Báo cáo giai đoạn 1

# Tìm hiểu về mongoDB

1. Giới thiệu

MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) mã nguồn mở, được phát triển bởi công ty MongoDB Inc. Nó được thiết kế để xử lý các tải công việc lớn, linh hoạt và có khả năng mở rộng dễ dàng. MongoDB sử dụng mô hình dữ liệu linh hoạt, gọi là BSON (Binary JSON), để lưu trữ dữ liệu trong các tài liệu JSON tương tự như các bản ghi hoặc tài liệu trong các cặp khóa-giá trị.

1. Ưu điểm

* Mô hình dữ liệu linh hoạt: MongoDB cho phép lưu trữ dữ liệu có cấu trúc và không cấu trúc một cách tự nhiên. Điều này cho phép phát triển ứng dụng mà có thể thay đổi cấu trúc dữ liệu một cách linh hoạt theo nhu cầu, mà không cần thực hiện các thay đổi cơ sở dữ liệu phức tạp.
* Tốc độ và hiệu suất cao: MongoDB được thiết kế để cung cấp hiệu suất cao và khả năng xử lý tải công việc lớn. Nó sử dụng mô hình lưu trữ dựa trên bộ nhớ và hỗ trợ các truy vấn song song, cho phép thực hiện các truy vấn nhanh chóng và xử lý hàng ngàn yêu cầu trên giây.
* Khả năng mở rộng dễ dàng: MongoDB hỗ trợ khả năng mở rộng ngang (horizontal scaling) một cách dễ dàng. Bạn có thể thêm các nút vào cụm MongoDB một cách linh hoạt để tăng khả năng xử lý và lưu trữ dữ liệu. Cơ sở dữ liệu có thể mở rộng theo nhu cầu, giúp đáp ứng tải công việc lớn và đảm bảo sẵn sàng cao.
* Truy vấn mạnh mẽ: MongoDB cung cấp một ngôn ngữ truy vấn linh hoạt và mạnh mẽ, cho phép thực hiện các truy vấn phong phú, bao gồm các truy vấn tiêu chí, truy vấn tìm kiếm văn bản đầy đủ và truy vấn địa lý. Các phép tính trên dữ liệu như tổng, trung bình, min, max cũng được hỗ trợ.

1. Nhược điểm

* Giới hạn về khả năng truy vấn: Mặc dù MongoDB cung cấp một ngôn ngữ truy vấn mạnh mẽ, nhưng so với hệ quản trị cơ sở dữ liệu SQL truyền thống, nó có một số giới hạn trong khả năng truy vấn phức tạp. Vì MongoDB không hỗ trợ các liên kết (joins) giữa các bảng, việc thực hiện các truy vấn liên quan đến nhiều bảng có thể trở nên phức tạp và không hiệu quả.
* Không phù hợp cho dữ liệu có quan hệ phức tạp: Nếu ứng dụng của bạn có dữ liệu có quan hệ phức tạp và yêu cầu các tính năng như khóa ngoại (foreign key) và ràng buộc liên bảng (table constraints), MongoDB có thể không phải là lựa chọn tốt nhất. Hệ quản trị cơ sở dữ liệu SQL truyền thống thường phù hợp hơn trong trường hợp này.
* Đòi hỏi tài nguyên lưu trữ và bộ nhớ: MongoDB có xu hướng sử dụng tài nguyên lưu trữ và bộ nhớ khá cao. Vì dữ liệu được lưu trữ dưới dạng các tài liệu BSON, nó có thể tốn nhiều dung lượng lưu trữ hơn so với hệ quản trị cơ sở dữ liệu SQL truyền thống.

# Tìm hiểu về NodeJS

Node.js là một môi trường chạy mã JavaScript phía máy chủ (server-side) xây dựng trên JavaScript Engine của Google Chrome (V8 Engine). Nó cho phép viết mã JavaScript để thực thi các tác vụ server-side, như xử lý yêu cầu HTTP, quản lý cơ sở dữ liệu, xử lý tác vụ đa luồng, và nhiều hơn nữa.

1. Lợi ích

* Hiệu suất cao: Node.js sử dụng mô hình non-blocking, sự kiện-driven (event-driven) và I/O không đồng bộ (asynchronous I/O), cho phép xử lý đồng thời hàng nghìn kết nối mà không bị chặn. Điều này tạo ra hiệu suất cao và khả năng mở rộng tốt.
* Giao diện lập trình đơn giản: Node.js sử dụng JavaScript, một ngôn ngữ lập trình phổ biến và dễ học, cho cả phía máy chủ và phía client. Việc sử dụng cùng một ngôn ngữ cho cả phía client và phía server giúp đơn giản hóa việc phát triển và duy trì ứng dụng.
* Hệ sinh thái mạnh mẽ: Node.js có một hệ sinh thái mạnh mẽ với hàng ngàn module được phát triển bởi cộng đồng, giúp giải quyết nhiều vấn đề thường gặp và tăng tốc quá trình phát triển ứng dụng.
* Phát triển nhanh: Node.js hỗ trợ mô hình phát triển mã nguồn mở và các công cụ phát triển mạnh mẽ như npm (Node Package Manager). Có nhiều module và thư viện sẵn có để tái sử dụng, giúp giảm thời gian và công sức phát triển.
* Scalability (khả năng mở rộng): Node.js được xây dựng với khả năng mở rộng dễ dàng. Với khả năng xử lý đồng thời cao, nó có thể xử lý hàng nghìn kết nối đồng thời mà không gặp vấn đề về hiệu suất.
* Real-time applications: Node.js rất phù hợp cho việc phát triển các ứng dụng thời gian thực (real-time applications) như ứng dụng chat, trò chơi trực tuyến, streaming media và ứng dụng cần cập nhật dữ liệu liên tục.

1. Nhược điểm

* Single-threaded: Node.js là một môi trường single-threaded, có nghĩa là nó chỉ sử dụng một luồng duy nhất để xử lý yêu cầu. Điều này có nghĩa rằng nếu có một tác vụ lâu dài hoặc chặn, nó có thể ảnh hưởng đến hiệu suất của ứng dụng
* Quản lý callback: Sử dụng các hàm gọi lại (callbacks) để xử lý các tác vụ không đồng bộ có thể khiến mã trở nên phức tạp và khó đọc. Tuy nhiên, có thể sử dụng các công nghệ như Promises hoặc async/await để giải quyết vấn đề này.
* Hạn chế về quản lý bộ nhớ: Node.js sử dụng bộ nhớ đơn giản và không có garbage collector tự động. Điều này có nghĩa là phải cẩn thận quản lý bộ nhớ để tránh rò rỉ bộ nhớ (memory leaks) và sử dụng các công cụ như Heapdump để phân tích và giải quyết vấn đề.

# Dự kiến công việc

* Viết các API : đăng nhập, đăng xuất, CRUD thông tin người dùng