BÁO CÁO THỰC HÀNH CHƯƠNG 2

Nhóm 8

|  |  |
| --- | --- |
| Họ & tên | MSSV |
| Nguyễn Hoàng Hiệp | 20205204 |
| Lê Chu Trung Hiếu | 20205229 |
| Lê Ngọc Đăng Khoa | 20205209 |
| Trịnh Phú Quang | 20205219 |
| Lê Hữu Tài | 20205221 |

## 1. Xây dựng một Chat room sử dụng socket.io

Câu 1: Tệp nào vừa xuất hiện trong thư mục ChatRoomApp? Nó được sử dụng để làm gì?

* package.json: sử dụng để định nghĩa các thông tin quan trọng như tên dự án, phiên bản, tác giả, miêu tả, quyền sở hữu và các thành phần phụ thuộc..
* package-lock.json: là một tệp tự động được tạo ra bởi npm khi cài đặt các phụ thuộc của dự án. Nó chứa thông tin chi tiết về cấu trúc cây phụ thuộc của dự án và các phiên bản chính xác của các gói đã được cài đặt, giúp đảm bảo tính nhất quán và đáng tin cậy trong việc cài đặt và triển khai phụ thuộc của dự án

Câu 2: Mở trình duyệt và gõ vào đó địa chỉ [*http://localhost:3000*](http://localhost:3000), bạn sẽ nhận được thông điệp gì?

\*\* Vì NodeJS được cài trên docker trên Debian, vậy nên cổng test mặc định của docker là 3000, thay vào đó trong file app.json và package.json, các thông số được đổi thành 3001 cho phù hợp vậy nên các câu hỏi liên quan tới cổng 3000 sẽ được đổi sang cổng 3001

A screen shot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated with medium confidence

Thông điệp hiện ra: Hello world

Câu 3: Bạn hãy thử reload (Ctrl-R) lại trình duyệt. Bạn có nhìn thấy gì mới xuất hiện trên cửa sổ không? Nếu không có gì xuất hiện hết thì là vì sao?

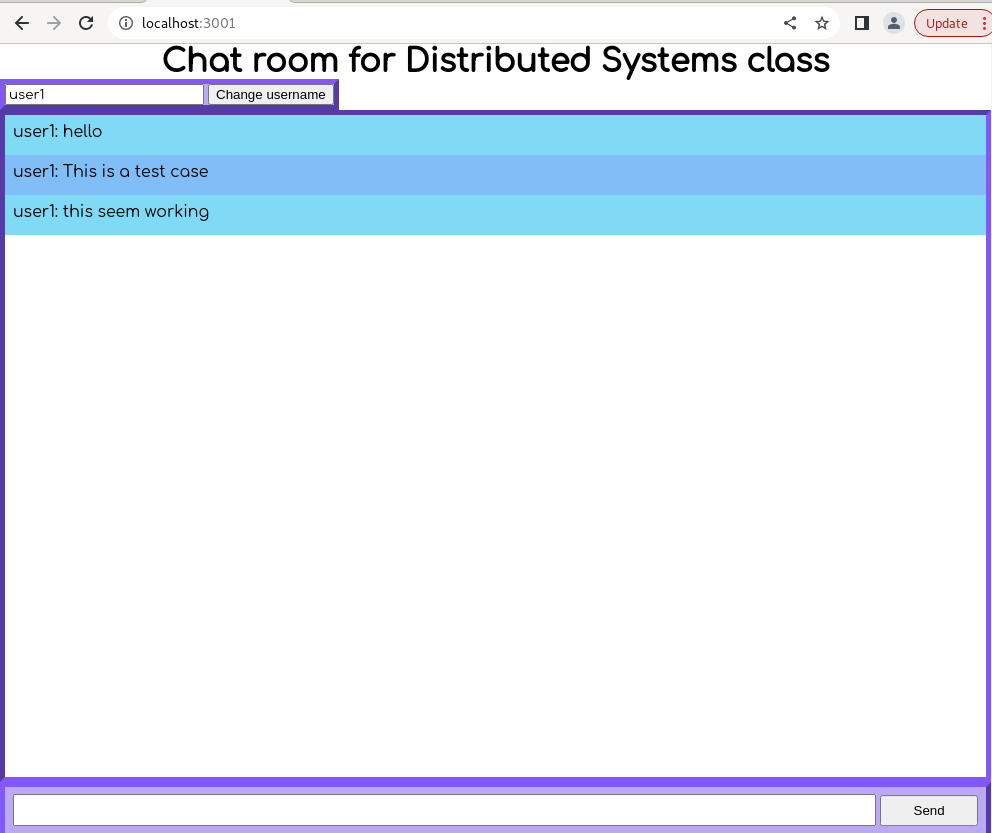
Chưa có gì mới xuất hiện trên cửa sổ bởi ta chưa xây dựng giao diện cho nó (bằng file HTML, CSS,...)

Câu 4: Refresh trang *localhost:3000*, bạn nhìn thấy thông điệp nào?

A screenshot of a computer

Description automatically generated with medium confidence

Câu 5: Bây giờ bạn hãy thử gõ gì đó lên một tab. Cùng lúc đó, nhìn sang tab khác của người dùng khác, bạn thấy gì?



## 2. Phát triển hệ thống RPC sử dụng RabbitMQ

Câu 6:

AMQP.BasicProperties replyProps = new AMQP.BasicProperties

.Builder()

.correlationId(delivery.getProperties().getCorrelationId())

.build();

Câu 7:

Trong đoạn code của class RPCClient, phương thức call() là phương thức được sử dụng để gửi yêu cầu đến Server thông qua hàng đợi rpc\_queue và đợi câu trả lời từ Server.

Đầu tiên, trong phương thức call(), Client sẽ tạo một corrId ngẫu nhiên để xác định yêu cầu của mình và tạo một hàng đợi mới để chờ câu trả lời từ Server bằng cách gọi channel.queueDeclare().getQueue(). Sau đó, Client tạo một đối tượng AMQP.BasicProperties để gửi cùng với yêu cầu, trong đó có chứa corrId và tên hàng đợi mới để Server có thể gửi câu trả lời.

Tiếp theo, Client sử dụng phương thức channel.basicPublish() để gửi yêu cầu lên Server thông qua hàng đợi rpc\_queue và chờ đợi câu trả lời từ Server.

Sau đó, Client sử dụng đối tượng CompletableFuture để đợi và nhận câu trả lời từ Server thông qua hàng đợi mới được tạo ra trước đó, trong đó sử dụng channel.basicConsume() để tiêu thụ và xử lý tin nhắn được gửi về từ Server. Khi nhận được câu trả lời từ Server, Client sẽ trả về kết quả đó.

Câu 8:

Câu lệnh rabbitmqctl list\_queues name messages\_ready messages\_unacknowledged được sử dụng để liệt kê thông tin về các hàng đợi (queues) hiện có trong RabbitMQ, bao gồm tên của hàng đợi, số lượng tin nhắn sẵn sàng (messages\_ready) và số lượng tin nhắn chưa được xác nhận (messages\_unacknowledged) trong hàng đợi.

Trong kết quả, cột đầu tiên là "name", chứa tên của các hàng đợi hiện có trong RabbitMQ. Cột thứ hai là "messages\_ready", cho biết số lượng tin nhắn đã sẵn sàng trong hàng đợi và có thể được dùng bởi các consumer của hàng đợi đó. Cột thứ ba là "messages\_unacknowledged", cho biết số lượng tin nhắn đã được gửi đến consumer, nhưng chưa được xác nhận nhận được bởi consumer đó.

Trong kết quả trên, có ba hàng đợi được liệt kê. Hai hàng đợi đầu tiên (amq.gen-XoBT9lJ1ktaa\_1YMaeOwTg và amq.gen-uK193oAYyYXFbItJAL3Ocg) là các hàng đợi được tạo tự động bởi RabbitMQ để hỗ trợ các tính năng như gửi tin nhắn trong mạng nội bộ (loopback). Cả hai hàng đợi này không có bất kỳ tin nhắn sẵn sàng hoặc chưa được xác nhận nào.

Hàng đợi thứ ba (rpc\_queue) chứa một tin nhắn đã sẵn sàng và một tin nhắn chưa được xác nhận. Tin nhắn đã sẵn sàng trong hàng đợi này có thể được dùng bởi một consumer hoặc một số consumer khác nhau, trong khi tin nhắn chưa được xác nhận đang được chờ đợi để xử lý bởi một consumer.