơ sở dữ liệu đang tồn tại trong MongoDB, người dùng sử dụng lệnh `show dbs`. Lệnh này hiển thị các cơ sở dữ liệu hiện có trong hệ thống.

**Cú pháp:**

```
show dbs
```

MongoDB cung cấp sẵn một cơ sở dữ liệu mặc định có tên là `test`. Trong trường hợp người dùng chưa tạo thêm cơ sở dữ liệu mới, các collection sẽ được lưu trữ trong cơ sở dữ liệu này.

Lưu ý rằng, một cơ sở dữ liệu mới được tạo nhưng chưa chứa bất kỳ collection nào sẽ không xuất hiện trong danh sách khi sử dụng lệnh `show dbs`. Cơ sở dữ liệu chỉ được hiển thị sau khi có ít nhất một collection được tạo và lưu dữ liệu.

### 2.2.4.4. Xóa cơ sở dữ liệu [4]

Trong MongoDB, để xóa một cơ sở dữ liệu đang tồn tại, người dùng sử dụng lệnh `db.dropDatabase()`. Lệnh này sẽ xóa toàn bộ cơ sở dữ liệu hiện hành mà người dùng đang chọn.

Trường hợp không chỉ định cơ sở dữ liệu cụ thể, MongoDB sẽ thực hiện xóa cơ sở dữ liệu mặc định `test`.

Trước khi tiến hành xóa, người dùng cần kiểm tra danh sách các cơ sở dữ liệu hiện có và chuyển sang cơ sở dữ liệu cần xóa.

Các bước thực hiện:

```
show dbs
use MyNoSQL
db.dropDatabase()
```

Sau khi thực hiện, MongoDB sẽ trả về thông tin xác nhận cơ sở dữ liệu đã được xóa, ví dụ:

```
{ "dropped" : "MyNoSQL", "ok" : 1 }
```

Cuối cùng, sử dụng lại lệnh `show dbs` để kiểm tra và xác nhận rằng cơ sở dữ liệu đã được xóa thành công.

#### 2.2.4.5. Tạo Collection [4]

Trong MongoDB, collection có thể được tạo bằng cách sử dụng phương thức `db.createCollection(name, options)`. Phương thức này cho phép người dùng tạo collection mới và tùy chọn cấu hình cho collection đó.

Cú pháp:

```
db.createCollection(name, options)
```

Trong đó:

`name`: là tên của collection cần tạo.

`options`: là một document (tùy chọn), dùng để xác định các cấu hình đặc biệt cho collection.

Bảng 2.1: Tham số options của Collection

Tham số	Kiểu	Miêu tả
Name	Chuỗi	Tên của Collection
Options	Document	(Tùy chọn) Xác định

Tham số options là tùy chọn, người dùng cần xác định tên của Collection. Dưới đây là danh sách các tùy chọn cho options người dùng có thể sử dụng:

Bảng 2.2. Danh sách các tùy chọn options của Collection

Trường	Kiểu	Miêu tả
capped	Boolean	(Tùy chọn) Nếu true, kích hoạt một Capped Collection. Đây là một Collection có kích thước cố định mà tự động ghi đè các entry cũ nhất khi nó tiếp cận kích cỡ tối đa. Nếu xác định là true, thì cần xác định tham số size>

Trường	Kiểu	Miêu tả
aouIndexID	Boolean	(Tùy chọn) Nếu true, tự động tạo chỉ mục trên các trường _id. Giá trị mặc định là false.
Size	Số	(Tùy chọn) Xác định kích thước tối đa (giá trị byte) cho một Capped Collection. Nếu tham số capped là true, thì cần xác định trường này.
Max	Số	(Tùy chọn) Xác định số Document tối đa được cho phép trong một Capped Colleciton.

Trong khi thực hiện việc chèn dữ liệu vào Document, đầu tiên MongoDB kiểm tra trường size của Capped Collection, sau đó nó kiểm tra trường max.

#### 2.2.4.6. Xóa Collection [4]

Trong MongoDB, để xóa một collection khỏi cơ sở dữ liệu, người dùng sử dụng phương thức `db.collection.drop()`. Phương thức này sẽ xóa hoàn toàn collection được chỉ định, bao gồm toàn bộ các document và chỉ mục liên quan.

**Cú pháp:**

```
db.<collection_name>.drop()
```

Sau khi thực hiện, MongoDB sẽ trả về kết quả cho biết thao tác xóa có thành công hay không. Nếu collection tồn tại và được xóa thành công, kết quả trả về là true; ngược lại, nếu collection không tồn tại, kết quả sẽ là false.

#### 2.2.4.7. Thêm mới Document [4]

Phương thức `insert()` thường được sử dụng phổ biến để thêm document mới vào collection. Khi thực hiện lệnh này, nếu document chưa có trường `_id`, MongoDB sẽ tự động tạo một giá trị `_id` duy nhất cho document.

**Cú pháp cơ bản:**

```
db.<COLLECTION_NAME>.insert(document)
```

Trong đó:

---

`COLLECTION_NAME` là tên collection cần thêm dữ liệu.

document là dữ liệu cần chèn, được biểu diễn dưới dạng một document BSON.

#### 2.2.4.8. Truy vấn Document [4]

Truy vấn document là thao tác dùng để lấy dữ liệu từ các collection trong MongoDB. MongoDB cung cấp phương thức `find()` và `findOne()` để thực hiện việc truy vấn dữ liệu một cách linh hoạt và hiệu quả.

```
db<COLLECTION_NAME>.find(query, projection)
```

#### 2.2.4.9. Cập nhật Document [4]

Phương thức `update()` hoặc `save()` trong MongoDB được sử dụng để cập nhật Document vào trong một Collection. Phương thức `update()` cập nhật các giá trị trong Document đang tồn tại trong khi phương thức `save()` thay thế Document đang tồn tại với Document đã truyền trong phương thức `save()` đó.

Phương thức `update()` cập nhật các giá trị trong Document đang tồn tại.

Cú pháp cơ bản của phương thức `update()` là như sau:

```
>db.<collectionname>.update(SELECTIOIN_CRITERIA, UPDATED_DATA)
```

#### 2.2.4.10. Xóa Document trong MongoDB [4]

- Phương thức `remove()` được sử dụng để xóa Document từ Collection.
- Phương thức `remove()` nhận hai tham số. Tham số đầu tiên deletion criteria xác định Document để xóa, và tham số thứ hai là `justOne`.

1. deletion criteria: Xác định Document để xóa.

2. `justOne` : Nếu được thiết lập là `true` (hoặc 1), thì chỉ xóa một Document.

Cú pháp cơ bản của phương thức `remove()` là như sau:

```
>db.COLLECTION_NAME.remove(DELETION_CRITTERIA)
```

#### 2.2.4.11. Phục hồi dữ liệu trong MongoDB [5]

Để phục hồi dữ liệu đã sao lưu trong MongoDB, bạn sử dụng lệnh `mongorestore`. Lệnh này phục hồi tất cả dữ liệu từ thư mục sao lưu.

Cú pháp cơ bản của lệnh `mongorestore` là:

>mongorestore

### 2.2.5. Cơ chế Index và Aggregation

Cơ chế Index (Đánh chỉ mục)

Index là một cấu trúc dữ liệu đặc biệt giúp MongoDB truy xuất dữ liệu một cách hiệu quả. Nếu không có Index, MongoDB phải thực hiện quét toàn bộ bộ sưu tập (Collection Scan) để tìm các tài liệu phù hợp, điều này gây tốn tài nguyên và kéo dài thời gian xử lý khi tập dữ liệu lớn [2].

Mục đích: Giảm thiểu số lượng tài liệu cần phải kiểm tra khi thực hiện truy vấn, từ đó tối ưu hóa tốc độ tìm kiếm và các thao tác sắp xếp dữ liệu [5].

Cấu trúc: Theo mặc định, MongoDB tự động tạo một chỉ mục duy nhất trên trường `_id` cho mọi bộ sưu tập để định danh tài liệu [2]. Để nâng cao hiệu năng thực tế, người dùng có thể chủ động tạo thêm các chỉ mục trên các trường thường xuyên được sử dụng trong bộ lọc tra cứu.

Cơ chế Aggregation (Xử lý dữ liệu tập trung)

Aggregation là một khung xử lý (Framework) mạnh mẽ cho phép thực hiện các phép toán thống kê và biến đổi dữ liệu thông qua quy trình đường ống (Pipeline) [6].

Định nghĩa: Thay vì chỉ trả về các tài liệu thô, Aggregation tiến hành xử lý các nhóm dữ liệu để trả về kết quả đã được tính toán như tổng (`$sum`), trung bình (`$avg`), hay đếm (`$count`) [2].

Aggregation Pipeline: Đây là cơ chế cốt lõi trong xử lý dữ liệu của MongoDB, nơi dữ liệu đi qua một chuỗi các giai đoạn xử lý (Stages). Kết quả đầu ra của giai đoạn này sẽ trở thành đầu vào của giai đoạn kế tiếp [3]:

`$match`: Thực hiện lọc các tài liệu thỏa mãn điều kiện cụ thể, tương đương với mệnh đề `WHERE` trong SQL truyền thống.

`$group`: Gom nhóm các tài liệu theo một khóa xác định (ví dụ: mã khoa, mã lớp) để thực hiện các phép toán thống kê.

`$merge`: Đây là giai đoạn quan trọng giúp ghi kết quả thống kê vào một bộ sưu tập mới. Cơ chế này giúp các báo cáo luôn sẵn sàng để truy xuất mà không cần phải thực hiện tính toán lại từ đầu trên toàn bộ dữ liệu gốc [2].

## CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

### 3.1. Cài đặt và chuẩn bị môi trường

#### 3.1.1. Hạ tầng phần mềm

Hệ điều hành: Windows 11 64-bit.

Hệ quản trị cơ sở dữ liệu: MongoDB Community Server phiên bản 6.0. Phiên bản này hỗ trợ mạnh mẽ các toán tử truy vấn trên mảng và document nhúng.

#### 3.1.2. Công cụ quản trị và thao tác

MongoDB Compass: công cụ giao diện đồ họa được sử dụng để thiết kế dữ liệu, nhập liệu mẫu và chỉnh sửa document.

MongoDB Shell: công cụ dòng lệnh được sử dụng để khởi tạo database và collection, thực thi các câu lệnh truy vấn phức tạp.

Collection	Thuộc tính cấp 1	Thuộc tính cấp 2	Thuộc tính cấp 3
NGUOI_DUNG	ma_nguoi_dung		
	ten_nguoi_dung		
	email		
	mat_khau		
	so_dien_thoai		
	Ngaytao_tai_khoa n		
SAN_PHAM	ma_san_pham		
	ten_san_pham		
	ten_danh_muc		
	mo_ta_san_pham		
	gia_san_pham		
	nha_san_xuat		
	danh_gia	nguoi_dung	ma_nguoi_dung
			ten_nguoi_dung
		diem_danh_gia	
		nhan_xet	
		ngay_danh_gia	
	tuong_tac	nguoi_dung	ma_nguoi_dung

Collection	Thuộc tính cấp 1	Thuộc tính cấp 2	Thuộc tính cấp 3
			ten_nguoi_dung
		loai_tuong_tac	
		tg_tuong_tac	
DM_SAN_PHAM	ma_danh_muc		
	ten_danh_muc		
	ten_san_pham		
TUONG_TAC	ma_tuong_tac		
	nguoi_dung		
	loai_tuong_tac		
	san_pham		
	tg_tuong_tac		
GOI_Y_SAN_PHAM	ma_goi_y		
	san_pham		
	diem_goi_y		
	nguoi_dung	ma_nguoi_dung	
		ten_nguoi_dung	
	ds_san_pham	ma_san_pham	
		ten_san_pham	
		diem_goi_y	
DANH_GIA	id_danh_gia		
	id_san_pham		
	id_nguoi_dung		
	nhan_xet		
	diem_danh_gia		
	thoi_gian_danh_gia		

### 3.2. Khởi tạo và cấu hình cơ sở dữ liệu

Các lệnh dưới đây dùng để khởi tạo các collection cần thiết cho hệ thống ngay từ đầu, thay vì để MongoDB tự động sinh ra khi chèn dữ liệu.

Chọn database

```
use he_thong_san_pham;
```

◆ Tạo collection cho sản phẩm

```
db.createCollection("san_pham", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["ten_san_pham", "ten_danh_muc", "gia", "nha_san_xuat"],
      properties: {
        ten_san_pham: { bsonType: "string" },
        ten_danh_muc: { bsonType: "string" },
        mo_ta: { bsonType: "string" },
        gia: { bsonType: "double" },
        nha_san_xuat: { bsonType: "string" },
        danh_gia: {
          bsonType: "array",
          items: {
            bsonType: "object",
            required: ["nguoi_dung", "diem", "binh_luan"],
            properties: {
              nguoi_dung: { bsonType: "string" },
              diem: { bsonType: "double" },
              binh_luan: { bsonType: "string" }
            }
          }
        },
        ds_goi_y: {
          bsonType: "array",
          items: { bsonType: "object" }
        }
      }
    }
  }
})
```



```
}  
});
```

◆ Tạo collection cho người dùng

```
db.createCollection("nguoi_dung", {  
  validator: {  
    $jsonSchema: {  
      bsonType: "object",  
      required: ["ten", "email", "mat_khau"],  
      properties: {  
        ten: { bsonType: "string" },  
        email: { bsonType: "string" },  
        mat_khau: { bsonType: "string" },  
        so_dien_thoai: { bsonType: "string" },  
        ngay_tao_tai_khoan: { bsonType: "date" }  
      }  
    }  
  }  
});
```

◆ Tạo collection cho danh mục sản phẩm

```
db.createCollection("danh_muc", {  
  validator: {  
    $jsonSchema: {  
      bsonType: "object",  
      required: ["ten_danh_muc"],  
      properties: {  
        ten_danh_muc: { bsonType: "string" },  
        mo_ta: { bsonType: "string" }  
      }  
    }  
  }  
});
```

```
})
```

### 3.3. Hiện thực hóa cấu trúc dữ liệu

#### 3.3.1. Chiến lược thiết kế

Mục tiêu: Nâng cao hiệu suất cho các thao tác truy vấn dữ liệu, đặc biệt là các hệ thống có tần suất đọc cao (read-heavy), như tra cứu thông tin sản phẩm, đánh giá người dùng và gợi ý sản phẩm.

Giải pháp: Thay vì lưu trữ các khóa tham chiếu và thực hiện phép Join giữa nhiều Collection (ví dụ: sản phẩm, đánh giá, danh mục, nhà sản xuất), thông tin liên quan đến sản phẩm, đánh giá của người dùng và gợi ý sản phẩm liên quan được nhúng trực tiếp vào Document sản phẩm. Thiết kế này cho phép truy xuất đầy đủ hồ sơ sản phẩm chỉ với một lần truy vấn duy nhất, từ đó giảm chi phí xử lý và cải thiện tốc độ phản hồi của hệ thống đề xuất sản phẩm.

#### 3.3.2. Chi tiết cấu trúc Document của các Collection

Collection `nguoi_dung` dùng để lưu trữ thông tin tài khoản của người dùng trong hệ thống đánh giá và đề xuất sản phẩm. Mỗi document đại diện cho một người dùng duy nhất.

```
{
  _id: ObjectId,
  ten_nguoi_dung: String,
  email: String,
  mat_khau: String,
  so_dien_thoai: String,
  ngay_tao: Date
}
```

Collection `san_pham` dùng để lưu trữ thông tin cơ bản của các sản phẩm trong hệ thống đánh giá và đề xuất.

```
{
  _id: ObjectId,
  ten_danh_muc: String,
```

```
ten_san_pham: String
}
```

Collection `tuong_tac` dùng để lưu trữ lịch sử tương tác của người dùng với sản phẩm trong hệ thống đánh giá và đề xuất sản phẩm. Mỗi document đại diện cho một hành vi tương tác cụ thể tại một thời điểm xác định.

```
{
  _id: ObjectId,
  nguoi_dung: String,
  loai_tuong_tac: String,
  san_pham: String,
  thoi_gian: Date
}
```

Collection `san_pham` lưu trữ thông tin chi tiết của sản phẩm, đồng thời nhúng trực tiếp dữ liệu đánh giá và tương tác của người dùng nhằm tối ưu hiệu suất truy vấn cho các hệ thống có tần suất đọc cao.

```
{
  _id: ObjectId,
  ten_san_pham: String,
  ten_danh_muc: String,
  mo_ta: String,
  gia: Number,
  nha_san_xuat: String,

  danh_gia: [
    {
      nguoi_dung: {
        id: ObjectId,
        ten: String
      },
      diem_danh_gia: Number,
      nhan_xet: String,
    }
  ]
}
```

```
    ngay_danh_gia: Date
  }
],

  tuong_tac: [
    {
      nguoi_dung: {
        id: ObjectId,
        ten: String
      },
      loai_tuong_tac: String,
      thoi_gian: Date
    }
  ]
}
```

Collection `danh_gia` dùng để lưu trữ các đánh giá của người dùng đối với sản phẩm trong hệ thống. Mỗi document đại diện cho một lượt đánh giá tại một thời điểm cụ thể.

```
{
  _id: ObjectId,
  id_san_pham: String,
  id_nguoi_dung: String,
  nhan_xet: String,
  diem: Number,
  thoi_gian: Date
}
```

Collection `goi_y` dùng để lưu trữ kết quả đề xuất sản phẩm cá nhân hóa cho từng người dùng. Mỗi document thể hiện mức độ phù hợp của một sản phẩm đối với một người dùng cụ thể, đồng thời chứa danh sách các sản phẩm liên quan được đề xuất.

```
{
  _id: ObjectId,
  san_pham: String,
```

```
diem_goi_y: Number,  
  
nguoi_dung: {  
  id: ObjectId,  
  ten: String  
},  
  
ds_goi_y: [  
  {  
    id: ObjectId,  
    ten: String,  
    diem_goi_y: Number  
  }  
]
```

---

## CHƯƠNG 4:

### KẾT QUẢ NGHIÊN CỨU

#### 4.1. Hiện thực hóa các nghiệp vụ

##### 4.1.1. Nghiệp vụ cập nhật dữ liệu

##### 4.1.1.1. Cập nhật hồ sơ cá nhân

Nghiệp vụ cập nhật dữ liệu cho phép hệ thống thêm mới, chỉnh sửa và cập nhật thông tin liên quan đến người dùng, sản phẩm, đánh giá, tương tác và kết quả gợi ý. Các thao tác cập nhật được thực hiện trực tiếp trên các collection trong MongoDB thông qua các lệnh như insert(), updateOne(), updateMany().

Ví dụ: Người dùng Nguyễn Văn An cập nhật số điện thoại liên hệ mới trong hệ thống.

```
db.nguoi_dung.updateOne(
  { ten_nguoi_dung: "Nguyễn Văn An" },
  { $set: { so_dien_thoai: "0123456789" } })
```

Kết quả trả về từ MongoDB Shell

```
{
  acknowledged: true,
  matchedCount: 1,
  modifiedCount: 1
}
```

##### 4.1.1.2. Cập nhật hàng loạt

Cập nhật nhà sản xuất mới cho toàn bộ các sản phẩm thuộc danh mục Laptop trong hệ thống.

```
db.san_pham.updateMany(
  { ten_danh_muc: "Laptop" },
  { $set: { nha_san_xuat: "Lenovo" } })
```

Kết quả trả về từ MongoDB Shell

```
acknowledged: true,
matchedCount: <3>,
```

```
modifiedCount: <3>
```

#### 4.1.2. Nghiệp vụ tra cứu và tìm kiếm

##### 4.1.2.1. Tra cứu chi tiết

Xem chi tiết hồ sơ người dùng có mã người dùng 675f100000000000000000115 trong hệ thống.

```
db.users.findOne(  
  { _id: ObjectId("675f100000000000000000115") }  
)
```

Kết quả trả từ Shell:

```
{  
  _id: ObjectId('675f100000000000000000115'),  
  ten_nguoi_dung: 'Trịnh Hoàng Nam',  
  email: 'nam.trinh@gmail.com',  
  mat_khau: 'nam321',  
  so_dien_thoai: '0808111222',  
  ngay_tao: 2024-02-15T18:25:00.000Z
```

##### 4.1.2.2. Tra cứu theo tiêu chí

Lấy danh sách tất cả người dùng đã đăng ký tài khoản trong hệ thống.

```
db.users.find()
```

Kết quả trả về từ Shell

```
{  
  _id: ObjectId('675f100000000000000000111'),  
  ten_nguoi_dung: 'Vũ Quốc Bảo',  
  email: 'quocbao@gmail.com',  
  mat_khau: 'baopass',  
  so_dien_thoai: '0877555444',  
  ngay_tao: 2024-02-28T11:20:00.000Z  
}..
```

---

#### 4.1.2.3. Tra cứu nâng cao

Tìm các sản phẩm có tên chứa từ khóa “watch” để phục vụ cho chức năng tìm kiếm và đề xuất sản phẩm.

```
db.products.find(  
  { ten_san_pham: { $regex: "watch", $options: "i" } }  
)
```

Kết quả trả về từ Shell

```
{  
  _id: ObjectId('675f30000000000000000105'),  
  ten_san_pham: 'Apple Watch S8',  
  ten_danh_muc: 'Đồng hồ',  
  mo_ta: 'Theo dõi sức khỏe, GPS',  
  gia: 9500000,  
  nha_san_xuat: 'Apple',  
  danh_gia: [  
    {  
      nguoi_dung: {  
        id: ObjectId('675f10000000000000000106'),  
        ten: 'Phạm Mỹ Dung'  
      },  
      diem_danh_gia: 4,  
      nhan_xet: 'Theo dõi sức khỏe chính xác, pin ổn.',  
      ngay_danh_gia: 2025-01-16T10:30:00.000Z  
    }  
  ],  
  tuong_tac: [  
    {  
      nguoi_dung: {  
        id: ObjectId('675f10000000000000000106'),  
        ten: 'Phạm Mỹ Dung'  
      },  
      loai_tuong_tac: 'Xem sản phẩm',  
    }  
  ]  
}
```



```
thoi_gian: 2025-01-16T09:10:00.000Z
},
{
  nguoi_dung: {
    id: ObjectId('675f10000000000000000106'),
    ten: 'Phạm Mỹ Dung'
  },
  loai_tuong_tac: 'Thêm vào yêu thích',
  thoi_gian: 2025-01-16T09:25:00.000Z
}
]
```

#### 4.1.3. Nghiệp vụ thống kê và báo cáo

Tổng hợp số lượng sản phẩm theo từng danh mục nhằm phục vụ công tác quản lý và phân tích cơ cấu sản phẩm trong hệ thống.

```
db.products.aggregate([
  {
    $group: {
      _id: "$ten_danh_muc",
      so_luong_san_pham: { $sum: 1 }
    }
  }
])
```

```
{
  _id: 'Tablet',
  so_luong_san_pham: 1
}
{
  _id: 'Điện thoại',
  so_luong_san_pham: 3
}
```

```
{
  _id: 'Laptop',
  so_luong_san_pham: 3
}
```

Thực hiện thống kê để xác định mỗi nhà sản xuất đang có bao nhiêu sản phẩm và thuộc những danh mục nào, phục vụ cho công tác quản lý và phân tích dữ liệu sản phẩm.

```
db.products.aggregate([
  {
    $group: {
      _id: {
        nha_san_xuat: "$nha_san_xuat",
        danh_muc: "$ten_danh_muc"
      },
      tong_so_san_pham: { $sum: 1 }
    }
  },
  {
    $group: {
      _id: "$_id.nha_san_xuat",
      ds_danh_muc: {
        $push: {
          ten_danh_muc: "$_id.danh_muc",
          so_luong: "$tong_so_san_pham"
        }
      },
      tong_cong: { $sum: "$tong_so_san_pham" }
    }
  }
])
```

```
{
  _id: 'Lenovo',
  ds_danh_muc: [
    {
      ten_danh_muc: 'Laptop',
      so_luong: 1
    }
  ],
  tong_cong: 1
}
{
  _id: 'Samsung',
  ds_danh_muc: [
    {
      ten_danh_muc: 'Máy tính bảng',
      so_luong: 1
    },
    {
      ten_danh_muc: 'Điện thoại',
      so_luong: 1
    }
  ],
  tong_cong: 2
}
```

#### 4.1.4. Nghiệp vụ: xóa hồ sơ

Xóa **hồ sơ người dùng thử nghiệm** hoặc **tài khoản không còn sử dụng** trong hệ thống

(ví dụ: xóa người dùng thử nghiệm có mã 675f10000000000000000000199 inclined).

```
db.users.deleteOne(
  { _id: ObjectId("675f10000000000000000000199") }
)
```

Kết quả trả về từ Shell:

```
{
  acknowledged: true,
  deletedCount: 1
}
```

## 4.2. Kết quả đạt được về mặt chức năng

Hệ thống đã hiện thực hóa đầy đủ các nghiệp vụ cốt lõi của một hệ thống đề xuất sản phẩm, bao gồm:

- Quản lý thông tin người dùng và sản phẩm;
- Lưu trữ và xử lý dữ liệu đánh giá (rating), nhận xét và tương tác của người dùng đối với sản phẩm;
- Quản lý dữ liệu đề xuất sản phẩm với điểm gợi ý (recommendation score);
- Hỗ trợ tra cứu, tìm kiếm sản phẩm theo nhiều tiêu chí khác nhau.

Các nghiệp vụ được thiết kế phù hợp với đặc thù dữ liệu lớn và hành vi người dùng thay đổi liên tục.

## 4.3. Đánh giá hiệu năng hệ thống

Kết quả thử nghiệm cho thấy:

- Thời gian truy vấn thông tin sản phẩm cùng đánh giá và tương tác được rút ngắn đáng kể;
- Hệ thống hoạt động ổn định khi số lượng người dùng và sản phẩm tăng lên;
- Các thao tác thống kê như tổng hợp điểm đánh giá, số lượt tương tác được thực hiện hiệu quả thông qua Aggregation Framework của MongoDB.

Điều này chứng minh MongoDB là lựa chọn phù hợp cho các hệ thống đề xuất sản phẩm có dữ liệu lớn và cấu trúc linh hoạt.

---

#### 4.4. Kết quả các giao diện chức năng trên MongoDB Compass

Quản lý Document chi tiết: MongoDB Compass cho phép xem và chỉnh sửa dữ liệu dưới dạng cây (tree view) hoặc bảng (table view). Các Document lồng nhau như thông tin sản phẩm, đánh giá người dùng, hoặc danh sách đề xuất được hiển thị trực quan, giúp người quản trị có cái nhìn tổng thể về một sản phẩm hoặc hồ sơ người dùng chỉ trên một màn hình duy nhất. Điều này rất hữu ích khi cần kiểm tra hoặc cập nhật dữ liệu phức tạp trong hệ thống đánh giá và đề xuất sản phẩm.

Thực thi bộ lọc (Filter): Giao diện truy vấn của MongoDB Compass hỗ trợ thực hiện các lệnh tìm kiếm từ đơn giản đến phức tạp. Người quản trị có thể lọc danh sách sản phẩm theo nhiều tiêu chí kết hợp như giá, đánh giá trung bình, nhà sản xuất, hoặc thể loại sản phẩm, từ đó nhanh chóng khai thác và phân tích dữ liệu phục vụ các chiến lược đề xuất.

Kiểm soát cập nhật (Update): Các thao tác updateOne và updateMany được Compass phản hồi tức thì về số lượng Document bị tác động, giúp đảm bảo tính chính xác và an toàn dữ liệu. Hệ thống cung cấp khả năng điều chỉnh thông tin sản phẩm, điểm đánh giá hay đề xuất mà không cần rời khỏi giao diện trực quan, tối ưu hóa quá trình quản lý dữ liệu.

#### 4.5. Đánh giá và Phân tích kết quả

##### 4.5.1. Về hiệu năng và tốc độ truy xuất

Tối ưu hóa đọc dữ liệu: Nhờ sử dụng kỹ thuật Embedding, tốc độ truy xuất thông tin một sản phẩm đầy đủ cùng các đánh giá và đề xuất liên quan nhanh hơn đáng kể so với SQL truyền thống, vì MongoDB không phải thực hiện các phép toán Join phức tạp giữa các bảng.

Tối ưu hóa báo cáo: Việc áp dụng Aggregation Pipeline kết hợp \$merge để tạo các Collection thống kê giúp các báo cáo doanh số, đánh giá sản phẩm hay đề xuất luôn sẵn sàng hiển thị, giảm tải việc tính toán lại dữ liệu mỗi khi người dùng truy cập dashboard.

#### **4.5.2. Về tính linh hoạt và khả năng mở rộng**

Linh hoạt lược đồ: Hệ thống cho phép thay đổi hoặc thêm các trường thông tin cho sản phẩm hoặc người dùng (ví dụ: thêm thuộc tính mới cho sản phẩm, điểm đánh giá chi tiết) mà không bắt buộc tất cả Document cũ phải có, tránh lãng phí không gian lưu trữ giá trị NULL.

Khả năng mở rộng: Cấu trúc dữ liệu hiện tại hoàn toàn tương thích với cơ chế sharding của MongoDB, sẵn sàng mở rộng khi số lượng sản phẩm, người dùng và đánh giá tăng lên hàng chục ngàn bản ghi.

#### **4.5.3. Trải nghiệm trong quá trình thực hiện**

Dễ thao tác với JSON: Việc sử dụng định dạng JSON giúp lập trình viên dễ dàng thao tác dữ liệu, vì cấu trúc Document tương đồng với các đối tượng trong nhiều ngôn ngữ lập trình hiện đại như JavaScript, Python.

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

### **5.1. Kết quả đạt được**

Đề tài “Thiết kế và cài đặt cơ sở dữ liệu NoSQL cho hệ thống đánh giá và đề xuất sản phẩm” đã hoàn thành các mục tiêu đề ra. Cụ thể, đề tài đã phân tích yêu cầu dữ liệu của hệ thống đánh giá và đề xuất sản phẩm; thiết kế mô hình cơ sở dữ liệu NoSQL theo hướng Document-oriented; cài đặt cơ sở dữ liệu bằng MongoDB và xây dựng các collection, document phục vụ lưu trữ thông tin người dùng, sản phẩm, đánh giá và dữ liệu đề xuất.

Đề tài đã đề xuất mô hình dữ liệu linh hoạt, áp dụng kỹ thuật nhúng và tham chiếu phù hợp, đáp ứng yêu cầu lưu trữ, truy xuất và mở rộng dữ liệu. Kết quả đạt được cho thấy cơ sở dữ liệu NoSQL phù hợp với các hệ thống có dữ liệu lớn và cấu trúc phức tạp như hệ thống đánh giá và đề xuất sản phẩm.

### **5.2. Hướng phát triển**

- Mở rộng mô hình dữ liệu để hỗ trợ thêm nhiều loại tương tác và hành vi người dùng.
- Kết hợp cơ sở dữ liệu NoSQL với hệ thống backend hoàn chỉnh và giao diện người dùng.
- Nghiên cứu tối ưu hiệu năng truy vấn và khả năng mở rộng trên tập dữ liệu lớn.
- Bổ sung các cơ chế bảo mật và phân quyền truy cập dữ liệu.
- Xây dựng hệ thống theo dõi và đánh giá chất lượng dữ liệu nhằm nâng cao độ tin cậy của kết quả đề xuất.

## **DANH MỤC TÀI LIỆU THAM KHẢO**

- [1] Meier and M. Kaufmann, SQL & NoSQL Databases: Models, Languages, Consistency Options and Architectures for Big Data Management. Springer Vieweg, 2019.
- [2] S. Bradshaw, E. Brazil, and K. Chodorow, MongoDB: The Definitive Guide: Powerful and Scalable Data Storage. O'Reilly Media, 2019.
- [3] Celko, Joe. Joe Celko's Complete Guide to NoSQL: What Every SQL Professional Needs to Know about Nonrelational Databases. Morgan Kaufmann, 2014.
- [4] Phan thị Phương Nam, Bài tập thực hành SQL và NoSQL, Trường Đại học Trà Vinh, 2024.
- [5] K. Banker et al., MongoDB in Action, 2nd ed. Manning Publications, 2016.
- [6] R. Copeland, MongoDB Applied Design Patterns. Sebastopol, CA: O'Reilly Media, Inc., 2013.