

TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP. HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN



TIÊU LUẬN ẢO HOÁ VÀ ĐIỆN TOÁN ĐÁM MÂY

ĐỀ TÀI:

**TÌM HIỂU VỀ NỀN TẢNG MONGODB – TRIỂN KHAI
ỨNG DỤNG WEB KẾT HỢP GITHUB**

GVHD: ThS. Nguyễn Quốc Sử – Nhóm thực hiện: Nhóm 3

- | | |
|---------------|-------------------|
| 1. 2033223948 | Nguyễn Minh Quân |
| 2. 2033223978 | Nguyễn Đình Quốc |
| 3. 2033220316 | Nguyễn Huy Bảo |
| 4. 2001221508 | Nguyễn Minh Hoàng |
| 5. 2033221134 | Trần Minh Hải |

TP. Hồ Chí Minh, tháng 11/2024

**BẢNG PHÂN CÔNG CÔNG VIỆC CHO CÁC THÀNH VIÊN
NHÓM 3**

STT	MSSV	HỌ TÊN	CÔNG VIỆC ĐƯỢC GIAO	ĐÁNH GIÁ
1	2033223948	Nguyễn Minh Quân		
2	2033223978	Nguyễn Đình Quốc		
3	2033220316	Nguyễn Huy Bảo		
4	2001221508	Nguyễn Minh Hoàng		
5	2033221134	Trần Minh Hải		

MỤC LỤC

MỞ ĐẦU	6
CHƯƠNG I. CƠ SỞ LÝ THUYẾT.....	8
I. GIỚI THIỆU	8
1. NoSQL là gì?	8
2. Tại sao cần phải có NoSQL?	8
3. NoSQL hoạt động như thế nào?	8
4. Phân loại NoSQL database.....	9
5. So sánh giữa NoSQL và SQL.....	9
6. MongoDB là gì? hình thành như nào?	10
7. Tại sao tạo ra MongoDB? mục đích tạo ra MongoDB?.....	11
8. Nếu không có MongoDB có thể thay thế bởi cái gì?	12
9. Ý nghĩa của MongoDB đối với ngành công nghệ thông tin	12
II. CÁC KHÁI NIỆM LIÊN QUAN	13
1. Các thành phần cốt lõi tạo nên MongoDB	13
2. JSON.....	13
3. Document.....	13
4. Collection.....	13
5. BSON.....	14
6. MongoDB Compass	14
7. MongoDB Database Tools	14
8. DataBase	14
III. MÔI TRƯỜNG SỬ DỤNG	14
1. Sử dụng trên máy chủ vật lý	14
2. Sử dụng trên môi trường đám mây.....	15
3. Sử dụng MongoDB Atlas (được MongoDB Inc quản lý)	15
IV. CÁC TÍNH NĂNG CƠ BẢN.....	15
1. Giới thiệu về mô hình dữ liệu.....	15
2. Tính mềm dẻo.....	15
3. Các cấu trúc của mô hình dữ liệu	15
4. Mô hình quan hệ giữa các document.....	16
5. Mô hình cây	18

6.	Các phương thức CRUD trong MongoDB.....	20
V.	CÁC TÍNH NĂNG NÂNG CAO.....	27
1.	Index	27
2.	Nhập và xuất dữ liệu.....	28
3.	Một số tính năng khác trên MongoDB	32
VI.	ỨNG DỤNG CỤ THỂ	32
1.	Ứng dụng Web	32
2.	Mobile App.....	33
3.	Phân tích dữ liệu thời gian thực.....	33
4.	Hệ Thống Phân Tán.....	33
5.	IoT	33
6.	Microservices	33
CHƯƠNG II. THỰC NGHIỆM VỚI MONGODB.....		34
I.	CÀI ĐẶT MONGODB.....	34
1.	Cấu hình phần cứng yêu cầu	34
2.	Các phiên bản	34
3.	Hướng dẫn cài MongoDB	34
II.	HƯỚNG DẪN SỬ DỤNG	42
1.	Tìm hiểu MongoDB và Mongo Shell.....	42
2.	Các truy vấn cơ bản trong MongoDB	44
3.	MongoDB Atlas.....	51
4.	Cài đặt Shell và MongoDB for VSCode.	67
III.	XÂY DỰNG ỨNG DỤNG WEB	77
1.	Triển khai ứng dụng web.....	77
2.	Sử dụng github quản lý code	98
3.	Deploy ứng dụng web lên internet	115
CHƯƠNG III. ĐÁNH GIÁ KẾT QUẢ.....		119
I.	KẾT QUẢ ĐẠT ĐƯỢC	119
1.	Kết quả thực nghiệm	119
2.	So sánh với mục tiêu ban đầu.....	119
II.	ĐÁNH GIÁ VÀ SO SÁNH	120
1.	Đánh giá ưu/nhược điểm của MongoDB	120

2.	So sánh MongoDB với MySQL	121
III.	HƯỚNG PHÁT TRIỂN	124
1.	Sức ảnh hưởng của MongoDB với hiện tại	124
2.	Định hướng cho tương lai của MongoDB	125
TÀI LIỆU THAM KHẢO		126

MỞ ĐẦU

Trong bối cảnh phát triển nhanh chóng của công nghệ, việc lưu trữ và quản lý dữ liệu lớn đã trở thành một thách thức quan trọng đối với các ứng dụng web hiện đại. Tiêu luận này nhằm cung cấp một cái nhìn toàn diện về nền tảng cơ sở dữ liệu NoSQL MongoDB, một giải pháp tối ưu cho việc lưu trữ và truy xuất dữ liệu linh hoạt, kết hợp với GitHub để triển khai ứng dụng web một cách hiệu quả.

Cơ sở lý luận của tiêu luận dựa trên nhu cầu thực tiễn về việc xử lý dữ liệu lớn và phi cấu trúc mà các hệ quản trị cơ sở dữ liệu quan hệ truyền thống không đáp ứng được. MongoDB, với khả năng mở rộng và lưu trữ dữ liệu theo dạng tài liệu, đã trở thành một lựa chọn phổ biến, giúp giải quyết các vấn đề như hiệu suất, khả năng mở rộng và tính linh hoạt.

Mục đích của tiêu luận là giúp người đọc hiểu rõ về MongoDB, từ lịch sử hình thành, các khái niệm cơ bản, đến cách thức triển khai ứng dụng thực tế. Phạm vi nghiên cứu bao gồm việc tìm hiểu và so sánh MongoDB với các hệ cơ sở dữ liệu truyền thống, cũng như cách tích hợp MongoDB vào quy trình phát triển ứng dụng thông qua GitHub.

Đối tượng nghiên cứu của tiêu luận là nền tảng cơ sở dữ liệu MongoDB, cùng với các phương pháp triển khai và tích hợp MongoDB vào hệ thống ứng dụng web thông qua các công cụ quản lý mã nguồn như GitHub.

Trước khi có sự xuất hiện của các nền tảng cơ sở dữ liệu hiện đại như MongoDB, việc lưu trữ dữ liệu thường dựa vào các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS), vốn có nhiều hạn chế như khó khăn trong việc mở rộng và xử lý dữ liệu phi cấu trúc. MongoDB ra đời đã khắc phục được những hạn chế này, cho phép lưu trữ và truy xuất dữ liệu linh hoạt, đồng thời cung cấp khả năng mở rộng không giới hạn mà không làm giảm hiệu năng.

Trong phương pháp nghiên cứu, tiêu luận sẽ áp dụng phương pháp phân tích tài liệu, so sánh lý thuyết với thực tiễn, đồng thời sử dụng các thử nghiệm cụ thể với MongoDB trong việc triển khai các ứng dụng web. Thông qua việc xây dựng và thử nghiệm các ứng dụng mẫu, người đọc sẽ thấy rõ được cách thức vận hành và ứng dụng thực tế của MongoDB.

Dự kiến kết quả đạt được là qua tiêu luận này, người đọc sẽ có một hiểu biết đầy đủ về MongoDB và biết cách tích hợp nền tảng này vào quy trình phát triển ứng dụng web. Cụ thể, kết quả mong đợi bao gồm việc nắm bắt lý thuyết cơ bản, thực hiện các thao tác cài đặt, triển khai, và quản lý dữ liệu với MongoDB, từ đó có thể xây dựng và phát triển các sản phẩm web hoàn chỉnh.

Ao hoá và điện toán đám mây – Nhóm 3

Trong tiêu luận này, nhóm 3 sẽ giới thiệu các cơ sở lý thuyết liên quan đến MongoDB và cung cấp hướng dẫn chi tiết về cách triển khai MongoDB kết hợp GitHub để tạo ra các ứng dụng web thực tế. Thông qua các chương trình cụ thể, người đọc sẽ nắm bắt được quy trình sử dụng MongoDB từ cài đặt đến phát triển ứng dụng, giúp việc vận dụng công nghệ này trở nên dễ dàng hơn.

Các chương chính bao gồm:

- Chương 1: Cơ sở lý thuyết về MongoDB, bao gồm các khái niệm và môi trường sử dụng.
- Chương 2: Thực nghiệm và hướng dẫn triển khai MongoDB trong các ứng dụng thực tế.
- Chương 3: Đánh giá kết quả và các bài học kinh nghiệm khi sử dụng MongoDB.

Hy vọng rằng thông qua tiêu luận này, các bạn sinh viên sẽ nắm bắt được cách thức vận dụng MongoDB để phát triển các ứng dụng một cách hiệu quả và khoa học.

CHƯƠNG I. CƠ SỞ LÝ THUYẾT

I. GIỚI THIỆU

1. NoSQL là gì?

Thuật ngữ NoSQL (not only SQL) được giới thiệu lần đầu vào năm 1998 sử dụng làm tên gọi chung cho các lightweight open source relational database nhưng không sử dụng SQL cho truy vấn. Vào năm 2009, Eric Evans, nhân viên của Rackspace giới thiệu lại thuật ngữ NoSQL trong một hội thảo về cơ sở dữ liệu nguồn mở phân tán. Thuật ngữ NoSQL đánh dấu bước phát triển của thế hệ database mới: distributed (phân tán) và non-relational (không ràng buộc). Đây là 2 đặc tính quan trọng nhất.

2. Tại sao cần phải có NoSQL?

Quản lý dữ liệu theo thời gian thực: MongoDB cho phép chúng ta cung cấp các đề xuất thời gian thực, cá nhân hóa và cải thiện trải nghiệm người dùng nhờ vào khả năng lưu trữ và xử lý dữ liệu phi cấu trúc nhanh chóng. Ví dụ, một hệ thống thương mại điện tử có thể sử dụng MongoDB để lưu trữ và phân tích dữ liệu người dùng trong thời gian thực, từ đó đưa ra các đề xuất sản phẩm dựa trên hành vi mua sắm gần nhất của họ. Nhờ vào kiến trúc không cần bảng và các chỉ mục linh hoạt, MongoDB giúp tăng hiệu suất khi xử lý các luồng dữ liệu lớn và thay đổi nhanh chóng.

Bảo mật trên đám mây: MongoDB hỗ trợ triển khai trên đám mây với nhiều tính năng bảo mật mạnh mẽ, bao gồm mã hóa dữ liệu, xác thực người dùng, và phân quyền chi tiết. Điển hình là MongoDB Atlas, một dịch vụ cơ sở dữ liệu đám mây được quản lý hoàn toàn, cung cấp các tính năng bảo mật toàn diện. MongoDB cho phép theo dõi và kiểm soát các mối quan hệ phức tạp giữa dữ liệu, giúp các tổ chức dễ dàng phát hiện và ngăn chặn các mối đe dọa tiềm ẩn trong môi trường đám mây.

Các ứng dụng có độ sẵn sàng cao MongoDB: là một hệ thống cơ sở dữ liệu NoSQL phân tán, phù hợp để xây dựng các ứng dụng có độ sẵn sàng cao và độ trễ thấp. Với tính năng replication (sao lưu) và sharding (phân đoạn), MongoDB giúp các ứng dụng có thể mở rộng dễ dàng và đảm bảo hoạt động liên tục ngay cả khi một trong các node gặp sự cố. Chẳng hạn, một mạng xã hội có thể sử dụng MongoDB để lưu trữ và xử lý hàng triệu lượt tương tác từ người dùng mỗi ngày mà không gặp vấn đề về hiệu suất hoặc độ trễ.

3. NoSQL hoạt động như thế nào?

Cơ sở dữ liệu NoSQL sử dụng nhiều mô hình dữ liệu để truy cập và quản lý dữ liệu. Các loại cơ sở dữ liệu này được tối ưu hóa dành riêng cho các ứng dụng yêu

cấu mô hình dữ liệu linh hoạt, lượng dữ liệu lớn và độ trễ thấp, có thể đạt được bằng cách giảm bớt một số hạn chế về tính nhất quán của dữ liệu của các cơ sở dữ liệu quan hệ. Có sự khác biệt trong việc thực hiện dựa trên mô hình dữ liệu. Tuy nhiên, nhiều cơ sở dữ liệu NoSQL sử dụng **Javascript Object Notation (JSON)**, một định dạng trao đổi dữ liệu mở đại diện cho dữ liệu dưới dạng tập hợp các cặp tên-giá trị.

4. Phân loại NoSQL database

Document Database: Lưu trữ dữ liệu dưới dạng tài liệu, thường sử dụng định dạng JSON hoặc BSON. Đây là sự lựa chọn linh hoạt cho các ứng dụng có dữ liệu có cấu trúc thay đổi như blog, bài viết, và thông tin cá nhân. Một vài cơ sở dữ liệu Document là Amazon Simple DB, Couch DB, MongoDB...

Key-Value Database: Dữ liệu được tổ chức thành các cặp khóa-giá trị, với mỗi giá trị được liên kết với một khóa. Thích hợp cho việc lưu trữ và truy xuất dữ liệu đơn giản như cache, phiên làm việc, và quản lý phiên đăng nhập... Một vài ví dụ về cơ sở dữ liệu Key-value là Redis, Dynamo, Riak...

Wide-Column Database: Dữ liệu được tổ chức thành các cột thay vì các hàng, tạo ra cơ sở dữ liệu có khả năng mở rộng ngang tốt. Thường được sử dụng trong các ứng dụng đòi hỏi xử lý lớn như hệ thống phân tích và lưu trữ dữ liệu dòng thời gian. Một vài cơ sở dữ liệu Wide-Column là HBase, Cassandra, Hypertable...

Graph Database: Dữ liệu được biểu diễn dưới dạng đồ thị với các đỉnh và cạnh. Phù hợp cho việc xử lý dữ liệu liên kết như mạng xã hội, quản lý quan hệ, và phân tích mối quan hệ... Một vài ví dụ về cơ sở dữ liệu Graph là Neo4j, OrientDB, FlockDB...

In-memory Database: Là loại cơ sở dữ liệu lưu trữ toàn bộ dữ liệu trong bộ nhớ thay vì trên ổ đĩa, giúp cho việc truy cập dữ liệu nhanh hơn, tiện hơn so với việc truy xuất dữ liệu từ ổ đĩa thông thường. Tuy nhiên, hạn chế là nguy cơ mất dữ liệu cao trong trường hợp gặp sự cố máy chủ vì dữ liệu không được lưu trữ ở nơi khác. Một ví dụ về cơ sở dữ liệu In-memory là IBM solidDB, Hazelcast,...

5. So sánh giữa NoSQL và SQL

Đặc điểm	SQL	NoSQL
Ngôn ngữ truy vấn	Sử dụng SQL (Structured Query Language).	Không sử dụng SQL; sử dụng ngôn ngữ truy vấn linh hoạt tùy thuộc vào loại cơ sở dữ liệu (VD: MongoDB sử dụng ngôn ngữ truy vấn có cú pháp

		tương tự với JavaScript).
Cấu trúc dữ liệu	Dữ liệu có cấu trúc, dựa trên mô hình bảng và quan hệ.	Có thể ở dạng cấu trúc, bán cấu trúc, phi cấu trúc, đa hình...
Quy mô (Scaling)	Chiều Dọc (Vertical Scaling): Tăng cường tài nguyên trên một máy chủ duy nhất.	Chiều Ngang (Horizontal Scaling): Mở rộng bằng cách thêm nhiều máy chủ, chia nhỏ công việc và dữ liệu giữa chúng.
Khả năng mở rộng (Scalability)	Vertical Scaling giới hạn bởi khả năng tăng cường tài nguyên trên một máy chủ.	Horizontal Scaling linh hoạt hơn vì có thể thêm máy chủ để chia sẻ công việc và dữ liệu, đáp ứng nhanh chóng với sự gia tăng.
Ứng dụng phổ biến	Thích hợp cho các ứng dụng có dữ liệu có quan hệ, ví dụ: hệ thống quản lý cơ sở dữ liệu truyền thống.	Lựa chọn phù hợp cho các ứng dụng có yêu cầu linh hoạt và khả năng mở rộng cao, chẳng hạn như các ứng dụng web có lượng người dùng lớn hoặc dữ liệu không có cấu trúc.

6. MongoDB là gì? hình thành như nào?

MongoDB là một trong những cơ sở dữ liệu NoSQL phổ biến nhất, là một cơ sở dữ liệu hướng tài liệu có mã nguồn mở. Thuộc loại cơ sở dữ liệu không quan hệ (NoSQL), MongoDB không dựa trên cấu trúc cơ sở dữ liệu quan hệ giống như bảng mà sẽ lưu trữ và truy xuất dữ liệu ở dạng document. Định dạng dữ liệu mới mà MongoDB sử dụng là BSON (khá giống với định dạng JSON). (What is MongoDB?, n.d.)

Khởi đầu: MongoDB bắt đầu được phát triển vào năm 2007 bởi công ty phần mềm 10gen, có trụ sở tại New York. Ban đầu, 10gen phát triển MongoDB như một phần của nền tảng dịch vụ PaaS (Platform as a Service) tương tự như Microsoft Azure.

Ra mắt: MongoDB được phát hành vào năm 2009 được viết bằng ngôn ngữ C++, C++ là ngôn ngữ gần với ngôn ngữ máy nên dễ dàng hiểu rằng MongoDB có thể

tính toán ở tốc độ cao hơn hẳn các hệ quản trị cơ sở dữ liệu khác. Đó cũng là lý do mà MongoDB luôn được các nhà phát triển đánh giá rất cao.

Phát triển và mở rộng: Sau khi ra mắt, MongoDB nhanh chóng trở nên phổ biến nhờ vào khả năng mở rộng và tính linh hoạt cao. Năm 2010, phiên bản 1.4 của MongoDB được phát hành, đánh dấu sự sẵn sàng của sản phẩm cho các ứng dụng thực tế.

Các phiên bản quan trọng:

- Năm 2014, phiên bản 2.4.9 được phát hành.
- Năm 2015, phiên bản 3.0 ra mắt, tiếp theo là phiên bản 3.2 vào cuối năm 2015, đi kèm với công cụ quản trị đồ họa MongoDB Compass.

MongoDB Atlas: Năm 2016, MongoDB Inc. giới thiệu MongoDB Atlas, một dịch vụ cơ sở dữ liệu đám mây, giúp người dùng dễ dàng triển khai và quản MongoDB trên các nền tảng đám mây như AWS, Microsoft Azure và Google Cloud Platform.

Hiện tại: MongoDB tiếp tục phát triển và cải tiến, trở thành một trong những cơ sở dữ liệu NoSQL phổ biến nhất, được sử dụng rộng rãi bởi nhiều công ty lớn trên toàn thế giới. Phiên bản mới nhất là 8.0.0.

MongoDB nhanh chóng trở thành một trong những cơ sở dữ liệu NoSQL phổ biến nhất, được sử dụng rộng rãi bởi các công ty lớn như Facebook, Google, và eBay.

MongoDB hỗ trợ đa nền tảng theo hướng đối tượng dùng lưu trữ dữ liệu có cấu trúc phức tạp. Không như các hệ quản trị cơ sở dữ liệu khác lưu trữ ở dạng bảng, MongoDB lưu trữ dữ liệu vào collection theo hướng tài liệu JSON. Đây là kiểu dữ liệu dạng Key-Value truy xuất nhanh và khả năng mở rộng không bị ràng buộc mới tạo khóa ngoại hay khóa chính.

7. Tại sao tạo ra MongoDB? mục đích tạo ra MongoDB?

MongoDB được tạo ra để giải quyết một số hạn chế của các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) truyền thống, đặc biệt là khi làm việc với các ứng dụng hiện đại yêu cầu khả năng mở rộng và linh hoạt cao. Dưới đây là một số mục đích chính khi tạo ra MongoDB:

- **Khả năng mở rộng:** MongoDB được thiết kế để dễ dàng mở rộng theo chiều ngang, nghĩa là có thể thêm nhiều máy chủ hơn để xử lý khối lượng dữ liệu lớn mà không gặp phải các vấn đề về hiệu suất.
- **Linh hoạt trong lưu trữ dữ liệu:** Thay vì lưu trữ dữ liệu trong các bảng và hàng như RDBMS, MongoDB lưu trữ dữ liệu dưới dạng tài liệu JSON, cho phép lưu trữ các cấu trúc dữ liệu phức tạp và thay đổi linh hoạt.
- **Hiệu suất cao:** MongoDB hỗ trợ các chỉ mục đa dạng và các truy vấn phức tạp, giúp tăng hiệu suất truy xuất dữ liệu.

- Dễ dàng tích hợp với các ứng dụng hiện đại: Với cấu trúc dữ liệu JSON, MongoDB dễ dàng tích hợp với các ứng dụng web và di động, nơi dữ liệu thường được trao đổi dưới dạng JSON.
- Khả năng xử lý dữ liệu lớn: MongoDB được thiết kế để xử lý các khối lượng dữ liệu lớn và không đồng nhất, phù hợp với các ứng dụng Big Data và phân tích dữ liệu. (mongodb get started, n.d.)

8. Nếu không có MongoDB có thể thay thế bởi cái gì?

Nếu không sử dụng MongoDB, chúng ta có thể thay thế bằng một số cơ sở dữ liệu NoSQL và SQL khác, tùy thuộc vào nhu cầu cụ thể của người dùng. Dưới đây là một số lựa chọn thay thế phổ biến:

- Apache Cassandra: Đây là một cơ sở dữ liệu NoSQL phân tán, được thiết kế để xử lý lượng dữ liệu lớn với khả năng mở rộng cao và độ tin cậy tốt.
- Redis: Một cơ sở dữ liệu NoSQL lưu trữ dữ liệu dưới dạng key-value, nổi tiếng với tốc độ truy xuất nhanh và hỗ trợ nhiều cấu trúc dữ liệu khác nhau.
- Amazon DynamoDB: Dịch vụ cơ sở dữ liệu NoSQL được quản lý hoàn toàn bởi
- Amazon Web Services (AWS), cung cấp khả năng mở rộng tự động và hiệu suất cao.
- CouchDB: Một cơ sở dữ liệu NoSQL sử dụng JSON để lưu trữ dữ liệu, JavaScript để truy vấn và MapReduce để xử lý dữ liệu.
- PostgreSQL: Một cơ sở dữ liệu quan hệ mã nguồn mở mạnh mẽ, hỗ trợ nhiều tính năng tiên tiến và có khả năng mở rộng tốt.
- ArangoDB: Một cơ sở dữ liệu đa mô hình, hỗ trợ cả tài liệu, đồ thị và key-value, giúp linh hoạt trong việc lưu trữ và truy vấn dữ liệu.

9. Ý nghĩa của MongoDB đối với ngành công nghệ thông tin

MongoDB có ý nghĩa rất lớn đối với ngành công nghệ thông tin, đặc biệt trong bối cảnh các ứng dụng hiện đại yêu cầu khả năng xử lý dữ liệu linh hoạt và hiệu quả. Dưới đây là một số điểm nổi bật về ý nghĩa của MongoDB:

- **Khả năng mở rộng và hiệu suất cao:** MongoDB cho phép mở rộng theo chiều ngang, giúp các doanh nghiệp dễ dàng quản lý và xử lý khối lượng dữ liệu lớn mà không gặp phải các vấn đề về hiệu suất.
- **Linh hoạt trong lưu trữ dữ liệu:** Với cấu trúc dữ liệu dạng JSON, MongoDB cho phép lưu trữ các cấu trúc dữ liệu phức tạp và thay đổi linh hoạt, phù hợp với các ứng dụng web và di động hiện đại.
- **Hỗ trợ phát triển ứng dụng nhanh chóng:** MongoDB giúp các nhà phát triển dễ dàng tích hợp và triển khai các ứng dụng, giảm thiểu thời gian và công sức so với việc sử dụng các cơ sở dữ liệu quan hệ truyền thống.

- Phù hợp với các ứng dụng Big Data và phân tích dữ liệu: MongoDB được thiết kế để xử lý các khối lượng dữ liệu lớn và không đồng nhất, giúp các doanh nghiệp khai thác và phân tích dữ liệu hiệu quả hơn.
- Cộng đồng và hệ sinh thái phát triển mạnh mẽ: Với một cộng đồng lớn và nhiều tài liệu hỗ trợ, MongoDB cung cấp nhiều tài nguyên và công cụ giúp các nhà phát triển dễ dàng học hỏi và áp dụng.
- Những yếu tố này đã giúp MongoDB trở thành một công cụ quan trọng trong ngành công nghệ thông tin, được sử dụng rộng rãi bởi nhiều công ty lớn như Facebook, Google, và eBay.

II. CÁC KHÁI NIỆM LIÊN QUAN

1. Các thành phần cốt lõi tạo nên MongoDB

Mongod: tiến trình lõi của MongoDB, nơi lưu trữ và quản lý dữ liệu.

Mongos: controller và query router, dùng cho trường hợp phân tán dữ liệu (sharded cluster).

Mongosh: giao diện shell để người dùng có thể tương tác thực hiện lệnh với cơ sở dữ liệu. Có thể dùng Mongosh để chạy kiểm tra các lệnh truy vấn và thao tác tùy biến dữ liệu lưu trong cơ sở dữ liệu.

Để khởi động MongoDB mặc định trên localhost, chạy lệnh mongosh trên command line. Tiến trình mongod khởi động, và mongosh sẽ kết nối với mongod và chạy trên localhost:27017. Lệnh mongosh tương đương với câu lệnh mongosh "mongodb://localhost:27017".

2. JSON

Viết tắt của JavaScript Object Notation. Con người có thể đọc được ở định dạng văn bản đơn giản thể hiện cho các dữ liệu có cấu trúc. Hiện tại JSON đang hỗ trợ rất nhiều ngôn ngữ lập trình.

3. Document

Trong MongoDB, một bản ghi được gọi là một document. Mỗi document chứa một cặp giá trị gồm trường dữ liệu và giá trị tương ứng của trường đó trong document. Document về hình thức tương tự như dữ liệu kiểu JSON. Giá trị của một trường có thể là số, văn bản, mảng, một document khác hoặc một mảng các document.

4. Collection

MongoDB lưu trữ các document trong collection. Collection trong MongoDB tương đương với bảng trong mô hình quan hệ. Để tạo một collection mới trong cơ sở dữ liệu, sử dụng phương thức **db.createCollection()**.

5. BSON

Là một định dạng dữ liệu tuân tự hóa dạng nhị phân ược sử dụng để lưu trữ document và thực hiện các cuộc gọi thủ tục từ xa trong MongoDB. Có nhiều kiểu dữ liệu BSON, có thể được sử dụng số hoặc chuỗi định danh để quy định kiểu dữ liệu cho một đối tượng.

6. MongoDB Compass

Là một giao diện GUI cho phép người dùng tương tác với mongod một cách trực quan thay vì dùng giao diện commandline. MongoDB Compass hiển thị các cơ sở dữ liệu, collection, document một cách trực quan, hỗ trợ truy vấn, thực hiện aggregating function và phân tích dữ liệu MongoDB.

7. MongoDB Database Tools

Là tập hợp các tiện ích dạng commandline được thiết lập sẵn, dùng để thực hiện các thao tác đặc biệt với MongoDB như nhập xuất, sao lưu, khôi phục dữ liệu (mongodump và mongorestore).

8. DataBase

Một mongod có thể chứa nhiều cơ sở dữ liệu. Một cơ sở dữ liệu chứa một hoặc nhiều collection.

Để chọn một cơ sở dữ liệu và tiến hành các thao tác với dữ liệu với cơ sở dữ liệu đó, trong mongosh dùng lệnh **use <tên cơ sở dữ liệu>**.

Để tạo một cơ sở dữ liệu mới trong mongod, trong trường hợp cơ sở dữ liệu không tồn tại, MongoDB sẽ tự động tạo cơ sở dữ liệu mới khi lưu trữ document lần đầu tiên cho cơ sở dữ liệu đó. Do đó, có thể dùng lệnh use để chuyển sang cơ sở dữ liệu không tồn tại và thực hiện thao tác thêm document mới trong collection mới trong mongosh. MongoDB sẽ tạo cơ sở dữ liệu mới kèm theo collection kèm document được chỉ định. Ví dụ:

use myNewDB

db.myNewCollection1.insertOne({ x: 1 })

III. MÔI TRƯỜNG SỬ DỤNG

1. Sử dụng trên máy chủ vật lý

Trường hợp nên sử dụng trên máy chủ vật lý:

Các doanh nghiệp có yêu cầu bảo mật cao và muốn giữ toàn quyền kiểm soát dữ liệu.

Các tổ chức đã có sẵn cơ sở hạ tầng vật lý và không muốn đầu tư vào đám mây.

2. Sử dụng trên môi trường đám mây

Trường hợp nên sử dụng trên môi trường đám mây (AWS, Azure, Google Cloud, ...):

Các doanh nghiệp muốn tận dụng tính linh hoạt và khả năng mở rộng của đám mây mà không muốn đầu tư nhiều vào phần cứng.

Ứng dụng yêu cầu triển khai trên quy mô lớn và có sự biến động về khối lượng dữ liệu.

3. Sử dụng MongoDB Atlas (được MongoDB Inc quản lý)

Trường hợp nên sử dụng MongoDB Atlas:

Các doanh nghiệp không muốn quản lý cơ sở hạ tầng hoặc không có đội ngũ kỹ thuật để quản lý MongoDB.

Ứng dụng yêu cầu tính sẵn sàng và khả năng mở rộng cao, với chi phí được cân nhắc kỹ lưỡng.

Các công ty cần triển khai nhanh chóng mà không cần quản lý phức tạp.

IV. CÁC TÍNH NĂNG CƠ BẢN

1. Giới thiệu về mô hình dữ liệu

Thách thức chính trong mô hình hóa dữ liệu là cân bằng giữa nhu cầu của ứng dụng, đặc điểm về hiệu suất của hệ quản trị cơ sở dữ liệu và các mẫu truy xuất dữ liệu. Khi thiết kế mô hình dữ liệu, cần phải luôn cân nhắc về việc sử dụng dữ liệu trong ứng dụng (bao gồm các truy vấn, cập nhật và xử lý dữ liệu) cũng như cấu trúc vốn có của chính dữ liệu đó.

2. Tính mềm dẻo

Khác với các cơ sở dữ liệu SQL, yêu cầu phải xác định và khai báo lược đồ của bảng trước khi thêm dữ liệu, các Collection của MongoDB không yêu cầu các document phải có cùng một mô hình dữ liệu: không nhất thiết phải có cùng một nhóm các trường, và kiểu dữ liệu cho một trường cũng có thể khác nhau giữa các document trong bộ dữ liệu. Để thay đổi cấu trúc của document trong Collection, chỉ cần cập nhật document theo cấu trúc dữ liệu mới.

Tính linh hoạt này tạo điều kiện thuận lợi cho việc ánh xạ các document tới một thực thể hoặc một đối tượng. Mỗi document có thể khớp với các trường dữ liệu của thực thể được đại diện, ngay cả khi bản ghi có sự thay đổi đáng kể so với các document khác trong Collection. Tuy nhiên trong thực tế, các document trong cùng một Collection thường có cấu trúc tương tự nhau. Đồng thời, người ta cũng sử dụng các quy tắc xác thực mô hình dữ liệu khỏi với quá trình thêm và sửa dữ liệu.

3. Các cấu trúc của mô hình dữ liệu

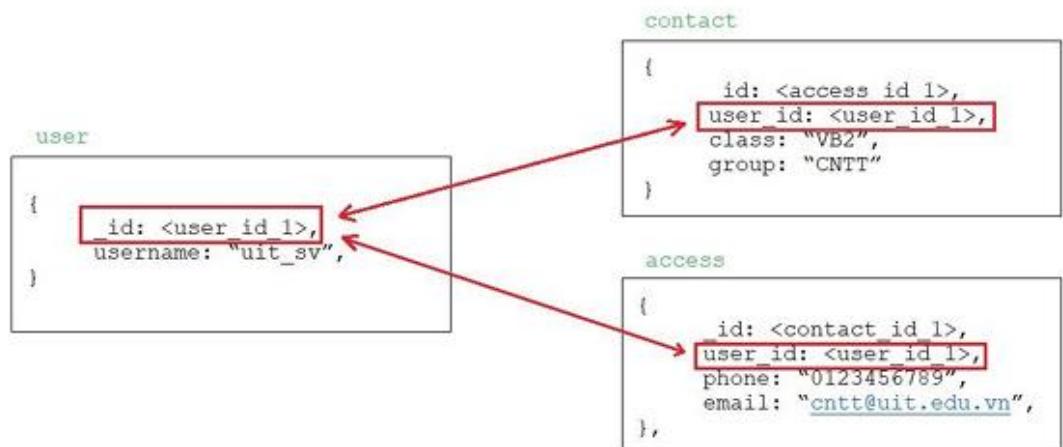
Vấn đề quan trọng trong việc thiết kế mô hình dữ liệu cho các ứng dụng sử dụng MongoDB xoay quanh cấu trúc của các Collection và cách ứng dụng thể hiện mối quan hệ giữa các dữ liệu.

- **Nhúng dữ liệu:** MongoDB cho phép nhúng tất cả dữ liệu liên quan trong một document duy nhất. Dữ liệu được nhúng thể hiện mối quan hệ giữa các dữ liệu bằng cách lưu trữ dữ liệu liên quan trong một cấu trúc tài liệu duy nhất. MongoDB hỗ trợ việc nhúng các cấu trúc dữ liệu vào một trường hoặc mảng trong document. Các mô hình dữ liệu không “chuẩn hóa” này cho phép các ứng dụng truy xuất và thao tác dữ liệu liên quan trong một thao tác cơ sở dữ liệu duy nhất. Ví dụ:

```
{
    _id: <user_id_1>,
    username: "uit_sv",
    contact: {
        phone: "0123456789",
        email: "cntt@uit.edu.vn",
    },
    access: {
        class: "VB2",
        group: "CNTT"
    }
}
```

Hình 1. Nhúng dữ liệu

- **Tham chiếu:** Còn được gọi là kiến trúc chuẩn hóa của mô hình dữ liệu. Các tham chiếu lưu trữ các mối quan hệ giữa các dữ liệu bằng cách thêm các liên kết hoặc tham chiếu giữa các document. Các ứng dụng có thể thông qua các tham chiếu này để truy cập dữ liệu liên quan. Đây là một dạng mô hình dữ liệu chuẩn hóa. Ở ví dụ dưới đây, trường “user_id” trong bộ dữ liệu “contact” và “access” được dùng để tham chiếu đến user có liên quan.



Hình 2. Tham chiếu

4. Mô hình quan hệ giữa các document

a. Quan hệ một – một

Xét ví dụ xây dựng sơ đồ mối quan hệ một – một giữa sinh viên và địa chỉ của sinh viên đó. Trong mô hình dữ liệu chuẩn hóa, document “DiaChi” sẽ chứa tham chiếu đến document “SinhVien”. Nếu việc sử dụng ứng dụng cần thường xuyên truy xuất dữ liệu địa chỉ cùng với dữ liệu sinh viên, thì với việc dùng tham chiếu sẽ đòi hỏi đưa ra nhiều truy vấn hơn để giải quyết tham

chiếu. Mô hình tốt hơn sẽ là nhúng dữ liệu “DiaChi” vào trong dữ liệu “SinhVien” như sau:

```
{  
    _id: "sv_uit_1",      name: "Nguyen Van A",      address: {      city:  
        "HCM",          district: "Thu Duc",          ward: "Linh Trung"  
    }  
}
```

Hình 3. Quan hệ một – một

b. Quan hệ một – nhiều sử dụng nhúng dữ liệu

Mô hình dữ liệu quan hệ một – nhiều sử dụng phương pháp nhúng dữ liệu được sử dụng trong trường hợp cần truy xuất nhiều thực thể của một dữ liệu nằm trong một dữ liệu khác. Tiếp tục với ví dụ ở trên, ta xét trong trường hợp một sinh viên có thẻ có nhiều địa chỉ liên hệ. Nếu nhu cầu truy xuất thông tin địa chỉ đồng thời với dữ liệu sinh viên xảy ra thường xuyên, thì phương pháp tối ưu hơn là nhúng các thực thể của dữ liệu địa chỉ vào trong dữ liệu sinh viên.

```
{  
    _id: "sv_uit_1",      name:  
    "Nguyen Van A",      address:  
    [      {      city:  
        "HCM",          district:  
        "Thu Duc",          ward:  
        "Linh Trung"  
    },      {  
        city: "HCM",  
        district: "Phu Nhuan",  
        ward: "Phuong 2"  
    }  
]
```

Hình 4. Quan hệ một – nhiều sử dụng nhúng dữ liệu

c. Quan hệ một – nhiều sử dụng tham chiếu

Xét ví dụ về mô hình dữ liệu thể hiện quan hệ giữa sinh viên và môn học đã đăng ký. Vì một môn học có rất nhiều sinh viên đăng ký, nên việc nhúng dữ liệu môn học vào trong dữ liệu sinh viên có thể dẫn đến việc lặp lại dữ liệu của môn học, gây lãng phí tài nguyên hệ thống. Để tránh việc lưu trữ lặp lại dữ liệu của môn học, có thể tạo ra một Collection riêng để chứa dữ liệu môn học. Cần lưu ý trong trường hợp này, một môn học sẽ có rất nhiều sinh viên đăng ký, nên việc để tham chiếu từ dữ liệu môn học sang dữ liệu sinh viên sẽ dẫn đến lưu trữ một mảng rất lớn:

```
// MonHoc
{
    _id: "mon_hoc_1",      name: "Quan ly thong tin",      students:
    ["sv_uit_1", "sv_uit_2", "sv_uit_3" ...]
}
```

Hình 5. Quan hệ một – nhiều sử dụng tham chiếu

Nên cân nhắc để tham chiếu đến dữ liệu môn học từ dữ liệu sinh viên như sau:

```
// MonHoc
{
    _id: "mon_hoc_1",
    name: "Quan ly thong tin",
}

// SinhVien
{
    _id: "sv_uit_1",
    name: "Nguyen Van A",
    class_id: "mon_hoc_1"
}

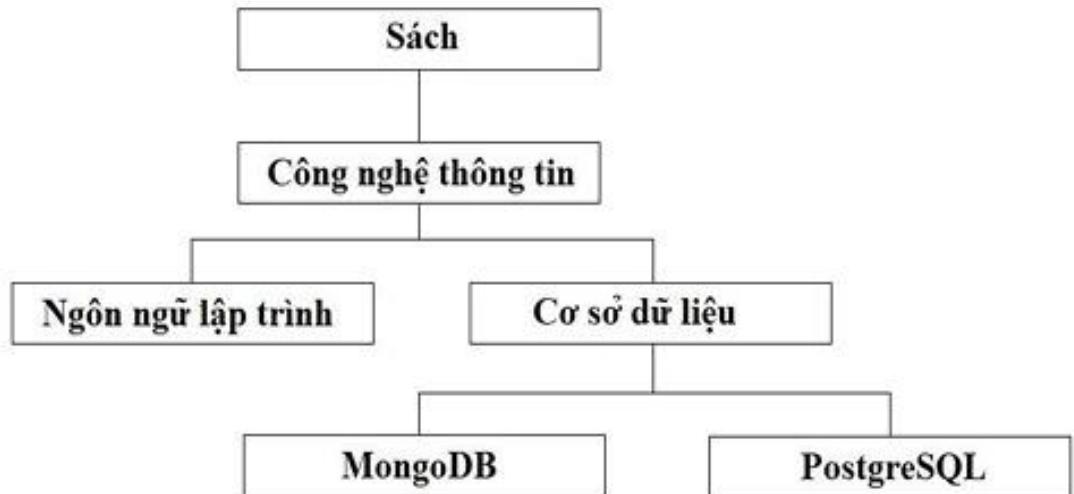
{
    _id: "sv_uit_2",
    name: "Nguyen Van B",
    class_id: "mon_hoc_1"
}
```

Hình 6. Cân nhắc tham chiếu khác

5. Mô hình cây

a. Sử dụng tham chiếu cha

Xét ví dụ về hệ thống danh mục sách được mô tả trong cấu trúc cây như sau:



Hình 7. Ví dụ hệ thống danh mục sách ược mô tả theo cấu trúc cây

```
{ _id: "MongoDB", parent: "CoSoDuLieu" }
{ _id: "PostgreSQL", parent: "CoSoDuLieu" }
{ _id: "CoSoDuLieu", parent: "CongNgheThongTin" }
{ _id: "NgonNguLapTrinh", parent: "CongNgheThongTin" } { _id: "CongNgheThongTin", parent: "Sach" }
{ _id: "Sach", parent: null }
```

Hình 8. Sử dụng tham chiếu cha

b. Sử dụng tham chiếu con

```
{ _id: "CoSoDuLieu", child: ["MongoDB", "PostgreSQL"] }
{ _id: "NgonNguLapTrinh", child: [] }
{ _id: "CongNgheThongTin", child: ["CoSoDuLieu", "NgonNguLapTrinh"] }
{ _id: "Sach", child: ["CongNgheThongTin"] }
```

Hình 9. Sử dụng tham chiếu con

c. Sử dụng danh sách gốc

```
{_id: "MongoDB", ancestors: ["Sach", "CongNgheThongTin", "CoSoDuLieu"], parent: "CoSoDuLieu"}
{_id: "PostgreSQL", ancestors: ["Sach", "CongNgheThongTin", "CoSoDuLieu"], parent: "CoSoDuLieu"}
{_id: "CoSoDuLieu", ancestors: ["Sach", "CongNgheThongTin"], parent:"CongNgheThongTin"}
{_id:"NgonNguLapTrinh", ancestors: ["Sach", "CongNgheThongTin"], parent:"CongNgheThongTin"}
{_id:"CongNgheThongTin", ancestors: ["Sach"], parent: "Sach"}
{_id: "Sach", ancestors: [], parent: null}
```

Hình 10. Sử dụng danh sách gốc

d. Sử dụng đường dẫn

Áo hóa và điện toán đám mây – Nhóm 3

```
{_id: "MongoDB", path: ",Sach,CongNgheThongTin,CoSoDuLieu,"}  
{_id: "PostgreSQL", path: ",Sach,CongNgheThongTin,CoSoDuLieu,"}  
{_id: "CoSoDuLieu", path: ",Sach,CongNgheThongTin,"}  
({_id:"NgonNguLapTrinh", path: ",Sach,CongNgheThongTin"}  
({_id:"CongNgheThongTin", path: ",Sach,"}  
{_id: "Sach", path: null})
```

Hình 11. Sử dụng đường dẫn

6. Các phương thức CRUD trong MongoDB

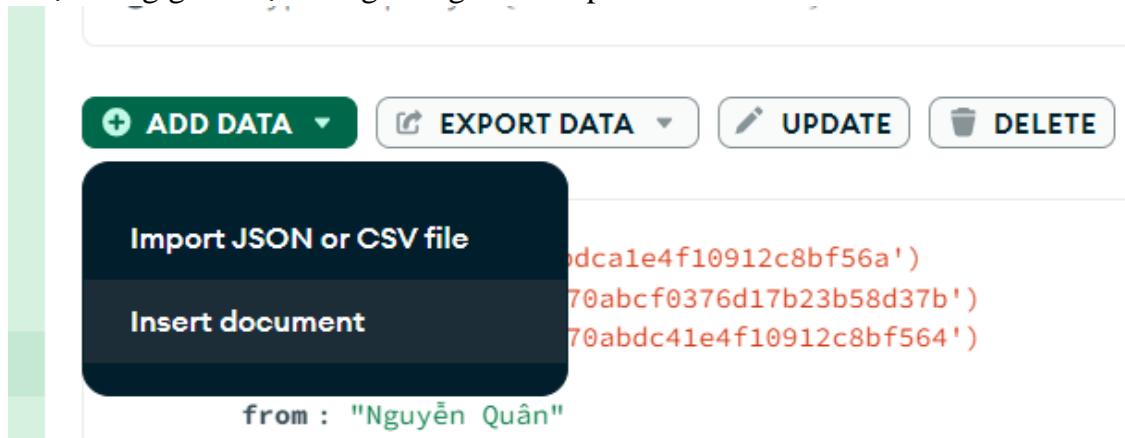
a. Thêm dữ liệu

Cú pháp db.collection.insertOne() được sử dụng để thêm một document vào Collection.

```
quanlythongtin_demo> db.students.insertOne({ name: "uit_sv", year: Int32(2020), major: "CNTT", gpa: Double(3.5), acknowledged: true, insertedId: ObjectId("634c01f7a6e48f56aa482685") })  
quanlythongtin_demo>
```

Hình 12. Dùng phương thức db.collection.insertOne() để thêm một sinh viên vào collection students

Hoặc dùng giao diện trong mongodb compass



Hình 15. giao diện insert tại compass

Câu lệnh trong ví dụ trên đã thêm thông tin của một sinh viên mới vào Collection “students”. Vì trong document không khai báo trường “_id”, MongoDB sẽ tự động gán một giá trị ObjectId cho trường này. Hàm **insertOne()** trả về kết quả thực thi thêm dữ liệu và giá trị của trường “_id” của document vừa được thêm vào.

Khi cần thêm nhiều document cùng một lúc, sử dụng cú pháp **db.collection.insertMany()** và truyền vào một mảng gồm các document.

```
quanlythongtin_demo> db.students.insertMany([
... { name: "ult_sv_1", year: Int32(2020), major: "CNTT", gpa: Double(3.3), address: { city: "HCM", street: "Duong so 1" } },
... { name: "ult_sv_2", year: Int32(2020), major: "CNTT", gpa: Double(3.7), address: { city: "HCM", street: "Duong so 2" } },
... { name: "ult_sv_3", year: Int32(2020), major: "CNTT", gpa: Double(3.6), address: { city: "HCM", street: "Duong so 3" } }
...
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("634c0659a6e48f56aa482686"),
    '1': ObjectId("634c0659a6e48f56aa482687"),
    '2': ObjectId("634c0659a6e48f56aa482688")
  }
}
quanlythongtin_demo>
```

Hình 13. Dùng phương thức `db.collection.insertMany()` để thêm các sinh viên vào collection `students`

Khi sử dụng câu lệnh `insert` để thêm dữ liệu, nếu thêm dữ liệu vào một Collection không tồn tại, thao tác thêm dữ liệu sẽ tự động tạo mới Collection. MongoDB cũng yêu cầu mỗi document cần có trường “`_id`” làm khóa chính. Nếu trong câu lệnh `insert` chưa khai báo trường này, thao tác thêm dữ liệu sẽ tự động tạo ra một giá trị `ObjectId` để gán cho “`_id`”, và trả về sau khi thêm dữ liệu thành công.

Ngoài ra, việc thêm dữ liệu cũng có thể được thực hiện bởi hàm `db.collection.bulkWrite()` hoặc một số hàm cập nhật, chỉnh sửa dữ liệu với tùy chọn `upsert=true`:

- `db.collection.updateOne()`
- `db.collection.updateMany()`
- `db.collection.findAndModify()`
- `db.collection.findOneAndUpdate()`
- `db.collection.findOneAndReplace()`

b. Truy vấn dữ liệu

Truy vấn tất cả document trong Collection: Sử dụng cú pháp `db.collection.find()` và truyền vào tham số là một đối tượng JSON rỗng {}.

```
quanlythongtin_demo> db.students.find()
[
  {
    _id: ObjectId("634c01f7a6e48f56aa482685"),
    name: 'uit_sv',
    year: 2020,
    major: 'CNTT',
    gpa: 3.8,
    address: { city: 'HCM', street: 'Dien Bien Phu' }
  },
  {
    _id: ObjectId("634c0659a6e48f56aa482686"),
    name: 'uit_sv_1',
    year: 2020,
    major: 'CNTT',
    gpa: 3.3,
    address: { city: 'HCM', street: 'Duong so 1' }
  },
  {
    _id: ObjectId("634c0659a6e48f56aa482687"),
    name: 'uit_sv_2',
    year: 2020,
    major: 'CNTT',
    gpa: 3.7,
    address: { city: 'HCM', street: 'Duong so 2' }
  },
  {
    _id: ObjectId("634c0659a6e48f56aa482688"),
    name: 'uit_sv_3',
    year: 2020,
    major: 'CNTT',
    gpa: 3.6,
    address: { city: 'HCM', street: 'Duong so 3' }
  }
]
```

Hình 14. Dùng phương thức `db.collection.find()` để lấy toàn bộ thông tin trong collection `students`

Sử dụng điều kiện bằng: Xác định các điều kiện tìm kiếm dưới dạng các cặp `<field>:<value>`, thể hiện dưới định dạng JSON và truyền vào hàm `db.collection.find()`.

Sử dụng toán tử truy vấn: MongoDB hỗ trợ tìm kiếm và lọc dữ liệu bằng các toán tử so sánh, logic, điều kiện.

Sử dụng điều kiện “AND” hoặc “OR: Ví dụ tìm kiếm tất cả sinh viên khoa CNTT có GPA > 3.7 hoặc < 3.6.

```
quanlythongtin_demo> db.students.find({ major: "CNTT", $or: [{gpa: {$gt: 3.7}}, {gpa: {$lt: 3.6}}] })
[
  {
    _id: ObjectId("634c01f7a6e48f56aa482685"),
    name: 'uit_sv',
    year: 2020,
    major: 'CNTT',
    gpa: 3.8,
    address: { city: 'HCM', street: 'Dien Bien Phu' }
  },
  {
    _id: ObjectId("634c0659a6e48f56aa482686"),
    name: 'uit_sv_1',
    year: 2020,
    major: 'CNTT',
    gpa: 3.3,
    address: { city: 'HCM', street: 'Duong so 1' }
  }
]
```

Hình 15. Dùng phương thức `db.collection.find()` để tìm sinh viên thỏa iều kiện cho trước

Thay thế giá trị trên một document: Sử dụng db.collection.replaceOne() để thay thế thông tin của sinh viên có tên “uit_sv”.

```
quanlythongtin_demo> db.students.replaceOne({  
... { name: "uit_sv" },  
... { name: "uit_sv_0", year: Int32(2020), major: "CNSP", gpa: Double(3.8), address: { city: "TP.HCM", street: "Hoàng Sa" } }  
... }  
acknowledged: true,  
insertedId: null,  
matchedCount: 1,  
modifiedCount: 1,  
upsertedCount: 0  
]  
  
quanlythongtin_demo> db.students.find({})  
{  
  {  
    _id: ObjectId("634c01f7a6e48f56aa482685"),  
    name: "uit_sv_0",  
    year: 2020,  
    major: "CNSP",  
    gpa: 3.8,  
    address: { city: "TP.HCM", street: "Hoàng Sa" }  
  },  
  {  
    _id: ObjectId("634c0659a6e48f56aa482686"),  
    name: "uit_sv_1",  
    year: 2020,  
    major: "CNTT",  
    gpa: 3.3,  
    address: { city: "TP.HCM", street: "Đường số 1" }  
  },  
  {  
    _id: ObjectId("634c0659a6e48f56aa482687"),  
    name: "uit_sv_2",  
    year: 2022,  
    major: "IOHT",  
    gpa: 3.7,  
    address: { city: "HCM", street: "Đường số 2" },  
    lastModified: ISODate("2022-10-16T16:35:47.679Z")  
  },  
  {  
    _id: ObjectId("634c0659a6e48f56aa482688"),  
    name: "uit_sv_3",  
    year: 2020,  
    major: "CNTT",  
    gpa: 3.6,  
    address: { city: "TP.HCM", street: "Đường số 3" }  
  }  
}
```

Hình 18. Dùng db.collection.replaceOne() thay thế thông tin của sinh viên có tên “uit_sv”

Hoặc dùng giao diện web của compass



Hình 19. Giao diện update

The screenshot shows the MongoDB update interface. At the top, it says "Update 7 documents". Below that is a "test.comments" collection. A "Filter" dropdown is set to "None". The "Update" section contains a code editor with the following query:

```
1 < [  
2 <   $set: {  
3 <     }  
4 <   },  
5 < }
```

Below the code editor is a "Preview" section showing two sample documents:

```
_id: ObjectId('670abdc1e4f10912c8bf56a')  
userId: ObjectId('670abcf0376d17b23b58d37b')  
postId: ObjectId('670abdc1e4f10912c8bf564')  
comment: "cm1"  
from: "Nguyễn Quân"  
> likes: Array  
> replies: Array  
createdAt: 2024-10-12T18:19:54.779+00:00  
updatedAt: 2024-10-12T18:20:00.627+00:00  
__v: 1
```

```
_id: ObjectId('670b7a0e5dab9a747bdcc2c84')  
userId: ObjectId('670b641aa58318609932be78')  
postId: ObjectId('670b79915dab9a747bdcc2c60')  
comment: "test server"  
from: "Bảo Nguyễn"  
> likes: Array  
> replies: Array  
createdAt: 2024-10-13T07:43:10.937+00:00  
updatedAt: 2024-10-13T07:43:10.937+00:00  
__v: 1
```

At the bottom are "Save", "Cancel", and "Update 7 documents" buttons.

Hình 20. Giao diện update

d. Xoá dữ liệu

Xóa tất cả document trong Collection: Sử dụng `db.collection.deleteMany()`, truyền vào tham số là một đối tượng JSON rỗng.

```
quanlythongtin_demo> db.students.deleteMany({})  
{ acknowledged: true, deletedCount: 4 }  
quanlythongtin_demo> db.students.find({})  
  
quanlythongtin_demo>
```

Hình 21. Dùng phương thức `db.collection.deleteMany()` xóa tất cả document trong Collection

Xóa tất cả document thỏa mãn điều kiện: Sử dụng `db.collection.deleteMany()`, với tham số truyền vào thể hiện điều kiện để tìm kiếm document thực thi lệnh xóa. Ví dụ xóa dữ liệu các sinh viên có GPA < 3.7 ra khỏi Collection students.

Áo hóa và điện toán đám mây – Nhóm 3

```
quanlythongtin_demo> db.students.deleteMany({gpa: {$lt: 3.7}})
{ acknowledged: true, deletedCount: 2 }
quanlythongtin_demo> db.students.find({})
[
  {
    _id: ObjectId("634c01f7a6e48f56aa482685"),
    name: 'uit_sv_0',
    year: 2020,
    major: 'CNTT',
    gpa: 3.8,
    address: { city: 'TP.HCM', street: 'Hoàng Sa' }
  },
  {
    _id: ObjectId("634c0659a6e48f56aa482687"),
    name: 'uit_sv_2',
    year: 2022,
    major: 'KHMT',
    gpa: 3.7,
    address: { city: 'HCM', street: 'Đường số 2' },
    lastModified: ISODate("2022-10-16T16:35:47.679Z")
  }
]
```

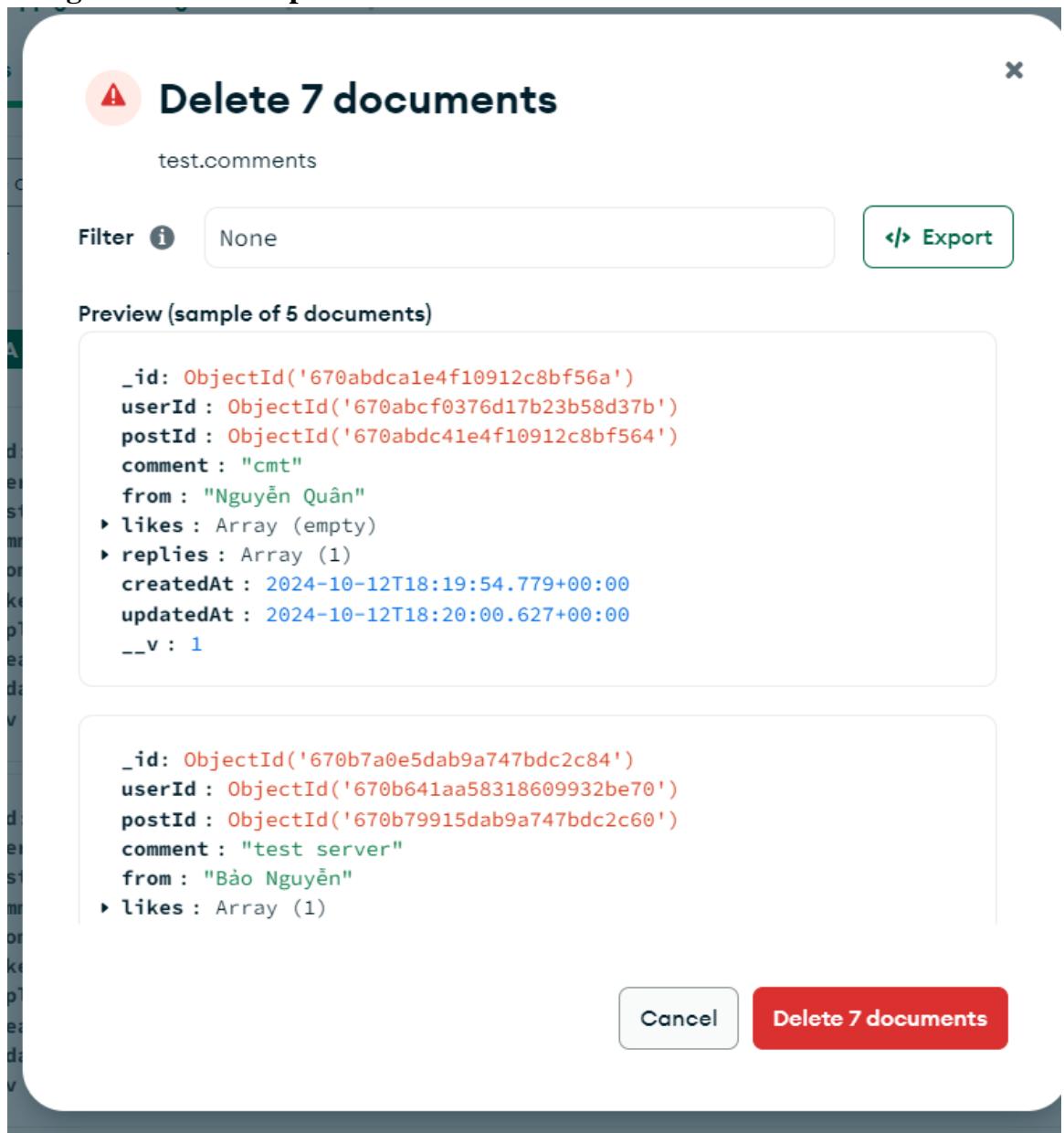
Hình 22. Dùng phương thức `db.collection.deleteMany()` xóa các document trong Collection theo điều kiện

Xóa một document thỏa điều kiện: Sử dụng cú pháp `db.collection.deleteOne()`, với tham số truyền vào là điều kiện tìm kiếm, lệnh này sẽ xóa document đầu tiên tìm thấy thỏa mãn điều kiện đã cho. Ví dụ, xóa dữ liệu sinh viên đầu tiên năm khóa 2020.

```
quanlythongtin_demo> db.students.deleteOne( {year: 2020})
{ acknowledged: true, deletedCount: 1 }
quanlythongtin_demo> db.students.find({})
[
  {
    _id: ObjectId("634c0659a6e48f56aa482686"),
    name: 'uit_sv_1',
    year: 2020,
    major: 'CNTT',
    gpa: 3.3,
    address: { city: 'TP.HCM', street: 'Đường số 1' }
  },
  {
    _id: ObjectId("634c0659a6e48f56aa482687"),
    name: 'uit_sv_2',
    year: 2022,
    major: 'KHMT',
    gpa: 3.7,
    address: { city: 'HCM', street: 'Đường số 2' },
    lastModified: ISODate("2022-10-16T16:35:47.679Z")
  },
  {
    _id: ObjectId("634c0659a6e48f56aa482688"),
    name: 'uit_sv_3',
    year: 2020,
    major: 'CNTT',
    gpa: 3.6,
    address: { city: 'TP.HCM', street: 'Đường số 3' }
  }
]
```

Hình 23. Dùng phương thức `db.collection.deleteOne()` xóa document đầu tiên trong Collection thỏa điều kiện

Dùng delete của compass



Hình 24. Xóa bằng compass

V. CÁC TÍNH NĂNG NÂNG CAO

1. Index

Giới thiệu về Index

Index là một cấu trúc dữ liệu đặc biệt, ứng dụng cây nhị phân để lưu trữ một phần của dữ liệu theo cách dễ duyệt và tìm kiếm. Index chứa dữ liệu của một trường hoặc một nhóm các trường cụ thể, và sắp xếp theo giá trị trong các trường đó. Việc sắp xếp các giá trị trong index giúp cho các truy vấn liên quan đến khoảng giá trị hoặc tìm giá trị khớp được hiệu quả hơn. Ngoài ra, MongoDB có thể trả về kết quả được sắp xếp sẵn khi dùng trình tự sắp xếp trong index. MongoDB có thể tận dụng index để giới hạn số lượng document cần kiểm tra.

Các loại index: MongoDB cung cấp các loại index khác nhau để phục vụ cho từng loại dữ liệu và câu truy vấn: Single Field, Compound Index, Multikey Index, Geospatial Index, Text Index, Hashed Index, Clustered Index.

Các thuộc tính của Index: Unique Index, Partial Index, Sparse Index, TTL Index, Hidden Index.

Tạo một index

Để tạo một index trong Mongo shell, dùng lệnh db.collection.createIndex() theo cú pháp sau: db.collection.createIndex(<tên trường dữ liệu và kiểu index kèm thông số>, <các tùy chọn khác>)

Ví dụ: tạo ra một index trên trường name, sắp xếp giá trị theo thứ tự giảm dần: db.collection.createIndex({ name: -1 })

Sử dụng Index để hỗ trợ truy vấn

Index có thể giúp cải thiện tốc độ truy vấn trong cơ sở dữ liệu. Khi truy vấn dữ liệu kèm theo chỉ định vùng cần tìm kiếm mà tương ứng với một index đã thiết lập, MongoDB sẽ trả về kết quả từ các giá trị trong index thay vì quét từng document hay đưa document vào vùng nhớ.

Ngoài ra, MongoDB có thể thực hiện Index Intersection bằng cách kết hợp các index đã thiết lập khác nhau để xử lý truy vấn nếu cho kết quả phù hợp với yêu cầu.

2. Nhập và xuất dữ liệu

MongoDB Compass hỗ trợ nhập và xuất dữ liệu JSON lẫn CSV.

Nhập dữ liệu vào Collection

Một số hạn chế khi nhập dữ liệu thông qua MongoDB Compass bao gồm chức năng nhập dữ liệu không ược hỗ trợ ở phiên bản Read Only của MongoDB Compass, và chức năng này sẽ không hoạt động nếu đang kết nối đến Data Lake. Trước khi nhập data vào MongoDB Compass phải chắc rằng dữ liệu đã ược định dạng đúng để Compass hiểu được.

Với định dạng file JSON: Mỗi dòng là một dữ liệu thông tin. Dấu phẩy ngăn cách dữ liệu trong một mảng (không ngắt dòng giữa các phần tử trong mảng).

Đối với định dạng file CSV: Dòng đầu tiên của file là tên cột phải ược ngăn cách bằng các dấu phẩy, những dòng tiếp theo là các giá trị của cột theo thứ tự.

Quy trình:

Bước 1: Kết nối với bộ tài liệu mà bạn muốn nhập dữ liệu.

Bước 2: Điều hướng đến bộ tài liệu muốn nhập.

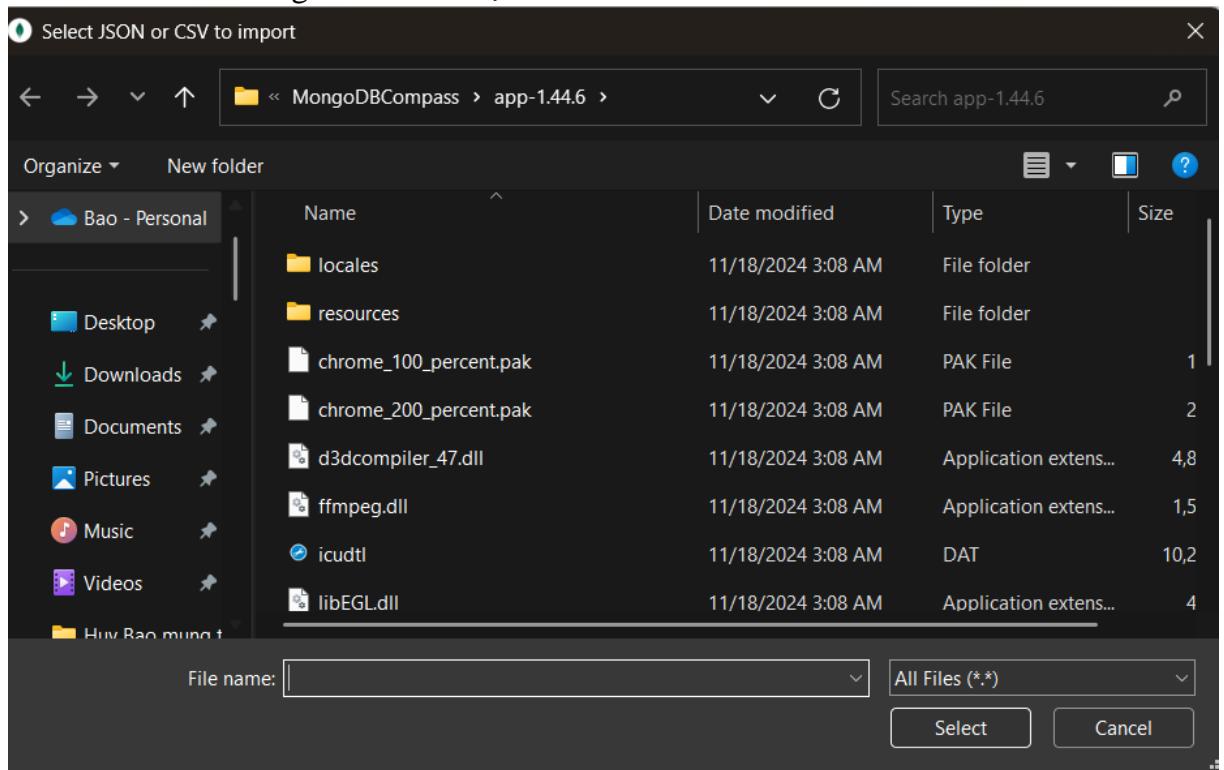
Bước 3: Chọn “Add Data” và chọn “import JSON or CSV file hoặc insert document”.

Áo hóa và điện toán đám mây – Nhóm 3

The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' sidebar lists 'clouds-mern-app.gx0nk.mongodb.net' with its collections: admin, config, local, and test. The 'test' collection is expanded, showing comments, friendrequests, passwordresets, posts, users, and verifications. The main area displays the 'comments' collection with 6 documents. A modal window titled 'ADD DATA' is open, showing options for 'Import JSON or CSV file' or 'Insert document'. Three JSON documents are listed under 'Insert document': one from 'Nguyễn Quân', one from 'Bảo Nguyễn', and one from 'Quan Nguyen'. Each document includes fields like _id, userId, postId, comment, from, likes, replies, createdat, and updatedat.

Hình 25. Click “Add Data” chọn import JSON or CSV file hoặc insert document

Bước 4: Chọn vị trí của tệp trong “Import File”, chọn loại tệp thích hợp với tệp muốn thêm vào bao gồm JSON hoặc CSV.



Hình 26. Chọn vị trí tệp và loại tệp cho loại tệp JSON

Bước 6: Chọn Import.

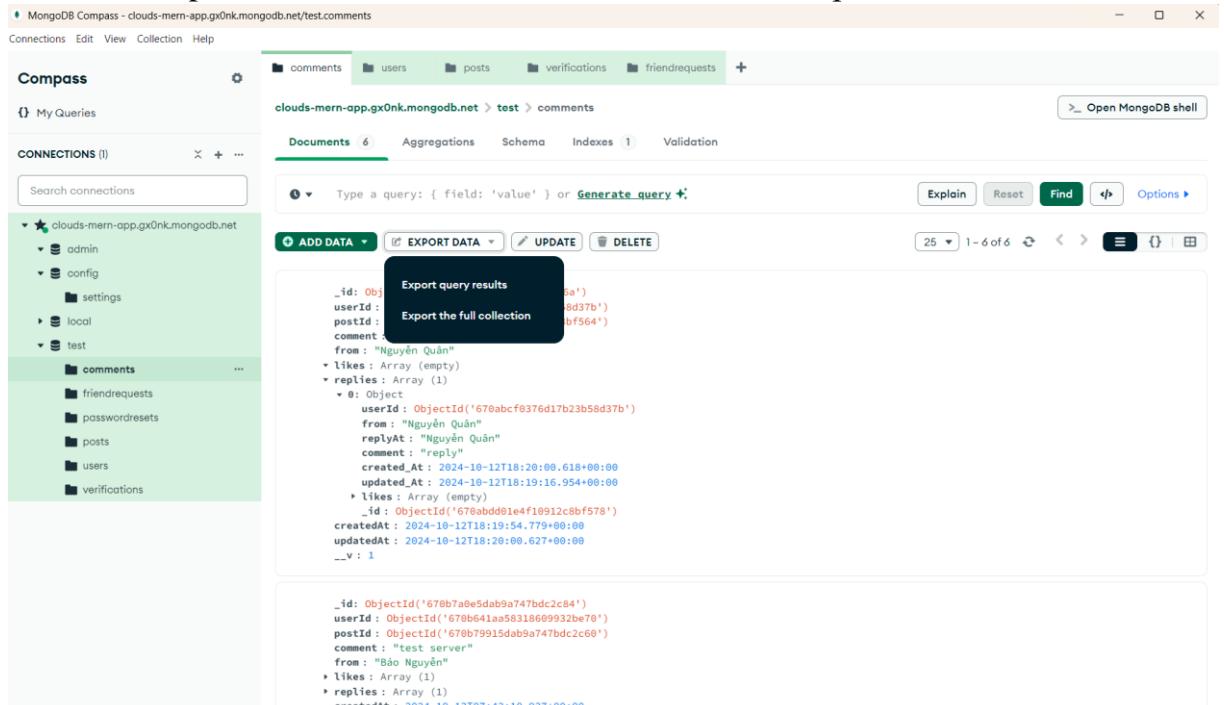
Xuất dữ liệu từ một Collection

MongoDB Compass có thể xuất dữ liệu dưới dạng tệp JSON hoặc CSV. Chỉ định bộ lọc hoặc đường dẫn tổng hợp cho nơi chứa dữ liệu của mình, Compass sẽ xuất các tài liệu phù hợp với kết quả truy vấn hoặc đường dẫn đã chỉ định. Cần lưu ý là không thể định hình lại tài liệu đã xuất bằng tài liệu dự án.

Để xuất toàn bộ bộ sưu tập sang một tệp:

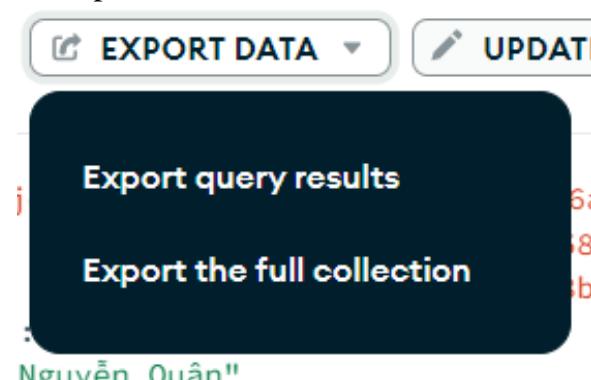
Bước 1: Kết nối với triển khai có chứa Collection muốn truy xuất.

Bước 2: Nhập vào Collection trên thanh menu và chọn “Export Collection”.



Hình 32. Chọn Export Collection

Compass sẽ xuất hiện hộp thoại như sau:



Hình 33. Hộp thoại chọn lựa cách xuất dữ liệu bằng cách query results hay xuất toàn bộ

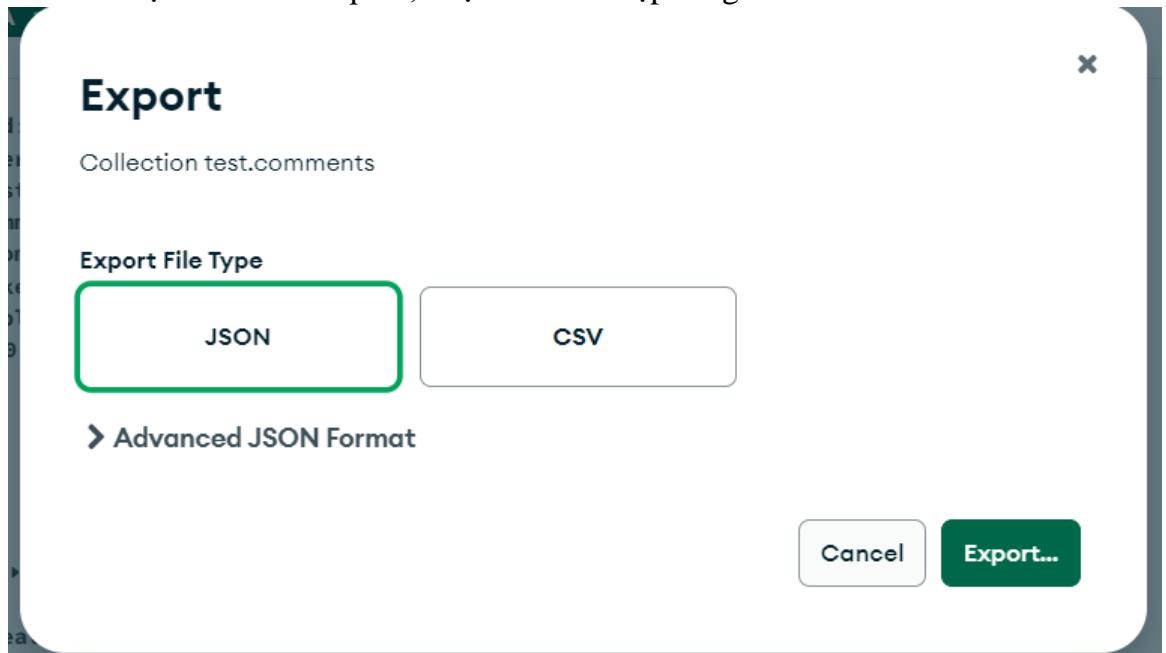
Hộp thoại hiển thị truy vấn mặc định sẽ được sử dụng để xuất dữ liệu. Nếu không có chỉnh sửa câu truy vấn, thao tác sẽ trả về tất cả các tài liệu trong collection. Nếu muốn bỏ qua bộ lọc và xuất toàn bộ dữ liệu thì chọn “Export Full Collection” và nhập vào “Select Fields”.

Trang 30

Áo hóa và điện toán đám mây – Nhóm 3

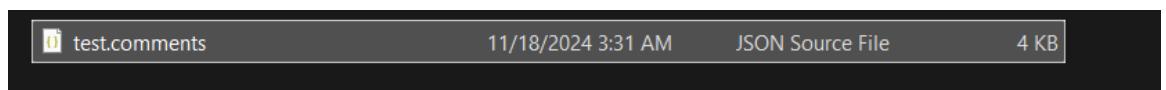
Bước 3: Chọn các trường tài liệu đưa vào tệp sẽ xuất. Thêm các trường khác bằng nút “Add Field” nếu trường đó không được chương trình tự động phát hiện.

Bước 4: Chọn loại tệp và vị trí xuất: Trong Chọn loại tệp xuất, chọn JSON hoặc CSV. Nếu chọn JSON, dữ liệu sẽ được xuất sang tệp đích dưới dạng một mảng. Sau đó chọn “Select Output”, chọn nơi xuất tệp sang.



Hình 34. nhấn Export để tiến hành xuất

Bước 5: Nhấp vào Export. Thanh tiến trình hiển thị trạng thái của quá trình xuất. Nếu xảy ra lỗi, thanh tiến trình sẽ chuyển sang màu đỏ và thông báo lỗi xuất hiện trong hộp thoại. Sau khi xuất thành công, hộp thoại sẽ đóng.



Hình 27. File comments

```
[{"_id": {"$oid": "670abdc41e4f10912c8bf56a"}, "userId": {"$oid": "670abcf0376d17b23b58d37b"}, "postId": {"$oid": "670abdc41e4f10912c8bf564"}, "comment": "cmt", "from": "Nguyễn Quân", "likes": [], "replies": [{"userId": {"$oid": "670abcf0376d17b23b58d37b"}, "from": "Nguyễn Quân", "replyAt": "Nguyễn Quân", "comment": "reply", "created At": {"$date": "2024-10-12T18:20:00.618Z"}, "updated At": {"$date": "2024-10-12T18:19:16.954Z"}, "likes": [], "id": {"$oid": "670abdd01e4f10912c8bf578"}}], "likes": []}]
```

Ln 1, Col 1 | 3,320 characters | 100% | Unix (LF) | UTF-8

Hình 28. Collection đã được ghi

3. Một số tính năng khác trên MongoDB

Change stream: Change stream cho phép các ứng dụng tiếp cận thông tin về các thay đổi trong dữ liệu theo thời gian thực thay vì phải theo dõi oplog. Các ứng dụng kết nối với cơ sở dữ liệu MongoDB có thể dùng change stream để có được các thay đổi về dữ liệu lưu trên một collection đơn lẻ, một cơ sở dữ liệu hay nguyên cả dự án, và dựa trên đó có thể ngay lập tức có các thao tác phản ứng lại với các thay đổi đó.

Time Series: Time series data (dữ liệu theo thời gian) là một chuỗi các dữ liệu thay đổi theo thời gian, từ sự thay đổi của dữ liệu có thể phân tích, nghiên cứu, đưa ra các nhận xét về dữ liệu, ngoài ra còn giúp cải thiện việc truy vấn và tối ưu vùng nhớ lưu trữ dữ liệu và các index. Collection time series có thể thực hiện các thao tác như thêm và truy vấn như các collection bình thường

Thực hiện một transaction trên nhiều document: Trong MongoDB, một thao tác lên một document có tính nguyên tử. Tuy nhiên, đối với các trường hợp cần tính nguyên tử trên thao tác đọc và ghi cùng lúc trên nhiều document (có thể trên một hoặc nhiều collection), MongoDB hỗ trợ thực hiện transaction trên nhiều document.

VI. ỨNG DỤNG CỦ THỂ

1. Ứng dụng Web

MongoDB thường được sử dụng trong các ứng dụng web, đặc biệt là những ứng dụng cần lưu trữ và truy vấn dữ liệu động. Với tính chất không có cấu trúc cố định của NoSQL, MongoDB rất phù hợp để lưu trữ dữ liệu người dùng, sản phẩm, hoặc nội dung động mà không cần quá nhiều quan tâm đến mô hình dữ liệu ban đầu.

2. Mobile App

MongoDB thường được sử dụng trong backend của các ứng dụng di động nhờ khả năng lưu trữ và truy vấn dữ liệu theo dạng JSON, giúp đơn giản hóa quá trình tích hợp dữ liệu giữa ứng dụng và cơ sở dữ liệu.

3. Phân tích dữ liệu thời gian thực

MongoDB hỗ trợ lưu trữ lượng dữ liệu lớn và có thể thực hiện truy vấn nhanh chóng, giúp nó trở thành lựa chọn phổ biến trong các hệ thống phân tích thời gian thực, thu thập và phân tích dữ liệu từ cảm biến, dữ liệu log hoặc dữ liệu người dùng.

4. Hệ Thống Phân Tán

Với khả năng hỗ trợ clustering, MongoDB có thể dễ dàng được triển khai trong các hệ thống phân tán để lưu trữ dữ liệu trên nhiều máy chủ khác nhau, đảm bảo tính sẵn sàng cao và khả năng mở rộng tốt.

5. IoT

MongoDB được sử dụng trong các hệ thống IoT để lưu trữ dữ liệu từ các thiết bị kết nối. Khả năng mở rộng và lưu trữ dữ liệu phi cấu trúc giúp nó trở thành lựa chọn lý tưởng cho các ứng dụng IoT.

6. Microservices

MongoDB là lựa chọn phổ biến cho kiến trúc microservices, nơi mỗi service có thể có một cơ sở dữ liệu riêng. Với khả năng hỗ trợ lưu trữ đa dạng kiểu dữ liệu, MongoDB giúp các microservices quản lý dữ liệu một cách linh hoạt và hiệu quả.

CHƯƠNG II. THỰC NGHIỆM VỚI MONGODB

I. CÀI ĐẶT MONGODB

1. Cấu hình phần cứng yêu cầu

Bộ xử lý (CPU)

- Đối với các phiên bản MongoDB từ 4.4 trở lên, chỉ hỗ trợ các phiên bản 64-bit của Windows trên kiến trúc x86_64.
- Khuyến nghị sử dụng CPU từ hai nhân trở lên để tối ưu hiệu suất.

Bộ nhớ (RAM)

- Tối thiểu: 2GB RAM.
- Khuyến nghị: Từ 4GB RAM trở lên để đảm bảo hiệu suất và khả năng xử lý.

Dung lượng ổ đĩa cứng (Storage)

- Dung lượng trống tối thiểu: 10GB.
- Khuyến nghị sử dụng ổ SSD để cải thiện tốc độ truy xuất dữ liệu.
- Chú ý: Cần dự trữ thêm không gian lưu trữ để đảm bảo đủ chỗ cho dữ liệu phát triển trong tương lai và các file nhật ký (log files).

Hệ điều hành

- Windows Server 2022
- Windows Server 2019
- Windows 11
- Chỉ hỗ trợ phiên bản 64-bit.

2. Các phiên bản

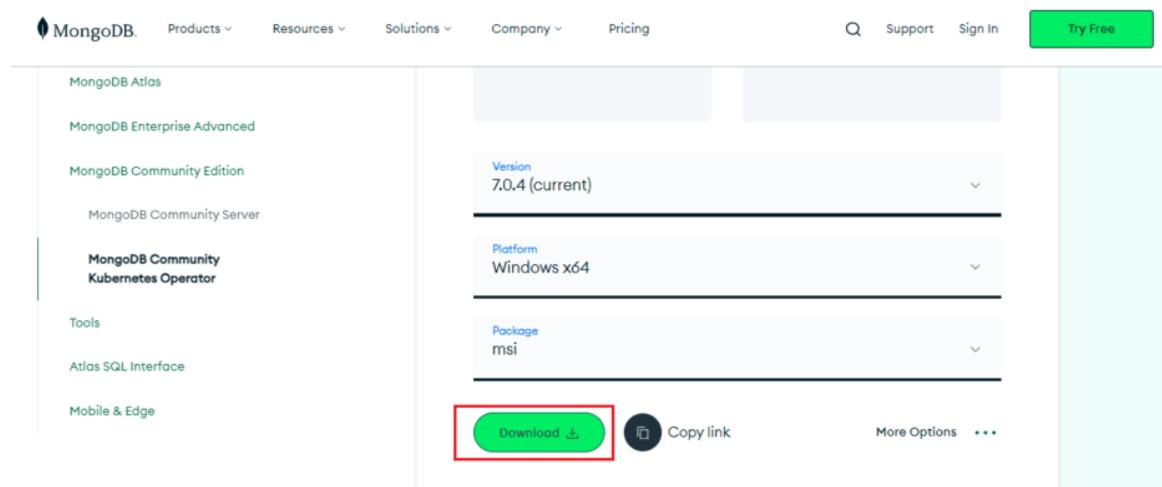
- MongoDB 7.0.4
- Windows 11 Pro 23H2 64-bit

3. Hướng dẫn cài MongoDB

Để cài đặt MongoDB trên Windows, đầu tiên, chúng ta tải MongoDB server và sau đó cài đặt MongoDB shell. Theo dõi các bước dưới đây để hoàn thành quá trình cài đặt MongoDB.

Bước 1: Truy cập trang [MongoDB Download Center](#) để cài đặt MongoDB Community Server:

Áo hóa và điện toán đám mây – Nhóm 3



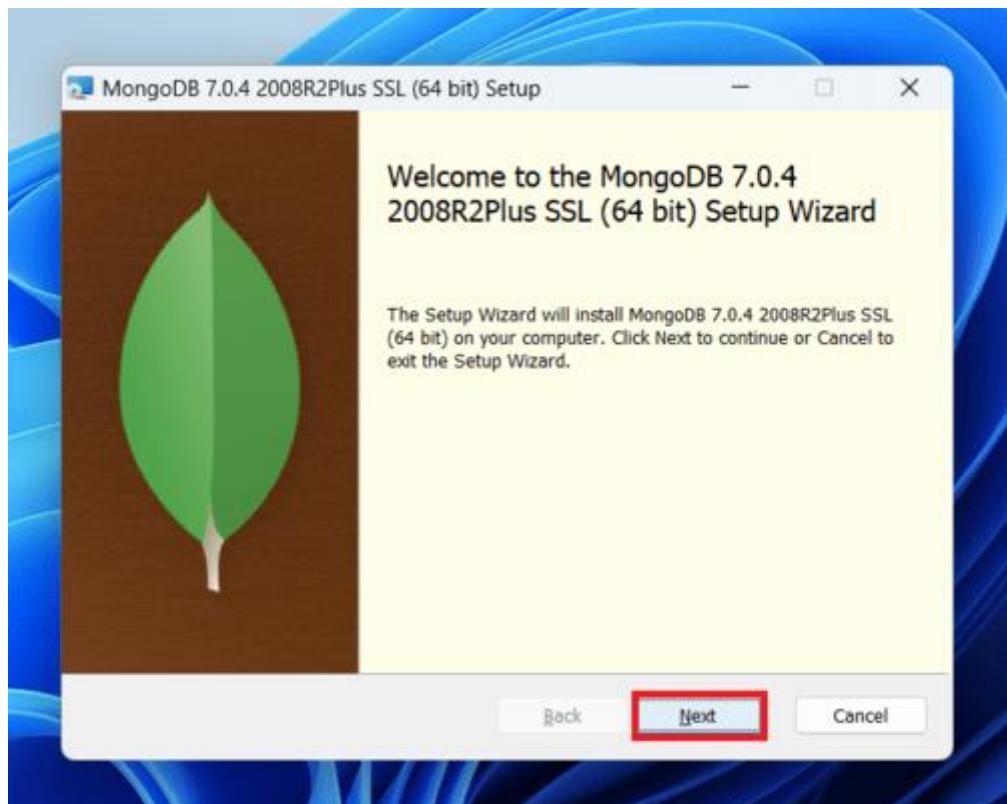
The screenshot shows the MongoDB website's product selection interface. On the left, there's a sidebar with options like MongoDB Atlas, MongoDB Enterprise Advanced, MongoDB Community Edition, MongoDB Community Server, MongoDB Community Kubernetes Operator, Tools, Atlas SQL Interface, and Mobile & Edge. The main area displays three dropdown menus: Version (set to 7.0.4 (current)), Platform (set to Windows x64), and Package (set to msi). At the bottom right of this section are buttons for 'Download', 'Copy link', and 'More Options'. A red box highlights the 'Download' button.

Hình 29. Tùy chọn các phiên bản phù hợp

Ở đây, bạn có thể tùy chọn các phiên bản, Windows, các package phù hợp. Ở đây, tôi sẽ chọn:

- Version: 7.0.4
- OS: Windows x64
- Package: msi

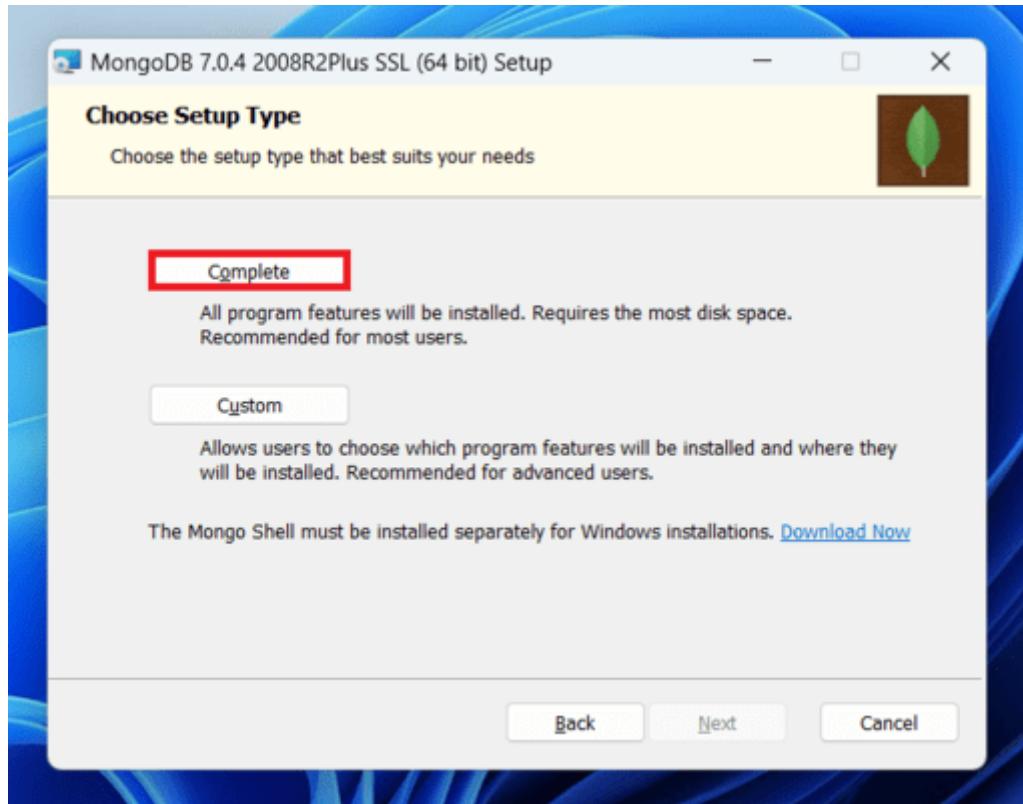
Bước 2: Khi tải xong, chúng ta mở file **msi** và chọn nút **next**:



Hình 30. Màn hình startup

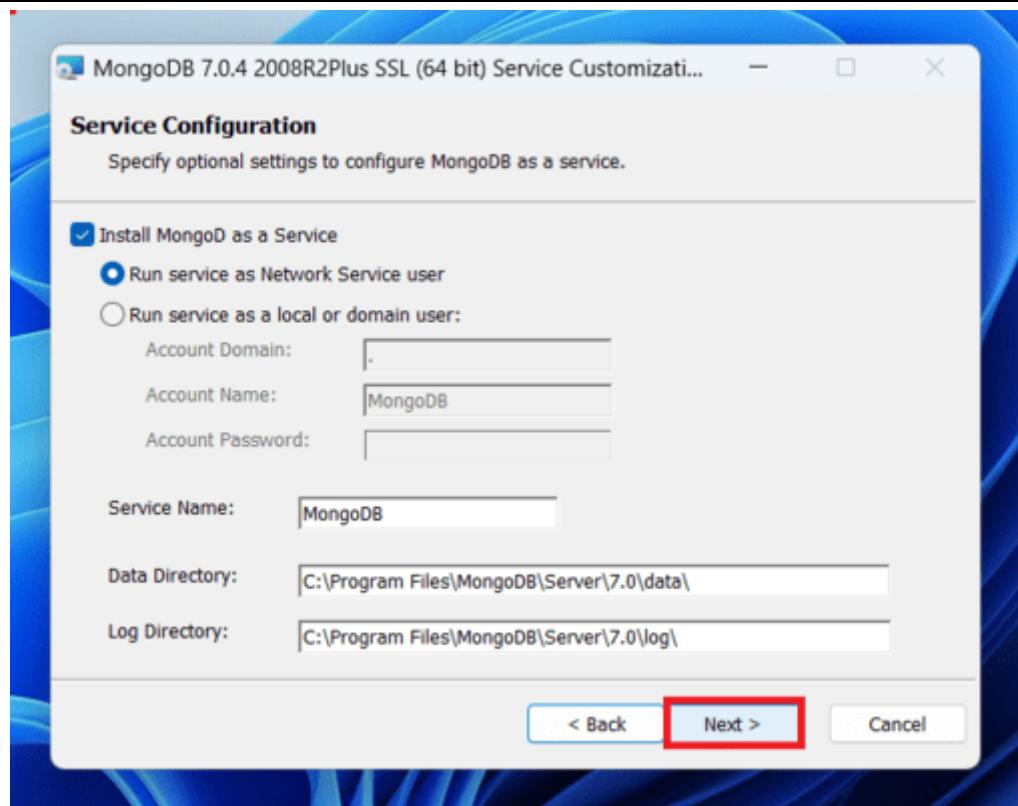
Bước 3: Chấp nhận **End-User License Agreement** và nhấn nút **next**

Bước 4: Bây giờ hãy chọn **complete option** để cài đặt tất cả các tính năng của chương trình. Ở đây, nếu bạn chỉ muốn cài đặt các tính năng chương trình đã chọn và muốn chọn vị trí cài đặt, hãy sử dụng **Custom option**:



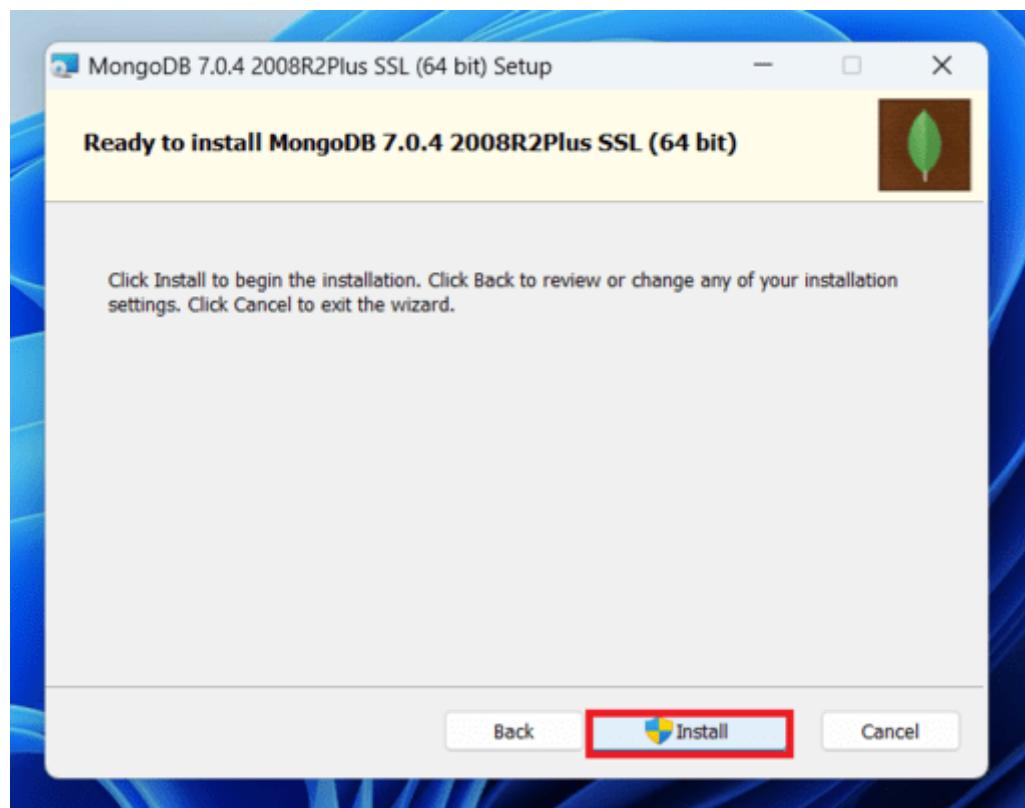
Hình 31. Tùy chọn cài đặt

Bước 5: Chọn “Run service as Network Service user” và sao chép đường dẫn của thư mục dữ liệu. Nhập vào Next :



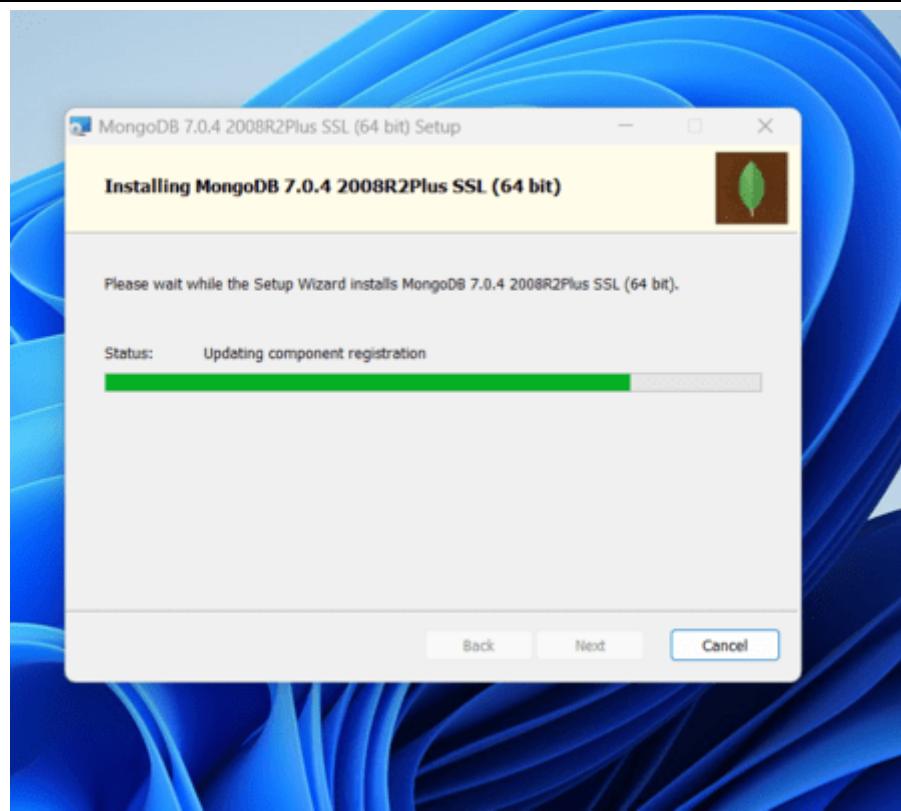
Hình 32. Cấu hình service

Bước 6: Nhấp vào nút **Install** để bắt đầu quá trình cài đặt MongoDB:



Hình 33. Tiến hành cài đặt

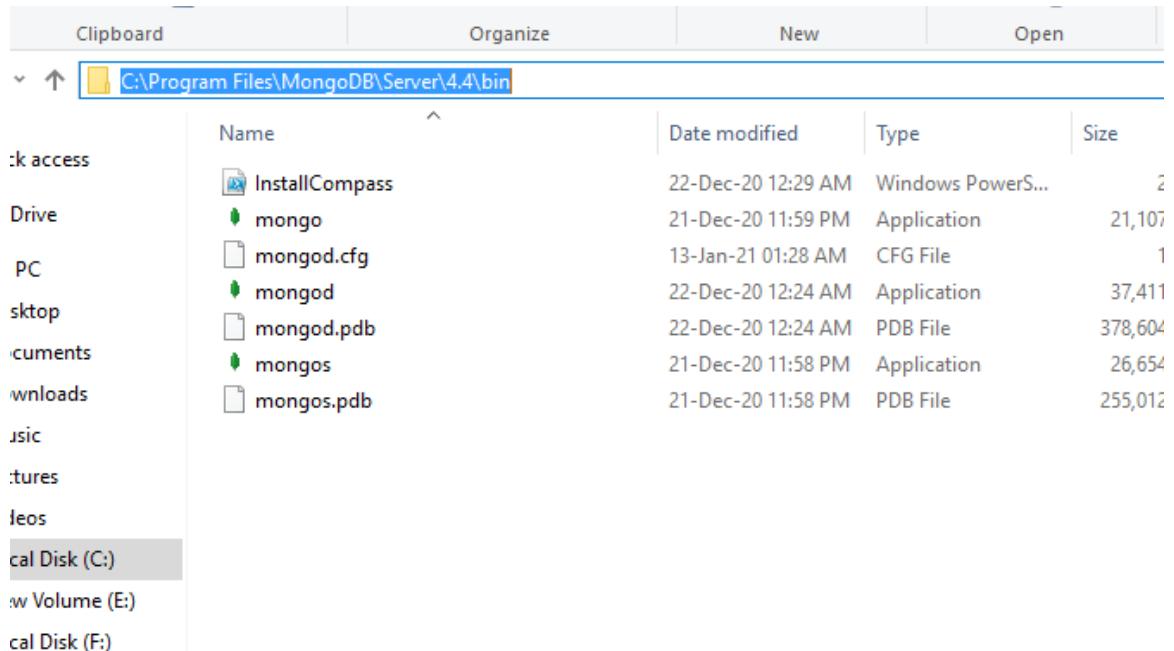
Bước 7: Sau khi nhấp vào nút cài đặt, quá trình cài đặt MongoDB sẽ bắt đầu:



Hình 34. Quá trình cài đặt đang được thực hiện

Bước 8: Bây giờ hãy nhấp vào nút **Kết thúc** để hoàn tất quá trình cài đặt MongoDB.

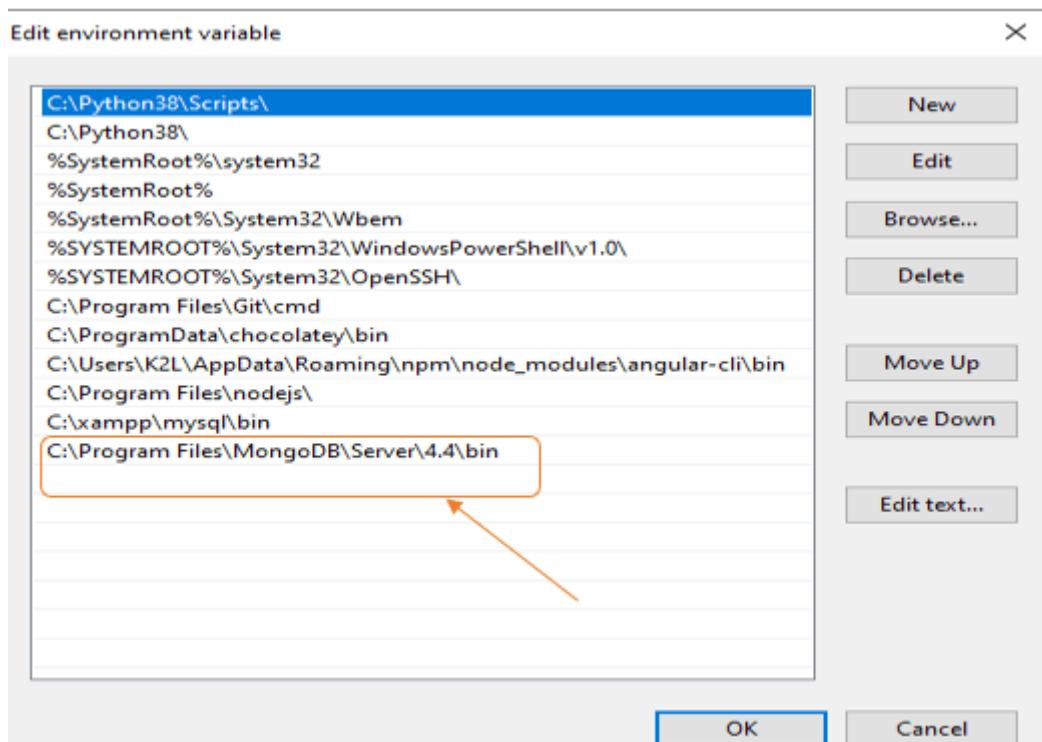
Bước 9: Bây giờ chúng ta đến vị trí MongoDB đã cài đặt ở bước 5 trong hệ thống của bạn và sao chép đường dẫn bin:



Hình 35. Sao chép đường dẫn bin

Ao hoá và điện toán đám mây – Nhóm 3

Bước 10: Bây giờ, để tạo biến môi trường, hãy mở system properties >> Environment Variable >> System variable >> path >> Edit Environment variable và dán liên kết đã sao chép vào hệ thống môi trường của bạn và nhấp vào Ok:



Hình 36. Tạo biến môi trường

Bước 11: Sau khi thiết lập biến môi trường, chúng ta sẽ chạy máy chủ MongoDB, tức là **mongod**. Vì vậy, hãy mở **command prompt** và chạy lệnh sau:



Hình 37. Chạy MongoDB server

Khi bạn chạy lệnh này, bạn sẽ nhận được lỗi như sau: **C:/data/db/ not found**.

Bước 12: Bây giờ, mở ô **C và tạo một thư mục có tên là “ data ”** bên trong thư mục này tạo một thư mục khác có tên là “db”. Sau khi tạo các thư mục này. Mở lại **command prompt** và chạy lệnh sau:



Hình 38. Chạy MongoDB server lại sau khi sửa lỗi

Bây giờ, lần này máy chủ MongoDB (tức là mongod) sẽ chạy thành công.

Bây giờ bạn có thể tạo một cơ sở dữ liệu, bộ sưu tập và tài liệu mới trong shell của mình. Dưới đây là ví dụ về cách tạo một cơ sở dữ liệu mới:

Lệnh **use Database_name** tạo một cơ sở dữ liệu mới trong hệ thống nếu nó không tồn tại, nếu cơ sở dữ liệu tồn tại thì sẽ sử dụng cơ sở dữ liệu đó:

```
use gfg
```

Hình 41. Chạy database của bạn

Bây giờ cơ sở dữ liệu của bạn đã sẵn sàng với tên gfg.

Lệnh **db.Collection_name** tạo một bộ sưu tập mới trong cơ sở dữ liệu gfg và phương thức **insertOne()** chèn tài liệu vào **student**:

```
db.student.insertOne({Akshay: 500})
```

Hình 42. Viết truy vấn cho database

Các bước cài đặt MongoDB trên Windows mà không cần quyền quản trị

Sau đây là quy trình từng bước về Cách cài đặt MongoDB trên Windows mà không cần Quyền quản trị viên –

Bước 1: Tải xuống tệp ZIP MongoDB từ trang web chính thức của MongoDB.

Bước 2: Giải nén nội dung của tệp ZIP vào một vị trí trên máy tính mà bạn có quyền ghi, chẳng hạn như thư mục người dùng.

Bước 3: Điều hướng đến thư mục MongoDB đã giải nén và định vị thư mục “bin”.

Bước 4: Mở cửa sổ **command prompt** và điều hướng đến thư mục “bin” trong thư mục MongoDB.

Bước 5: Chạy máy chủ MongoDB bằng cách thực hiện lệnh có đường dẫn đến thư mục mà bạn muốn lưu trữ các tệp dữ liệu MongoDB. Đảm bảo sử dụng vị trí mà bạn có quyền ghi.

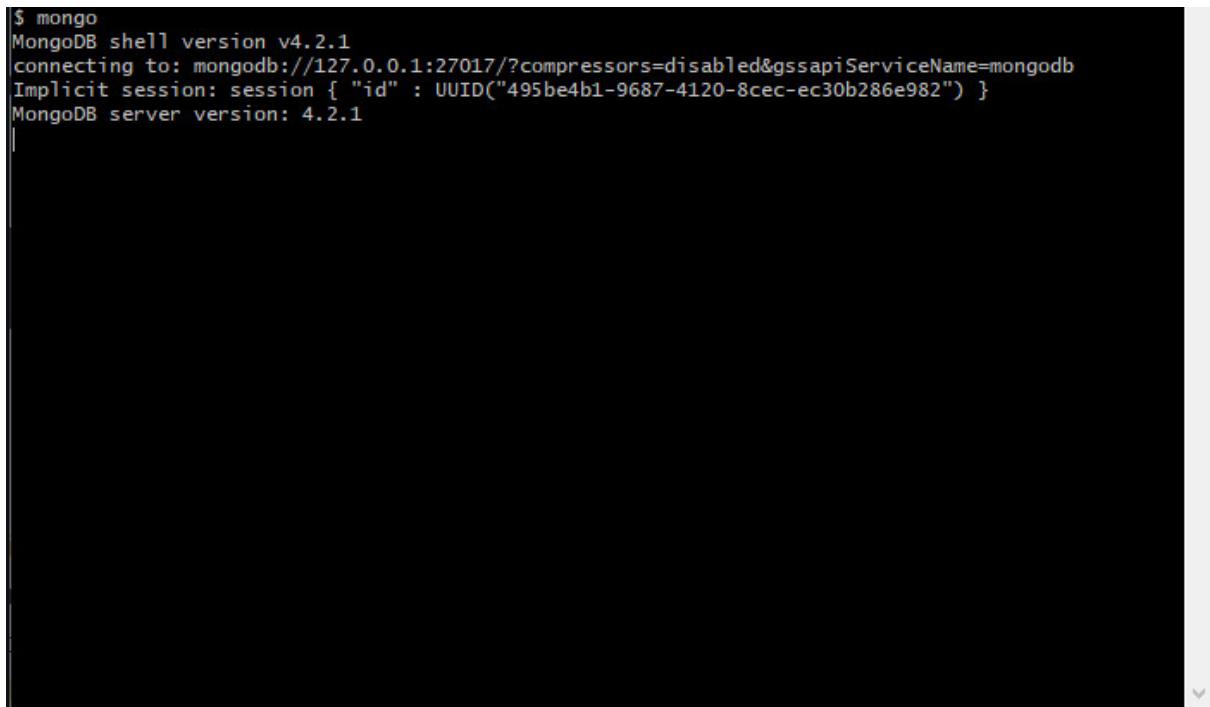
```
Command: mongod.exe --dbpath=path\to\data\directory, replacing  
"path\to\data\directory"
```

Hình 43. Đường dẫn đến nơi lưu trữ

Bước 6: MongoDB bây giờ sẽ chạy cục bộ trên hệ thống Windows của bạn mà không cần quyền quản trị. Bạn có thể tương tác với MongoDB bằng cách sử dụng MongoDB shell bằng cách chạy lệnh **mongo.exe** trong cửa sổ **command prompt** khác.

Bước 5: Để tương tác với mongoDB qua command line. Thì phải cài đặt biến môi trường. Tìm tới folder cài đặt, mặc định thì sẽ là **C:\Program Files\MongoDB\Server\4.2\bin**.

Sau đó, mở **cmd** hoặc **Git bash** lên gõ lệnh **mongo** để kiểm tra:



```
$ mongo
MongoDB shell version v4.2.1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("495be4b1-9687-4120-8cec-ec30b286e982") }
MongoDB server version: 4.2.1
```

Hình 45. Kiểm tra bằng lệnh mongo

Một số câu lệnh cơ bản:

- show dbs: dùng để hiển thị danh sách các database đang tồn tại. Mặc định ban đầu sẽ có 3 database của hệ thống là: admin, config và local.
- show collections: hiển thị các collection đang tồn tại
- db: hiển thị danh sách database đang sử dụng. Vì bạn chưa có database nào nên mặc định database mặc định cho bạn sẽ là test.
- use DATABASE_NAME: chuyển sang sử dụng database với tên là DATABASE_NAME (tên này tùy bạn chọn).
- db.dropDatabase(): dùng để xóa database hiện tại
- db.createCollection(name, options): dùng để tạo mới một Collection trong MongoDB (cái này tương tự như Table trong MySQL).

Cách select, insert, update collections . Mọi thứ bắt đầu từ db.<collections>...

- db.COLLECTION_NAME.insert(document): thêm mới một Document vào MongoDB (cái này tương tự như Row trong MySQL)
- db.COLLECTION_NAME.save(document): thêm mới hoặc cập nhật một document trong collection

2. Các truy vấn cơ bản trong MongoDB

a) Mongo Shell

Mongo Shell là một JavaScript interface tương tác với MongoDB. Bạn có thể sử dụng mongo shell để truy vấn và cập nhật dữ liệu cũng như thực hiện các hoạt động quản trị.

Thực hiện một số lệnh truy vấn cơ bản trong mongo shell. Đầu tiên, tôi sẽ sử dụng lệnh mongoimport --help để xem tất cả documents trong mongo shell.

```
$ mongoimport --help
Usage:
  mongoimport <options> <file>

  Import CSV, TSV or JSON data into MongoDB. If no file is provided, mongoimport reads from stdin.

  See http://docs.mongodb.org/manual/reference/program/mongoimport/ for more information.
```

Hình 46. Tất cả documents trong mongo shell.

Chúng ta sẽ thực hiện một số thao tác CRUD trong trình Mongo Shell. Để làm như vậy, ta có thể sử dụng các lệnh import trong Mongo. Các lệnh import Mongo có thể thực hiện công việc import dữ liệu nằm trong file .tsv, .csv, .json, v.v.

b) Insert

Cách 1: Tạo một collections có tên Numbers và insert 26 000 row vào đó.

```
> show dbs
admin      0.000GB
config     0.000GB
local      0.000GB
mongotest   0.001GB
mylearning  0.000GB

> use mongo test
switched to db mongotest

> for (i = 0; i <= 26000; i++) {
    db.Numbers.insert({
        "number": i
    })
}
WriteResult({ "nInserted" : 1 })
```

Hình 47. Tạo collections

Import data từ file JSON. Tạo một file json với tên colors.json có cấu trúc như sau:

```
[  
  {  
    "color": "black",  
    "category": "hue",  
    "type": "primary",  
    "code": {  
      "rgba": [  
        255,  
        255,  
        255,  
        1  
      ],  
      "hex": "#000"  
    }  
  },  
  {  
    "color": "white",  
    "category": "value",  
    "code": {  
      "rgba": [  
        0,  
        0,  
        0,  
        1  
      ],  
      "hex": "#FFF"  
    }  
  },  
  {  
    "color": "red",  
    "category": "hue",  
    "type": "primary",  
    "code": {  
      "rgba": [  
        255,  
        0,  
        0,  
        1  
      ],  
      "hex": "#FF0"  
    }  
  },  
]
```

Hình 48. File colors.json (1)

```
},
{
  "color": "blue",
  "category": "hue",
  "type": "primary",
  "code": {
    "rgba": [
      0,
      0,
      255,
      1
    ],
    "hex": "#00F"
  }
},
{
  "color": "yellow",
  "category": "hue",
  "type": "primary",
  "code": {
    "rgba": [
      255,
      255,
      0,
      1
    ],
    "hex": "#FF0"
  }
},
{
  "color": "green",
  "category": "hue",
  "type": "secondary",
  "code": {
    "rgba": [
      0,
      255,
      0,
      1
    ],
    "hex": "#0F0"
  }
}
]
```

Hình 49. File colors.json (2)

Sau đó thực hiện import nó vào mongo. Open terminal tại nơi chứa file colors.json.

```
$ mongoimport --db mongo_color --collection colors --jsonArray --file colors.json
```

Hình 50. Import file vào mongo

Kết quả:

```
connected to: mongodb://localhost/
6 document(s) imported successfully. 0 document(s) failed to import.
```

Hình 51. Database đã được kết nối thành công

- --db mongo_color: tên database
- --collection colors: tên collection
- --jsonArray: nguồn đầu vào là một mảng JSON

- --file colors.json: tên file data

Hoặc cách insert từ mongo shell:

```
$ mongo
> use mongo_color
switched to db mongo_color
> db.colors.insert({
  "color": "custome",
  "category": "hue",
  "type": "primary",
  "code": {
    "rgba": [255, 1, 255, 1],
    "hex": "#FF1"
  }
})
```

Hình 52. Insert từ mongo shell

c) Update

Update một row trong mongoDB thì như thế nào?

Chúng ta sẽ thêm một property mới manuallyCreated với value là true vào color custome đã insert ở trên.

```
> db.colors.update({
  "color": "custome"
},
{
  $set: {
    "manuallyCreated": "True"
  }
})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Hình 53. Thêm một property mới

Truy vấn các color có manuallyCreated là true.

```
> db.colors.find({"manuallyCreated":"True"})
{ "_id" : ObjectId("5ddca6b6993d1592c2ace363"), "color" : "custome", "category" : "hue", "type" : "primary",
"code" : { "rgba" : [ 255, 1, 255, 1 ], "hex" : "#FF1" }, "manuallyCreated" : "True" }
```

Hình 54. Truy vấn các color có manuallyCreated là true

Update color name thành blackpink, điều kiện là một là ObjectId:

```
> db.colors.update({
  "_id": ObjectId("5ddca6b6993d1592c2ace363")
}, {
  $set: {
    "color": "blackpink"
  }
})

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.colors.find({"manuallyCreated":"True"})

{ "_id" : ObjectId("5ddca6b6993d1592c2ace363"), "color" : "blackpink", "category" : "hue", "type" : "primary", "code" : { "rgba" : [ 255, 1, 255, 1 ], "hex" : "#FF1" }, "manuallyCreated" : "True" }
```

Hình 55. Update color name

Trường hợp không set một value cho property cụ thể, thì nó bị ghi đè:

```
db.colors.update({"color":"red"}, {"color":"test"})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.colors.find({"color":"test"})
{ "_id" : ObjectId("5ddca636c0ca555ba6529b59"), "color" : "test" }
```

Hình 56. Trường hợp bị ghi đè

Một số ví dụ về update user collections:

```
db.users.update(
  {
    age: {
      $gt: 25
    }
  },
  {
    $set: {
      status: "Active"
    }
  },
  {
    multi: true
  }
)

db.users.update(
  {
    status: "Active"
  },
  {
    $inc: {
      age: 3
    }
  },
  {
    multi: true
  }
)
```

Hình 57. Ví dụ về update user collections

- multi: true: nếu có dữ liệu phù hợp điều kiện, thì sẽ update tất cả thay vì một
- \$gt: greater than
- \$lt: less than

Áo hóa và điện toán đám mây – Nhóm 3

- \$inc: ví dụ age = age + 3 để tăng giá trị của một field được chỉ định.

d) Truy vấn select trong MongoDB

i. Find() & Pretty format trong mongo shell:

```
> db.COLLECTION_NAME.find()
```

Hình 58. Cú pháp

```
db.colors.find()
```

Hình 59. Ví dụ về Find()

```
db.colors.find().pretty()
{
    "_id" : ObjectId("5ddcd6a60bd1004f4f9e83d5"),
    "color" : "blue",
    "category" : "hue",
    "type" : "primary",
    "code" : {
        "rgba" : [
            0,
            0,
            255,
            1
        ],
        "hex" : "#00F"
    }
}
```

Hình 60. Sử dụng pretty formating

ii. Toán tử

Xem docs - <https://docs.mongodb.com/manual/reference/operator/query/>

Operation	Syntax	Example	SQL
Equality	{<key>:<value>}	{status: "Active"}	where kind = 'rat'
Less Than	{<key>:{\$lt:<value>}}	{age: {\$lt: 2}}	where age < 2
Less Than Equals	{<key>:{\$lte:<value>}}	{age: {\$lte: 2}}	where age <= 2
Greater Than	{<key>:{\$gt:<value>}}	{age: {\$gt: 2}}	where age > 2
Greater Than Equals	{<key>:{\$gte:<value>}}	{age: {\$gte: 2}}	where age >= 2
Not Equals	{<key>:{\$ne:<value>}}	{age: {\$ne: 2}}	where age != 2
In	{<key>:{\$in:[<value1>, <value2>, ...]}}	{age: {\$in: [1, 2, 3]}}	where age in (1, 2, 3)

Hình 61. Các toán tử trong mongodb

iii. Columns

```
db.COLLECTION_NAME.find(QUERY, COLUMNS)
```

Hình 62. Cú pháp của columns

```
db.users.find(  
  {  
    status: "A"  
  },  
  {  
    user_id: 1,  
    status: 1,  
    _id: 0  
  }  
)
```

Hình 63. Ví dụ

iv. AND

```
db.COLLECTION_NAME.find({key1:value1, key2:value2})
```

Hình 64. Cú pháp của AND

```
db.users.find({  
  age: {  
    $gte: 25  
  },  
  status: 'Active'  
})
```

Hình 65. Ví dụ

v. OR

```
db.COLLECTION_NAME.find({$or: [{key1: value1}, {key2:value2}]})
```

Hình 66. Cú pháp của OR

```
db.users.find({  
  $or: [{  
    status: 'Active'  
  }, {  
    status: 'UnActive'  
  }]  
)
```

Hình 67. Ví dụ về OR

vi. Limit & Offset

```
db.users.find().limit(1)
```

Hình 68. Ví dụ về limit

```
db.pets.find().skip(1).limit(1)
```

Hình 69. Ví dụ về limite và offset

vii. Sort

Sắp xếp tăng dần

```
db.users.find({
    status: "Active"
}).sort({
    name: 1
})
```

Hình 70. Ví dụ sắp xếp tăng dần

Sắp xếp giảm dần

```
db.users.find({
    status: "Active"
}).sort({
    name: -1
})
```

Hình 71. Ví dụ sắp xếp giảm dần

e) Backup database trong MongoDB

Tạo một folder để chứa file backup.

- Command backup tất cả database:

```
$ mongodump --out D:\data\mongodb_backup
2019-11-26T17:18:38.604+0700      writing admin.system.users to
2019-11-26T17:18:38.612+0700      done dumping admin.system.users (1 document)
2019-11-26T17:18:38.612+0700      writing admin.system.version to
2019-11-26T17:18:38.617+0700      done dumping admin.system.version (2 documents)
2019-11-26T17:18:38.618+0700      writing mongotest.Numbers to
2019-11-26T17:18:38.618+0700      writing mongo_color.colors to
2019-11-26T17:18:38.618+0700      writing mongo_color.users to
2019-11-26T17:18:38.618+0700      writing mongotest.users to
2019-11-26T17:18:38.624+0700      done dumping mongotest.users (1 document)
2019-11-26T17:18:38.925+0700      done dumping mongo_color.users (3 documents)
2019-11-26T17:18:38.925+0700      done dumping mongo_color.colors (6 documents)
2019-11-26T17:18:39.163+0700      done dumping mongotest.Numbers (52002 documents)
```

Hình 72. Command backup database

- Command backup với một database cụ thể:

```
$ mongodump --db mongo_color --out d:\data\mongodb_backup
2019-11-26T17:11:45.650+0700      writing mongo_color.colors to
2019-11-26T17:11:45.652+0700      writing mongo_color.users to
2019-11-26T17:11:45.659+0700      done dumping mongo_color.users (3 documents)
2019-11-26T17:11:45.964+0700      done dumping mongo_color.colors (6 documents)
```

Hình 73. Chọn database cụ thể để backup

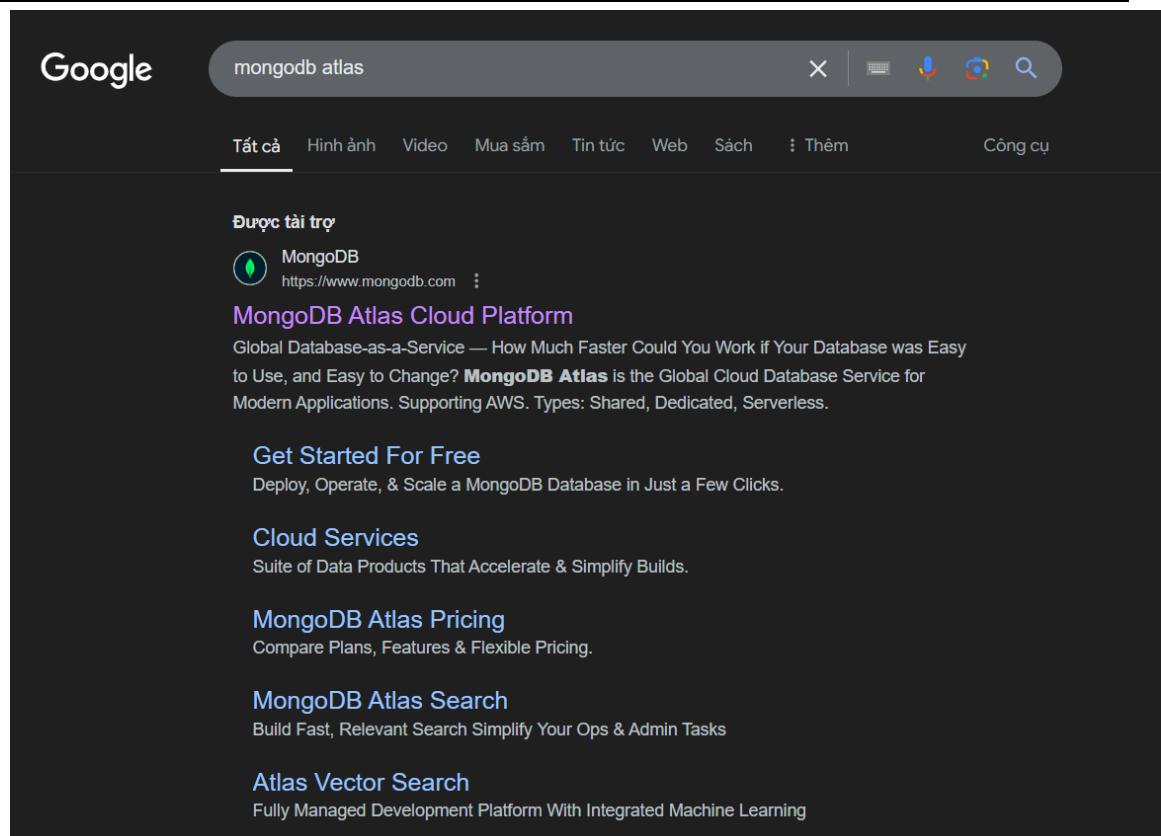
f) Restore

```
mongorestore --db mongo_color d:\data\mongodb_backup\mongo_color
mongorestore --db l4_analytics C:\wamp64\www\568E\roi\l4\data2\l4_analytics
```

Hình 74. Lệnh restore một database đã được backup:

3. MongoDB Atlas

Bước 1: tìm kiếm trên web theo từ khóa MongoDB Atlas.



Hình 75. Tìm kiếm Atlas trên internet

Bước 2: Tạo tài khoản

Get started free

First Name*
Last Name*

Company

Email*

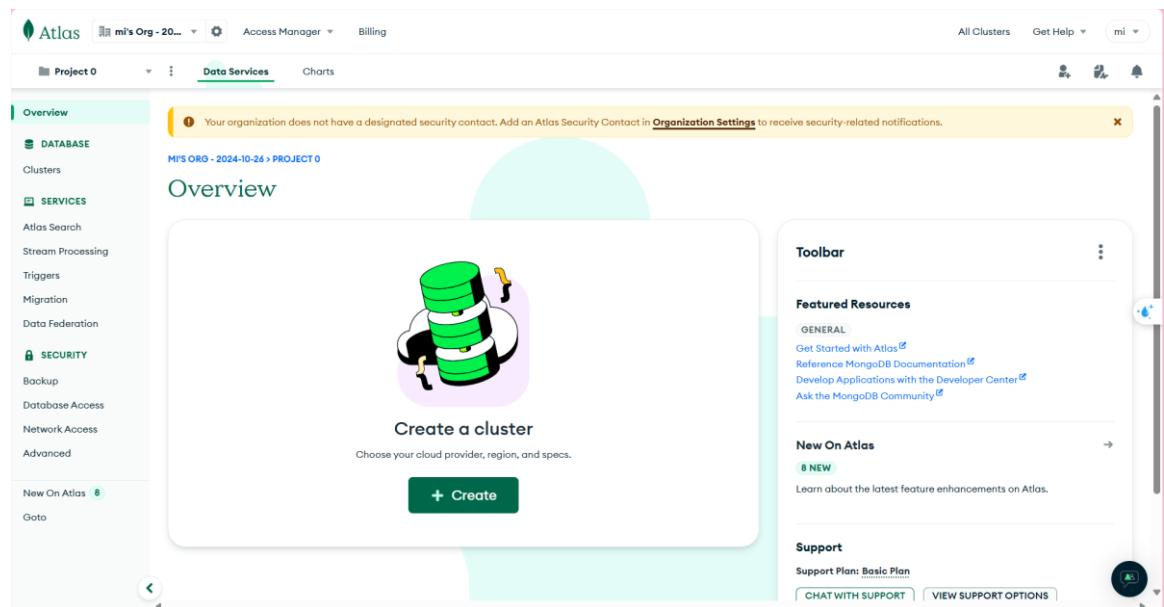
Password*

I agree to the [Terms of Service](#) and [Privacy Policy](#).

Create your Atlas account

Hình 76. Màn hình tạo tài khoản

Bước 3: Tạo một cluster:



Hình 77. Tạo cluster

Giải thích: cluster ở đây có nghĩa là một dịch vụ cơ sở dữ liệu NoSQL dưới dạng dịch vụ trên đám mây công cộng được quản lý bởi MongoDB (có sẵn trong Microsoft Azure, Google Cloud Platform, Amazon Web Services).

Bước 4: Tạo một cluster bằng cách ấn vào **Create** sau khi ấn xong sẽ có giao diện để điều chỉnh các tùy chọn cho cluster của các bạn.

Tiếp theo sẽ có các gói:

- Gói 1: M10 (\$0.10/giờ):
 - Đây là cụm máy chủ chuyên dụng (Dedicated Cluster) dành cho các môi trường phát triển và ứng dụng có lưu lượng truy cập thấp.
 - Cấu hình bao gồm: 10 GB dung lượng lưu trữ, 2 GB RAM và 2 vCPU.
 - Chi phí là 0.10 đô la mỗi giờ. Người dùng sẽ bị tính phí theo mức sử dụng, có thể kết thúc cụm bất kỳ lúc nào.
- Gói 2: Serverless:
 - Cụm này thích hợp cho phát triển ứng dụng, thử nghiệm hoặc các ứng dụng có lưu lượng truy cập biến động.
 - Cụm tự động điều chỉnh quy mô (Auto-scale) theo nhu cầu về dung lượng lưu trữ (lên đến 1 TB), RAM và vCPU.
 - Chi phí là 0.12 đô la cho 1 triệu lệnh đọc. Người dùng sẽ bị tính phí theo mức sử dụng, có thể kết thúc cụm bất kỳ lúc nào.

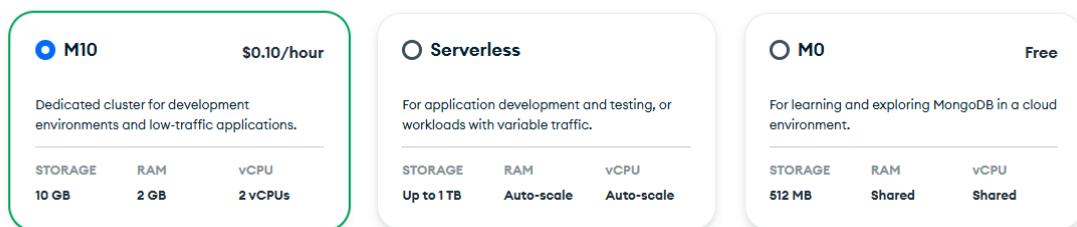
Áo hóa và điện toán đám mây – Nhóm 3

- Phù hợp cho những ứng dụng cần tính linh hoạt cao trong việc thay đổi tài nguyên, tính phí theo lượng sử dụng.
- Gói 3: M0 (Free):
 - Đây là cụm miễn phí, phù hợp cho mục đích học tập và khám phá MongoDB trên môi trường đám mây và chạy trên Local.
 - Cấu hình có dung lượng lưu trữ 512 MB, RAM và vCPU là dạng "Shared" (chia sẻ tài nguyên với các người dùng khác).
 - Phù hợp cho người mới bắt đầu hoặc các dự án nhỏ không yêu cầu tài nguyên lớn.
 - Trong tiểu luận này, nhóm 3 sẽ sử dụng gói M0.

Chú ý: Tất cả các gói này đều có tính phí riêng cho các dịch vụ phụ như chuyển dữ liệu, sao lưu và thuê.

Deploy your cluster

Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.



Hình 78. Các gói dịch vụ có thể đăng ký

Phản thông tin về máy chủ:

Name
You cannot change the name once the cluster is created.

Automate security setup [i](#)

Preload sample dataset [i](#)

Provider

AWS 

Google Cloud 

Azure 

Region

Hong Kong (ap-east-1) 

[★ Recommended \[i\]\(#\)](#) [!\[\]\(e0bac2501a7cd4a4d1e9d8c455795756_img.jpg\) Low carbon emissions \[i\]\(#\)](#)

Tag (optional)
Create your first tag to categorize and label your resources; more tags can be added later. [Learn more.](#)

:

Hình 79. Chọn Provider và Region

Name:

Name
You cannot change the name once the cluster is created.

Automate security setup [i](#)

Preload sample dataset [i](#)

Hình 80. Đặt tên cluster

Các bạn có thể đặt tên cụm máy chủ của mình theo tên mình muốn.

Trang 55

Provider:

Các nhà cung cấp dịch vụ. Các bạn có thể dùng bất cứ cái nào tùy theo yêu cầu của các bạn nếu như các bạn sử dụng các dịch vụ của Amazon thì chọn AWS còn của Google Cloud thì sử dụng Google Cloud và Azure cũng vậy.

Region

Là nơi các bạn muốn đặt cụm máy chủ của mình (nên để các khu vực gần thì nó sẽ ổn định hơn) tuy nhiên việc lựa chọn là của các bạn, các bạn có thể chọn US, Canada...

Bước 5: Khởi tạo cluster:

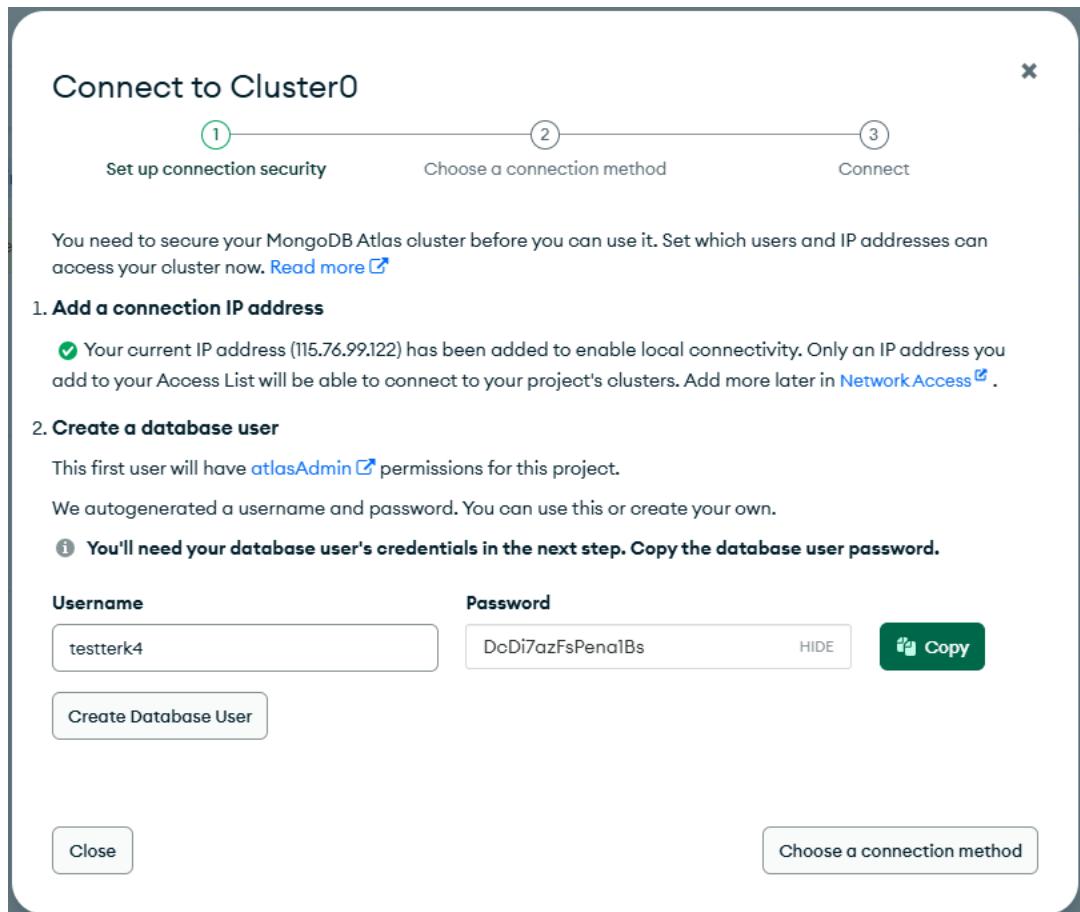


Hình 81. Khởi tạo cluster

Advanced Configuration:

Click chọn vào **Create Deployment**.

Giờ là lúc thiết lập kết nối:



Hình 82. Giao diện tổng thể cluster

Ao hoá và điện toán đám mây – Nhóm 3

Ở phần này các bạn chỉ có thể sử dụng được cluster này với địa chỉ IP này (địa chỉ này được cấp cho Local của các bạn) nếu các bạn muốn thêm các địa IP khác thì có thể thêm sau.

1. Add a connection IP address

Your current IP address (115.76.99.122) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in [Network Access](#).

Hình 83. Thiết lập kết nối

Tại phần này có một cái quan trọng là mật khẩu của các bạn hãy lưu lại nó ở đâu vì nó cần cho việc kết nối sau này nếu mất sẽ không kết nối được, ngoài ra các bạn cũng có thể thay đổi username, password của mình cho dễ nhớ và thuận tiện cho việc quản lí.

Username	Password
testterk4	DcDi7azFsPendoBs
Create Database User	

Hình 84. Thông tin username và password của cluster

Sau khi các bạn đã thiết lập xong thì nhấn **Create Database User**.

2. Create a database user

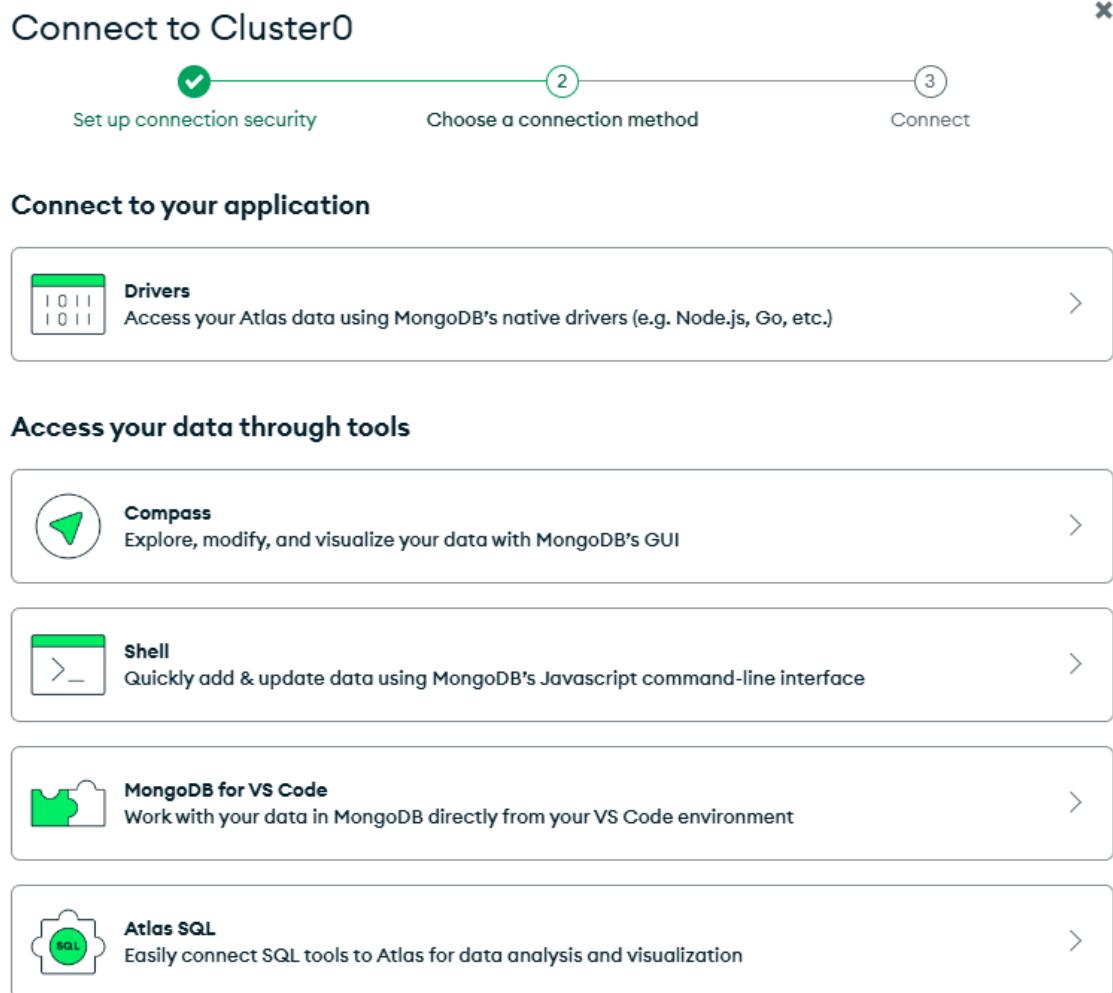
A database user has been added to this project. Create another user later in [Database Access](#). You'll need your database user's credentials in the next step.

Hình 85. Tạo database user

Như này là các bạn đã thành công tạo user.

Bước 6: chọn các phương thức kết nối “choose a connection method”

Tiếp theo, Ở đây có 5 loại kết nối:



Hình 86. Các loại kết nối có thể sử dụng

Drivers (Nhóm mình lựa chọn phương thức này cho web demo)

Cho phép các bạn truy cập dữ liệu trên MongoDB Atlas bằng các driver chính thức của MongoDB. Bạn có thể sử dụng các ngôn ngữ như Node.js, Go, Python, Java, và nhiều ngôn ngữ lập trình khác để kết nối và tương tác với cơ sở dữ liệu. Cách này phù hợp nếu các bạn muốn tích hợp trực tiếp vào ứng dụng.

Lợi ích:

- Cho phép tích hợp trực tiếp MongoDB vào ứng dụng bằng nhiều ngôn ngữ lập trình phổ biến như Node.js, Python, Java, Go, v.v.
- Hỗ trợ toàn diện cho các thao tác CRUD (Create, Read, Update, Delete) và các truy vấn phức tạp.
- Hiệu quả cho việc xây dựng ứng dụng sản xuất và các hệ thống xử lý dữ liệu lớn, vì không yêu cầu giao diện trung gian.

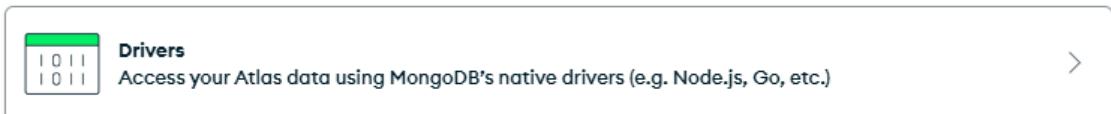
Hạn chế:

Ao hoá và điện toán đám mây – Nhóm 3

- Cần kiến thức lập trình và kỹ thuật kết nối cơ sở dữ liệu, không phù hợp cho những người không chuyên về kỹ thuật.
- Quá trình thiết lập phức tạp hơn các công cụ GUI.
- Có thể gặp khó khăn trong việc cấu hình bảo mật và quản lý kết nối. (Nhóm bọn mình gặp phải rất nhiều vấn đề trong việc gửi dữ liệu từ FE về BE và một số kết nối API khác).

Cách kết nối:

Connect to your application



*Hình 87. Lựa chọn kết nối **Drivers***

Bấm chọn sẽ dẫn tới:

Connect to Cluster0

Set up connection security Choose a connection method Connect (3)

Connecting with MongoDB Driver

1. Select your driver and version
We recommend installing and using the latest driver version.

Driver	Version
Node.js	5.5 or later

2. Install your driver
Run the following on the command line
`npm install mongodb`

[View MongoDB Node.js Driver installation instructions.](#)

3. Add your connection string into your application code

Use this connection string in your application
 View full code sample Show Password [i](#)

```
mongodb+srv://testterk4:DcDi7azFsPenalBs@cluster0.svqkc.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0
```

The password for **testterk4** is included in the connection string for your first time setup. **This password will not be available again after exiting this connect flow.**

RESOURCES

[Get started with the Node.js Driver](#) [Node.js Starter Sample App](#)
[Access your Database Users](#) [Troubleshoot Connections](#)

[Go Back](#) [Done](#)

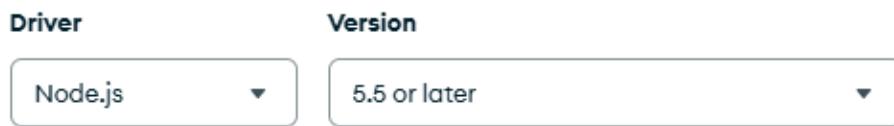
Hình 88. Hoàn thành tạo cluster

Ở phần 1 là chọn các kiểu kết nối cho MongoDB (ở đây nhóm bạn mình sử dụng Node.js sử dụng Javascript và version mới nhất để kết nối và giao tiếp với MongoDB) ngoài ra nếu các bạn sử dụng Python, Ruby thì các bạn có thể ấn vào đó và chọn ngôn ngữ mà MongoDB hỗ trợ.

Connecting with MongoDB Driver

1. Select your driver and version

We recommend installing and using the latest driver version.



Hình 89. Lựa chọn driver và version

Cách cài đặt và kết nối từ frontend về backend:

Bước 1: Cài đặt các công cụ cần thiết:

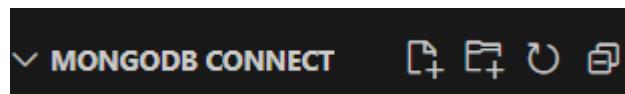
Trước khi bắt đầu, hãy đảm bảo bạn đã cài đặt các công cụ sau:

- Cài đặt Nodejs [tại đây](#).
- MongoDB
- npm (đi kèm với Node.js)
- Express
- React
- Mongoose (để kết nối MongoDB)
- IDE (như VSCode)

Bước 2: tải và cài đặt VS code [tại đây](#).

Sau khi tải xong thì bắt đầu:

- Tạo một dự án mới trên VSCode:
- Mở VSCode.
- Chọn File -> Open Folder.
- Tạo một thư mục mới cho dự án của bạn.



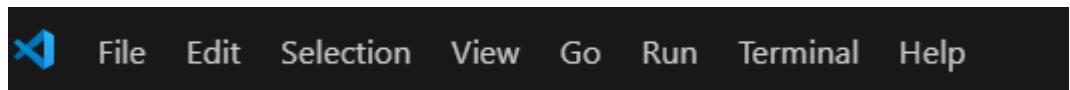
Hình 90. Ví dụ thư mục

Tiếp theo tạo thêm 2 thư mục con cho phía frontend (mình đặt là client) và backend (mình đặt là server).

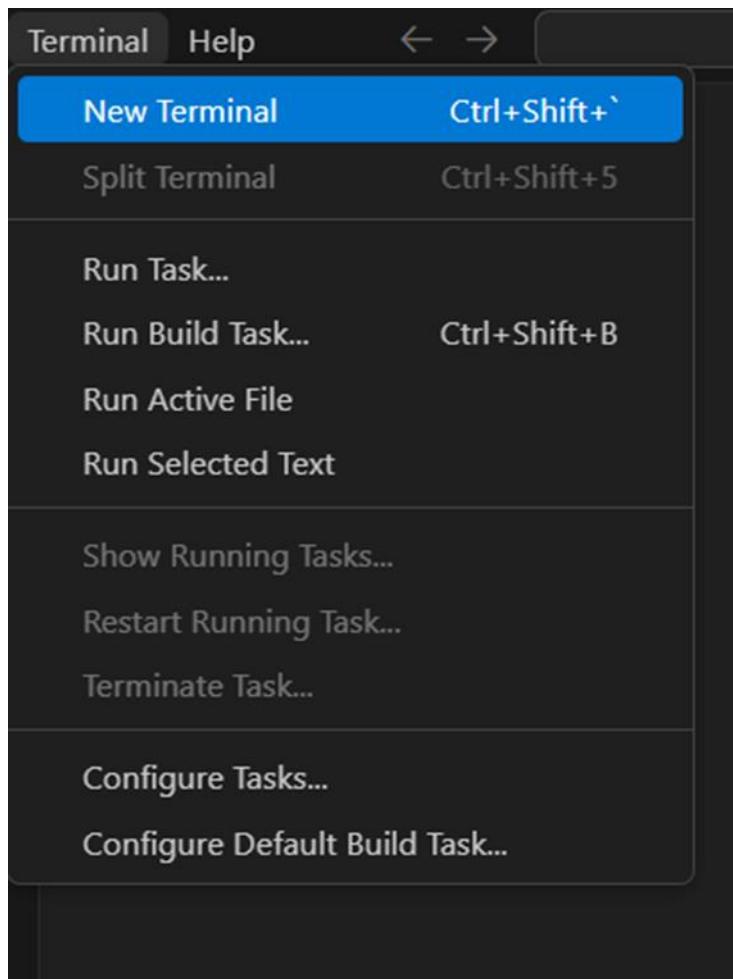
Các bạn có thể tạo trên UI bằng cách click vào **tạo thư mục** và gõ backend và frontend.

Hoặc có thể sử dụng lệnh **mkdir** backend, frontend để tạo bằng lệnh các bạn có thể sử dụng lệnh bằng cách.

Trên góc trái màn hình chọn **Terminal** nhấn chọn **New Terminal**.

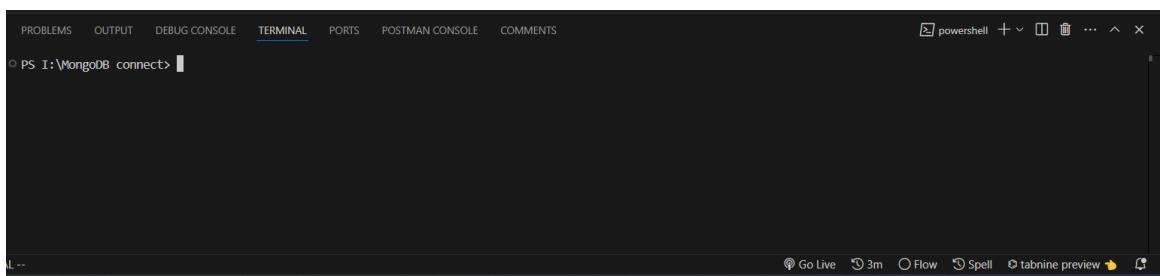


Hình 91. Chọn terminal trên taskbar



Hình 92. Chọn new terminal

Sau đó ở dưới góc màn hình sẽ hiện ra:



Hình 93. Bảng terminal mới

Bây giờ hãy nhập lệnh vào và ấn enter:

- PS G:\MONGODB CONNECT> `mkdir backend, frontend`

Directory: G:\MONGODB CONNECT

Mode	LastWriteTime	Length	Name
---	---	---	---
d-----	10/26/2024 5:18 PM		backend
d-----	10/26/2024 5:18 PM		frontend

Hình 94. Tạo hai thư mục backend và frontend

Sau khi hoàn thành thì chúng ta sẽ tải các package cần thiết.

Tiếp theo dùng lệnh `npm init -y` để tạo file **package.json** cho Node.js.

Đây là tệp tin quan trọng trong một dự án Node.js, nó chứa thông tin về dự án, bao gồm:

- Tên dự án: Tên của ứng dụng hoặc thư viện.
- Phiên bản: Phiên bản hiện tại của dự án.
- Mô tả: Một mô tả ngắn gọn về dự án.
- Điểm vào (entry point): Tệp tin chính mà ứng dụng sẽ bắt đầu (mặc định là `index.js`).
- Script: Các lệnh script có thể chạy bằng npm, chẳng hạn như `test`, `start`, v.v.
- Thư viện phụ thuộc: Danh sách các thư viện mà dự án phụ thuộc vào.

Tiếp theo là tải các gói cần thiết bằng lệnh:

“npm install express jsonwebtoken mongoose cookie-parser dotenv axios bcryptjs”

Express:

- Framework web nhẹ cho Node.js, giúp xây dựng ứng dụng web và API với định tuyến, middleware và hỗ trợ các phương thức HTTP.

Mongoose:

- Thư viện cho MongoDB trong Node.js, giúp tương tác với cơ sở dữ liệu, định nghĩa schema và thực hiện các truy vấn dễ dàng.

dotenv:

- Thư viện quản lý biến môi trường trong Node.js. Giúp lưu trữ thông tin nhạy cảm trong tệp `.env` mà không cần cứng hóa trong mã nguồn.

Bcrypt:

- Thư viện mã hóa và so sánh mật khẩu. Sử dụng thuật toán mạnh mẽ để bảo vệ thông tin người dùng bằng cách lưu trữ mật khẩu đã mã hóa.

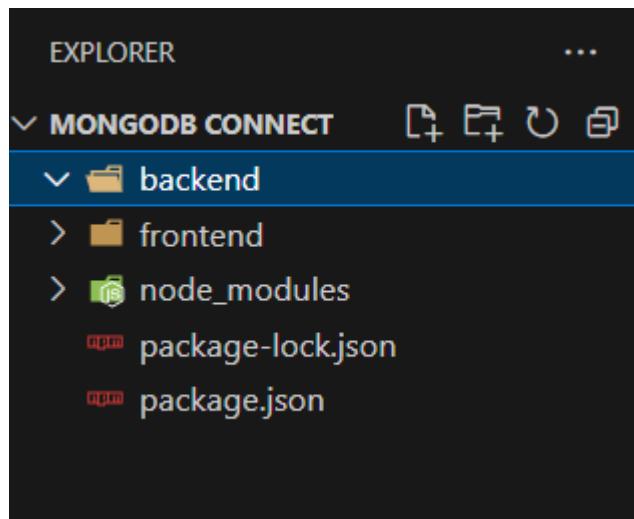
jsonwebtoken:

- Thư viện cho phép tạo và xác minh JSON Web Tokens (JWT). Thường dùng để xác thực người dùng, giúp kiểm tra quyền truy cập khi gửi yêu cầu đến server.

Dán tất cả vào terminal và nhấn Enter:

```
● PS G:\MONGODB CONNECT>
  >> npm install express jsonwebtoken mongoose cookie-parser dotenv axios bcryptjs
    added 110 packages, and audited 111 packages in 26s
    16 packages are looking for funding
      run `npm fund` for details
    found 0 vulnerabilities
○ PS G:\MONGODB CONNECT>
```

Hình 95. Các package sẽ tự động tải về



Hình 96. Cây thư mục sau khi cài đặt

Bây giờ tại mục backend tạo một file server.js.

Tại thư mục server.js.

```
import express from "express";

const app = express();

Tabnine | Edit | Test | Explain | Document | Ask
app.listen(5000, () =>{
    console.log("Server started at http://localhost:5000");
});
```

Hình 97. Ví dụ tạo server express

Ngoài ra bạn có thể quay lại file packet.json để chỉnh sửa cho đúng với các đường dẫn.

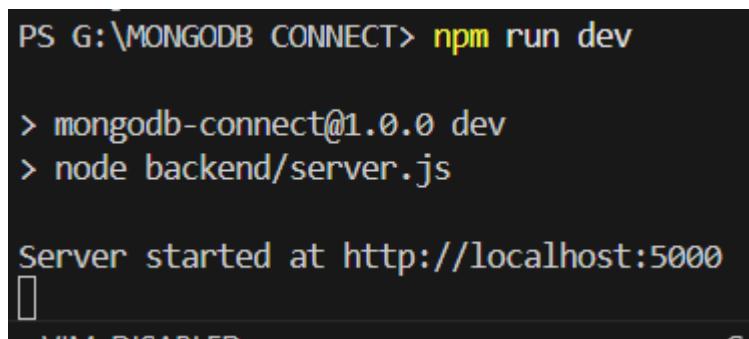
```
package.json > {} scripts > dev
1  {
2      "name": "mongodb-connect",
3      "version": "1.0.0",
4      "main": "backend/server.js",
5      "scripts": [
6          "dev": "node backend/server.js"
7      ],
8      "keywords": [],
9      "author": "",
10     "license": "ISC",
11     "description": "",
12     "dependencies": {
13         "axios": "^1.7.7",
14         "bcryptjs": "^2.4.3",
15         "cookie-parser": "^1.4.7",
16         "dotenv": "^16.4.5",
17         "express": "^4.21.1",
18         "jsonwebtoken": "^9.0.2",
19         "mongoose": "^8.7.3"
20     }
}
```

Hình 98. Định dạng file package.json

Sửa lại các đường dẫn đến đúng file server của các bạn và thêm “type”: “module” vào để sử dụng ES model.

Áo hóa và điện toán đám mây – Nhóm 3

Sau khi chỉnh sửa xong các bạn có thể dùng lệnh npm run dev coi đã kết nối được chưa.



```
PS G:\MONGODB CONNECT> npm run dev

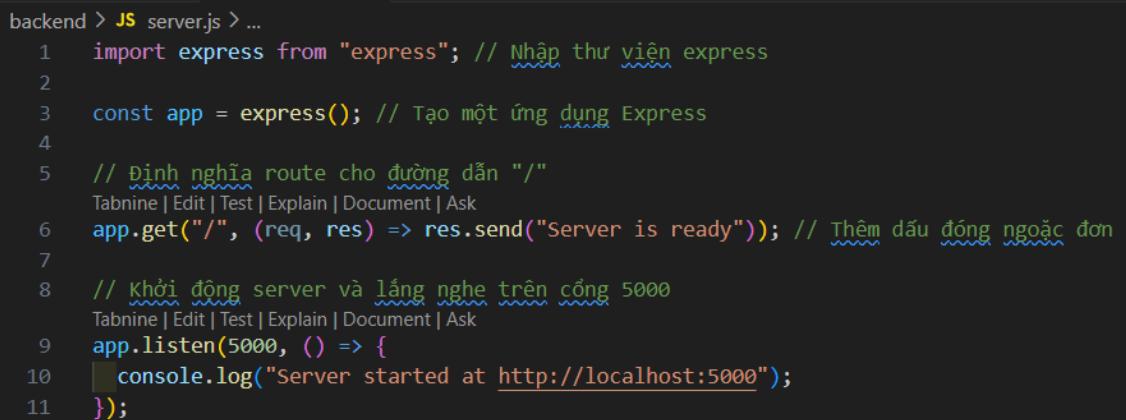
> mongodb-connect@1.0.0 dev
> node backend/server.js

Server started at http://localhost:5000

```

Hình 99. Đã kết nối

Tiếp theo thì bạn có thể thử đưa một nội dung lên local của mình.



```
backend > JS server.js > ...
1 import express from "express"; // Nhập thư viện express
2
3 const app = express(); // Tạo một ứng dụng Express
4
5 // Định nghĩa route cho đường dẫn "/"
6 app.get("/", (req, res) => res.send("Server is ready")); // Thêm dấu đóng ngoặc đơn
7
8 // Khởi động server và lắng nghe trên cổng 5000
9 app.listen(5000, () => {
10   console.log("Server started at http://localhost:5000");
11 });
12
```

Hình 100. Script đưa nội dung lên local

Nhưng khi bạn lên thì không thấy nội dung bạn vừa đổi, vấn đề ở đây là local không thể tự động cập nhật và bạn phải chạy lại lệnh **npm run dev** một lần nữa và để có thể tự động cập nhật mỗi khi chỉnh sửa chúng ta có thể dùng một gói cài đặt của **nodemon**.

Cú pháp: **npm install nodemon -D** (D ở đây là mức độ DEV).

Sau khi cài xong các bạn hay quay lại file **package.json** để chỉnh sửa lại dòng.

Và chạy lại **npm run dev** một lần nữa và chỉnh sửa nội dung trong lúc đó xem nó cập nhật không.

```
PS G:\MONGODB CONNECT> npm run dev

> netflix-clone@1.0.0 dev
> nodemon backend/server.js

[nodemon] 3.1.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node backend/server.js`
Server started at http://localhost:3000
[]
```

Hình 101. Kiểm tra lại server

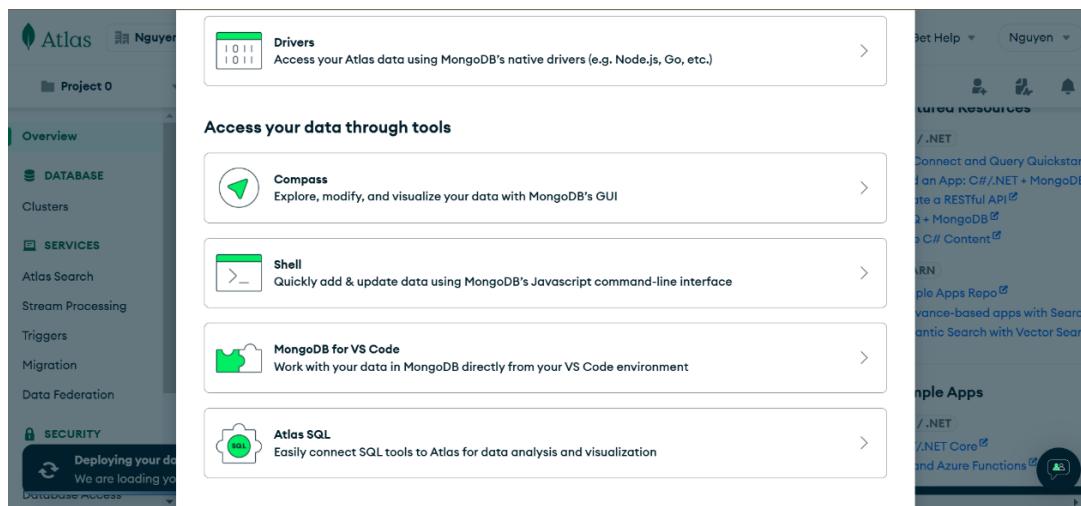
```
Server started at http://localhost:3000
[nodemon] restarting due to changes...
[nodemon] starting `node backend/server.js`
[nodemon] restarting due to changes...
[nodemon] starting `node backend/server.js`
Server started at http://localhost:3000
[]
```

Hình 102. Mỗi khi bạn ấn save thì nó sẽ tự động cập nhật

4. Cài đặt Shell và MongoDB for VSCode.

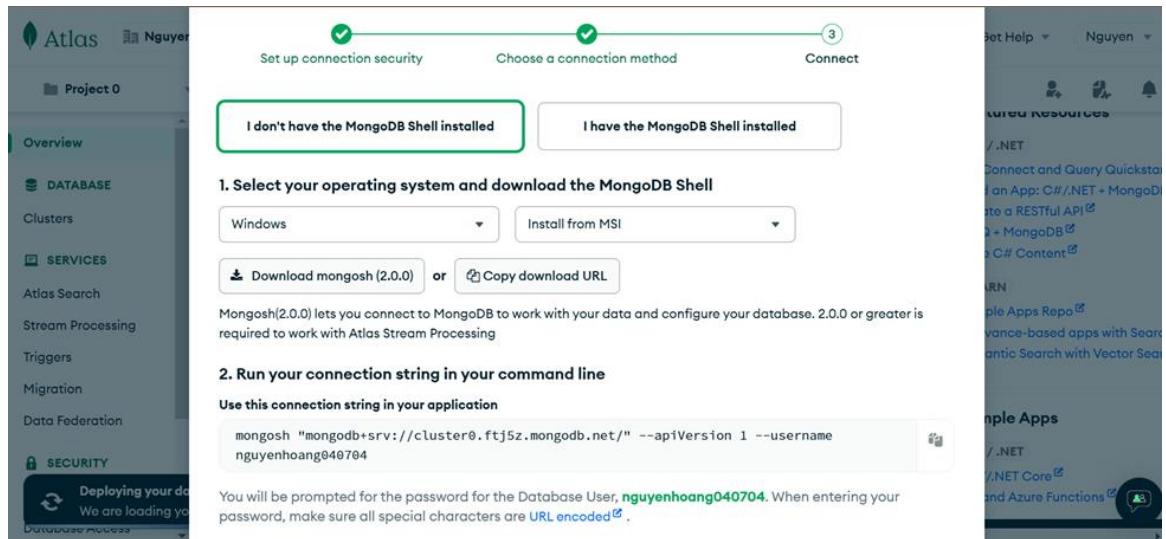
Shell: dùng dòng lệnh kết nối và truy vấn Database trên MongoDB Atlas.

Bước 1: Sau khi tạo database, vào phần connect:



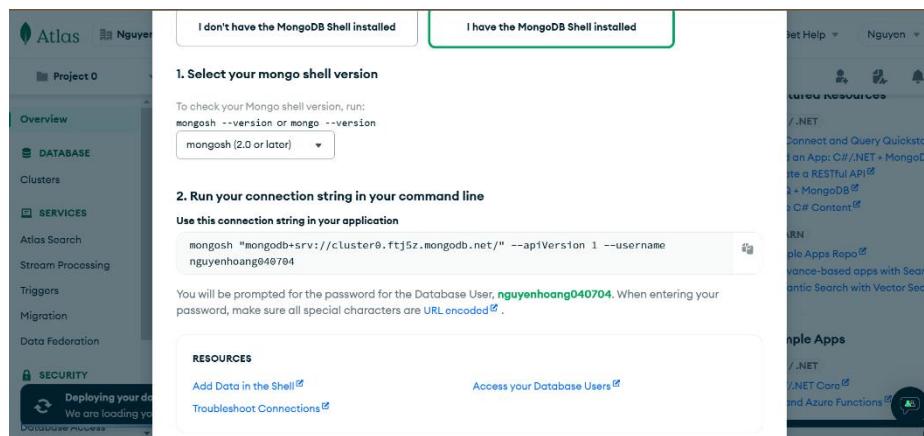
Hình 103. Chọn kết nối

Bước 2: Chọn Shell



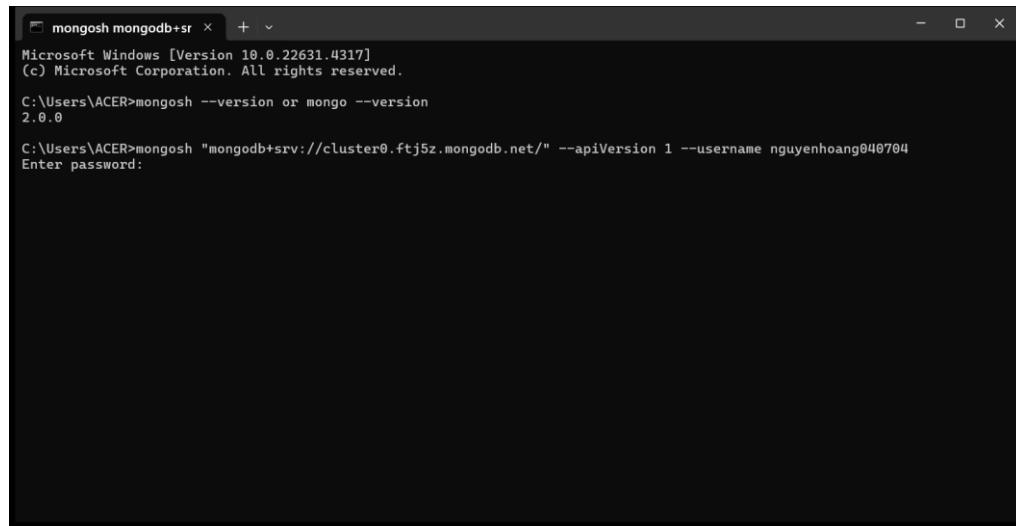
Hình 104. Lựa chọn MongoDB Shell

Tại đây nếu bạn chưa tải MongoDB shell installer thì lựa chọn bản mongosh phù hợp sau đó tải về, tiếp theo copy connection string và chạy trên cmd.



Hình 105. Copy connection string

Áo hóa và điện toán đám mây – Nhóm 3



```
mongosh mongodb+srv + ~
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

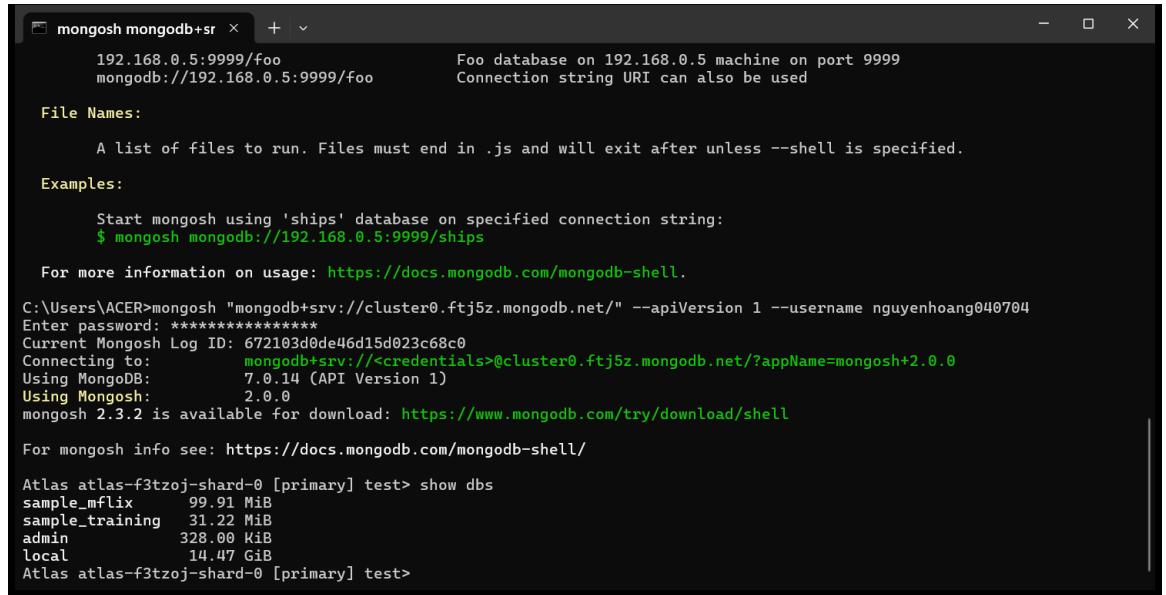
C:\Users\ACER>mongosh --version or mongo --version
2.0.0

C:\Users\ACER>mongosh "mongodb+srv://cluster0.ftj5z.mongodb.net/" --apiVersion 1 --username nguyenhoang040704
Enter password:
```

Hình 106. Kiểm tra mongosh

Nếu bạn đã cài đặt mongosh, tại cmd gõ lệnh **mongosh --version or mongo --version** để xem version của **mongosh**, chọn phiên bản phù hợp, copy **connection string** chạy ở **cmd**.

Nhập mật khẩu cho username đã tạo (ở đây là **nguyenhoang040704**), nhấn enter.



```
mongosh mongodb+srv + ~
192.168.0.5:9999/foo          Foo database on 192.168.0.5 machine on port 9999
                               Connection string URI can also be used

File Names:
A list of files to run. Files must end in .js and will exit after unless --shell is specified.

Examples:
Start mongosh using 'ships' database on specified connection string:
$ mongosh mongodb://192.168.0.5:9999/ships

For more information on usage: https://docs.mongodb.com/mongodb-shell/

C:\Users\ACER>mongosh "mongodb+srv://cluster0.ftj5z.mongodb.net/" --apiVersion 1 --username nguyenhoang040704
Enter password: *****
Current Mongosh Log ID: 672103d0de46d15d023c68c0
Connecting to:      mongodb+srv://<credentials>@cluster0.ftj5z.mongodb.net/?appName=mongosh+2.0.0
Using MongoDB:     7.0.14 (API Version 1)
Using Mongosh:     2.0.0
mongosh 2.3.2 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Atlas atlas-f3tzoj-shard-0 [primary] test> show dbs
sample_flix      99.91 MiB
sample_training   31.22 MiB
admin            328.00 KiB
local             14.47 GiB
Atlas atlas-f3tzoj-shard-0 [primary] test>
```

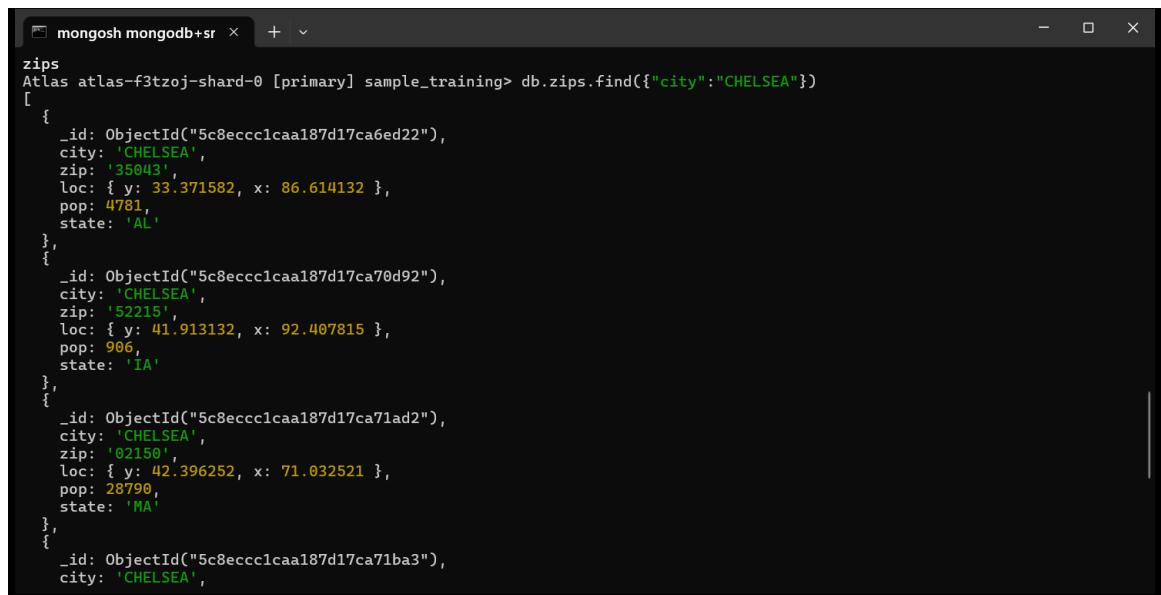
Hình 107. Setup mongosh trên cmd

Demo:

- Dùng lệnh **show dbs** để truy vấn danh sách các database.
- **use sample_training** để hướng tới database sample_training.
- **show collections** để hiển thị các collection trong database.

Ví dụ tìm danh sách các document có CITY=CHELSEA trong collection zips.

db.zips.find({“city”:”CHELSEA”})



```
mongosh mongodb+srv
Atlas atlas-f3tzoj-shard-0 [primary] sample_training> db.zips.find({"city": "CHEALSEA"})
[
  {
    _id: ObjectId("5c8ecc1caa187d17ca6ed22"),
    city: 'CHEALSEA',
    zip: '35043',
    loc: { y: 33.371582, x: 86.614132 },
    pop: 4781,
    state: 'AL'
  },
  {
    _id: ObjectId("5c8ecc1caa187d17ca70d92"),
    city: 'CHEALSEA',
    zip: '52215',
    loc: { y: 41.913132, x: 92.407815 },
    pop: 906,
    state: 'IA'
  },
  {
    _id: ObjectId("5c8ecc1caa187d17ca71ad2"),
    city: 'CHEALSEA',
    zip: '02150',
    loc: { y: 42.396252, x: 71.032521 },
    pop: 28790,
    state: 'MA'
  },
  {
    _id: ObjectId("5c8ecc1caa187d17ca71ba3"),
    city: 'CHEALSEA',
  }
]
```

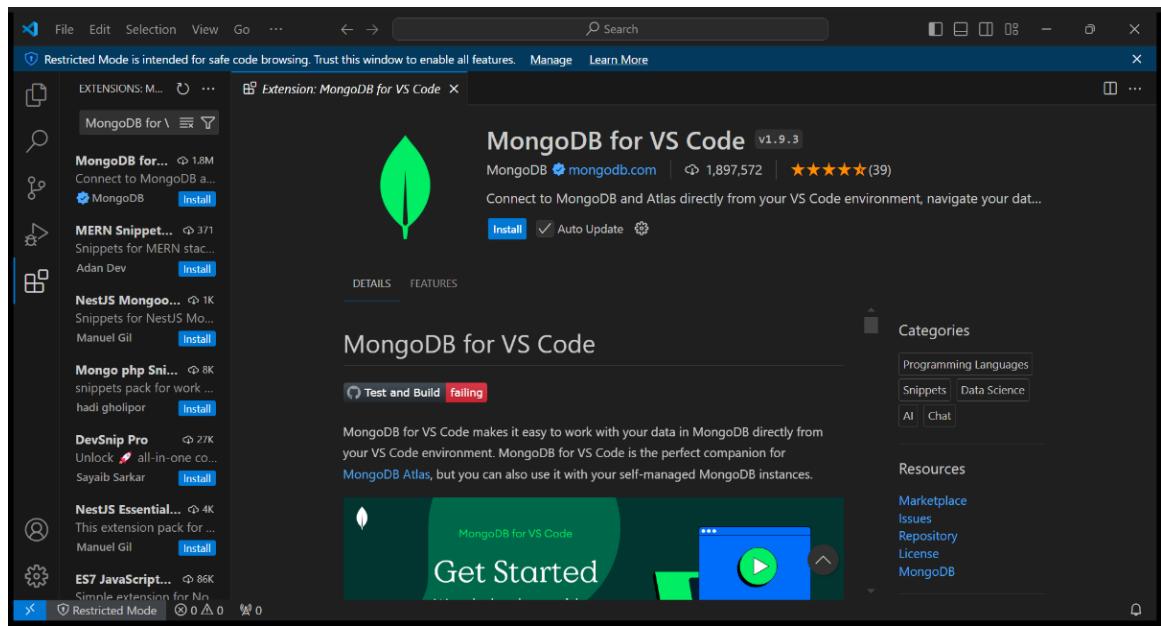
Hình 108. Tìm kiếm thành phố CHEALSEA

Kết nối thành công bằng Shell.

MongoDB for VSCode

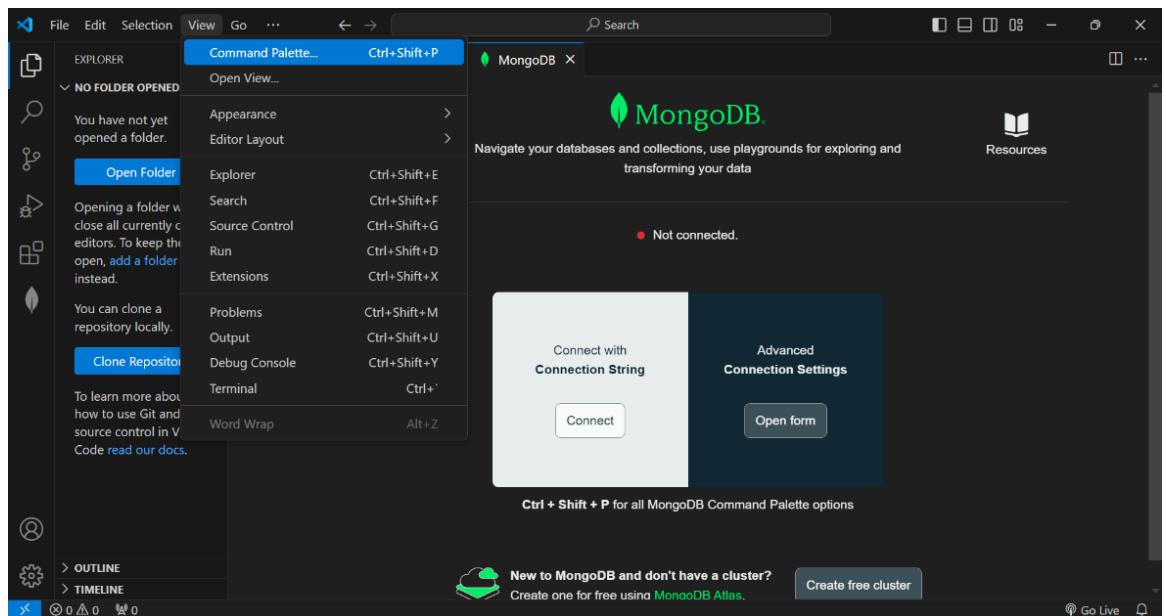
MongoDB for VSCode: MongoDB kết nối với VSCode bằng Extensions “MongoDB for VS Code”.

Bước 1: Mở VSCode, bấm Extensions, tìm “MongoDB for VS Code”, và nhấn install.



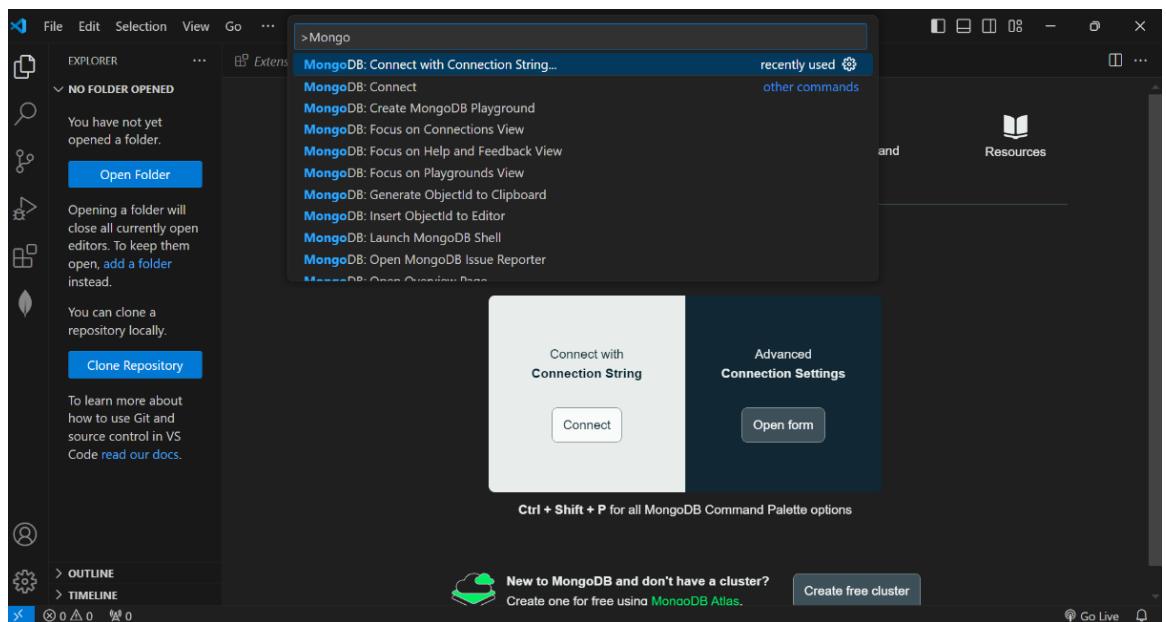
Hình 109. Tải extension về máy

Bước 2: Tại VSCode, chọn View, chọn Command Palette.



Hình 110. Chọn Command Palette

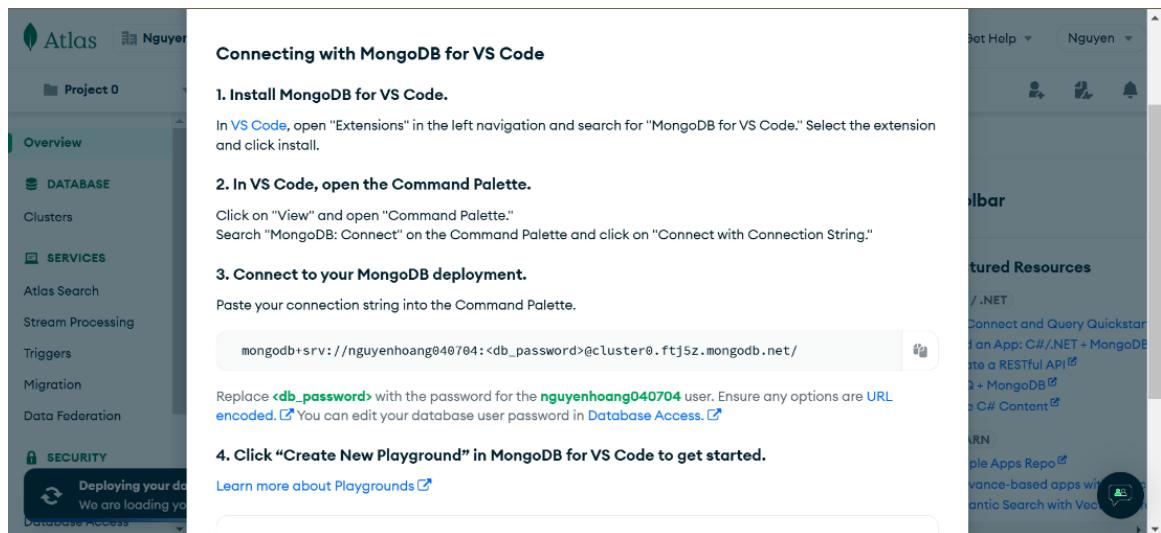
Bước 3: Tìm kiếm "MongoDB: Connect" trên Command Palette và nhấp vào "Connect with Connection String".



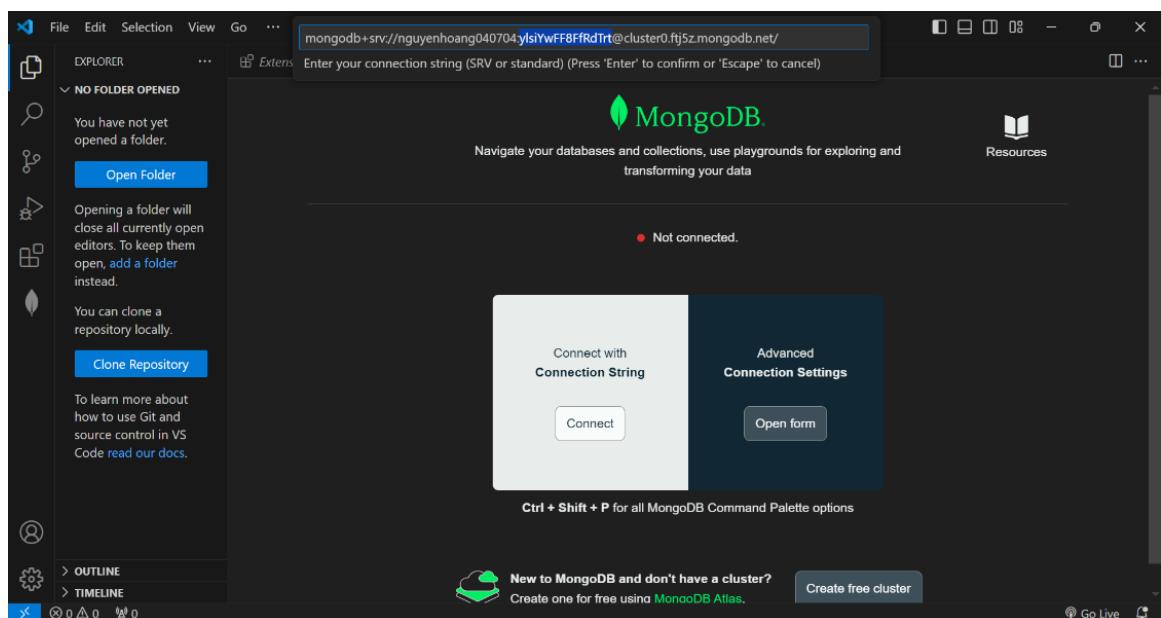
Hình 111. Chọn Connect with Connection String

Bước 4: Copy connection string vào Command Palette, nhấn Enter.

Áo hóa và điện toán đám mây – Nhóm 3



Hình 112. Lấy connection string



Hình 113. Thay vào đây

Lưu ý quan trọng: Thay thế chuỗi <db_password> trong connection string bằng password của username

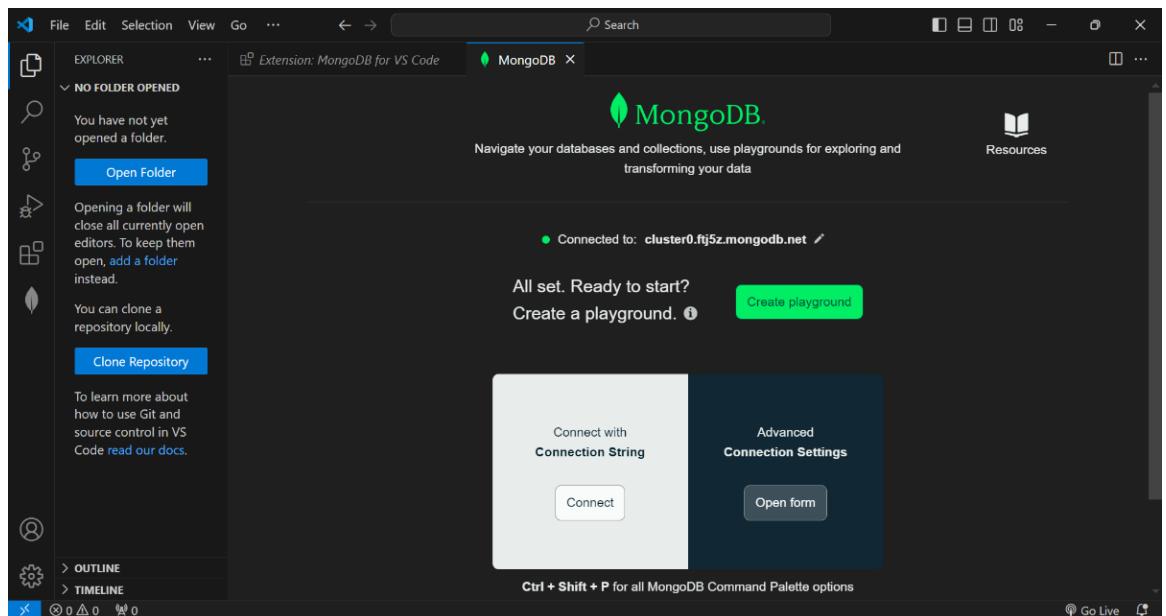
Ví dụ:

mongodb+srv://nguyenhoang040704:<db_password>@cluster0.ftj5h.mongodb.net/ thành

mongodb+srv://nguyenhoang040704:123456789@cluster0.ftj5h.mongodb.net/

Màn hình kết quả kết nối thành công:

Áo hóa và điện toán đám mây – Nhóm 3



Hình 114. Thành công

Demo: truy vấn thử ví dụ tìm danh sách các document có CITY= CHELSEA trong collection zips.

db.zips.find({“city”:”CHELSEA”}).

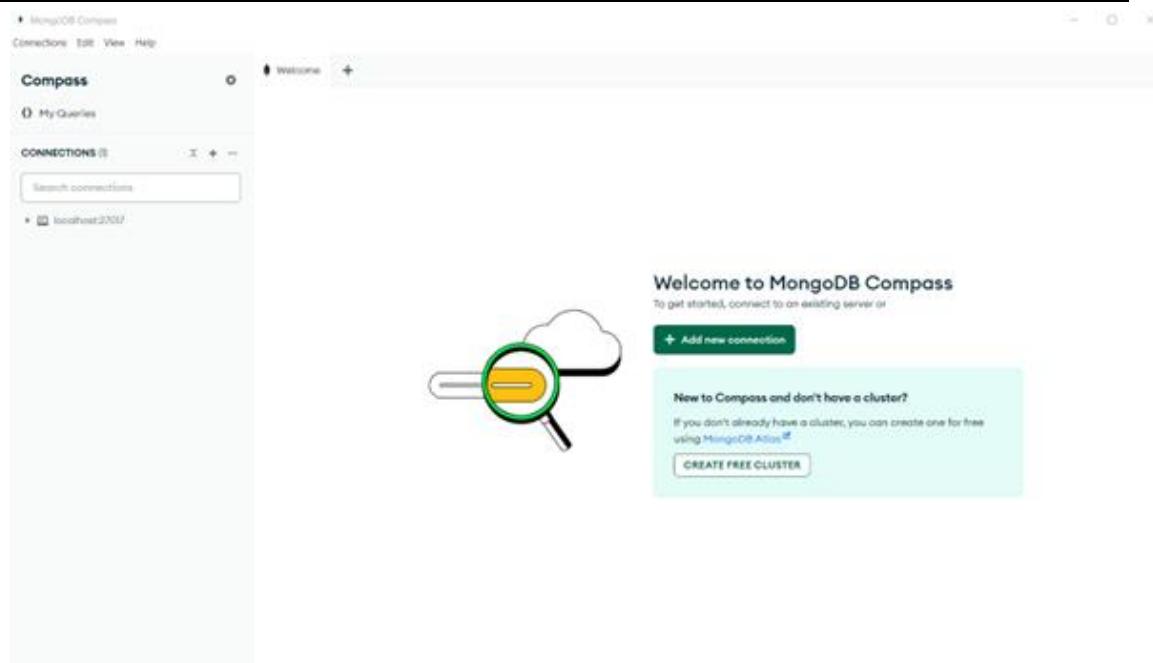
A screenshot of the MongoDB playground in VS Code. The sidebar shows a list of databases and collections, including "MONGDB" and "PLAYGROUNDS". In the playground editor, a JavaScript code block contains the command "db.zips.find({“city”:”CHELSEA”})". To the right, the "Playground Result" panel displays the results of the query as an array of documents. Each document includes fields like _id, \$oid, city, zip, loc, pop, and state. The results show two entries for the city "CHELSEA". At the bottom of the playground editor, there are status indicators for line and column numbers, and a Go Live button.

Hình 115. Demo thành công

5. MongoDB Compass

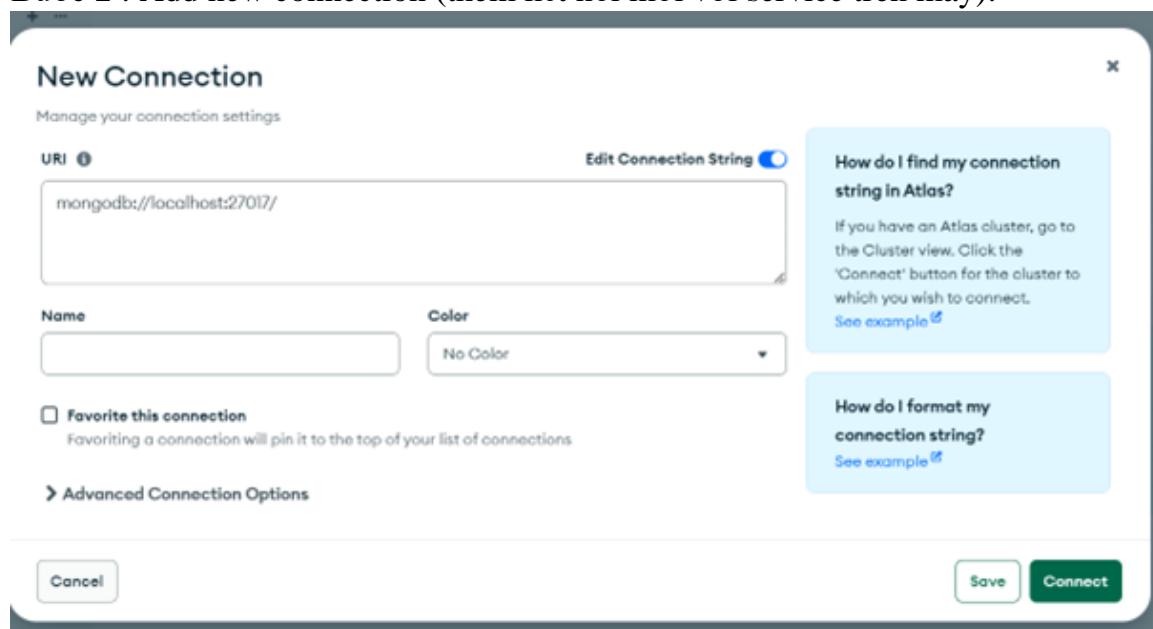
Tải MongoDB như hướng dẫn ở trên.

Bước 1 : Mở MongoDB Compass:



Hình 116. Giao diện MongoDB Compass

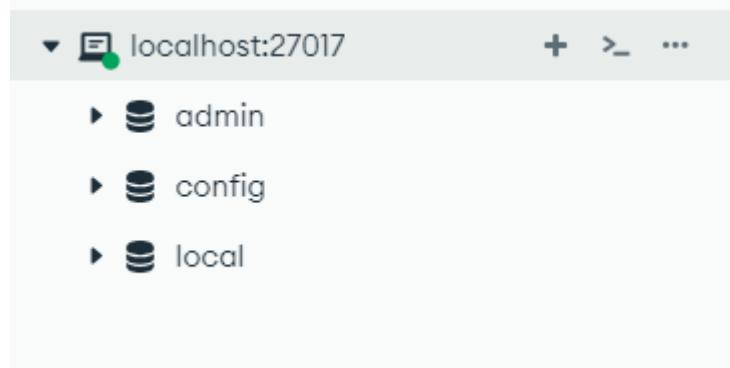
Bước 2 : Add new connection (thêm kết nối mới với service trên máy):



Hình 117. Thêm mới connection

config lại thông tin và bấm connect.

Bước 3:



Hình 118. Thông tin localhost

Connect thành công

Ví dụ cụ thể: Thiết kế DB:

Các file json setup:

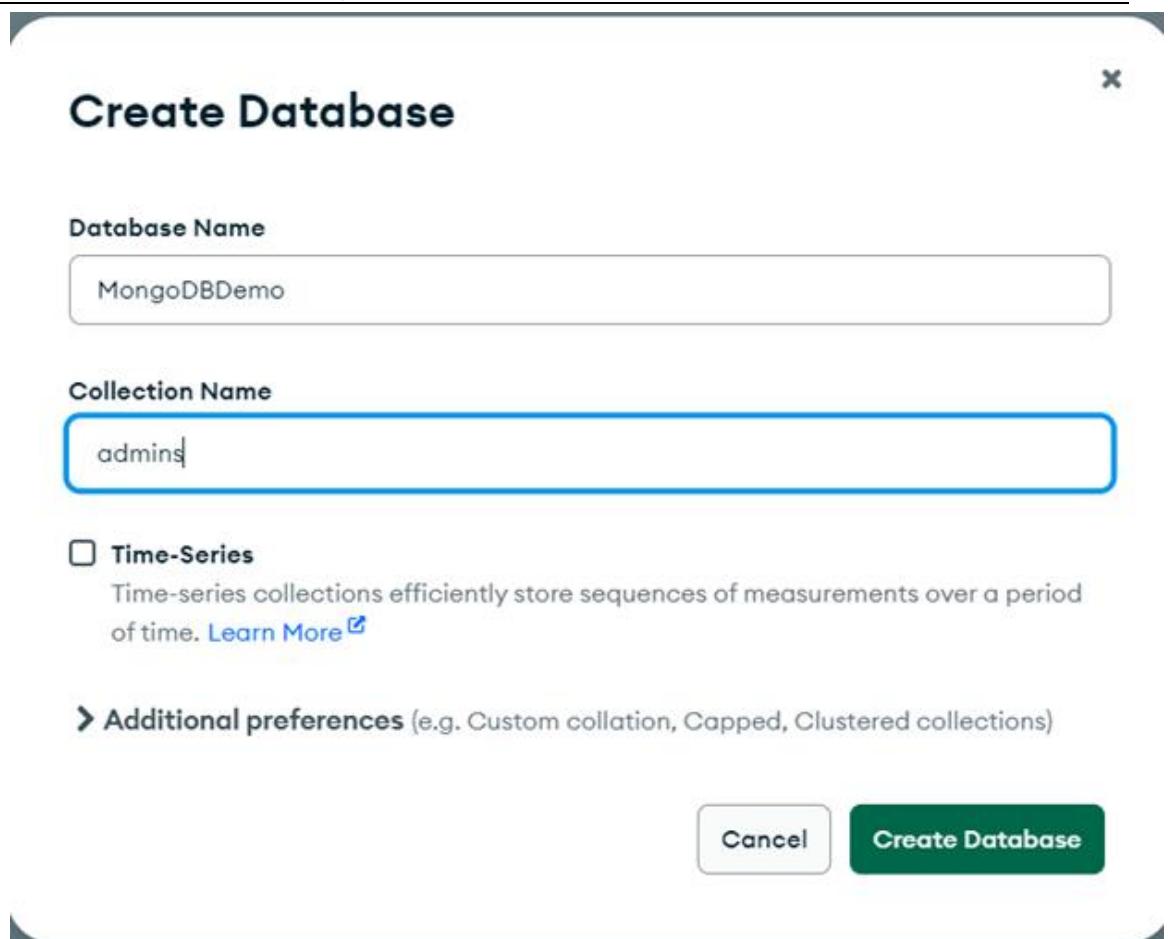
- admins.json
- coursecategories.json
- courses.json
- coursetopics.json
- facebookusers.json
- lecturers.json
- localusers.json
- topweeks.json

Hình 119. Các file json setup

Setup Database

Bước 1: Vào MongoDB Compass.

Bước 2: Tạo db mới kèm theo 1 collection khởi tạo.



Hình 120. Tạo database

Bước 3: Thực hiện import json.

The screenshot shows the MongoDB interface with the 'CustomerCareDB' database selected. In the top right, there are 'ADD DATA' and 'EXPORT DATA' buttons. A modal window is open with the title 'Import JSON or CSV file' and an 'Insert document' button.

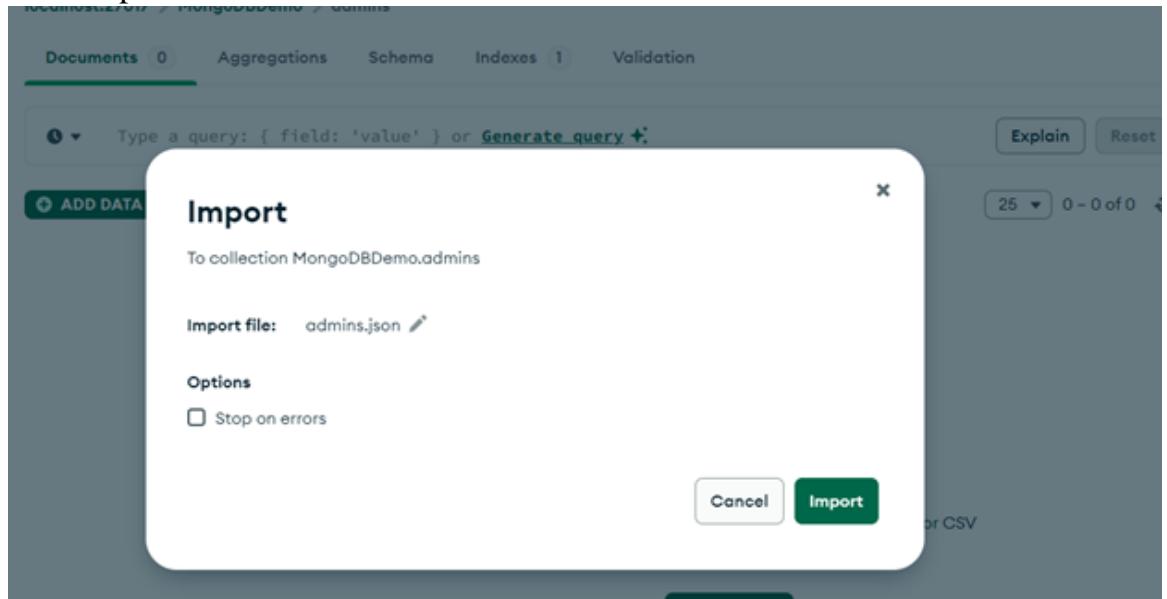
Hình 121. Import file json

Bước 4 : Chọn file admins.json.

Name		Date modified	Type
	admins.json	2/14/2021 5:32 PM	JSON Source File
	coursecategories.json	2/14/2021 5:32 PM	JSON Source File
	courses.json	2/14/2021 5:32 PM	JSON Source File
	coursetopics.json	2/14/2021 5:32 PM	JSON Source File
	facebookusers.json	2/14/2021 5:32 PM	JSON Source File
	lecturers.json	2/14/2021 5:32 PM	JSON Source File
	localusers.json	2/14/2021 5:32 PM	JSON Source File
	topweeks.json	2/14/2021 5:32 PM	JSON Source File

Hình 122. Chọn file json

Bấm import.



Hình 123. Import lên

Thực hiện thành công.

Bước 5: Lần lượt tạo các collection tiếp theo.

III. XÂY DỰNG ỨNG DỤNG WEB

1. Triển khai ứng dụng web

Trong tiểu luận này, nhóm chúng em sử dụng công nghệ MERN stack để xây dựng ứng dụng web mạng xã hội. (mern-stack, 2023)

MERN stack là nguyên bộ combo open source các công nghệ đều liên quan đến Javascript là cũng hot nhất hiện nay: MongoDB, ExpressJS, React, NodeJS. Người ta dùng MERN stack để xây dựng React Universal App. Trong đó:

MongoDB:

- Là một noSQL database hot nhất hiện nay. MongoDB thường đi với Mongoose – một library để giao tiếp với MongoDB dễ dàng hơn.
- Nghiên cứu cách làm việc với collection, thực thi các câu query để truy vấn dữ liệu.

Node.js:

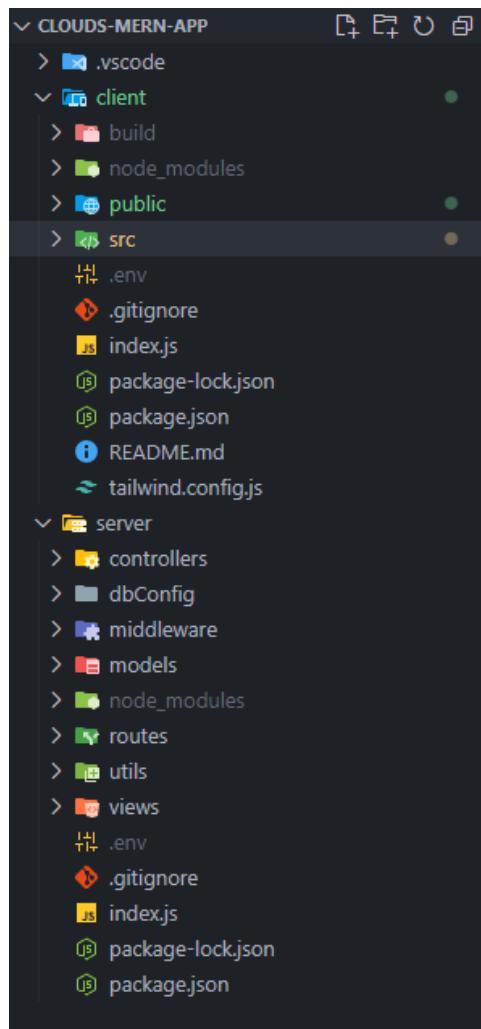
- Nghiên cứu cú pháp, câu lệnh của javascript nói chung và của NodeJS nói riêng, sử dụng npm để install các package cần thiết cho ứng dụng web.

Express

- Là web framework được xây dựng bằng Javascript chạy trên nền Node.js.
- Nghiên cứu cách sử dụng các tính năng như routing, middlewares, ... để thiết kế các API và xây dựng backend.

React:

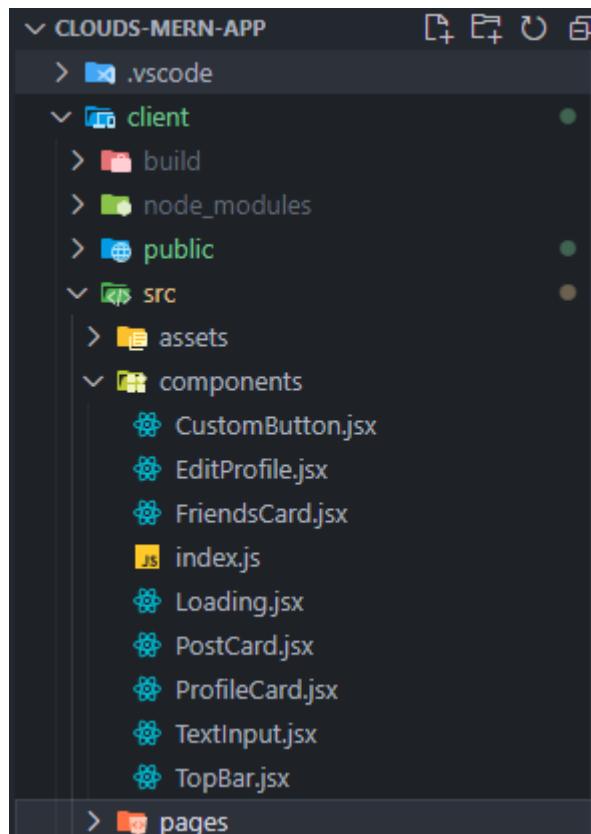
- Nghiên cứu cách sử dụng một component để render html, hiểu được vòng đời của component, quản lý state, props của component, ...



Hình 124. Cấu trúc của ứng dụng web

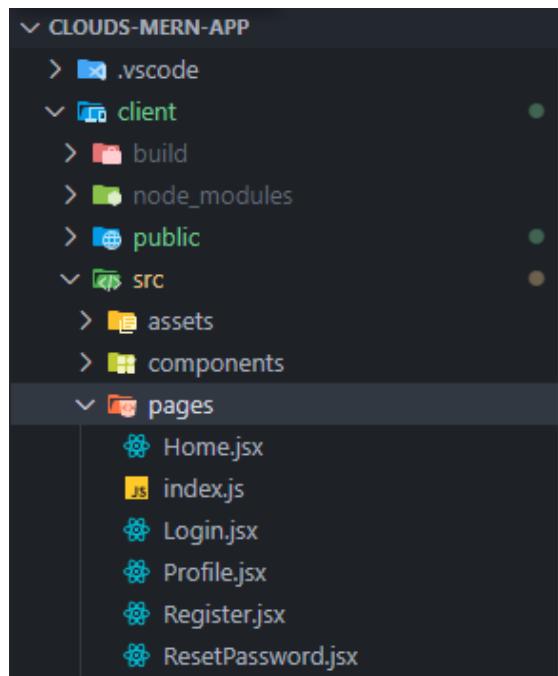
a. Client

Thư mục components đảm nhiệm các thành phần giao diện trong trang, chẳng hạn như các nút, thanh công cụ trên, hình đại diện, giao diện chờ, ...



Hình 125. Thư mục components

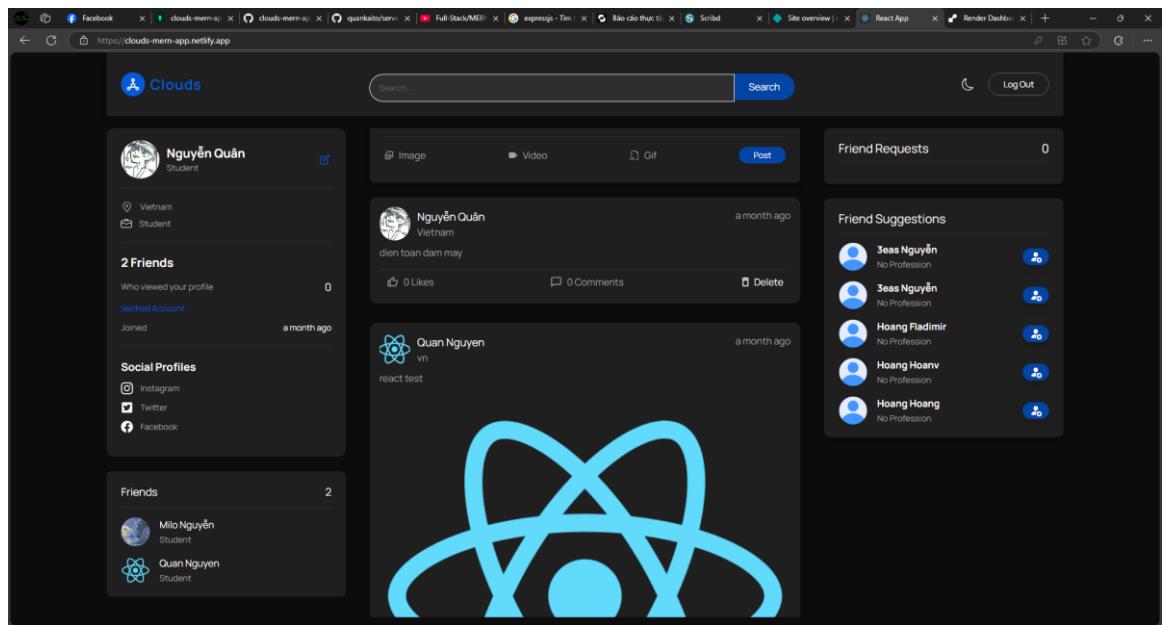
Thư mục pages đảm nhiệm tạo giao diện các trang khác nhau để người dùng chuyển qua lại, bắt đầu với trang đăng ký/đăng nhập, sau đó dẫn người dùng đến trang home và các trang khác.



Hình 126. Thư mục pages

Thư mục này dùng để xây dựng front-end (giao diện người dùng).

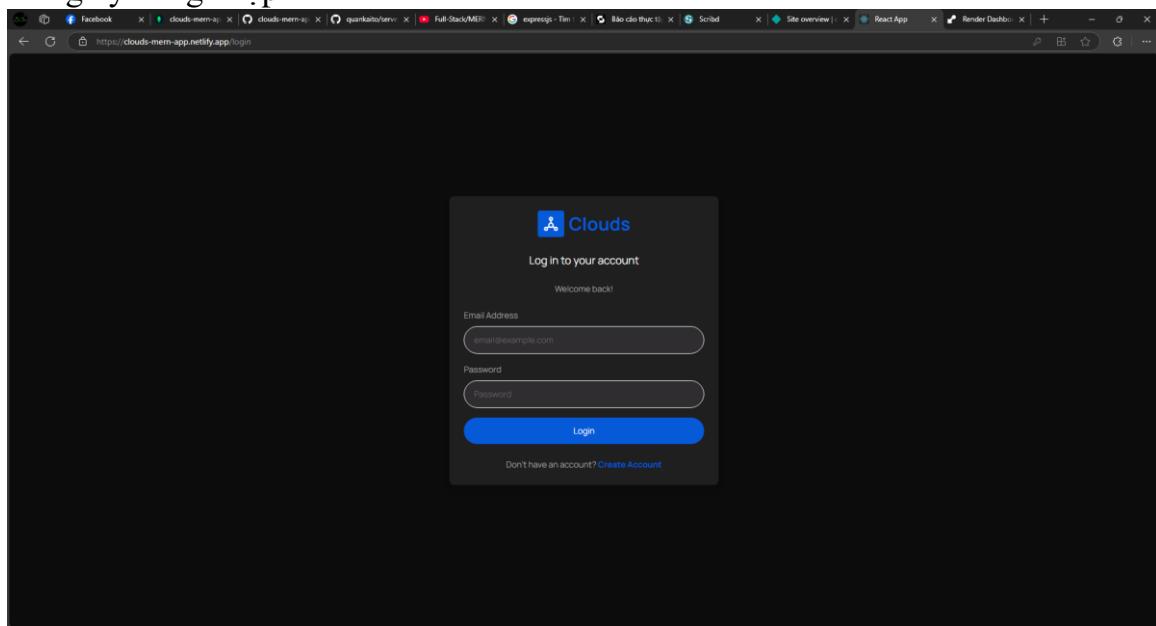
Ao hoá và điện toán đám mây – Nhóm 3



Hình 127. Giao diện chính của trang web

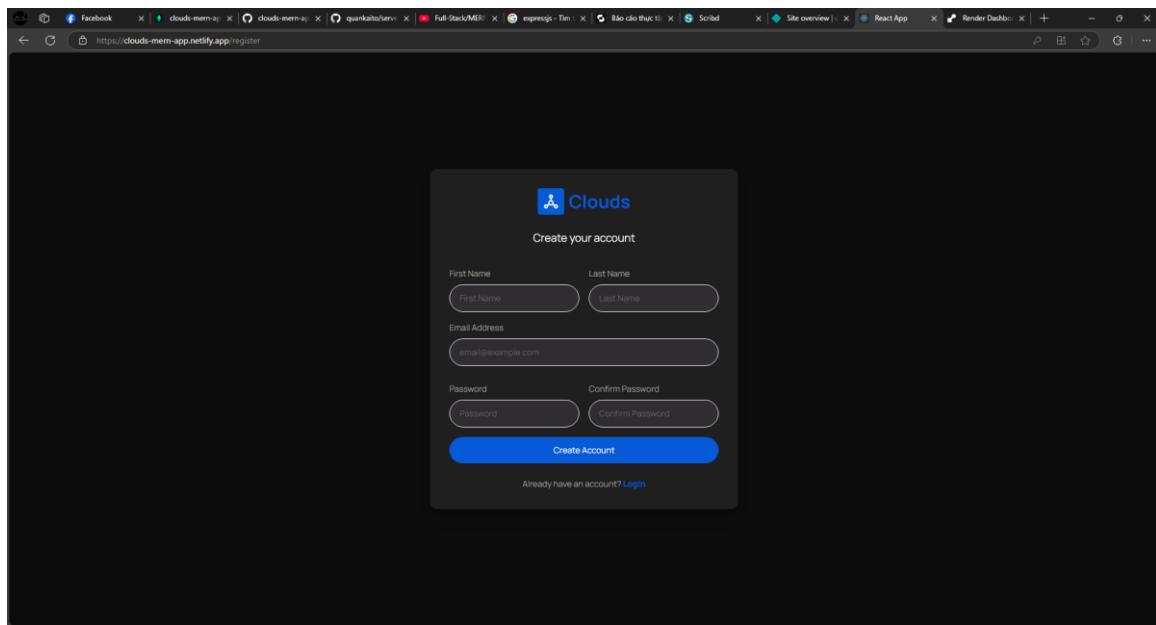
Ứng dụng web này có các chức năng chính:

- Đăng ký/đăng nhập



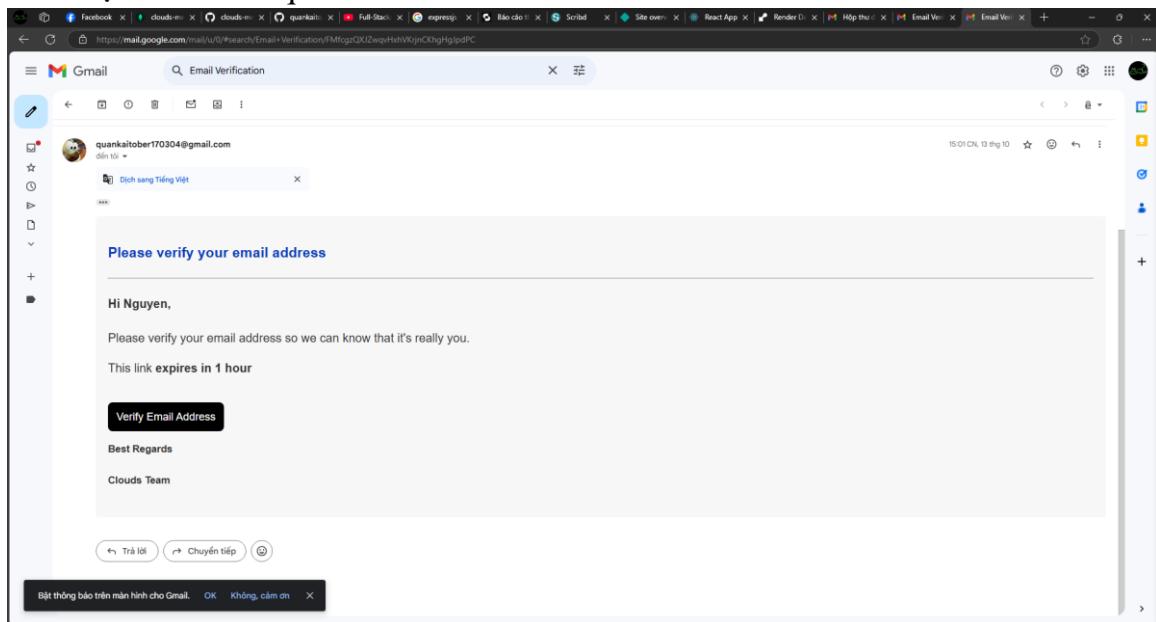
Hình 128. Giao diện đăng nhập

Áo hóa và điện toán đám mây – Nhóm 3



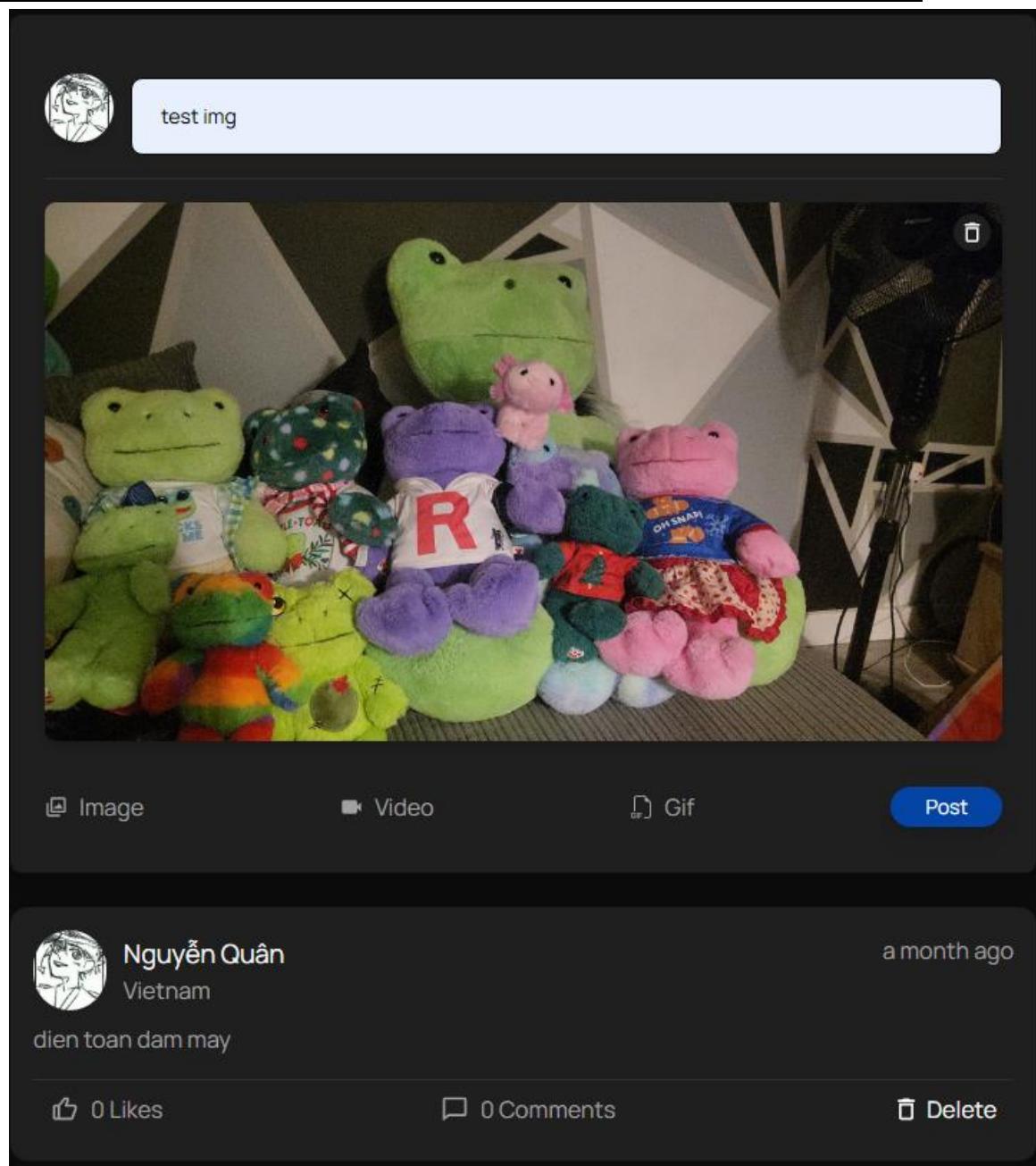
Hình 129. Giao diện đăng ký

- Xác thực tài khoản qua email



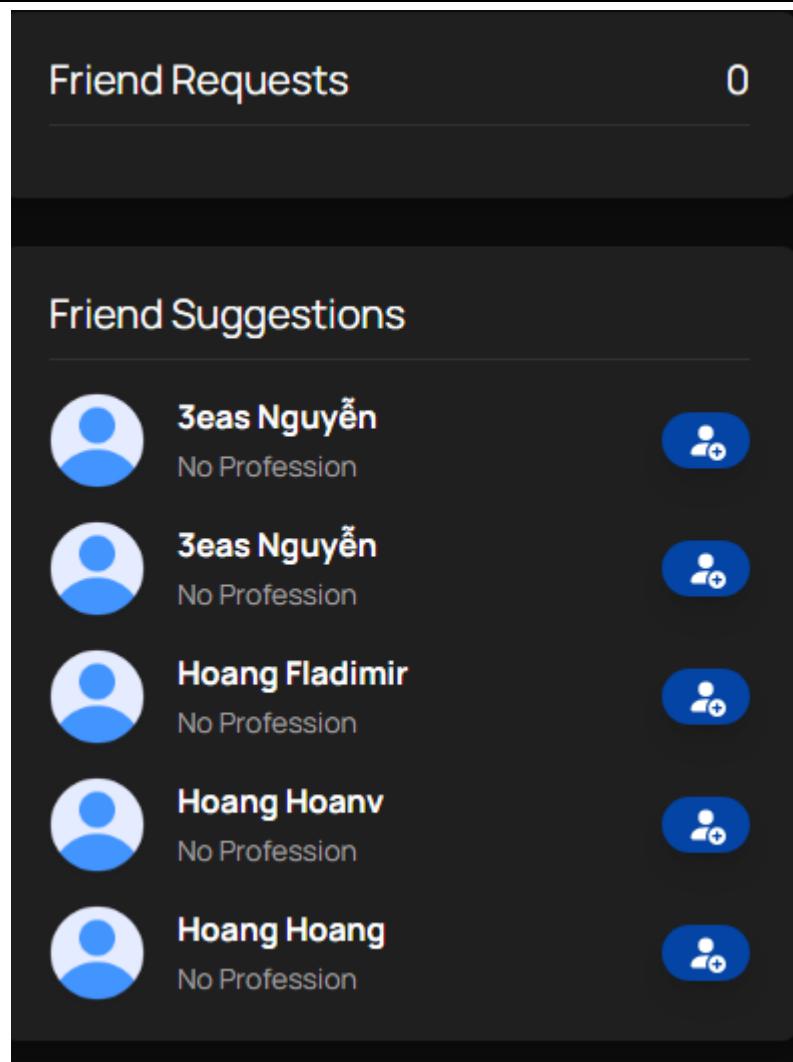
Hình 130. Email xác thực đăng ký tài khoản

- Đăng bài (có thể kèm hình ảnh, video)



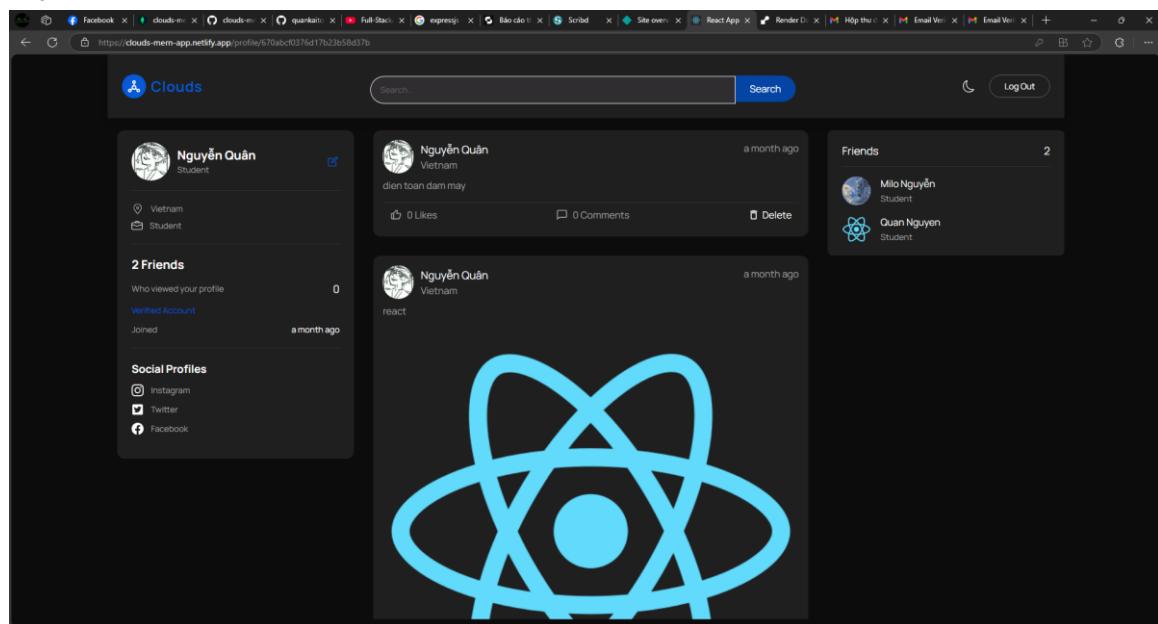
Hình 131. Chức năng đăng bài

- Kết bạn



Hình 132. Chức năng kết bạn

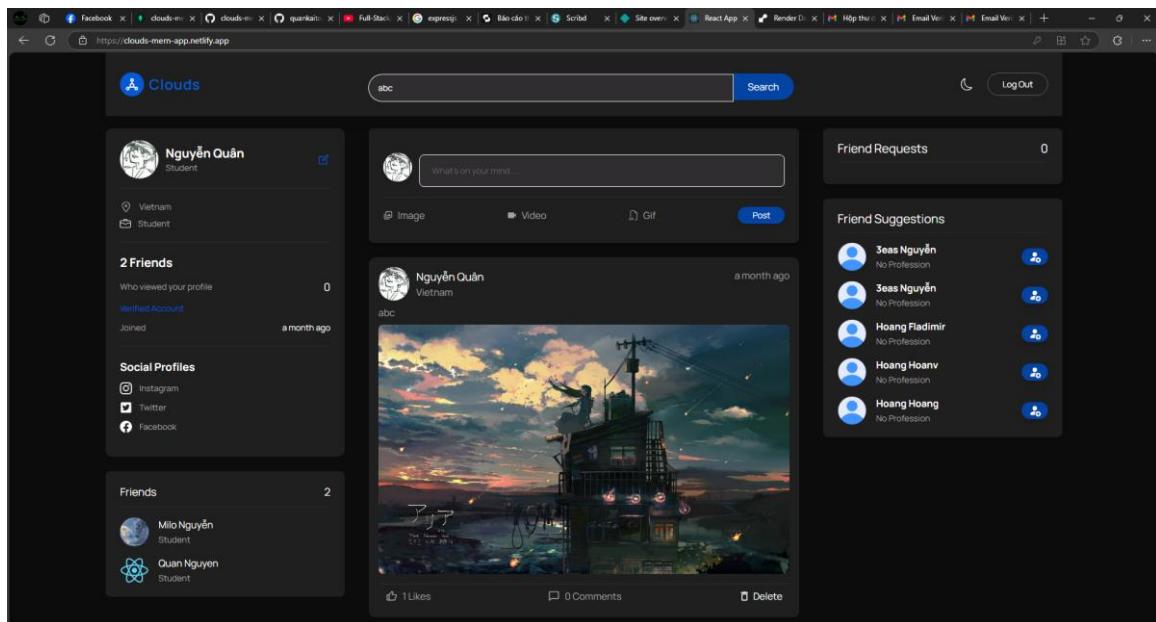
- Xem hồ sơ



Hình 133. Chức năng xem hồ sơ người dùng

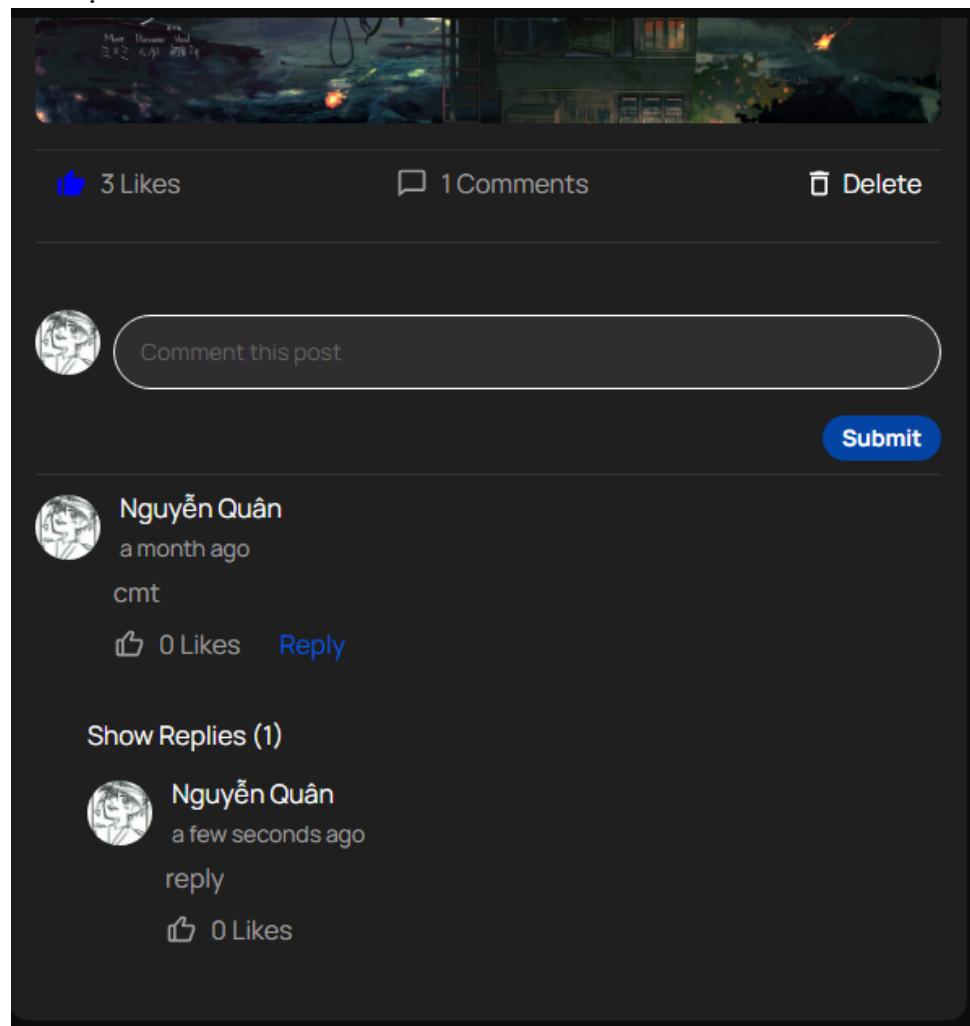
Áo hóa và điện toán đám mây – Nhóm 3

- Tìm kiếm



Hình 134. Tìm kiếm

- Bình luận



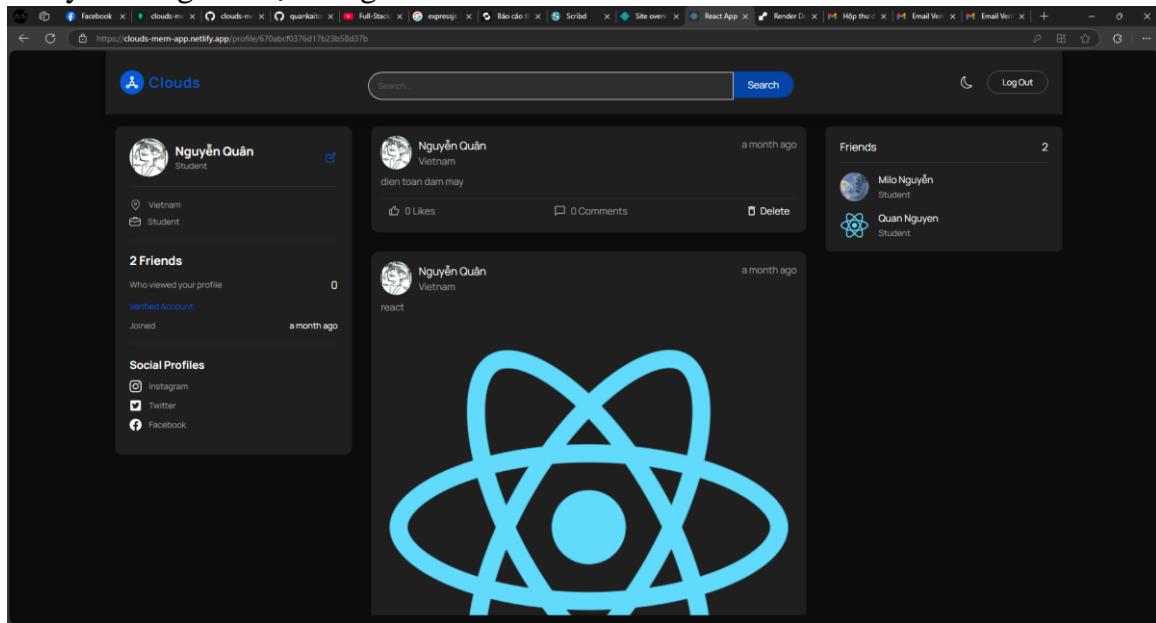
Hình 135. Chức năng bình luận

Trang 85

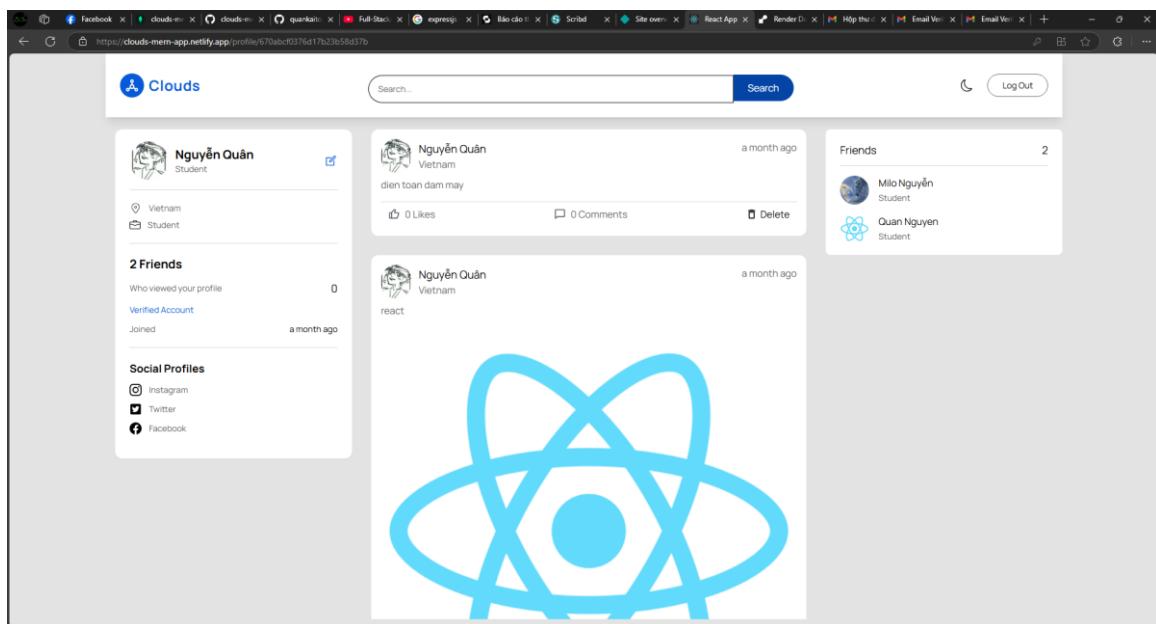
Tìm hiểu về nền tảng MongoDB

Áo hóa và điện toán đám mây – Nhóm 3

- Chuyển đổi giao diện sáng/tối



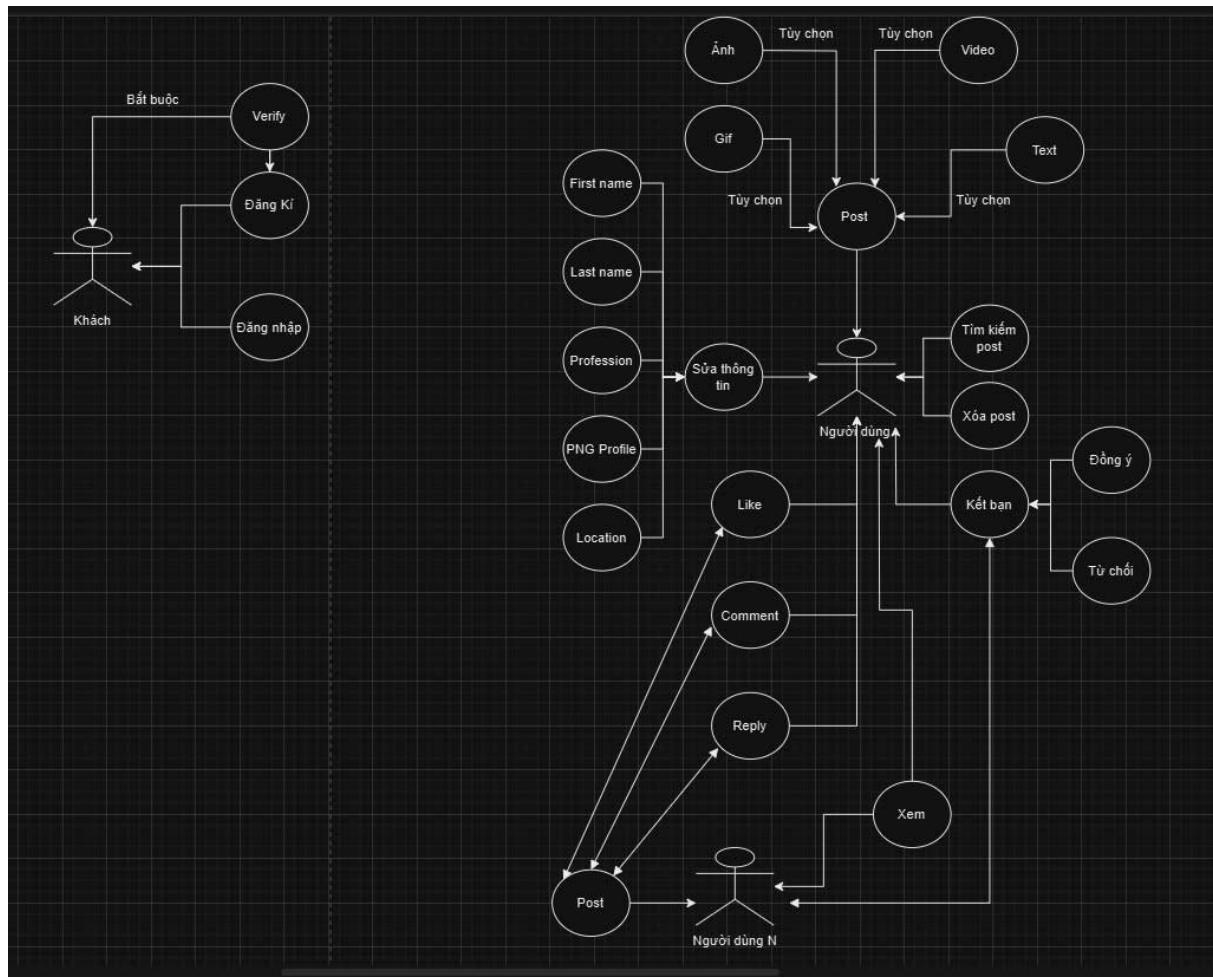
Hình 136. Giao diện tối



Hình 137. Giao diện sáng

b. Server

Database trên MongoDB Atlas



Hình 138. Use case

Comment Model (Mô hình bình luận)

Mục đích: Quản lý các bình luận liên quan đến bài đăng. Bao gồm các trường như ID người dùng, ID bài đăng, nội dung bình luận. Hỗ trợ thêm phản hồi (replies) dưới dạng đối tượng lồng nhau và theo dõi lượt thích cho cả bình luận và phản hồi.

Chi tiết:

- Phản hồi được tổ chức dưới dạng đối tượng với các trường như ID phản hồi, thông tin người dùng, thời gian tạo/cập nhật, và lượt thích.
- Tích hợp tính năng tự động ghi lại thời gian tạo (createdAt) và cập nhật (updatedAt).



```
commentModel.js ×
server > models > commentModel.js > ...
Minh Quan, last month | 1 author (Minh Quan)
1 import mongoose, { Schema } from "mongoose";  870.1k (gzipped: 233k)
2
3 const commentSchema = new mongoose.Schema(
4   {
5     userId: { type: Schema.Types.ObjectId, ref: "Users" },
6     postId: { type: Schema.Types.ObjectId, ref: "Posts" },
7     comment: { type: String, required: true },
8     from: { type: String, required: true },
9     replies: [
10       {
11         rid: { type: mongoose.Schema.Types.ObjectId },
12         userId: { type: Schema.Types.ObjectId, ref: "Users" },
13         from: { type: String },
14         replyAt: { type: String },
15         comment: { type: String },
16         created_At: { type: Date, default: Date.now() },
17         updated_At: { type: Date, default: Date.now() },
18         likes: [{ type: String }],
19       },
20     ],
21     likes: [{ type: String }],
22   },
23   { timestamps: true }
24 );
25
26 const Comments = mongoose.model("Comments", commentSchema);
27
28 export default Comments;
29
```

Hình 139. Mô hình bình luận

```

_id: ObjectId('670abdc4e4f10912c8bf56a')
userId : ObjectId('670abcf0376d17b23b58d37b')
postId : ObjectId('670abdc41e4f10912c8bf564')
comment : "comment"
from : "Nguyễn Quán"
likes : Array (empty)
replies : Array (1)
createdAt : 2024-10-12T18:19:54.779+00:00
updatedAt : 2024-10-12T18:20:06.627+00:00
__v : 1

_id: ObjectId('670b7a0e5dab9a747bdc2c84')
userId : ObjectId('670b641aa5831869932b70')
postId : ObjectId('670b79915dab9a747bdc2c60')
comment : "test server"
from : "Bùi Nguyễn"
likes : Array (1)
replies : Array (1)
createdAt : 2024-10-13T07:43:18.937+00:00
updatedAt : 2024-10-13T07:43:22.039+00:00
__v : 1
  
```

Hình 140. Database bình luận

Friend Request Model (Mô hình yêu cầu kết bạn)

Mục đích: Quản lý các yêu cầu kết bạn giữa người dùng, bao gồm thông tin người gửi, người nhận, và trạng thái yêu cầu.

Chi tiết:

- Trạng thái mặc định là "Pending" (Chờ xử lý) khi yêu cầu được tạo.
- Hỗ trợ tự động ghi lại thời gian tạo và cập nhật yêu cầu.

```

server > models > friendRequest.js > ...
Minh Quan, last month | 1 author (Minh Quan)
1 import mongoose, { Schema } from "mongoose"; 870.1k (gzipped: 233k)
2
3 const requestSchema = Schema(
4   {
5     requestTo: { type: Schema.Types.ObjectId, ref: "Users" },
6     requestFrom: { type: Schema.Types.ObjectId, ref: "Users" },
7     requestStatus: { type: String, default: "Pending" },
8   },
9   { timestamps: true }
10 );
11
12 const FriendRequest = mongoose.model("FriendRequest", requestSchema);
13
14 export default FriendRequest;
15
  
```

Hình 141. Mô hình friendRequest

Áo hóa và điện toán đám mây – Nhóm 3

```
_id: ObjectId('670b6525a58318609932be9d')
requestTo : ObjectId('670abc0f376d17b23b58d37b')
requestFrom : ObjectId('670b641aa58318609932be70')
requestStatus : "Accepted"
createdAt : 2024-10-13T06:13:57.431+00:00
updatedAt : 2024-10-13T06:32:01.776+00:00
__v : 0

_id: ObjectId('670b79f25dab9a747bdcc2c75')
requestTo : ObjectId('670b776237aaef3cd1b0674b')
requestFrom : ObjectId('670b641aa58318609932be70')
requestStatus : "Pending"
createdAt : 2024-10-13T07:42:42.662+00:00
updatedAt : 2024-10-13T07:42:42.662+00:00
__v : 0

_id: ObjectId('670c9640fd8e842e3b29515')
requestTo : ObjectId('670c9072fd8e842e3b29481')
requestFrom : ObjectId('670b7e525180706a0f498d876')
requestStatus : "Pending"
```

Hình 142. Database friendRequest

Post Model (Mô hình bài đăng)

Mục đích: Quản lý các bài đăng của người dùng, bao gồm mô tả, hình ảnh và tương tác như bình luận, lượt thích.

Chi tiết:

- Tham chiếu tới mô hình Users để xác định chủ bài đăng và mô hình Comments để liên kết các bình luận.
- Mảng likes lưu trữ danh sách người dùng đã thích bài đăng.

```
server > models > postModel.js > ...
Minh Quan, last month | 1 author (Minh Quan)
1 import mongoose, { Schema } from "mongoose";  870.1k (gzipped: 233k)
2
3 //schema
4 const postSchema = new mongoose.Schema(
5   {
6     userId: { type: Schema.Types.ObjectId, ref: "Users" },
7     description: { type: String, required: true },
8     image: { type: String },
9     likes: [{ type: String }],
10    comments: [{ type: Schema.Types.ObjectId, ref: "Comments" }],
11  },
12  { timestamps: true }
13 );
14
15 const Posts = mongoose.model("Posts", postSchema);
16
17 export default Posts;
18
```

Hình 143. Mô hình đăng bài

Trang 90

The screenshot shows the MongoDB Compass interface with the following details:

- Collection:** test.posts
- Storage Size:** 36KB
- Logical Data Size:** 2.56KB
- Total Documents:** 10
- Indexes Total Size:** 36KB

Toolbar buttons include: Find, Indexes, Schema Anti-Patterns, Aggregation, Search Indexes, and INSERT DOCUMENT.

Search bar: Type a query: { field: 'value' }

Buttons: Filter, Apply, Options ▾, Reset, and INSERT DOCUMENT.

Query results: 1-10 OF 10

```
_id: ObjectId('670abdc41e4f10912c8bf564')
userId: ObjectId('670abcfc9376d17b23b58d37b')
description: "img"
image: "https://res.cloudinary.com/do20aodkx/image/upload/v1728757187/xkqdagei..."
likes: Array (3)
comments: Array (1)
createdAt: 2024-10-12T18:19:48.363+00:00
updatedAt: 2024-10-14T03:33:43.455+00:00
__v: 0

_id: ObjectId('670b648ea58318609932be96')
userId: ObjectId('670b641aa58318609932be70')
description: "test server"
likes: Array (1)
comments: Array (empty)
createdAt: 2024-10-13T06:11:26.218+00:00
updatedAt: 2024-10-14T03:33:41.796+00:00
__v: 0

_id: ObjectId('670b64be9efdad27d6c7dd8a')
```

Hình 144. Database đăng bài

User Model (Mô hình người dùng)

Mục đích: Định nghĩa cấu trúc dữ liệu người dùng, bao gồm thông tin cá nhân, cài đặt tài khoản và kết nối xã hội.

Chi tiết:

- Xác thực (validation) cho các trường như firstName, email, và password để đảm bảo tính toàn vẹn dữ liệu.
- Hỗ trợ tùy chỉnh hồ sơ với các trường như profileUrl (đường dẫn ảnh đại diện) và profession (nghề nghiệp).
- Quản lý mối quan hệ bạn bè thông qua mảng friends và theo dõi lượt xem (views).
- Trường verified dùng để xử lý trạng thái xác thực tài khoản.



```
js userModel.js ×
server > models > userModel.js > ...
Minh Quan, last month | 1 author (Minh Quan)
1 import mongoose, { Schema } from "mongoose";  870.1k (gzipped: 233k) Mi...
2
3 //schema
4 const userSchema = new mongoose.Schema(
5   {
6     firstName: {
7       type: String,
8       required: [true, "First Name is Required!"],
9     },
10    lastName: {
11      type: String,
12      required: [true, "Last Name is Required!"],
13    },
14    email: {
15      type: String,
16      required: [true, " Email is Required!"],
17      unique: true,
18    },
19    password: {
20      type: String,
21      required: [true, "Password is Required!"],
22      minlength: [6, "Password length should be greater than 6 character"],
23      select: true,
24    },
25    location: { type: String },
26    profileUrl: { type: String },
27    profession: { type: String },
28    friends: [{ type: Schema.Types.ObjectId, ref: "Users" }],
29    views: [{ type: String }],
30    verified: { type: Boolean, default: false },
31  },
32  { timestamps: true }
33);
34
35 const Users = mongoose.model("Users", userSchema);
36
37 export default Users;
38
```

Hình 145. Mô hình quản lý người dùng

Áo hóa và điện toán đám mây – Nhóm 3

The screenshot shows the MongoDB Compass interface. At the top, it displays the database name 'test.users' with storage details: STORAGE SIZE: 36KB, LOGICAL DATA SIZE: 2.59KB, TOTAL DOCUMENTS: 8, and INDEXES TOTAL SIZE: 72KB. Below this are tabs for 'Find', 'Indexes', 'Schema Anti-Patterns', 'Aggregation', and 'Search Indexes'. A button for 'INSERT DOCUMENT' is on the right. A search bar at the top says 'Generate queries from natural language in Compass'. Below the search bar is a 'Filter' dropdown and a text input field 'Type a query: { field: 'value' }'. A 'Reset' button and 'Apply' button are to the right of the filter. The results section shows 'QUERY RESULTS: 1-8 OF 8'. Two documents are listed:

```
_id: ObjectId('670abc0376d17b23b58d37b')
firstName: "Nguyễn"
lastName: "Quân"
email: "quancaitober1703@gmail.com"
password: "$2a$10$u0m0ldcnrytIETULtEBZWVu0xzeZAF00uPjXAIQ3rY2aznCken7lC"
friends: Array (2)
views: Array (empty)
verified: true
createdAt: 2024-10-12T18:16:16.983+00:00
updatedAt: 2024-10-14T05:53:17.193+00:00
location: "Vietnam"
profession: "Student"
profileUrl: "https://res.cloudinary.com/do20aodkx/image/upload/v1728757214/r76kszlt_"

_id: ObjectId('670b641aa58318609932be70')
firstName: "Milo"
lastName: "Nguyễn"
email: "baosino2468@gmail.com"
password: "$2a$10$avzD7Dde3T/TqX3b0kpphurK4ctwiowZHE3dvsR8fp56c2DTL7hVv"
friends: Array (2)
views: Array (empty)
```

Hình 146. Database quản lý người dùng

Email Verification Model (Mô hình xác minh email)

Mục đích: Quản lý việc xác minh email của tài khoản người dùng. Lưu trữ mã xác minh (token) tạm thời và thời gian hết hạn.

Chi tiết:

- Các trường lưu trữ ID người dùng, mã xác minh và thời gian tạo hoặc hết hạn.
- Đảm bảo bảo mật cho quy trình xác minh email.

```
JS emailVerification.js X
server > models > emailVerification.js > ...
Minh Quan, last month | 1 author (Minh Quan)
1 import mongoose, { Schema } from "mongoose"; 870.1k (gzipped: 233k) Minh Quan
2
3 const emailVerificationSchema = Schema({
4   userId: String,
5   token: String,
6   createdAt: Date,
7   expiresAt: Date,
8 });
9
10 const Verification = mongoose.model("Verification", emailVerificationSchema);
11
12 export default Verification;
13
```

Hình 147. Mô hình xác thực email người dùng

Áo hóa và điện toán đám mây – Nhóm 3

The screenshot shows the MongoDB Compass interface with the following details:

- Collection:** test.verifications
- Storage Size:** 36KB
- Logical Data Size:** 1,04KB
- Total Documents:** 6
- Indexes Total Size:** 36KB

Below the header, there are tabs: Find, Indexes, Schema Anti-Patterns, Aggregation, and Search Indexes. A button labeled "INSERT DOCUMENT" is located in the top right corner.

A search bar at the top says "Generate queries from natural language in Compass". Below it, a "Filter" dropdown and a "Type a query" input field with the placeholder "field: 'value'" are present. Buttons for "Reset", "Apply", and "Options" are also visible.

The results section shows "QUERY RESULTS: 1-6 OF 6". Three documents are listed:

```
_id: ObjectId('670b6715dfd9ale98db18430')
userId: "670b6715dfd9ale98db1842e"
token: "$2a$18$CNzptpFI5W8s6PaB502tBuWLjQ0z3Tu.JU0rb3Hy6GvtT1j7tTau"
createdAt: 2024-10-13T06:22:13.895+00:00
expiresAt: 2024-10-13T07:22:13.895+00:00
__v: 0

_id: ObjectId('670b776237aa6f3cd1b8674d')
userId: "670b776237aa6f3cd1b8674b"
token: "$2a$10$Lcts3QxMfyRbEOvzRF30uKTqSG0MAYql4J8WjFgobAlzf.TABSGG"
createdAt: 2024-10-13T07:31:46.778+00:00
expiresAt: 2024-10-13T08:31:46.778+00:00
__v: 0

_id: ObjectId('670b7a3d5dab9a747bdc2cad')
userId: "670b7a3c5dab9a747bdc2cab"
token: "$2a$10$sp4useZPwdehy8BYVqZmuE3AbohY8huPu15fdW2qzsUNjLhy9tS"
createdAt: 2024-10-13T07:43:57.544+00:00
expiresAt: 2024-10-13T08:43:57.544+00:00
__v: 0
```

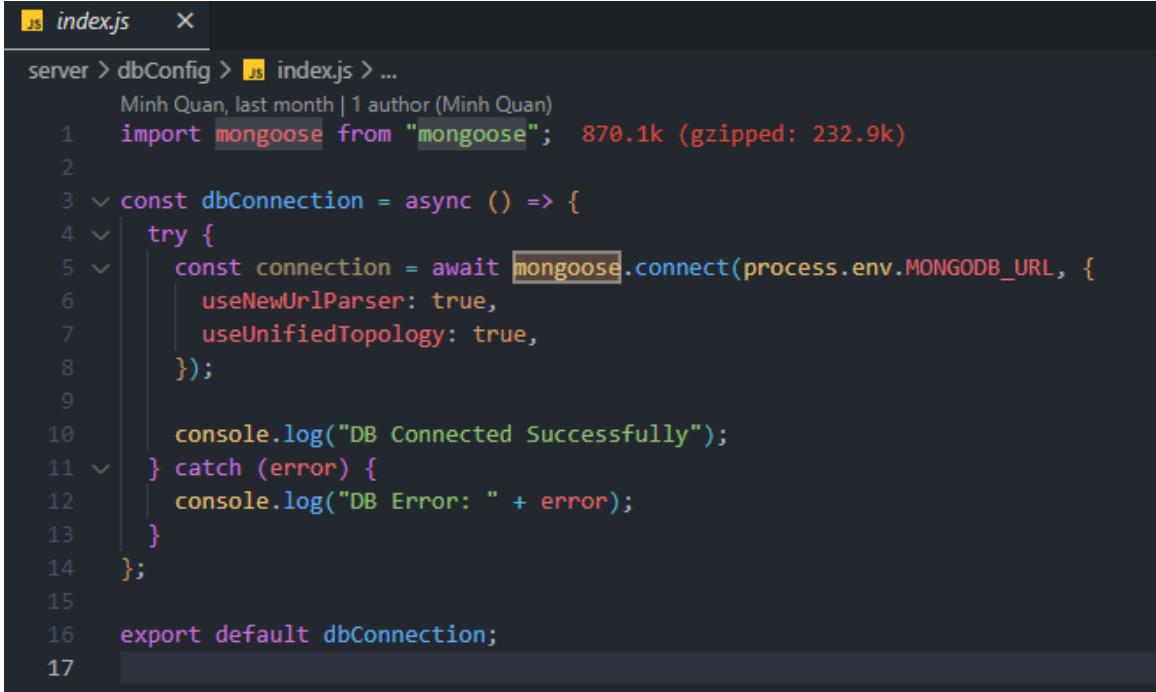
Hình 148. Database xác thực email người dùng

Phân tích:

Mục đích: File này chịu trách nhiệm thiết lập kết nối giữa ứng dụng Node.js và cơ sở dữ liệu MongoDB.

Chi tiết:

- Sử dụng mongoose.connect để thiết lập kết nối tới MongoDB thông qua URL được lưu trong biến môi trường (process.env.MONGODB_URL).
- Hai tùy chọn useNewUrlParser và useUnifiedTopology được bật để đảm bảo khả năng tương thích và hiệu suất tối ưu khi sử dụng MongoDB mới.
- Nếu kết nối thành công, log "DB Connected Successfully" sẽ được hiển thị.
- Nếu kết nối thất bại, lỗi sẽ được log ra console với thông báo chi tiết.



```
index.js
server > dbConfig > index.js > ...
Minh Quan, last month | 1 author (Minh Quan)
1 import mongoose from "mongoose"; 870.1k (gzipped: 232.9k)
2
3 const dbConnection = async () => {
4   try {
5     const connection = await mongoose.connect(process.env.MONGODB_URL, {
6       useNewUrlParser: true,
7       useUnifiedTopology: true,
8     });
9
10    console.log("DB Connected Successfully");
11  } catch (error) {
12    console.log("DB Error: " + error);
13  }
14};
15
16 export default dbConnection;
17
```

Hình 149. Thiết lập kết nối giữa ứng dụng với database

Phân tích:

Mục đích: Đây là tệp chính khởi chạy ứng dụng, chịu trách nhiệm cấu hình server và middleware.

Chi tiết:

Modules cài đặt:

- dotenv: Đọc các biến môi trường từ tệp .env (bao gồm thông tin như PORT, MONGODB_URL).
- express: Framework dùng để xây dựng server.
- helmet: Tăng cường bảo mật bằng cách thiết lập các HTTP headers.
- cors: Cho phép chia sẻ tài nguyên từ các domain khác (CORS policy).
- morgan: Ghi lại log chi tiết của các request, hữu ích trong môi trường phát triển.
- body-parser và express.json: Xử lý dữ liệu đầu vào từ các request (dạng JSON hoặc URL-encoded).

Cấu hình tĩnh:

- express.static được dùng để phục vụ các file tĩnh từ thư mục views/build, thường là frontend đã được build (React, Angular, v.v.).
- Kết nối cơ sở dữ liệu:
- Gọi hàm dbConnection từ file dbConfig/index.js để thiết lập kết nối MongoDB.

Middleware và router:

- Các middleware được sử dụng để bảo mật, phân tích dữ liệu, và xử lý lỗi.
- Router chính (router) chịu trách nhiệm quản lý các route của ứng dụng.

Xử lý lỗi:

- Middleware errorMiddleware được dùng để quản lý và phản hồi lỗi một cách tập trung.
- Khởi chạy server:
- Server được chạy trên cổng chỉ định trong biến môi trường PORT, hoặc mặc định là 8800.



```
JS index.js ×
server > JS index.js > ...
Minh Quan, last month | 1 author (Minh Quan)
1 import express from "express";
2 import dotenv from "dotenv"; 6.8k (gzipped: 3k)
3 import cors from "cors"; 5k (gzipped: 2.1k)
4 import morgan from "morgan"; 14.2k (gzipped: 5.3k)
5 import bodyParser from "body-parser"; 487.2k (gzipped: 212.2k)
6 import path from "path";
7 //security packges
8 import helmet from "helmet"; 11.9k (gzipped: 3.1k)
9 import dbConnection from "./dbConfig/index.js";
10 import errorMiddleware from "./middleware/errorMiddleware.js";
11 import router from "./routes/index.js";
12
13 const __dirname = path.resolve(path.dirname(""));
14
15 dotenv.config();
16
17 const app = express();
18
19 app.use(express.static(path.join(__dirname, "views/build")));
20
21 const PORT = process.env.PORT || 8800;
22
23 dbConnection();
24
25 app.use(helmet());      Minh Quan, last month • first commit
26 app.use(cors());
27 app.use(bodyParser.json());
28 app.use(bodyParser.urlencoded({ extended: true }));
29 app.use(express.json({ limit: "10mb" }));
30 app.use(express.urlencoded({ extended: true }));
31
32 app.use(morgan("dev"));
33 app.use(router);
34
35 //error middleware
36 app.use(errorMiddleware);
37
38 app.listen(PORT, () => {
39   console.log(`Server running on port: ${PORT}`);
40 });
41
```

Hình 150. Khởi chạy ứng dụng

2. Sử dụng github quản lý code

Đầu tiên, chúng ta vào trang [github](#) để tạo repository mới. Bạn đăng nhập vào GitHub. Tại giao diện chính, nhấp chọn vào dấu + ở góc phải màn hình và chọn **New repository** từ menu đó xuống để tạo repository mới.

Nhập tên cho **repository** mới tại ô **Repository name** > Chọn **Public** hoặc **Private** cho **repository** > Nhấp vào nút **Create repository**.

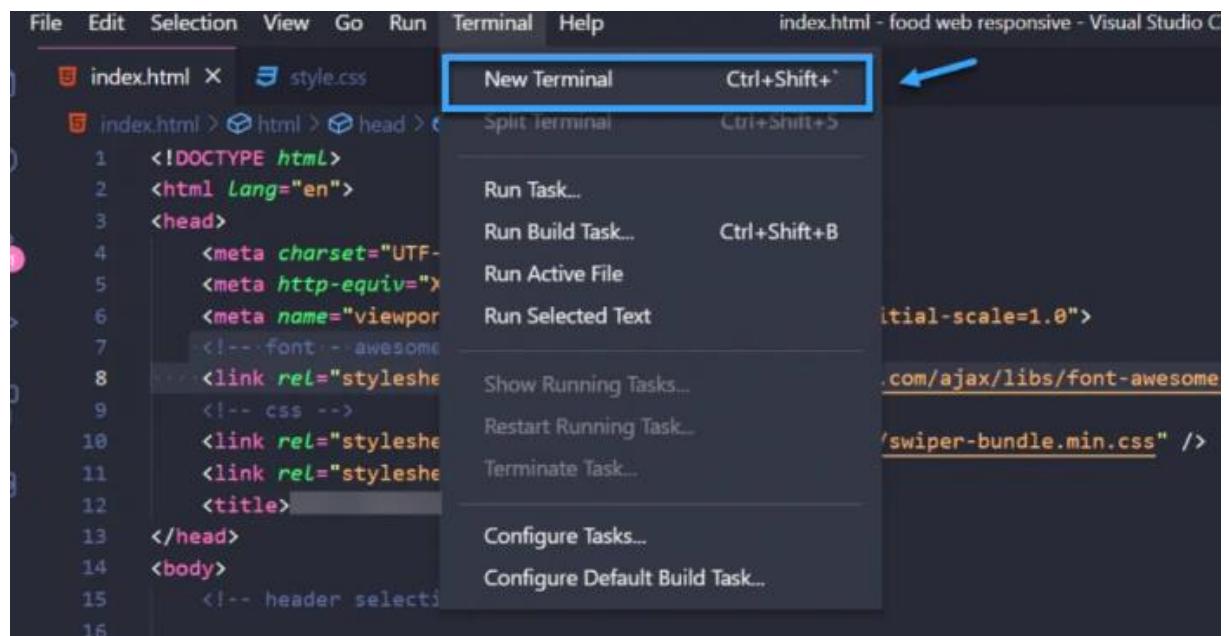
Liên kết kho lưu trữ: Trong Terminal/Command Prompt, nhập lệnh sau, thay thế <https://github.com/> bằng URL kho lưu trữ GitHub của bạn:

```
git remote add origin https://github.com/
```

Đẩy code lên GitHub:

```
git push origin master
```

Mở [VScode](#) và truy cập vào thư mục mà bạn muốn đưa code lên GitHub. Tiếp theo vào **Terminal** > Chọn **New Terminal**.

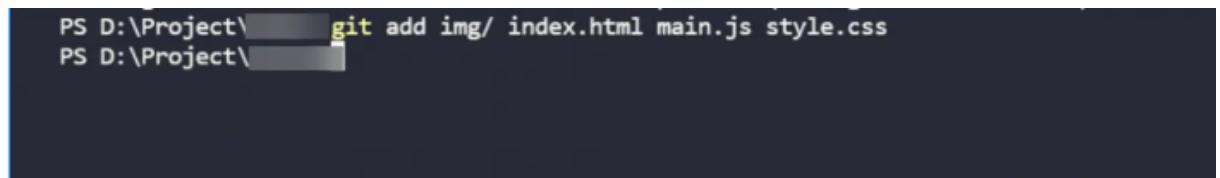


Hình 151. Chọn New Terminal

Gõ lệnh **git init** để tạo 1 git repository project mới hoặc đã có sẵn. Nếu gặp lỗi ở bước này, có thể là do bạn chưa tải Git về hoặc cài đặt không thành công.

Tiếp theo bạn gõ lệnh **git status** để kiểm tra các file chưa được đưa lên hoặc các file đã thay đổi trong thư mục.

Gõ lệnh **git add** để thêm những file cần đẩy lên GitHub. Dùng **git add .** để add nhanh toàn bộ.



```
PS D:\Project\ git add img/ index.html main.js style.css  
PS D:\Project\
```

Hình 152. Dùng lệnh git add để thêm file lên GitHub

Gõ lệnh **git status** để kiểm tra xem đã thêm thành công hay chưa.

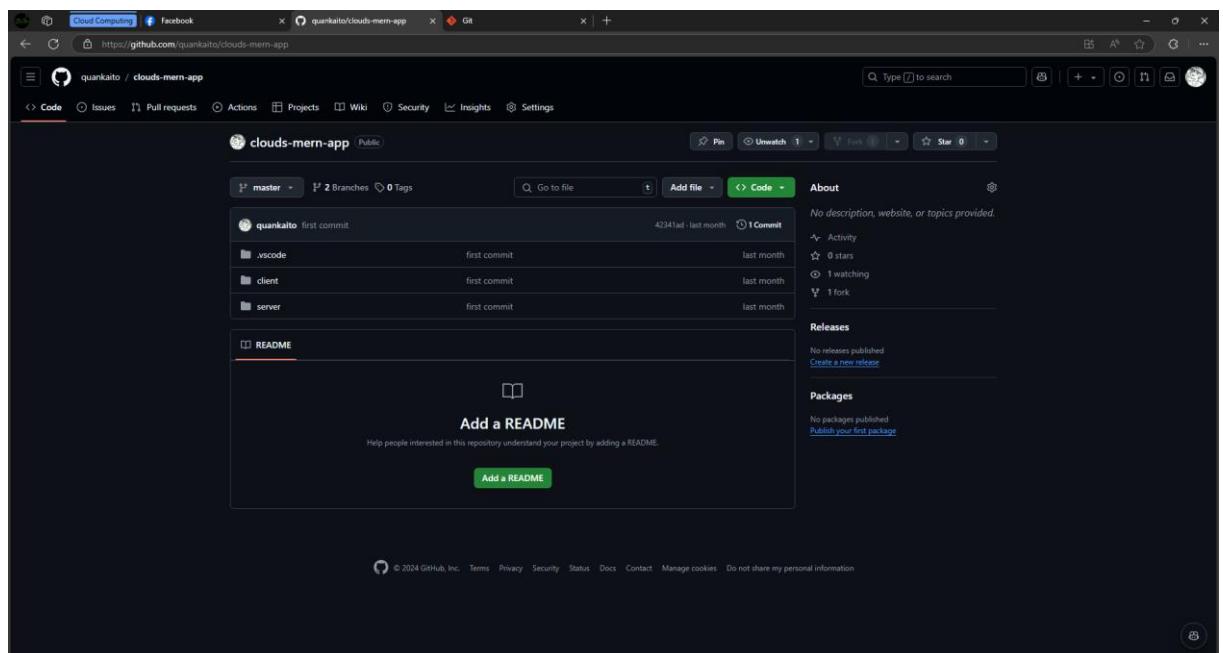
Gõ lệnh **git commit -m "Add new project"** để ghi chú các thay đổi, thuận tiện cho việc theo dõi về sau.

Gõ lệnh **git branch -M main**.

Gõ lệnh **git remote add origin link github**.

Bạn gõ lệnh **git push -u origin main** để đẩy code lên GitHub.

Tải lại trang GitHub để kiểm tra code đã được đẩy lên thành công hay chưa.

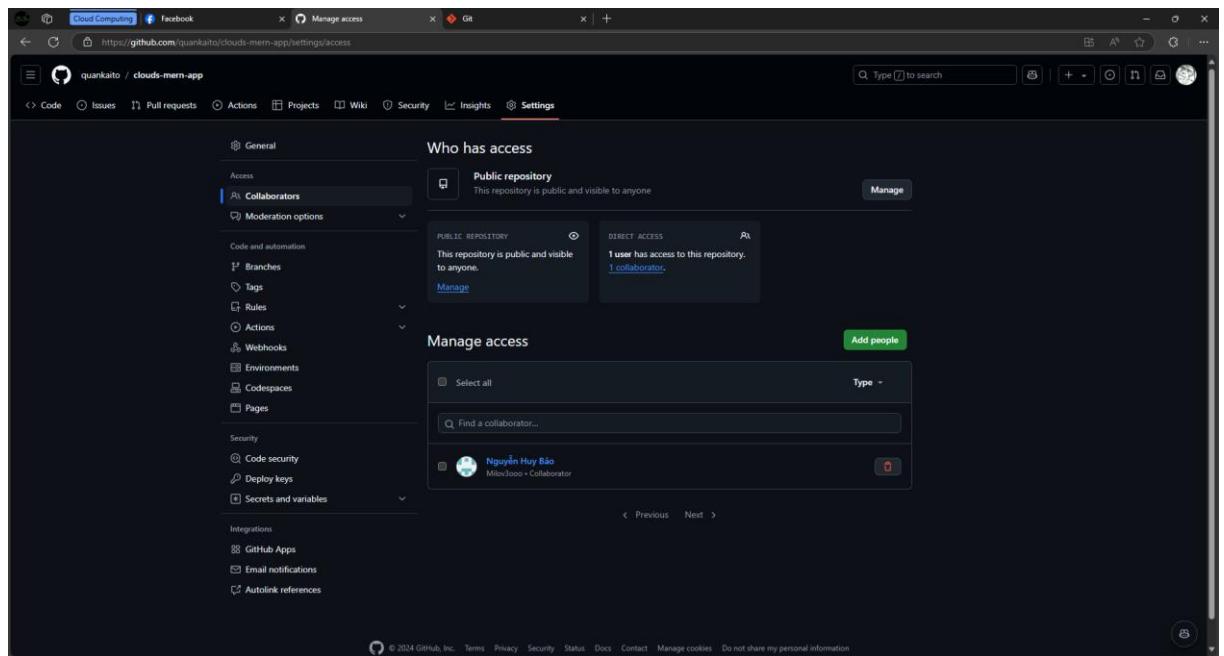


Hình 153. Hình ảnh khi thành công

Tại đây, chúng ta có thể dễ dàng quản lý, đồng bộ code cho cả nhóm cùng làm.

Bây giờ, chúng ta sẽ tiến hành mời đồng đội trong nhóm vào để code cùng nhau và dễ dàng làm việc. Đầu tiên, bấm vào nút **settings** trên taskbar, chọn **Collaborators** ở mục **Access** và tiến hành **Add People**.

Áo hóa và điện toán đám mây – Nhóm 3



Hình 154. Dự án nãy đã thêm một người

Nếu chưa có git trong máy, chúng ta hãy lên [Git](#) để tải về và set-up.

Bước 1: Vào [trang web tải Git](#). Bạn lựa chọn mục **Downloads**.



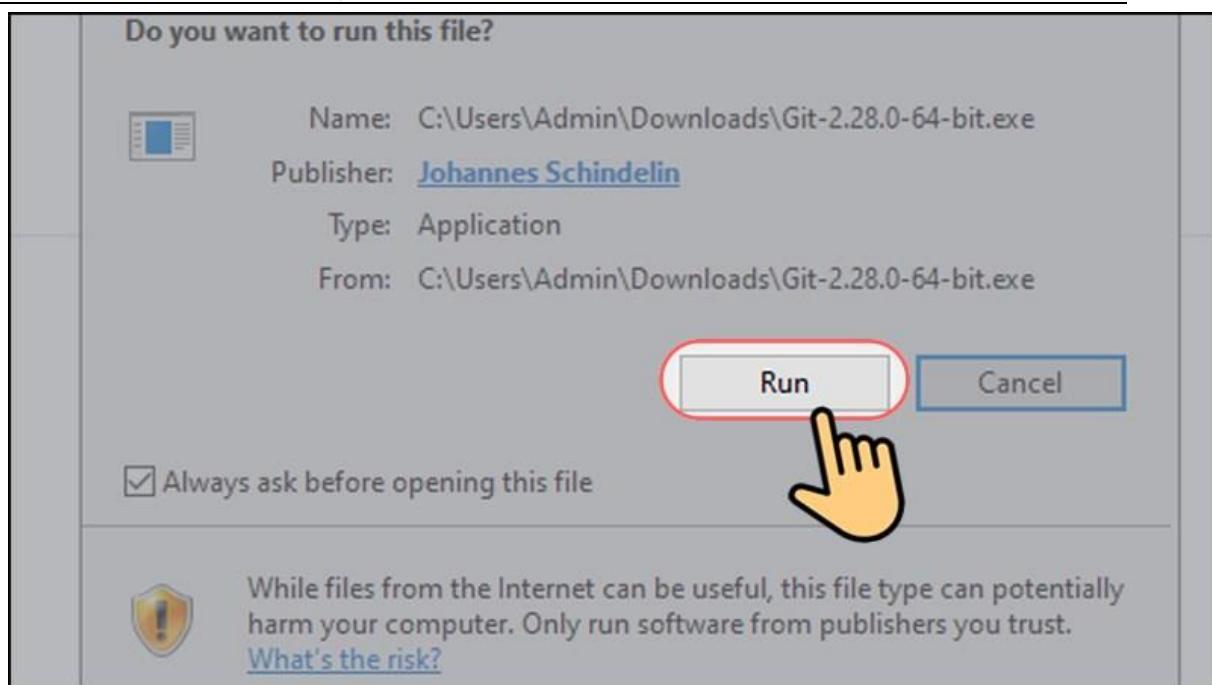
Hình 155. Vào trang web tải Git. Bạn lựa chọn mục **Downloads**

Bước 2: Sẽ có rất nhiều phiên bản phù hợp với các hệ điều hành khác nhau để bạn lựa chọn. **Bạn chọn phiên bản dành cho Windows.**



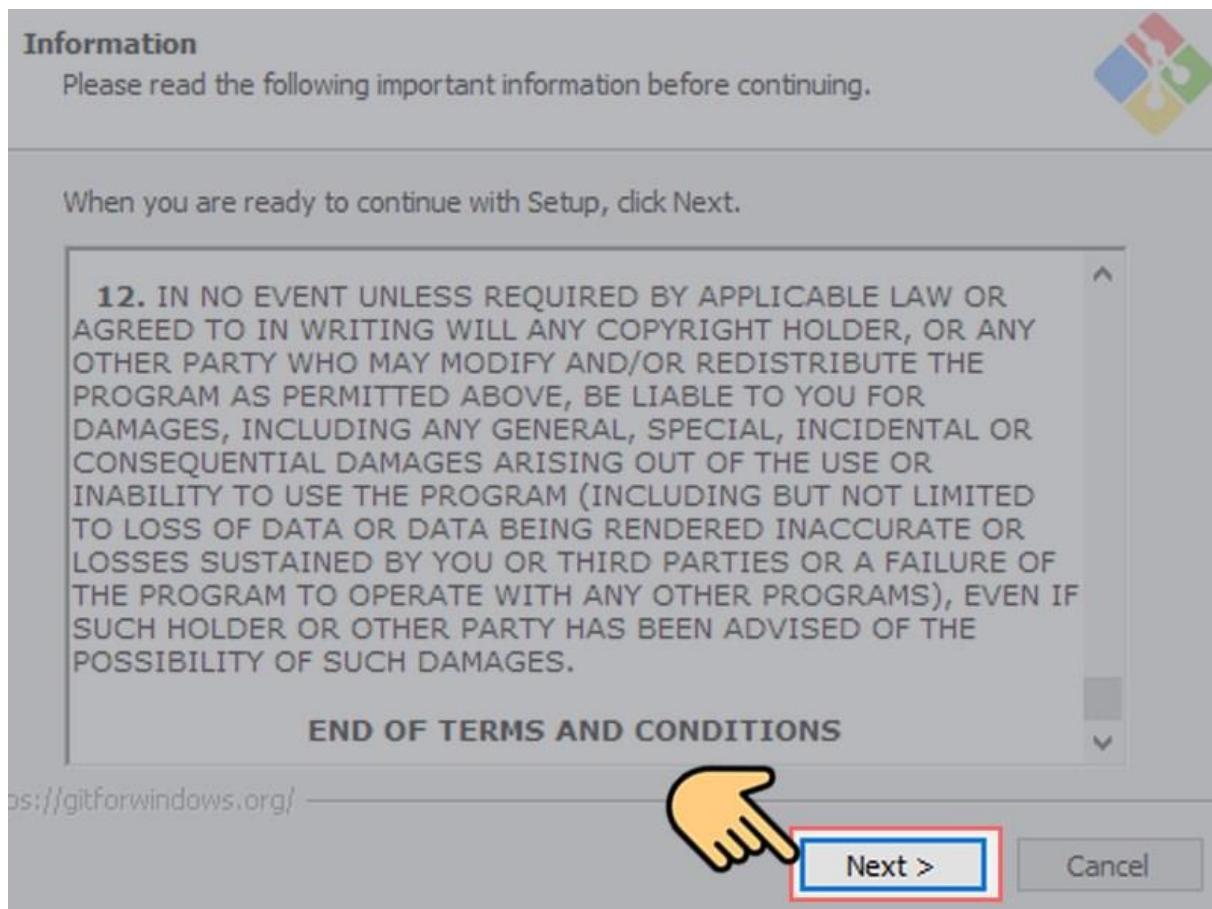
Hình 156. Bạn chọn phiên bản dành cho Windows.

Bước 3: Mở file Git bạn vừa mới tải xuống **chọn Run**.



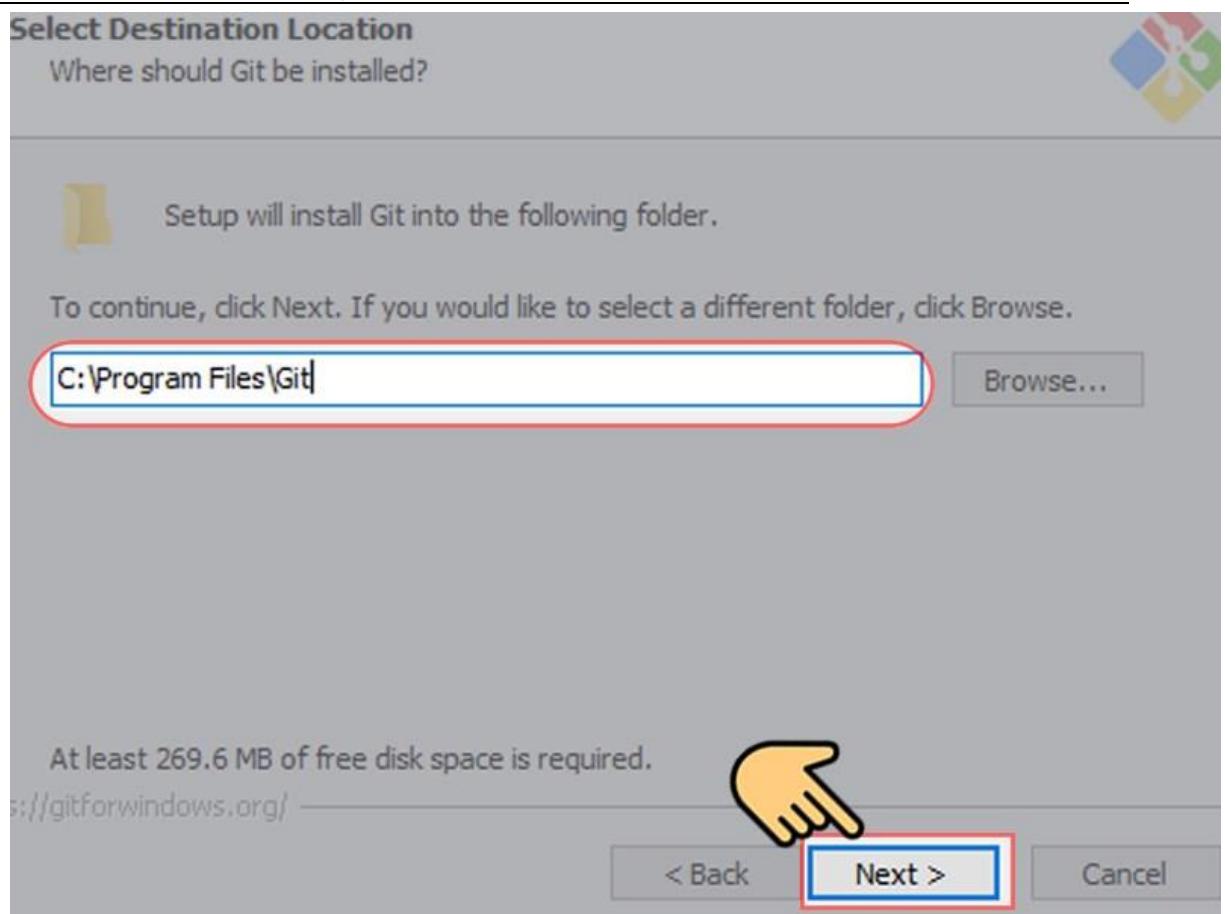
Hình 157. Mở file Git bạn vừa mới tải xuống chọn Run.

Bước 4: Một cửa sổ cài đặt hiện ra bạn chọn Next.



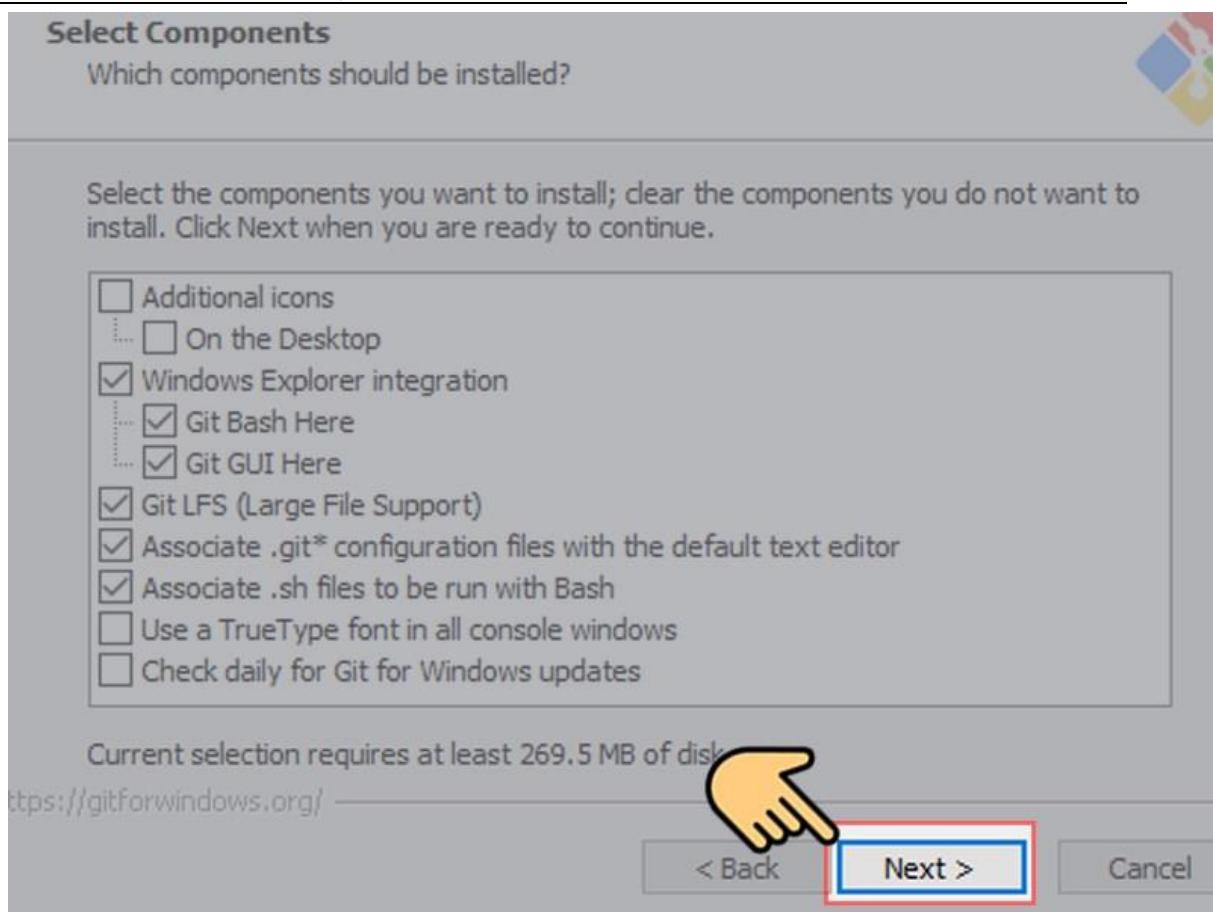
Hình 158. Một cửa sổ cài đặt hiện ra bạn chọn Next

Bước 5: Lựa chọn vị trí lưu mà bạn mong muốn sau đó chọn Next.



Hình 159. Lựa chọn vị trí lưu mà bạn mong muốn sau đó chọn Next.

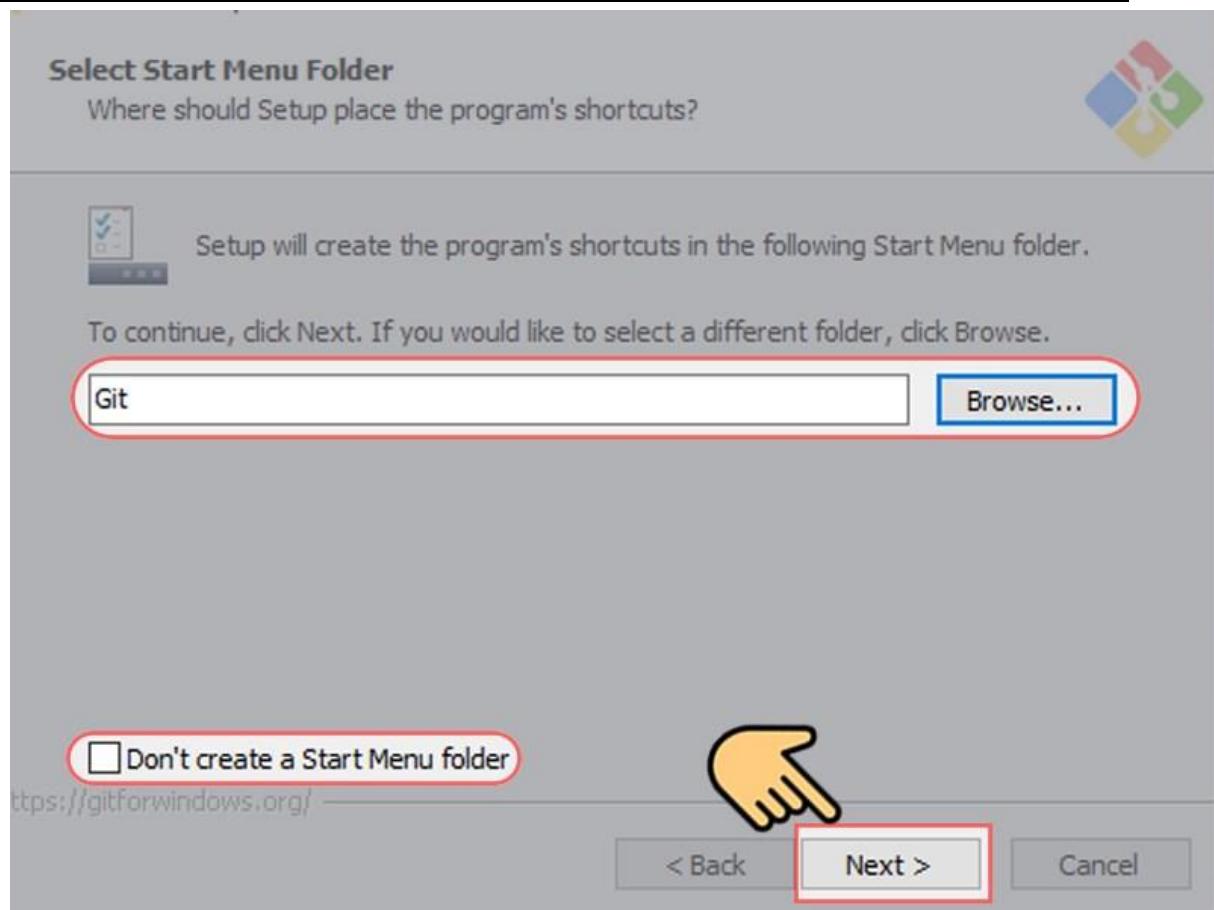
Bước 6: Cửa sổ tiếp theo đã có những lựa chọn mặc định cho bạn, bạn có thể chọn thêm các mục khác theo nhu cầu bản thân. Hoặc cài đặt như bản mặc định của nhà phát triển. Bạn **chọn Next** để tiếp tục cài đặt.



Hình 160. Bạn chọn Next để tiếp tục cài đặt

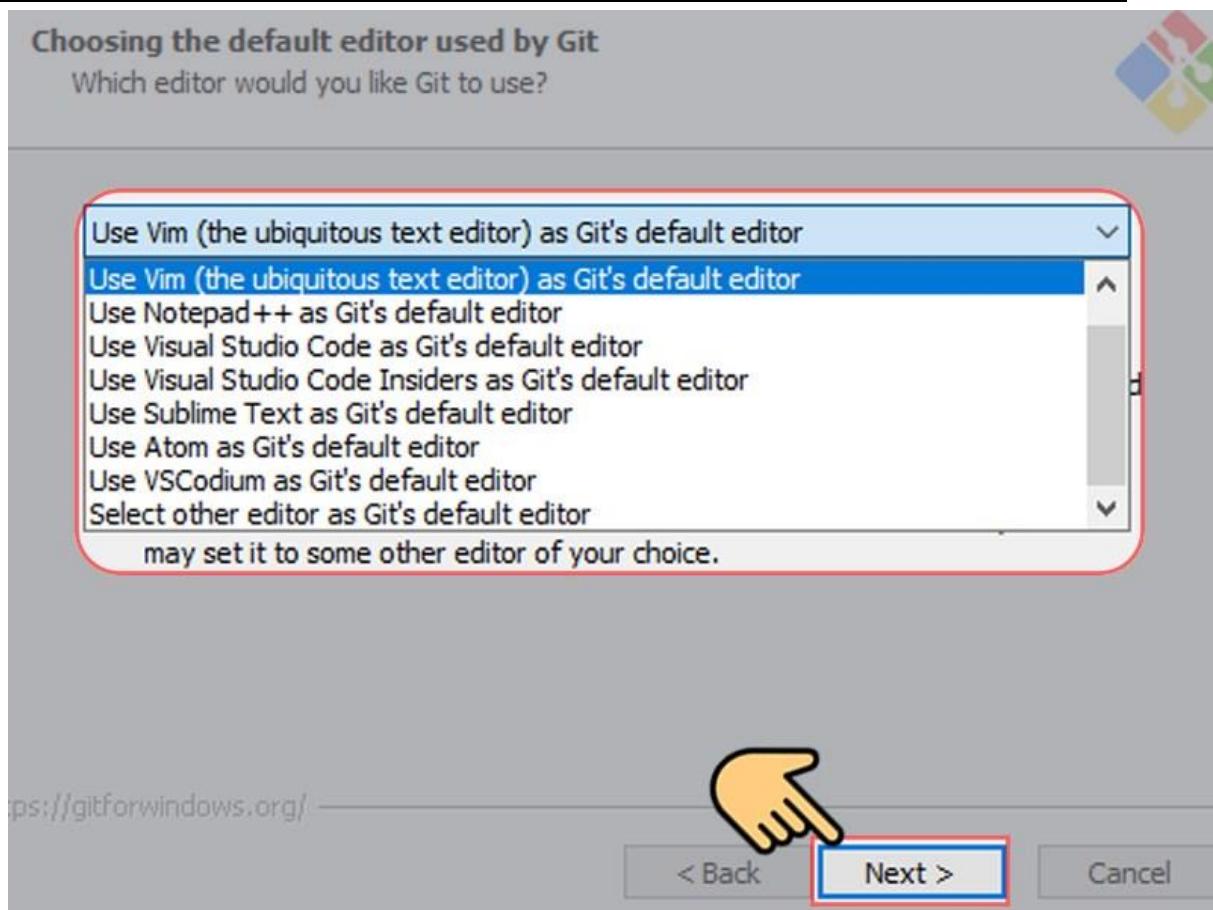
Bước 7: Tiếp đến là lựa chọn đặt vị trí Git vào Start Menu trong mục **Browse**.

Bạn có thể chọn **Don't create a Start Menu folder** nếu không muốn tạo ra thư mục của Git trên Start Menu hoặc bạn cũng có thể để mặc định, sau đó **chọn Next**.



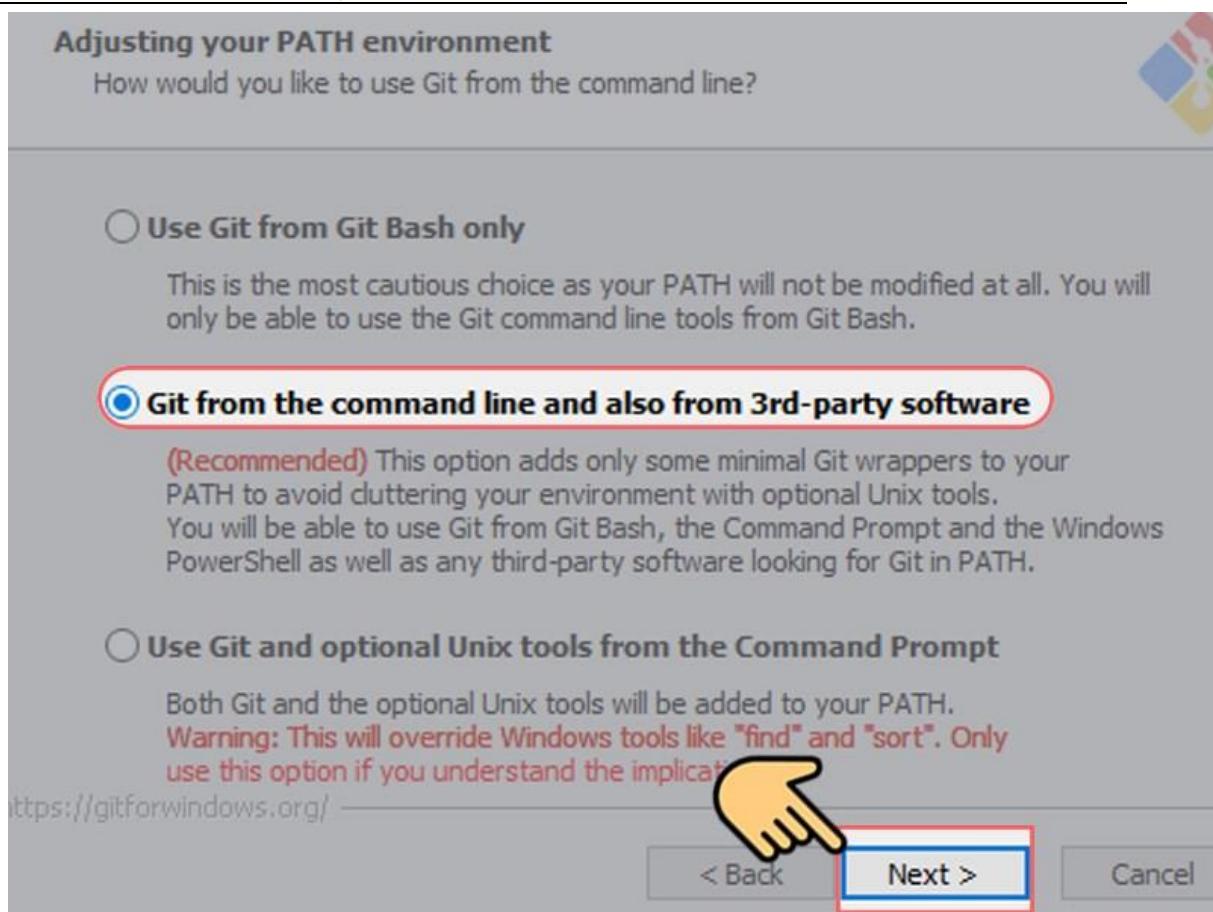
Hình 161. Chọn Next để tiếp tục cài đặt.

Bước 8: Tiếp đến chương trình sẽ yêu cầu bạn **lựa chọn một chương trình soạn thảo để biên tập lệnh cho Git bash**. Có rất nhiều chương trình khác nhau để bạn lựa chọn, nhưng nếu sử dụng Win 10, bạn nên để **trình mặc định Vim mà phần mềm đã chọn sẵn**, sau đó bạn **chọn Next** để tiếp tục cài đặt.



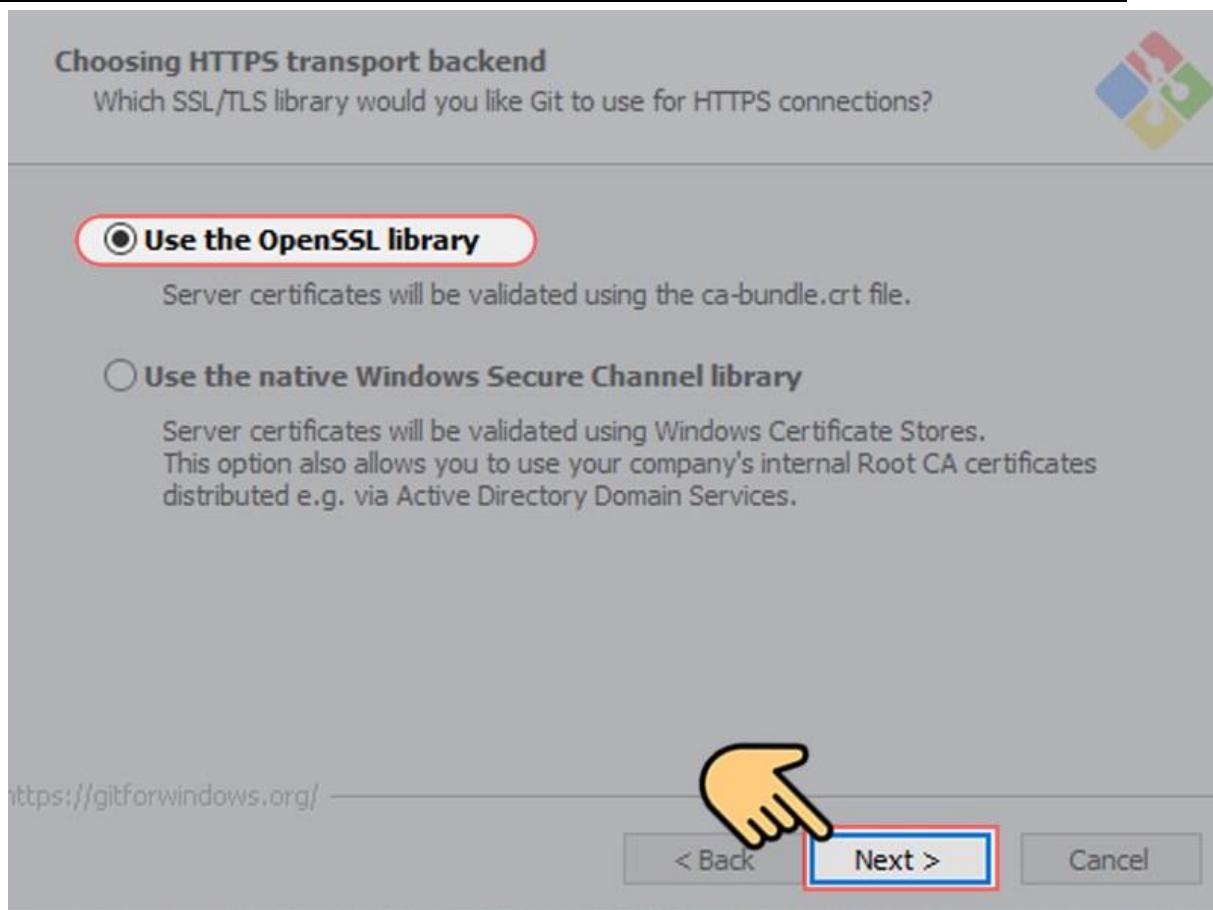
Hình 162. Lựa chọn một chương trình soạn thảo biên tập lệnh cho Git bash

Bước 9: Tiếp theo, phần cài đặt Git yêu cầu bạn **lựa chọn cài đặt về biến môi trường**. Khi cài trên Windows 10 bạn nên sử dụng lựa chọn mặc định **Git from the command line and also from 3rd-party software** (Git từ dòng lệnh và cả từ phần mềm của bên thứ ba). Sau đó **chọn Next**.



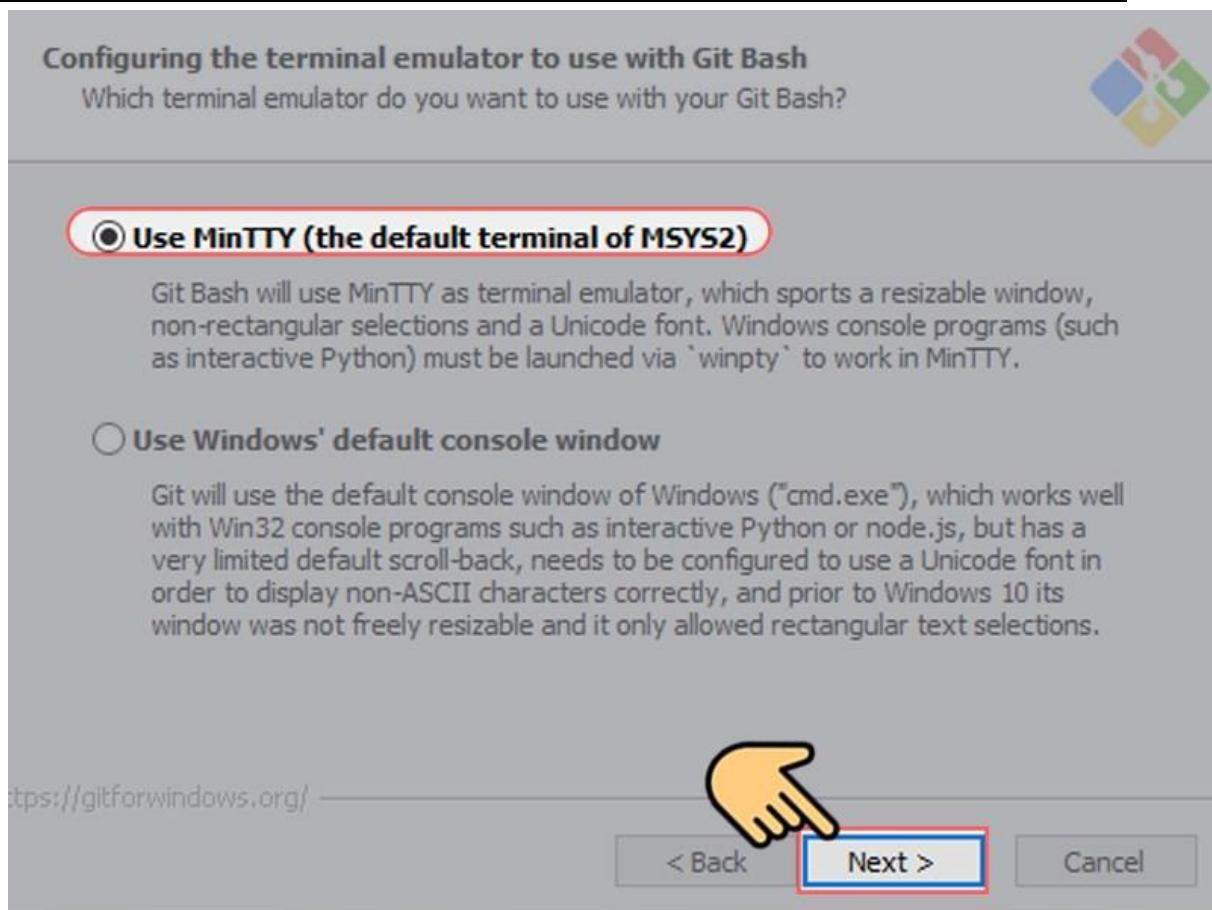
Hình 163. Sử dụng lựa chọn mặc định của chương trình và chọn Next

Bước 10: Tiếp theo là lựa chọn phương thức bảo mật. Bạn có thể lựa chọn phương thức mình mong muốn nhưng tốt nhất là nên giữ lại lựa chọn mặc định là Use the OpenSSL library (sử dụng thư viện OpenSSL) của phần mềm. Chọn Next để tiến hành bước tiếp theo.



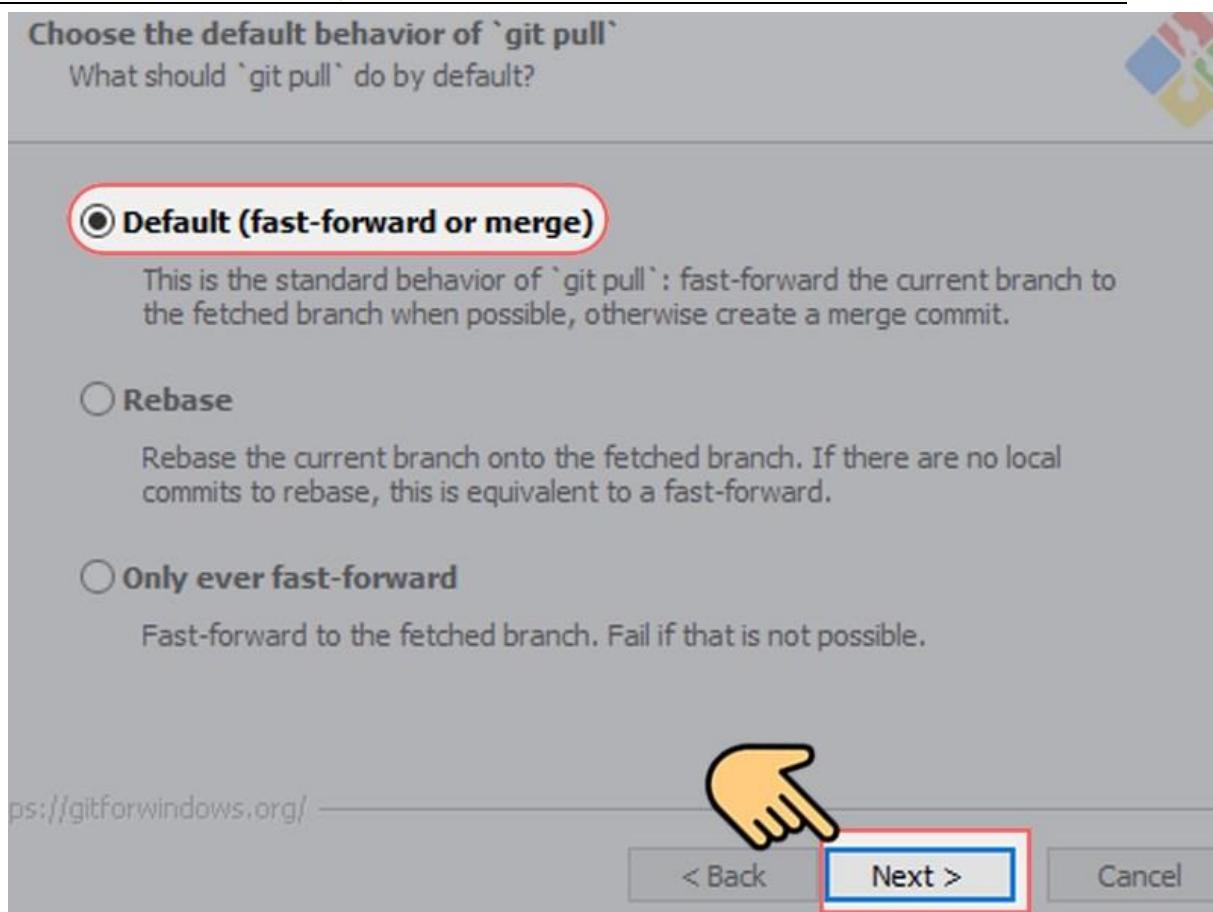
Hình 164. Chọn Next để tiến hành bước tiếp theo.

Bước 11: Cửa sổ tiếp theo yêu cầu chúng ta lựa chọn chương trình hiển thị cửa sổ dòng lệnh, để tương tác với chúng ta về sau. Bạn có thể lựa chọn **Use Windows' default console window** (sử dụng cửa sổ bảng điều khiển mặc định của Windows) để chọn màn hình đen chữ trắng mặc định của Windows.



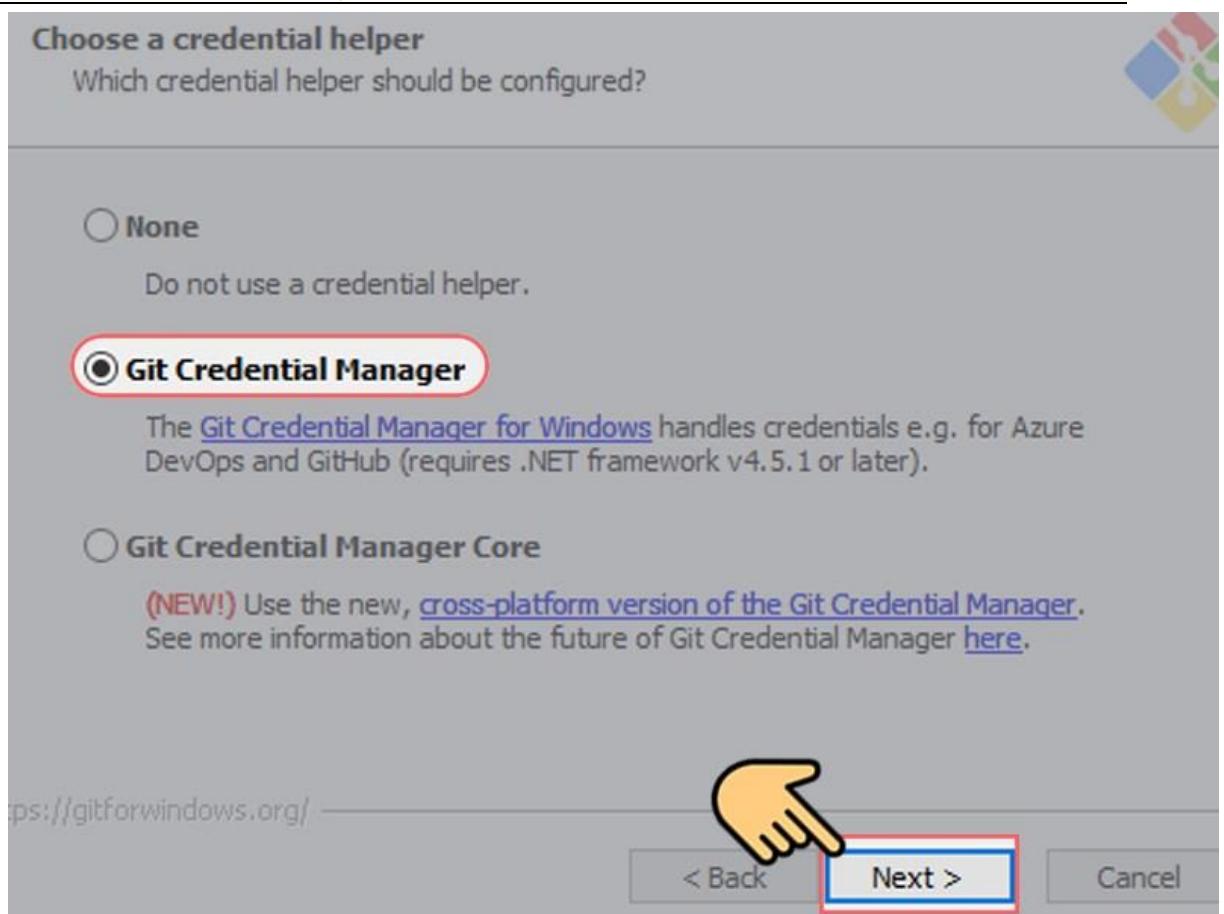
Hình 165. Giữ nguyên mục Use MinTTY (the default terminal of MSYS2).

Bước 12: Tiếp đến bạn sẽ lựa chọn hành vi mặc định của git. Bạn để nguyên lựa chọn mặc định là **Default (fast-forward or merge)** là chế độ mặc định của phần mềm. Chọn **Next**.



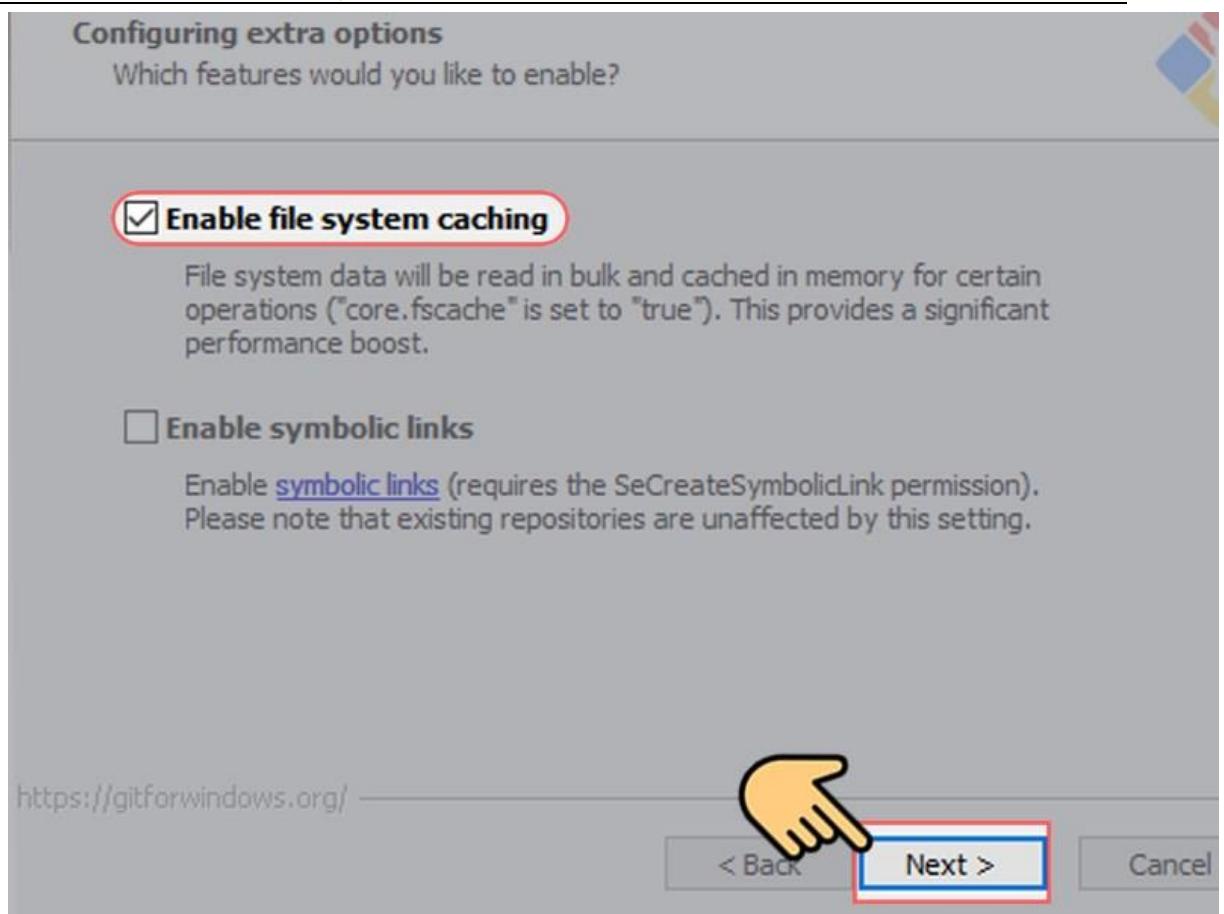
Hình 166. Tiếp theo là một số cài đặt mở rộng không quá quan trọng bạn cứ để lựa chọn mặc định

Bước 13: Chọn mục trợ giúp chứng chỉ, để nguyên lựa chọn mặc định là **Git credential Manager** (Trình quản lý thông tin xác thực Git). Sau đó **chọn Next**.



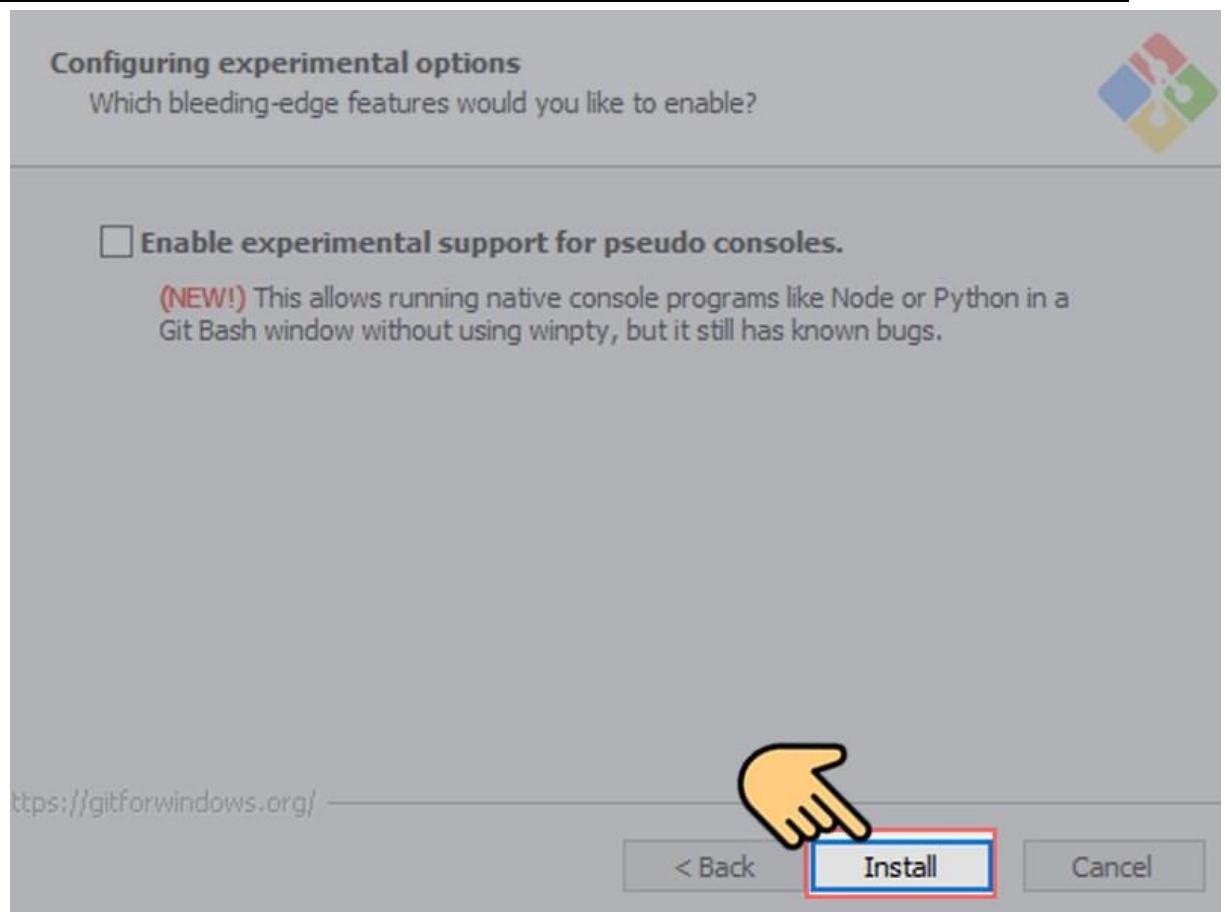
Hình 167. Giữ nguyên các mục lựa chọn

Bước 14 : Tiếp đến bạn sẽ lựa chọn cấu hình tùy chọn bổ sung, bạn để mặc định là Enable file system caching. Chọn Next.



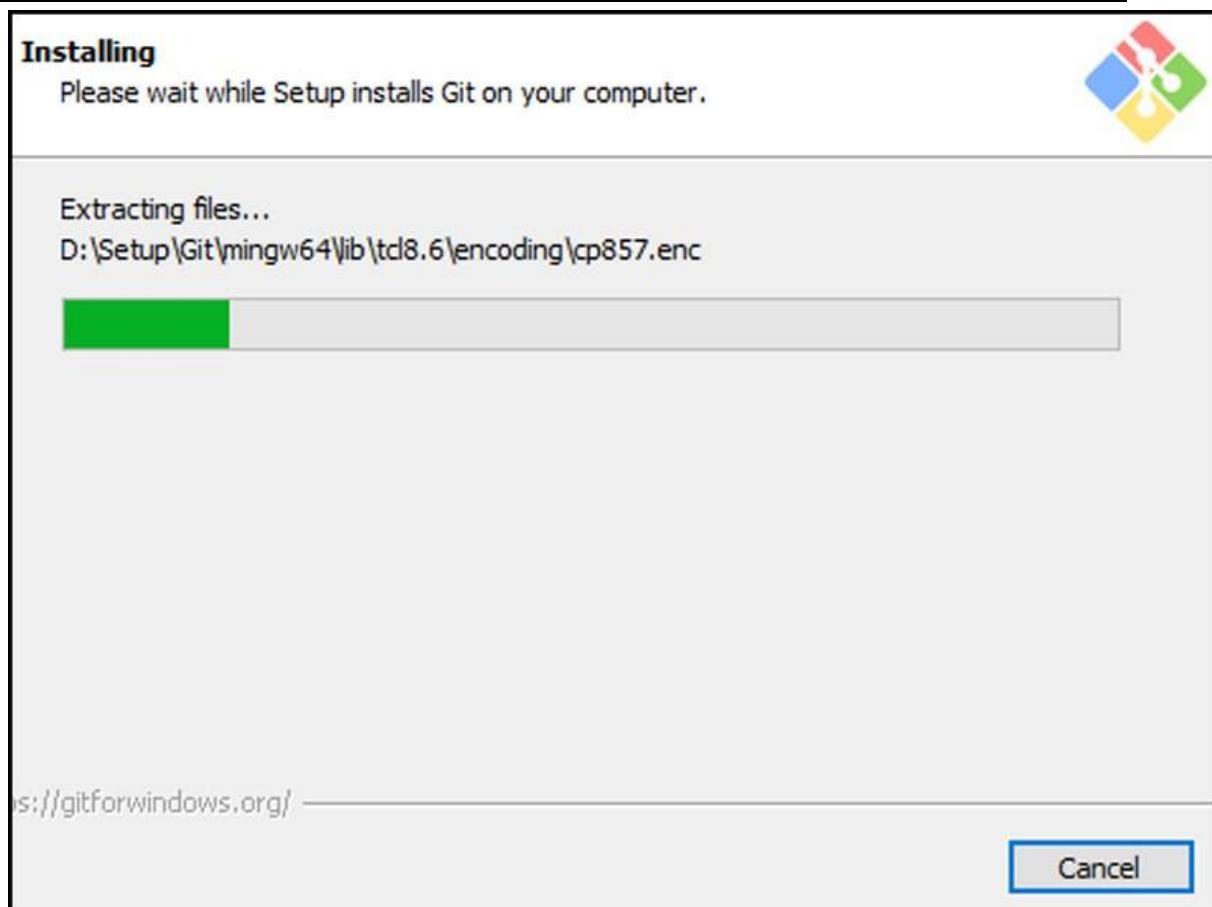
Hình 168. Chọn Next để tiến hành cài đặt

Bước 15: Cuối cùng bạn lựa chọn **Install** để kết thúc quá trình cài đặt. Bạn có thể chọn **Enable experimental support for pseudo consoles** (bật hỗ trợ thử nghiệm cho bảng điều khiển giả).



Hình 169. Cuối cùng bạn lựa chọn Install để kết thúc quá trình cài đặt.

Bây giờ việc của bạn chỉ là chờ đợi chương trình cài đặt vào máy nữa là xong.



Hình 170. Chờ đợi chương trình cài đặt vào máy

Bước 16: Cuối cùng bạn sẽ có 2 mục để bạn lựa chọn:

- Mục **Launch Git Bash**: để mở cửa sổ dòng lệnh của Git lên ngay sau đó.
- Mục **View Release Notes**: để xem ghi chú về các thay đổi của phiên bản này so với các phiên bản trước đó.

Các bạn có thể lựa chọn tùy theo nhu cầu của bản thân. Và **chọn Finish** để hoàn tất cài đặt.



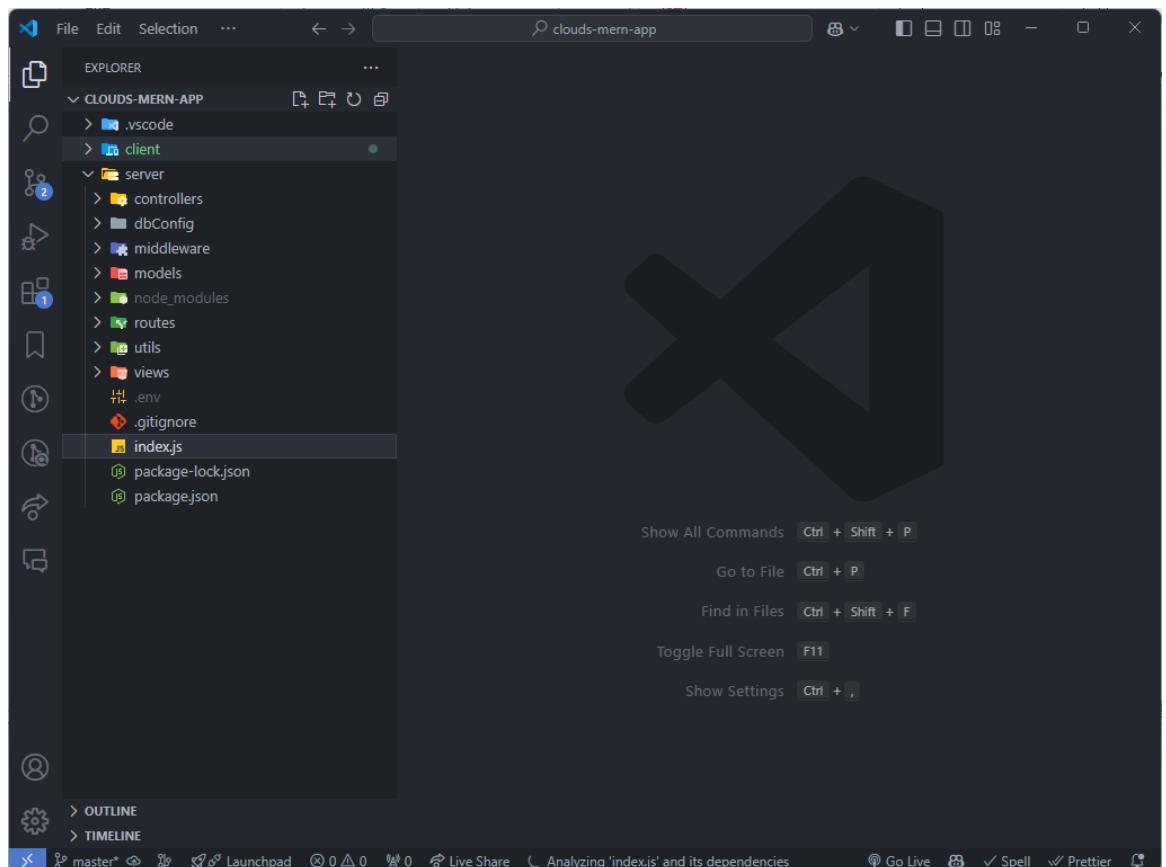
Hình 171. Chọn Finish để hoàn tất cài đặt.

3. Deploy ứng dụng web lên internet

Cách sử dụng render để deploy ứng dụng web lên internet. Ở đây, chúng em dùng render để đưa server của trang web lên.

Bước 1: Chuẩn bị mã nguồn ứng dụng Node.js.

Tô chúc code với server lưu ở file index.js:

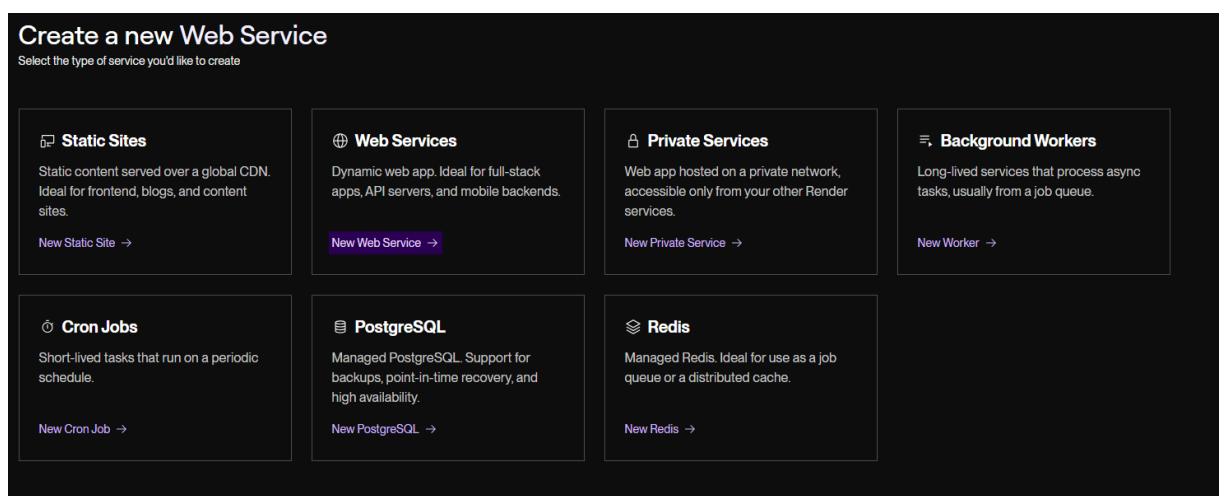


Hình 172. Tô chúc file như hình

Bước 2: Tạo tài khoản Render.

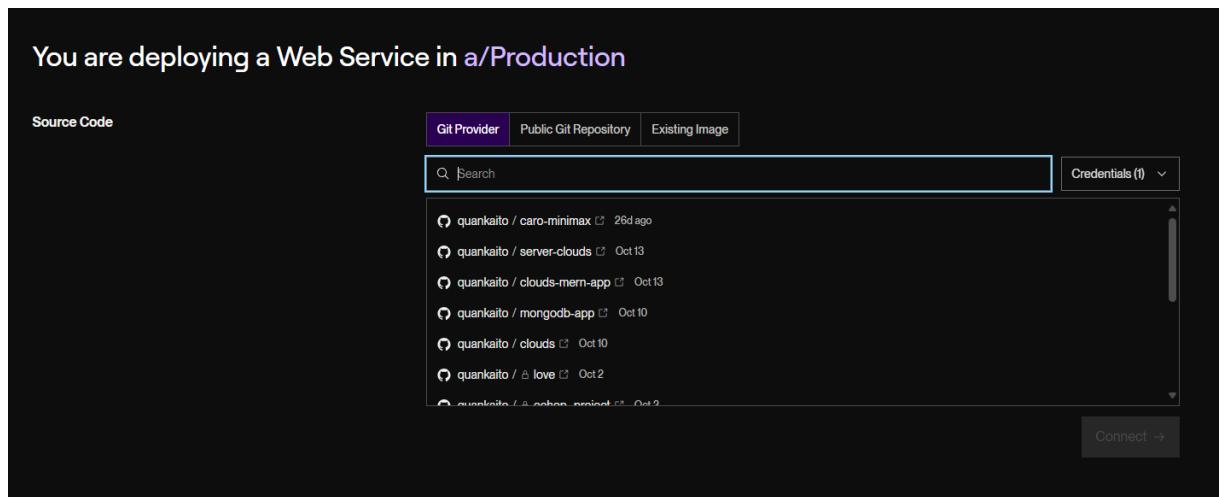
Truy cập vào link: <https://render.com/>

Sau khi Login thành công tài khoản thì tiến hành tạo Web Service bằng cách chọn New ->Web Service.



Hình 173. Chọn web service theo ý bạn

Bước 3: Kết nối với Github và Repository của dự án muốn deploy.



Hình 174. Chọn project

Bước 4: Config Web Service.

Tiến hành điền các thông tin tương ứng, ở đây hoàn toàn có thể sử dụng branch khác với master. Region thì mặc định để ở nơi gần mình nhất để tối ưu hiệu suất.

Điền start command và build command cho dự án nodejs, ở đây mình dùng npm install và npm start để tối ưu hiệu suất.

A screenshot of the Render web service configuration interface. It shows various settings: Project (Optional, dropdown), Language (Node), Branch (master), Region (Singapore, highlighted with a purple background), Root Directory (e.g., SRC), Build Command (\$ npm install), Start Command (\$ npm start), and Instance Type (For hobby projects, Free, 512 MB RAM).

Hình 175. Config web service

Add các environment variables vào mục advanced.

Environment Variables

Set environment-specific config and secrets (such as API keys), then read those values from your code. [Learn more](#)

Key	Value
APP_URL
AUTH_EMAIL
AUTH_PASSWORD
JWT_SECRET_KEY
MONGODB_URL

Hình 176. Thêm các biến môi trường vào

Chọn Create Web Service, đợi một vài phút để Build, sau đó ứng dụng của bạn đã được deploy.

WEB SERVICE

server-clouds Node Free Upgrade your instance →

Connect Manual Deploy

Events Logs Disks Environment Shell Previews Jobs Metrics Scaling Settings

All logs Search

Nov 16 02:37:06 PM A (node:88) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version

Nov 16 02:37:06 PM A Server running on port: 106998

Nov 16 02:37:07 PM A Connected Successfully

Nov 16 02:37:09 PM A POST /users/get-friend-request 200 37.144 ms - 26

Nov 16 02:37:09 PM A POST /users/suggested-friends 200 54.812 ms - 482

Nov 16 02:37:09 PM A POST /users/get-user 200 230.982 ms - 1217

Nov 16 02:37:09 PM A POST /posts 200 230.274 ms - 4885

Nov 16 03:11:33 PM A *** Running 'node index.js'

Nov 16 03:11:38 PM A (node:88) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version

Nov 16 03:11:38 PM A (Use 'node --trace-warnings ...' to show where the warning was created)

Nov 16 03:11:38 PM A (node:88) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will be removed in the next major version

Nov 16 03:11:38 PM A Server running on port: 106998

Nov 16 03:11:39 PM A DB Connected Successfully

Hình 177. Deploy thành công

CHƯƠNG III. ĐÁNH GIÁ KẾT QUẢ

I. KẾT QUẢ ĐẠT ĐƯỢC

1. Kết quả thực nghiệm

Trong quá trình thực hiện đề tài, nhóm đã xây dựng thành công một ứng dụng web mạng xã hội nhỏ sử dụng MERN stack (MongoDB, Express.js, React.js, Node.js). Đây là kết quả của việc áp dụng lý thuyết vào thực tế, minh họa rõ ràng cho tính hiệu quả của MongoDB khi được tích hợp vào quy trình phát triển ứng dụng. Một số kết quả nổi bật bao gồm:

a. Xây dựng cơ sở dữ liệu:

- Dữ liệu người dùng, bài viết, và bình luận, đăng ký, đăng nhập, xác thực, bạn bè được lưu trữ dưới dạng tài liệu trong MongoDB, cho phép thiết kế cấu trúc linh hoạt mà không cần tuân thủ schema cố định.
- Sử dụng MongoDB Atlas để triển khai cơ sở dữ liệu trên đám mây, giúp việc quản lý và mở rộng trở nên dễ dàng.

b. Hiệu năng hoạt động:

- Truy xuất dữ liệu nhanh chóng, đảm bảo thời gian phản hồi ngắn cho các thao tác như hiển thị bài viết, thêm bình luận, và cập nhật trạng thái người dùng.
- Thực nghiệm với khối lượng dữ liệu lớn hơn dự kiến ban đầu, ứng dụng vẫn hoạt động ổn định nhờ khả năng mở rộng theo chiều ngang của MongoDB.

c. Tích hợp với MERN stack:

- Dữ liệu từ MongoDB được xử lý hiệu quả thông qua API RESTful trên nền tảng Node.js và Express.js.
- Frontend React hiển thị dữ liệu từ cơ sở dữ liệu với các tính năng tương tác thời gian thực như cập nhật trạng thái và thông báo.

d. Quy trình phát triển:

- Nhóm sử dụng GitHub để quản lý mã nguồn, theo dõi lịch sử thay đổi và hỗ trợ làm việc nhóm hiệu quả.
- Triển khai CI/CD giúp đảm bảo ứng dụng luôn được cập nhật, giảm thiểu lỗi trong quá trình phát triển.

2. So sánh với mục tiêu ban đầu

Mục tiêu ban đầu:

Áo hóa và điện toán đám mây – Nhóm 3

- Xây dựng một ứng dụng web nhỏ với chức năng cơ bản của mạng xã hội: đăng bài, bình luận, và quản lý người dùng.
- Tích hợp MongoDB để lưu trữ dữ liệu một cách linh hoạt và hiệu quả.
- Sử dụng GitHub để quản lý mã nguồn và hỗ trợ làm việc nhóm.

So sánh với kết quả thực tế:

Về chức năng:

- Ứng dụng đã đáp ứng đầy đủ các mục tiêu đề ra. Hơn nữa, nhóm đã bổ sung một số tính năng mở rộng như cập nhật trạng thái người dùng theo thời gian thực và hệ thống thông báo cơ bản.

Về cơ sở dữ liệu:

- MongoDB hoạt động vượt mong đợi với khả năng lưu trữ và xử lý linh hoạt, đặc biệt khi khối lượng dữ liệu tăng. Việc tích hợp MongoDB Atlas giúp nhóm quản lý cơ sở dữ liệu dễ dàng hơn so với các phương án triển khai truyền thống.

Về quy trình phát triển:

- Sử dụng GitHub đã tối ưu hóa việc công tác nhóm. Tất cả thành viên có thể làm việc đồng bộ, theo dõi thay đổi, và giải quyết xung đột mã nguồn nhanh chóng.

Hiệu năng:

- Ứng dụng vận hành ổn định ngay cả khi thử nghiệm với lượng lớn người dùng giả định (sử dụng công cụ giả lập), nhờ sự hỗ trợ từ MongoDB và khả năng mở rộng của MERN stack.

II. ĐÁNH GIÁ VÀ SO SÁNH

1. Đánh giá ưu/nhược điểm của MongoDB

Ưu Điểm:

- MongoDB có nhiều ưu điểm hơn so với những loại khác. Ưu điểm đầu tiên của MongoDB chính là sử dụng lưu trữ dữ liệu dưới dạng Document JSON. Nhờ có nó nên mỗi một collection đều sẽ có các kích cỡ và các Document khác nhau. Sự linh hoạt trong việc lưu trữ dữ liệu của MongoDB là rất hữu dụng. Chính vì vậy, các bạn hoàn toàn có thể sử dụng MongoDB để Insert dữ liệu bất cứ lúc nào.
- Ưu điểm thứ hai của MongoDB đó chính là nó không có sự ràng buộc lẫn nhau trong dữ liệu. Các bạn sẽ không cần phải join như trong RDBMS. Nên khi sử dụng insert hay xóa, update sẽ không cần tốn nhiều thời gian. Chúng ta cũng không cần phải chờ xem nó có thỏa mãn các ràng buộc dữ liệu hay không để tiến hành insert.

- Sử dụng MongoDB có thể mở rộng dễ dàng hơn. Trong nền tảng này có một khái niệm cluster là cụm các node chứa dữ liệu giao tiếp với nhau. Khi bạn muốn mở rộng một hệ thống, các bạn chỉ cần thêm một node vào cluster. Đây chính là sự nhanh nhẹn khi dùng MongoDB.
- Trường dữ liệu “_id” luôn tự động đánh chỉ mục index ở MongoDB để tốc độ truy vấn thông tin nhanh nhất. Khi có một truy vấn dữ liệu, bản ghi của cached sẽ cho lên bộ nhớ Ram. Từ đó phục vụ lần lượt các truy vấn của người dùng, diễn ra nhanh hơn mà không cần đọc từ ổ cứng.
- Ngoài ra, sử dụng MongoDB còn hỗ trợ hiệu năng cao cho người dùng. Ví dụ như tốc độ truy vấn find, update, insert hay delete. Tất cả đều được tối ưu nhanh hơn so với các hệ thống quản trị dữ liệu quan hệ khác. Từ những thử nghiệm cho thấy, tốc độ của MongoDB có thể nhanh gấp 100 lần so với MySQL.

Nhược điểm:

- Mặc dù MongoDB có nhiều ưu điểm như vậy, tuy nhiên nó vẫn chưa phải hoàn hảo. Vẫn sẽ có những nhược điểm nhất định khi sử dụng.
- Ưu điểm đôi khi là nhược điểm. Mặc dù ưu điểm khi sử dụng MongoDB đó là không có quá nhiều ràng buộc như trong RDBMS. Tuy nhiên, chính điều này khiến cho nhiều người dùng lo lắng. Khi thao tác trên MongoDB các bạn cần phải cẩn thận hơn vì không có sự ràng buộc.
- MongoDB bị nhiều người dùng đánh giá là tồn bộ nhớ do lưu dữ liệu dưới dạng key – value, collection. Nền tảng này các dữ liệu chỉ khác nhau về value, do đó, key vẫn sẽ bị lặp lại nhiều lần. Vì không hỗ trợ Join nên dễ dẫn đến dư thừa dữ liệu.
- Khi sử dụng MongoDB có nguy cơ gây mất dữ liệu khi chưa hoàn thành bản lưu. Điều này hoàn toàn có thể xảy ra bởi quá trình insert hay update, remove bản ghi của MongoDB không cập nhật ngay xuống ổ cứng. Phải mất khoảng 60 giây nền tảng này mới thực hiện ghi toàn bộ dữ liệu thay đổi từ Ram vào ổ cứng. Nếu có sự cố như mất điện xảy ra trong trường hợp này, thì người dùng có thể bị mất dữ liệu.

2. So sánh MongoDB với MySQL

MongoDB và MySQL là hai nền tảng cơ sở dữ liệu phổ biến nhưng phục vụ các mục đích khác nhau. Sự khác biệt chính nằm ở cấu trúc dữ liệu, mô hình hoạt động và tính năng của chúng.

Mô hình cơ sở dữ liệu

MongoDB:

- Là cơ sở dữ liệu NoSQL, sử dụng mô hình lưu trữ tài liệu (document-based).
- Dữ liệu được lưu dưới dạng BSON (một dạng nhị phân của JSON).
- Không cần schema cố định, cho phép cấu trúc dữ liệu linh hoạt.

MySQL:

- Là cơ sở dữ liệu quan hệ (Relational Database Management System - RDBMS).
- Dữ liệu được tổ chức thành các bảng với các hàng và cột.
- Yêu cầu schema chặt chẽ, đảm bảo dữ liệu có cấu trúc rõ ràng.

Khả năng mở rộng

MongoDB:

- Hỗ trợ mở rộng theo chiều ngang (horizontal scaling) bằng cách phân mảnh dữ liệu (sharding).
- Phù hợp với các hệ thống lớn, cần mở rộng nhanh chóng.

MySQL:

- Chủ yếu hỗ trợ mở rộng theo chiều dọc (vertical scaling) bằng cách nâng cấp phần cứng.
- Có thể mở rộng theo chiều ngang, nhưng việc triển khai phức tạp hơn MongoDB.

Xử lý dữ liệu

MongoDB:

- Thích hợp cho dữ liệu phi cấu trúc hoặc bán cấu trúc.
- Hỗ trợ lưu trữ dữ liệu dạng JSON, rất tiện lợi cho các ứng dụng web hiện đại.
- Dễ dàng thêm hoặc sửa đổi các trường dữ liệu mà không làm gián đoạn hệ thống.

MySQL:

- Phù hợp với dữ liệu có cấu trúc chặt chẽ và mối quan hệ giữa các bảng.
- Tối ưu cho các hệ thống yêu cầu tính toàn vẹn dữ liệu cao nhờ các khóa chính, khóa ngoại.

Tốc độ và hiệu năng

MongoDB:

- Hiệu năng tốt hơn khi xử lý khôi lượng lớn dữ liệu phi cấu trúc hoặc bán cấu trúc.
- Tốc độ truy xuất nhanh nhờ kiến trúc không cần schema và lưu trữ tài liệu.

MySQL:

- Hiệu năng cao hơn khi làm việc với dữ liệu có cấu trúc chặt chẽ.
- Tối ưu cho các truy vấn phức tạp và các giao dịch (transactions).

Hỗ trợ giao dịch

Trang 122

MongoDB:

- Hỗ trợ giao dịch ACID (Atomicity, Consistency, Isolation, Durability) từ phiên bản 4.0 trở đi, nhưng chưa tối ưu cho các hệ thống yêu cầu nhiều giao dịch phức tạp.

MySQL:

- Hỗ trợ giao dịch ACID tốt và đáng tin cậy.
- Phù hợp với các ứng dụng yêu cầu bảo toàn dữ liệu cao như ngân hàng, quản lý tài chính.

Hệ sinh thái và công cụ hỗ trợ

MongoDB:

- Cung cấp các công cụ như MongoDB Compass, Atlas (dịch vụ đám mây) giúp quản lý và triển khai dễ dàng.
- Cộng đồng đang phát triển mạnh, hỗ trợ các ngôn ngữ lập trình hiện đại như Node.js, Python.

MySQL:

- Có hệ sinh thái lâu đời với nhiều công cụ như MySQL Workbench, phpMyAdmin.
- Cộng đồng lớn và tài liệu phong phú do được sử dụng rộng rãi từ lâu.

Tính năng bảo mật

MongoDB:

- Hỗ trợ cơ chế mã hóa dữ liệu, quyền truy cập (role-based access control).
- Tích hợp tốt với các dịch vụ đám mây để bảo mật dữ liệu.

MySQL:

- Hỗ trợ bảo mật mạnh mẽ thông qua user authentication, SSL, và cơ chế cấp quyền truy cập chặt chẽ.

Ứng dụng thực tế

MongoDB:

- Phù hợp với các ứng dụng yêu cầu lưu trữ dữ liệu linh hoạt và quy mô lớn như ứng dụng mạng xã hội, hệ thống IoT, thương mại điện tử.

MySQL:

- Thích hợp với các hệ thống yêu cầu tính toàn vẹn dữ liệu cao và quy trình quản lý chặt chẽ như hệ thống ERP, CRM, hoặc quản lý ngân hàng.

Tóm tắt so sánh MongoDB và MySQL:

Tiêu chí	MongoDB	MySQL
Loại cơ sở dữ liệu	NoSQL (document-based)	RDBMS (relational)
Linh hoạt schema	Có	Không
Mở rộng	Theo chiều ngang	Theo chiều dọc
Dữ liệu phù hợp	Phi cấu trúc, bán cấu trúc	Có cấu trúc
Hỗ trợ giao dịch	Hạn chế hơn	Tốt
Hiệu năng	Tốt với dữ liệu lớn, phi cấu trúc	Tốt với dữ liệu có cấu trúc
Ứng dụng chính	Web hiện đại, IoT, Big Data	ERP, CRM, ngân hàng

Lựa chọn nào phù hợp?

- MongoDB: Khi cần xử lý dữ liệu phi cấu trúc, mở rộng linh hoạt, hoặc phát triển nhanh các ứng dụng hiện đại.
- MySQL: Khi cần hệ thống ổn định, quản lý dữ liệu có cấu trúc rõ ràng, yêu cầu giao dịch phức tạp.

III. HƯỚNG PHÁT TRIỂN

1. Sức ảnh hưởng của MongoDB với hiện tại

MongoDB hiện là một trong những cơ sở dữ liệu NoSQL phổ biến và có ảnh hưởng lớn trên toàn cầu, đặc biệt trong bối cảnh công nghệ hiện đại yêu cầu khả năng xử lý dữ liệu lớn và phi cấu trúc. Những sức ảnh hưởng nổi bật của MongoDB hiện nay bao gồm:

- **Khả năng xử lý dữ liệu phi cấu trúc và đa dạng:** MongoDB cho phép lưu trữ và xử lý dữ liệu dưới dạng tài liệu JSON, mang lại sự linh hoạt cao trong việc xử lý dữ liệu phi cấu trúc hoặc bán cấu trúc. Điều này giúp các doanh nghiệp dễ dàng lưu trữ thông tin phức tạp như hình ảnh, video, hoặc dữ liệu từ IoT.
- **Hiệu suất cao và khả năng mở rộng:** Với kiến trúc không cần schema (schema-less), MongoDB hỗ trợ mở rộng theo chiều ngang (horizontal scaling) thông qua kỹ thuật sharding, cho phép hệ thống hoạt động ổn định kể cả khi khối lượng dữ liệu tăng lên nhanh chóng.
- **Ứng dụng trong các ngành công nghiệp lớn:** MongoDB được áp dụng trong nhiều ngành như thương mại điện tử, y tế, giáo dục, và các ứng dụng AI. Những doanh nghiệp hàng đầu như eBay, Uber, và Forbes sử dụng MongoDB để xử lý dữ liệu thời gian thực và tối ưu hóa trải nghiệm người dùng.

- Tích hợp với các công cụ phát triển hiện đại: Nhờ sự hỗ trợ từ cộng đồng và hệ sinh thái phong phú, MongoDB dễ dàng tích hợp với các framework phổ biến như Node.js, React, hoặc các dịch vụ đám mây như AWS, Azure, và Google Cloud. Điều này làm tăng tính linh hoạt trong việc triển khai các ứng dụng web.
- Khả năng đồng bộ hóa và cộng tác: Khi kết hợp với GitHub, MongoDB không chỉ hỗ trợ việc quản lý cơ sở dữ liệu mà còn tạo điều kiện thuận lợi cho các nhóm phát triển hợp tác, triển khai và quản lý mã nguồn một cách hiệu quả.

2. Định hướng cho tương lai của MongoDB

Sự phát triển không ngừng của MongoDB hứa hẹn những cải tiến và định hướng mới trong tương lai nhằm đáp ứng tốt hơn nhu cầu của người dùng và sự thay đổi của công nghệ. Một số định hướng chính bao gồm:

- Cải tiến hiệu năng và khả năng mở rộng: MongoDB tiếp tục tối ưu hóa các thuật toán xử lý dữ liệu, đặc biệt trong việc hỗ trợ các hệ thống dữ liệu lớn và phân tán. Điều này nhằm đảm bảo tính ổn định và hiệu năng cao ngay cả trong những hệ thống có yêu cầu khắt khe.
- Tích hợp mạnh mẽ với trí tuệ nhân tạo (AI) và học máy (ML): Với xu hướng phát triển mạnh mẽ của AI, MongoDB có thể được cải thiện để tối ưu hóa cho các ứng dụng AI và ML, bao gồm việc hỗ trợ dữ liệu huấn luyện quy mô lớn và xử lý thời gian thực.
- Bảo mật nâng cao: Trước bối cảnh ngày càng nhiều rủi ro an ninh mạng, MongoDB sẽ tập trung phát triển các tính năng bảo mật như mã hóa dữ liệu toàn phần (end-to-end encryption), phát hiện truy cập bất thường, và bảo vệ dữ liệu nhạy cảm.
- Hỗ trợ môi trường đa đám mây (Multi-Cloud): MongoDB sẽ cải tiến khả năng tích hợp với nhiều nền tảng đám mây cùng lúc, giúp doanh nghiệp tối ưu hóa chi phí và tận dụng tối đa sức mạnh của các dịch vụ đám mây khác nhau.
- Thúc đẩy cộng đồng và mã nguồn mở: Việc tăng cường sự tham gia của cộng đồng mã nguồn mở sẽ giúp MongoDB phát triển nhanh hơn và có thêm nhiều tính năng phù hợp với nhu cầu thực tiễn.
- Tăng cường hỗ trợ phân tích dữ liệu lớn: MongoDB dự kiến mở rộng các công cụ phân tích dữ liệu trực tiếp trên nền tảng, giúp các doanh nghiệp thực hiện báo cáo và đưa ra quyết định chiến lược nhanh chóng hơn.
- Phát triển công cụ hỗ trợ dành cho lập trình viên: Các công cụ như MongoDB Compass, Atlas Search sẽ tiếp tục được phát triển, giúp lập trình viên dễ dàng hơn trong việc thiết kế, triển khai và quản lý cơ sở dữ liệu.

TÀI LIỆU THAM KHẢO

- [1] *mern-stack*. (2023, November 13). Retrieved from geeksforgeeks:
<https://www.geeksforgeeks.org/mern-stack/>
- [2] *mongodb get started*. (n.d.). Retrieved from w3schools:
https://www.w3schools.com/mongodb/mongodb_get_started.php
- [3] *What is MongoDB?* (n.d.). Retrieved from mongodb:
<https://www.mongodb.com/docs/manual/>