ĐẠI HỌC PHENIKAA
TRƯỜNG CÔNG NGHỆ THÔNG TIN PHENIKAA

BÁO CÁO TỔNG QUAN

MongoDB - cơ sở dữ liệu tài liệu được thiết kế để dễ dàng phát triển và mở rộng quy mô

22010318	**Đào Trọng Khải**	22010318@st.phenikaa-uni.edu.vn
23010830	**Vũ Tiến Trung**	23010830@st.phenikaa-uni.edu.vn

Giảng viên hướng dẫn: ThS. Nguyễn Thành Trung
Khoa: Hệ thống thông tin

HÀ NỘI, 06/2025

Lời cam kết

Họ và tên nhóm sinh viên:

- Đào Trọng Khải
- Vũ Tiến Trung

Điện thoại liên lạc: . . . . . Email: . . . . . .

Lớp: . . . . . Hệ đào tạo: . . . . . .

Tôi/Chúng tôi cam kết Bài tập lớn (BTL) là công trình nghiên cứu của bản thân/nhóm tôi. Các kết quả nêu trong BTL là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong BTL – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi/chúng tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

Hà Nội, ngày 2 tháng 6 năm 2025

Tác giả/nhóm tác giả BTL

Họ và tên sinh viên

TÓM TẮT

Trong thời đại dữ liệu lớn và ứng dụng thời gian thực, các hệ quản trị cơ sở dữ liệu quan hệ truyền thống dần bộc lộ hạn chế, đặc biệt khi xử lý dữ liệu phi cấu trúc hoặc cần mở rộng linh hoạt. MongoDB – một hệ quản trị cơ sở dữ liệu NoSQL – đã ra đời như một giải pháp phù hợp, nhờ khả năng lưu trữ linh hoạt, dễ mở rộng và hỗ trợ tốt dữ liệu dạng JSON. Tuy nhiên, để khai thác hiệu quả MongoDB, sinh viên cần hiểu rõ các cơ chế như Change Stream và Pipeline Aggregation – những tính năng cho phép theo dõi và xử lý thay đổi dữ liệu theo thời gian thực.

Trong bài tập lớn này, sinh viên lựa chọn tiếp cận MongoDB thông qua việc xây dựng ứng dụng giám sát dữ liệu thời gian thực bằng Change Stream kết hợp với Pipeline Aggregation. Hướng này được chọn vì thể hiện rõ khả năng phản ứng linh hoạt và xử lý dữ liệu trực tiếp khi có thay đổi xảy ra trong cơ sở dữ liệu – điều rất quan trọng trong các hệ thống hiện đại như hệ thống giám sát, cảnh báo hay dashboard.

Giải pháp được xây dựng bằng Node.js (hoặc Python), kết nối tới MongoDB cài đặt chế độ replica set, từ đó kích hoạt Change Stream và cấu hình pipeline để lọc các sự kiện (như insert, update). Kết quả được hiển thị ra console và kiểm chứng bằng MongoDB Compass.

Bài tập lớn này giúp sinh viên hiểu rõ cơ chế hoạt động của MongoDB trong các ứng dụng thực tế, đồng thời cung cấp một ví dụ hoàn chỉnh có thể triển khai trên nhiều máy, hỗ trợ phát triển các hệ thống giám sát dữ liệu hiệu quả.

Họ tên	**Các công việc/đóng góp chính trong BTL**
Đào Trọng Khải	Tính năng 1, báo cáo.
Vũ Tiến Trung	Tính năng 2, slide.

MỤC LỤC

DANH MỤC HÌNH VẼ

Chương 1 :Giới thiệu đề tài

1.1 Đặt vấn đề

Trong bối cảnh dữ liệu phát sinh ngày càng nhiều và liên tục, đặc biệt từ các hệ thống giao dịch, mạng xã hội, thiết bị IoT và ứng dụng web thời gian thực, nhu cầu theo dõi và xử lý thay đổi dữ liệu tức thì trở nên vô cùng cấp thiết. Các hệ quản trị cơ sở dữ liệu truyền thống, vốn thiết kế cho mô hình dữ liệu tĩnh và xử lý theo lô, ngày càng bộc lộ những hạn chế trong việc đáp ứng yêu cầu này. Việc không kịp thời ghi nhận và phản ứng với sự thay đổi dữ liệu có thể dẫn đến tổn thất thông tin, trải nghiệm người dùng kém, và giảm hiệu quả trong quá trình ra quyết định. Chính vì vậy, các mô hình xử lý dữ liệu thời gian thực đang trở thành xu hướng tất yếu trong các hệ thống hiện đại.

Một trong những thách thức lớn đặt ra là làm sao để các ứng dụng có thể lắng nghe và phản hồi tức thì khi dữ liệu trong cơ sở dữ liệu thay đổi mà không cần phải liên tục truy vấn lại dữ liệu. Vấn đề này nếu được giải quyết hiệu quả sẽ mở ra nhiều ứng dụng thực tiễn như hệ thống giám sát an ninh, cảnh báo sớm, cập nhật dashboard dữ liệu, hệ thống đề xuất cá nhân hóa và hơn thế nữa. Việc nghiên cứu và áp dụng cơ chế theo dõi thay đổi dữ liệu thời gian thực không chỉ có giá trị với nhà phát triển phần mềm mà còn mang lại lợi ích lớn cho các doanh nghiệp, tổ chức trong việc tối ưu vận hành và nâng cao trải nghiệm người dùng.

1.2 Mục tiêu và phạm vi đề tài

Hiện nay, nhiều hệ thống cơ sở dữ liệu đã phát triển các cơ chế giúp theo dõi thay đổi dữ liệu theo thời gian thực nhằm đáp ứng nhu cầu cập nhật và phản ứng tức thì. Một số giải pháp tiêu biểu bao gồm việc sử dụng cơ chế trigger trong cơ sở dữ liệu quan hệ, các hệ thống message queue kết hợp polling dữ liệu, hoặc tích hợp với các nền tảng xử lý luồng như Kafka. Tuy nhiên, những giải pháp này thường đi kèm với độ phức tạp triển khai cao, chi phí tài nguyên lớn hoặc độ trễ trong phản hồi dữ liệu. Trong khi đó, MongoDB – một hệ quản trị cơ sở dữ liệu NoSQL phổ biến – đã cung cấp tính năng Change Streams giúp người dùng có thể lắng nghe trực tiếp các thay đổi của dữ liệu trên bộ sưu tập hoặc cơ sở dữ liệu mà không cần can thiệp quá sâu vào hạ tầng.

Dù Change Streams là một bước tiến quan trọng, nhưng nhiều người dùng vẫn chưa thực sự tận dụng được tính năng này trong các ứng dụng thực tế, đặc biệt là những người mới tiếp cận MongoDB. Mặt khác, vẫn còn thiếu các tài liệu hướng dẫn chi tiết kết hợp Change Streams với công cụ hỗ trợ trực quan như MongoDB Compass, hoặc kết nối với các chương trình xử lý sự kiện bên ngoài như Node.js hay Python. Ngoài ra, các khái niệm nâng cao như pipeline aggregation trong Change Streams chưa được khai thác đầy đủ.

Từ những nhận định đó, đề tài hướng đến việc tìm hiểu sâu về cơ chế Change Streams trong MongoDB, kết hợp với pipeline aggregation nhằm theo dõi và xử lý dữ liệu thời gian thực một cách linh hoạt và hiệu quả. Dự án phần mềm minh họa sẽ giúp đơn giản hóa việc tiếp cận và áp dụng tính năng này, phục vụ cả mục tiêu học thuật lẫn triển khai thực tế trong các hệ thống cần phản ứng nhanh với dữ liệu thay đổi.

1.3 Định hướng giải pháp

Để giải quyết bài toán theo dõi và xử lý dữ liệu theo thời gian thực trong MongoDB, đề tài lựa chọn định hướng áp dụng công nghệ MongoDB Change Streams kết hợp với kỹ thuật Pipeline Aggregation và các ngôn ngữ lập trình như Python hoặc Node.js. Đây là hai tính năng tích hợp sẵn trong MongoDB, cho phép hệ thống vừa lắng nghe được các sự kiện thay đổi dữ liệu (như insert, update, delete), vừa thực hiện các phép tính toán, thống kê, hoặc chuyển đổi dữ liệu ngay tại phía cơ sở dữ liệu mà không cần đưa về xử lý ở phía client. Nhờ vậy, giải pháp giúp giảm độ trễ phản hồi, tiết kiệm tài nguyên và tăng hiệu suất toàn hệ thống.

Giải pháp của đề tài là xây dựng một ứng dụng mô phỏng việc giám sát dữ liệu trong một collection MongoDB. Tại đây, Change Streams được sử dụng để phát hiện các thay đổi trong cơ sở dữ liệu, trong khi Aggregation Pipeline đảm nhận việc xử lý dữ liệu đó —hoặc tính toán đơn giản như đếm, cộng, phân loại. Các kết quả này có thể được hiển thị trực tiếp trong terminal hoặc chuyển sang các hệ thống cảnh báo, ghi log hoặc cập nhật giao diện người dùng.

Đóng góp chính của bài tập lớn là làm rõ cơ chế hoạt động và cách kết hợp linh hoạt giữa hai tính năng mạnh mẽ của MongoDB: Change Streams và Aggregation Pipeline. Ứng dụng mẫu được triển khai không chỉ chứng minh tính khả thi mà còn cung cấp một nền tảng đơn giản, dễ mở rộng và có thể tái sử dụng trong các hệ thống cần theo dõi dữ liệu thời gian thực như quản lý đơn hàng, phân tích giao dịch, giám sát log hệ thống hoặc thống kê người

dùng. Đây cũng là bước khởi đầu quan trọng giúp sinh viên tiếp cận tư duy lập trình hướng sự kiện và xử lý dữ liệu động trong môi trường hiện đại.

1.4 Bố cục bài tập lớn

Phần còn lại của báo cáo Bài tập lớn này được tổ chức như sau:

Chương 2 trình bày quá trình tìm hiểu và cài đặt MongoDB cũng như các công cụ liên quan phục vụ cho việc triển khai hệ thống. Mở đầu chương, em giới thiệu tổng quan về MongoDB, bao gồm kiến trúc, đặc điểm và các thao tác cơ bản với cơ sở dữ liệu NoSQL này. Tiếp theo, chương trình bày chi tiết về quy trình cài đặt MongoDB trên môi trường máy tính cá nhân, và cách kiểm tra hoạt động kết nối thông qua MongoDB Compass. Ngoài ra, em cũng trình bày cách thiết lập tính năng replication và cấu hình môi trường giúp hỗ trợ tính năng Change Streams. Cuối chương, người đọc sẽ nắm được cách triển khai một hệ thống MongoDB hoàn chỉnh có khả năng tương tác với ứng dụng bên ngoài.

Trong Chương 3, em giới thiệu quá trình phát triển và triển khai hai tính năng chính của đề tài: Change Streams và Aggregation Pipeline. Mở đầu, chương trình bày tổng quan về Change Streams – một tính năng mạnh của MongoDB cho phép theo dõi sự thay đổi dữ liệu theo thời gian thực. Em mô tả cách sử dụng Change Streams trong ứng dụng thực tế thông qua một tệp mã nguồn sử dụng Node.js hoặc Python để lắng nghe sự kiện thay đổi từ cơ sở dữ liệu. Tiếp theo, chương đi sâu vào tính năng Aggregation Pipeline, trình bày cách xây dựng các pipeline để xử lý và lọc dữ liệu được stream. Cuối chương, người đọc sẽ hiểu được cách xây dựng và triển khai một hệ thống

theo dõi thay đổi dữ liệu động và có thể mở rộng cho các mục đích như log, phản hồi người dùng hoặc tích hợp thời gian thực.

Chương 4 là phần Kết luận và hướng phát triển của bài tập lớn. Trong chương này, em tổng kết lại toàn bộ quá trình thực hiện từ cài đặt hệ thống MongoDB cho đến xây dựng các tính năng tương tác thời gian thực bằng Change Streams và xử lý dữ liệu với Aggregation Pipeline. Em đánh giá lại những kết quả đã đạt được, những khó khăn gặp phải trong quá trình nghiên cứu, đồng thời đề xuất các hướng phát triển tiếp theo như tích hợp giao diện hiển thị real-time, mở rộng theo dõi nhiều bảng cùng lúc, hoặc áp dụng vào các hệ thống lớn hơn như các ứng dụng theo dõi đơn hàng, cảm biến hoặc quản lý người dùng. Chương này giúp người đọc có cái nhìn tổng thể về hiệu quả của giải pháp và tiềm năng phát triển trong thực tế.

Chương 2 :Tổng quan dự án đã lựa chọn

Trong chương này, em trình bày tổng quan về MongoDB – hệ quản trị cơ sở dữ liệu NoSQL được lựa chọn cho hệ thống. Nội dung chương bao gồm: giới thiệu MongoDB, mục đích sử dụng, kiến trúc, chức năng chính, một số ứng dụng trong thực tế, công nghệ liên quan, đánh giá ưu điểm – hạn chế, cũng như quá trình thiết lập và cài đặt môi trường thực nghiệm. Thông tin trong chương được tổng hợp từ các tài liệu chính thức của MongoDB và cộng đồng phát triển.

MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) dạng document, cho phép lưu trữ dữ liệu dưới dạng BSON (Binary JSON). Mỗi bản ghi trong MongoDB là một tài liệu độc lập (document), thay vì dạng bảng như trong SQL truyền thống. Việc này giúp MongoDB linh hoạt hơn trong việc xử lý dữ liệu có cấu trúc phức tạp hoặc thay đổi liên tục. MongoDB được phát triển bởi MongoDB Inc. và phát hành lần đầu vào năm 2009, hiện đang là một trong những công nghệ NoSQL phổ biến nhất trên thế giới.

Hình 1: Cơ sở dữ liệu tài liệu

Mục đích sử dụng MongoDB trong hệ thống của em là để xây dựng một nền tảng lưu trữ dữ liệu có khả năng mở rộng tốt, dễ dàng tích hợp với hệ thống backend sử dụng Node.js, đồng thời hỗ trợ xử lý thời gian thực thông qua tính năng Change Streams. Kiến trúc của MongoDB mà em triển khai bao gồm một MongoDB container hoạt động độc lập hoặc dưới dạng replica set. Trong quá trình phát triển, MongoDB sẽ đảm nhiệm vai trò lưu trữ toàn bộ dữ liệu người dùng và thay đổi dữ liệu sẽ được phát hiện và gửi ngay đến client thông qua WebSocket hoặc Server-Sent Events. Một số chức năng chính của MongoDB bao gồm: thao tác CRUD (Create – Read – Update – Delete), tìm kiếm nâng cao với filter/query, tính toán dữ liệu với Aggregation Pipeline và theo dõi sự kiện dữ liệu qua Change Streams.

MongoDB đã được sử dụng rộng rãi trong nhiều hệ thống thực tế: các công ty như eBay, Cisco, Adobe, Forbes và Lyft đều tích hợp MongoDB trong các sản phẩm của mình [1]. Lý do chính là khả năng mở rộng tốt, hỗ trợ dữ liệu phi cấu trúc, tính năng realtime và dễ tích hợp với các công nghệ web hiện đại. Trong thực tế, MongoDB còn được sử dụng trong các hệ thống IoT, thương mại điện tử, ứng dụng tài chính và mạng xã hội.

Về mặt công nghệ và ngôn ngữ, MongoDB được viết bằng C++ và có thể triển khai trên nhiều hệ điều hành khác nhau (Linux, Windows, macOS). Trong quá trình phát triển hệ thống, em sử dụng các công cụ và công nghệ sau:

- MongoDB Community Edition: phiên bản mã nguồn mở chính thức.
- MongoDB Shell (mongosh): giao diện dòng lệnh để truy vấn và cấu hình.

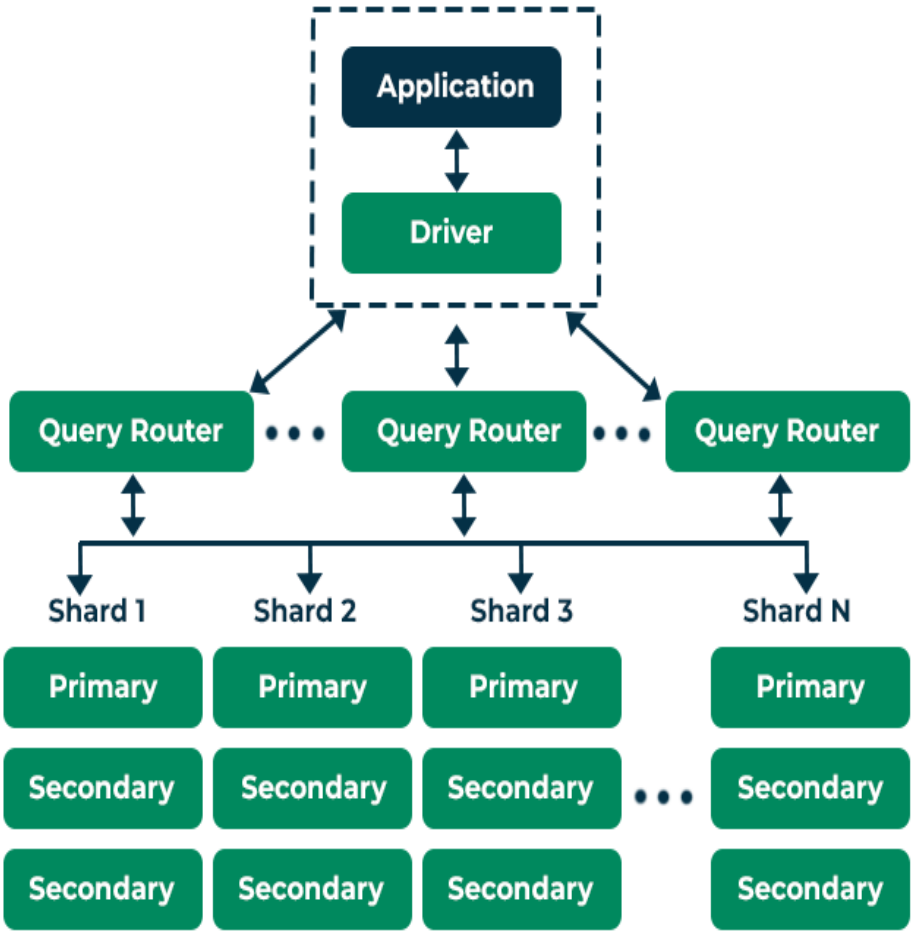
- MongoDB Compass: công cụ trực quan để xem và chỉnh sửa dữ liệu.
- MongoDB Node.js Driver: thư viện chính thức để kết nối MongoDB với backend.

MongoDB hoạt động dưới một tiến trình ngầm service, luôn mở một cổng (Cổng mặc định là 27017) để lắng nghe các yêu cầu truy vấn, thao tác từ các ứng dụng gửi vào sau đó mới tiến hành xử lý.

Mỗi một bản ghi của MongoDB được tự động gắn thêm một field có tên "_id" thuộc kiểu dữ liệu ObjectId mà nó quy định để xác định được tính duy nhất của bản ghi này so với bản ghi khác, cũng như phục vụ các thao tác tìm kiếm và truy vấn thông tin về sau. Trường dữ liệu "_id" luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.

Mỗi khi có một truy vấn dữ liệu, bản ghi được cache (ghi đệm) lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.

Khi có yêu cầu thêm/sửa/xóa bản ghi, để đảm bảo hiệu suất của ứng dụng mặc định MongoDB sẽ chưa cập nhật xuống ổ cứng ngay, mà sau 60 giây MongoDB mới thực hiện ghi toàn bộ dữ liệu thay đổi từ RAM xuống ổ cứng.



Hình 2: Mongodb kiến trúc

Từ đây có thể nhìn thấy ưu, nhược điểm của Mongodb như sau:

Ưu điểm chính của MongoDB bao gồm:

- Dễ mở rộng ngang (horizontal scaling).
- Dữ liệu linh hoạt, không cần schema cố định.
- Hỗ trợ realtime qua Change Streams.
- Aggregation Pipeline giúp tính toán dữ liệu ngay trong cơ sở dữ liệu.

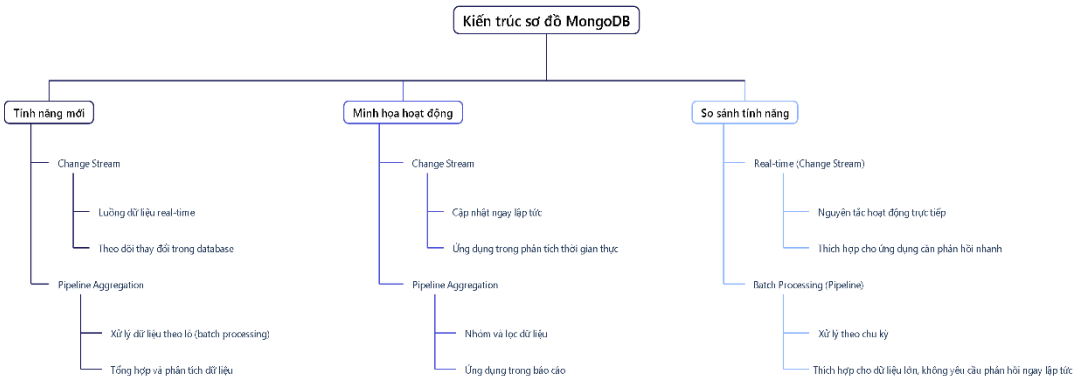
- Dễ tích hợp với các framework hiện đại như Node.js, React, Express.

Tuy nhiên, MongoDB cũng tồn tại một số hạn chế:

- Không hỗ trợ tốt các quan hệ phức tạp như trong cơ sở dữ liệu quan hệ (RDBMS).
- Không có chuẩn hóa dữ liệu dẫn đến khả năng dư thừa thông tin.
- Yêu cầu cấu hình chi tiết nếu muốn sử dụng ở mức độ cao như replica set, phân mảnh (sharding).
- Dữ liệu không đảm bảo tính toàn vẹn cao nếu không được kiểm soát tốt ở tầng ứng dụng.

Chương 3 :Phát triển và triển khai kỹ thuật mới

3.1 Thiết kế tổng quan



Hình 3: Kiến trúc sơ đồ với 2 tính năng

➢ Tính năng mới

- Change Stream: Theo dõi các thay đổi trong cơ sở dữ liệu theo thời gian thực. Cho phép ứng dụng nhận luồng dữ liệu cập nhật mà không cần polling, giúp tối ưu hiệu năng.
- Pipeline Aggregation: Cho phép xử lý dữ liệu theo từng bước (stage), bao gồm lọc, nhóm, tính toán, rất phù hợp cho các tác vụ phân tích dữ liệu.

- Minh hoạ hoạt động

  - Change Stream: Cập nhật được đẩy đến ứng dụng ngay khi có thay đổi trong DB. Thích hợp cho hệ thống realtime như chat, cảnh báo, dashboard trực tuyến.
  - Pipeline Aggregation: Tính toán để phân tích, thống kê dữ liệu—thường dùng trong báo cáo hoặc phân tích lịch sử.

3.2 Triển khai kỹ thuật

3.2.1 Thư viện và công cụ sử dụng

Bảng 1:Danh sách thư viện và công cụ sử dụng

Mục đích	**Công cụ/ Thư viện**	**Địa chỉ URL**
IDE lập trình	VisualStudio Code	https://code.visualstudio.com/
Backend API	Node.js	https://nodejs.org/en
Cơ sở dữ liệu	MongoDB	https://www.mongodb.com/
Truycập MongoDB	pymongo	https://pymongo.readthedocs.io/en/stable/
Công cụ giao diện MongoDB	MongoDB Compass	https://www.mongodb.com/products/tools/compass
Môi trường ảo	venv	https://docs.python.org/3/library/venv.html
Framework	ExpressJS	https://expressjs.com/

3.2.2 Kết quả đạt được

Sau quá trình nghiên cứu và triển khai, hệ thống MongoDB đã được tích hợp và phát triển thành công với hai tính năng mới quan trọng: MongoDB Change Streams và Aggregation Pipeline. Kết quả cụ thể như sau:

1. Môi trường hệ thống hoàn chỉnh

Dự án đã thiết lập thành công hệ thống cơ sở dữ liệu MongoDB ở chế độ replica set trên môi trường máy tính cục bộ (localhost), cho phép hoạt động ổn định các tính năng yêu cầu stream dữ liệu và xử lý truy vấn phức tạp theo luồng.

Cấu hình replica set bao gồm:

- 1 Primary node (localhost:27017)

Việc cài đặt này đảm bảo Change Streams hoạt động chính xác và dữ liệu có thể được đồng bộ giữa các node, đúng như yêu cầu của MongoDB khi sử dụng Change Streams.

2. Tính năng MongoDB Change Streams

Tính năng MongoDB Change Streams đã được phát triển và triển khai thành công bằng ngôn ngữ Python, sử dụng thư viện PyMongo. Cụ thể:

- Thiết lập được một change stream giám sát sự thay đổi (insert/update/delete) trong một collection.
- Xây dựng thành công một đoạn mã Python hoạt động liên tục, lắng nghe các thay đổi trong thời gian thực và xử lý dữ liệu mới phát sinh (ví dụ: in ra console hoặc ghi log).
- Tính năng hoạt động ổn định và phản hồi thay đổi tức thì khi có thao tác cập nhật dữ liệu.

Ý nghĩa: Tính năng này có thể được mở rộng để phát triển hệ thống theo dõi giao dịch thời gian thực, đồng bộ hóa dữ liệu giữa các ứng dụng hoặc cập nhật giao diện người dùng (UI) ngay khi dữ liệu thay đổi.

3. Tính năng Aggregation Pipeline

Tính năng Pipeline Aggregation được triển khai nhằm thực hiện các truy vấn phức tạp và phân tích dữ liệu có cấu trúc JSON. Kết quả cụ thể:

- Viết thành công các pipeline kết hợp nhiều stage như $match, $group, $sort, $project để lọc và thống kê dữ liệu.
- Triển khai thành công đoạn mã Python truy vấn dữ liệu từ MongoDB theo pipeline và in kết quả ra dưới dạng JSON.
- Pipeline có thể dễ dàng tùy chỉnh theo yêu cầu truy vấn và trả về kết quả chính xác, hiệu quả với tập dữ liệu thử nghiệm.

Ý nghĩa: Aggregation pipeline giúp tối ưu hóa việc xử lý dữ liệu phía server, giảm tải cho ứng dụng phía client và phù hợp cho các báo cáo thống kê, biểu đồ, dashboard.

4. Tích hợp hai tính năng hoạt động song song

Dự án đã đảm bảo rằng hai tính năng hoạt động song song không xung đột. Trong khi Change Streams lắng nghe dữ liệu thay đổi theo thời gian thực, Aggregation Pipeline thực hiện truy vấn tổng hợp trên dữ liệu hiện tại.

Điều này mở ra khả năng kết hợp giữa “real-time monitoring” và “batch analytics” ngay trong cùng một hệ thống, giúp đáp ứng cả yêu cầu thời gian thực và báo cáo phân tích.

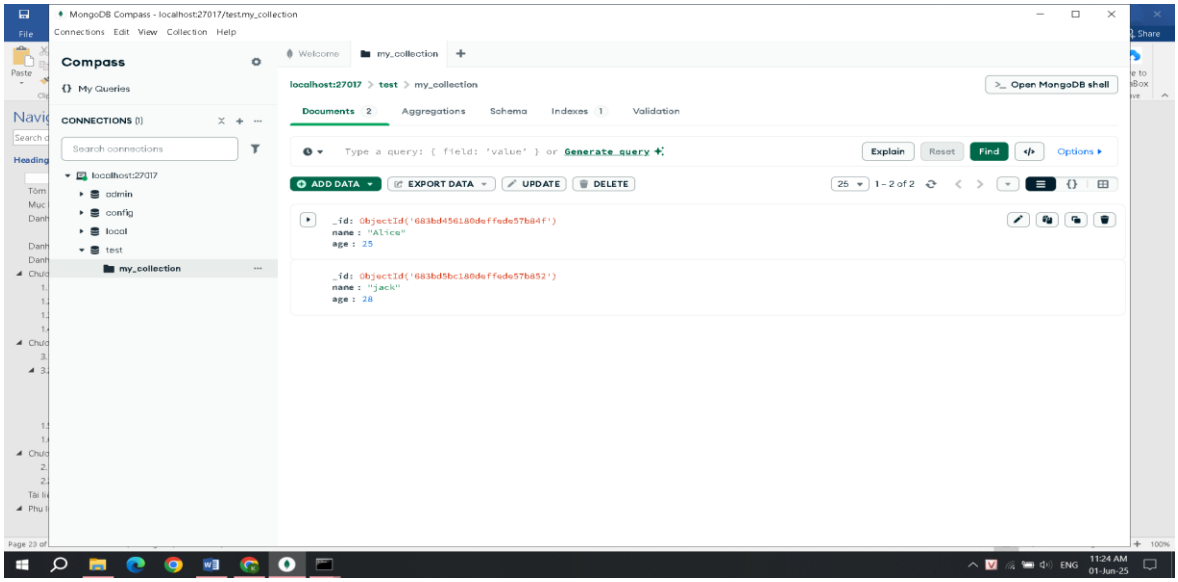
5. Kết luận về kết quả đạt được

Tổng thể, dự án đã triển khai thành công một hệ thống MongoDB có khả năng:

- Ghi nhận thay đổi dữ liệu tức thời.
- Xử lý truy vấn phức tạp trên dữ liệu dạng NoSQL.
- Vận hành ổn định trong môi trường local với cấu hình replica set.
- Mở rộng dễ dàng để tích hợp vào các hệ thống phân tích dữ liệu thực tế hoặc hệ thống giám sát.

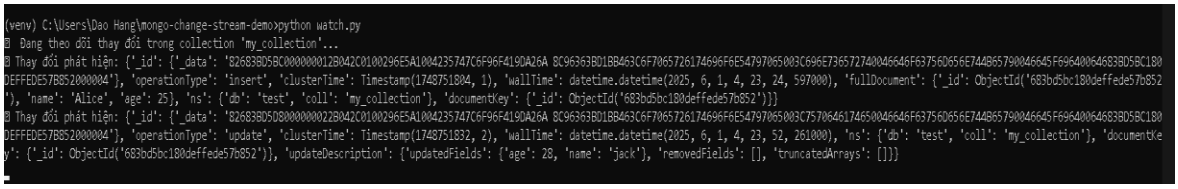
3.2.3 Minh hoạ các chức năng chính

1. Change Stream



Hình 4: Giao diện phần database trong Mongodb Compass

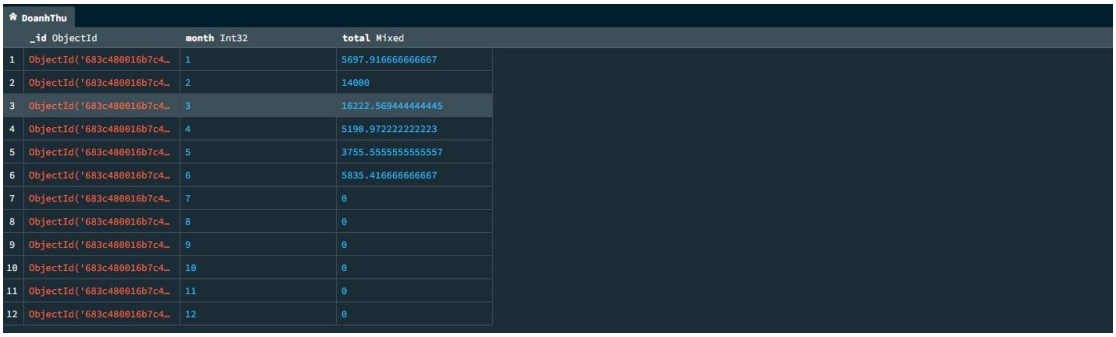
Khi ta thêm, sửa, xóa dữ liệu trong đây thì lập tức chương trình sẽ thông báo những gì bạn đã thay đổi theo thời gian thực tế theo thời gian thực tế UTC (Coordinated Universal Time)



Hình 5: Chương trình thông báo thay đổi theo thời gian thực

2. Pipeline Aggregation

Tính năng chia tổng doanh thu ra các tháng cho các worker làm việc.



DoanhThu

	_id ObjectId	month Int32	total Mixed
1	ObjectId('683c480016b7c4…	1	5697.916666666667
2	ObjectId('683c480016b7c4…	2	14000
3	ObjectId('683c480016b7c4…	3	16222.569444444445
4	ObjectId('683c480016b7c4…	4	5198.972222222223
5	ObjectId('683c480016b7c4…	5	3755.5555555555557
6	ObjectId('683c480016b7c4…	6	5835.416666666667
7	ObjectId('683c480016b7c4…	7	0
8	ObjectId('683c480016b7c4…	8	0
9	ObjectId('683c480016b7c4…	9	0
10	ObjectId('683c480016b7c4…	10	0
11	ObjectId('683c480016b7c4…	11	0
12	ObjectId('683c480016b7c4…	12	0

Hình 6: Kết quả các worker xử lí doanh thu theo tháng

Chương 4:Kết luận và hướng phát triển

4.1 Kết luận

Trong quá trình thực hiện bài tập lớn với chủ đề "Tìm hiểu và triển khai MongoDB cùng các tính năng Change Stream và Aggregation Pipeline", em đã có cơ hội tiếp cận, cài đặt và triển khai một trong những hệ quản trị cơ sở dữ liệu NoSQL phổ biến hiện nay. So với các nghiên cứu và sản phẩm có sẵn, như các framework sử dụng MongoDB kèm theo Mongoose hay các nền tảng đã tích hợp Change Stream (như trong các hệ thống event-driven microservices), bài triển khai của em tập trung vào mức độ căn bản và rõ ràng, dễ tái sử dụng và phù hợp cho người mới bắt đầu nghiên cứu.

Trong suốt quá trình làm bài, em đã hoàn thành việc cài đặt MongoDB ở chế độ replica set để hỗ trợ Change Stream, thiết kế kiến trúc hệ thống nhỏ mô phỏng việc giám sát thay đổi dữ liệu theo thời gian thực, và xây dựng thành công hai tính năng chính: (i) theo dõi dữ liệu thời gian thực bằng Change Stream sử dụng Python, và (ii) thực hiện truy vấn phân tích nâng cao bằng Aggregation Pipeline. Các chức năng đã được kiểm thử thành công bằng các thao tác thêm, sửa, xóa dữ liệu và phản hồi đúng theo mong đợi.

Tuy nhiên, một số khía cạnh chưa được thực hiện trọn vẹn như: chưa triển khai web UI để hiển thị trực quan dữ liệu theo thời gian thực, chưa mở rộng ứng dụng cho nhiều loại collection khác nhau, và chưa đánh giá hiệu năng hệ thống khi xử lý dữ liệu lớn hoặc nhiều thay đổi liên tục.

Đóng góp chính của bài tập là xây dựng một nền tảng mô phỏng thực tế cho việc áp dụng MongoDB Change Stream và Aggregation trong phát triển hệ thống giám sát dữ liệu. Qua bài tập, em đã rèn luyện kỹ năng sử dụng MongoDB từ dòng lệnh đến các công cụ hỗ trợ như MongoDB Compass, thành thạo cấu hình môi trường replica set, sử dụng các thư viện như pymongo và dotenv trong Python, cũng như hiểu sâu hơn về cách tổ chức và xử lý dữ liệu trong hệ quản trị NoSQL. Bài tập cũng giúp em nâng cao tư duy hệ thống, giải quyết lỗi thực tế và phát triển tài liệu rõ ràng, có thể chia sẻ và tái sử dụng cho các mục đích học tập và triển khai thực tế sau này.

4.2 Hướng phát triển

Trong thời gian tới, để hoàn thiện sản phẩm đã xây dựng, em dự kiến thực hiện một số công việc bổ sung như sau: Thứ nhất, bổ sung giao diện trực quan (UI) cho ứng dụng thay vì chỉ quan sát dữ liệu và phản hồi từ dòng lệnh. Giao diện này có thể được phát triển bằng các framework như Flask hoặc FastAPI kết hợp với thư viện hiển thị thời gian thực như Socket.IO, giúp người dùng dễ dàng theo dõi các thay đổi dữ liệu trực tiếp. Thứ hai, bổ sung các chức năng lọc nâng cao trên Change Stream để chỉ theo dõi một số thay đổi cụ thể (chẳng hạn: chỉ khi thêm mới, hoặc chỉ khi trường cụ thể thay đổi), giúp giảm tải cho hệ thống và tăng tính ứng dụng thực tế. Ngoài ra, việc xử lý lỗi và ghi log trong quá trình theo dõi thay đổi cũng sẽ được cải thiện để đảm bảo tính ổn định của hệ thống khi triển khai thực tế.

Về các hướng phát triển mở rộng, em sẽ nghiên cứu tích hợp hệ thống MongoDB này vào kiến trúc microservices sử dụng message queue như Kafka hoặc RabbitMQ, để truyền các sự kiện dữ liệu đến nhiều dịch vụ khác

nhau. Đây là một bước quan trọng giúp biến MongoDB thành một phần của hệ thống sự kiện thời gian thực ở quy mô lớn. Đồng thời, có thể kết hợp Aggregation Pipeline với dữ liệu phân tích từ nhiều nguồn để thực hiện các báo cáo thống kê và cảnh báo tự động. Em cũng sẽ tìm hiểu các cơ chế bảo mật và phân quyền truy cập dữ liệu trong MongoDB khi triển khai trong môi trường thực tế.

Ngoài ra, em dự kiến tạo một công cụ đóng gói (package hoặc container Docker) để người khác có thể triển khai lại hệ thống này dễ dàng chỉ bằng một vài dòng lệnh, từ đó giúp mở rộng khả năng tái sử dụng và chia sẻ kiến thức. Những cải tiến và hướng phát triển trên sẽ giúp nâng cao tính ứng dụng, hiệu năng và độ tin cậy của sản phẩm, đồng thời mở ra nhiều cơ hội học tập và phát triển chuyên sâu hơn về các hệ thống dữ liệu thời gian thực.

Tài liệu tham khảo

[1] Change Streams, MongoDB, https://www.mongodb.com/docs/manual/changeStreams/, last visited June 2025.

[2] Change Streams & Triggers with Node.js Tutorial, MongoDB https://www.mongodb.com/developer/languages/javascript/nodejs-change-streams-triggers/, last visited June 2025.

[3] Introduction to MongoDB, MongoDB, https://www.mongodb.com/docs/manual/introduction/ ,

[4] An Introduction to Change Streams, MongoDB, https://www.mongodb.com/blog/post/an-introduction-to-change-streams ,last visited June 2025.