**ĐỒ ÁN MÔN HỌC**

**Học phần: Hệ thống phân tán**

**Chat GRPC**

**19126049 – Nguyễn Hoàng Anh Khôi**

**19126050 – Hoàng Lân**

**Mục lục**

[Hướng dẫn sử dụng 3](#_Toc132835796)

[Demo 5](#_Toc132835797)

[Hạn chế 14](#_Toc132835798)

[Nguồn tham khảo 15](#_Toc132835799)

# Hướng dẫn sử dụng

**Thực hiện các bước sau:**

* Khởi chạy Server bằng cách run file **Server.java** (src/main/java). Sau khi khởi chạy sẽ hận được thông báo đã khởi chạy tại port nào.
* Khởi chạy file **Client.java** (src/main/java) 5 lần (hoặc nhiều hơn tương ứng với số lượng client muốn khởi tạo). Sau khi khởi tạo sẽ hiện lên yêu cầu nhập tên người dùng.
* Sau khi nhập tên, bạn đã tham gia vào channel chat, tin nhắn của bạn có thể gửi đến các thành viên khác trong channel.
* Để thoát chương trình Client: nhập ***LEAVE*** để thoát và dừng chương trình.
* Để dừng Server, dùng ‘Ctrl + C’ nếu trong Terminal hoặc nhấn biểu tượng ‘stop’ trong IDLE đang dùng.

**Syntax cho Client:**

* Khi có ai đó gửi tin nhắn vào channel, những người khác sẽ nhận được tin nhắn có dạng:

***<Tên người gửi> (<messageID>): <nội dung tin nhắn>***

* Khi có ai đó tham gia vào channel, mọi người trong channel sẽ được thông báo về điều đó.
* Mỗi tin nhắn phải nhận được LIKE từ ít nhất 2 người trong channel thì người gửi mới có thể gửi tin nhắn tiếp theo.
* Để LIKE 1 tin nhắn, client nhập cú pháp:

***LIKE <messageID>***

* Người gửi không thể tự LIKE tin nhắn của mình.
* Cliet nhập ***LEAVE*** để rời channel và dừng chương trình.

# Demo

Khởi chạy Server, ta nhận được thông báo Server đã khởi chạy tại port:

Text

Description automatically generated

Khởi chạy Client, ta nhận được thông báo yêu cầu nhập tên người dùng:

A screenshot of a computer

Description automatically generated

Khi chạy các client khác, ta nhận được thông báo có client mới đã tham gia vào channel:

Graphical user interface, text

Description automatically generated

Bây giờ, ta có thể chat gì đó và mọi người đều sẽ nhận được.

Giả sử người dùng KHOI gửi một tin nhắn:Text

Description automatically generated

Các thành viên khác đều sẽ nhận được tin nhắn đó:

Text

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

Giả sử 1 người dùng khác gửi tin nhắn:

Text

Description automatically generated

Tất cả người dùng trong channel đều nhận được tin nhắn tương tự:

Text

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

Khi tin nhắn trước của 1 client chưa được like bởi 2 memmber khác, họ không thể gửi tin nhắn của mình và tất cả thành viên khác đều không thể thấy tin nhắn này.

Text

Description automatically generated

Giả sử có 2 client khác LIKE tin nhắn này, đều đó được broadcast đến mọi người trong channel.

Text

Description automatically generated Graphical user interface, text

Description automatically generated

Bây giờ, hay quay lại Server, ta sẽ thấy các dòng thông báo tương ứng mỗi khi Client gửi Request đến Server:

Text

Description automatically generated

(Những dòng WARNING thấy trên là do chương trình đã tự động đóng các blockingStub được khởi tạo giữa Client và Server khi Client gửi 1 LIKE Request, kết thúc hàm thì stub trên tự động đóng.)

Tiếp tục, khi 1 tin nhắn gần nhất của 1 người dùng có đủ 2 LIKE, họ có thể gửi tiếp 1 tin nhắn khác và mọi người trong channel sẽ thấy được nó.

Text

Description automatically generated

Text

Description automatically generated Text

Description automatically generated

Text

Description automatically generated Text

Description automatically generated

Nếu một người dùng gửi LIKE cho 1 message 2 lần hoặc tự LIKE message của chính mình, đều sẽ nhận được thông báo lỗi.

Text

Description automatically generated

A picture containing text, indoor

Description automatically generated

Và Server cũng in ra thông báo tương ứng.

Text

Description automatically generated

Bây giờ các Client muốn thoát channel và dừng chương trình:

Text

Description automatically generated

Tương tự, Server cũng sẽ đưa ra thông báo đã đóng 1 stub với mỗi client đã thoát.

Text

Description automatically generated

Và những thông tin trên đề được ghi lại ở file log của Server.

Text

Description automatically generated

# Hạn chế

1. Hiện tại, chương trình chưa thể đưa ra thông báo cho người dùng khi có ai đó rời chatroom.
2. Hiện tại, chương trình Demo nói trên thực thi thành công bằng cách thực thi Run của IDLE đang dùng (hiện tại là IntelliJ). Mặt khác, khi chạy trên Terminal, chương trình của Client xuất hiện lỗi thông báo rằng hàm nextLine() của Scanner (thư viện dùng để đọc dữ liệu nhập vào) nhận dữ liệu trống và trả ra lỗi (người dùng chưa kịp nhập bất kì dữ liệu nào, chương trình build trên Terminal, trả ra lỗi không nhận được đầu vào và lập tức thoát chương trình.) Lỗi này chưa tìm được cách khắc phục, dù chạy bằng Run của IDLE thì vẫn thực hiện thành công.

# Nguồn tham khảo

[Basics tutorial | Java | gRPC](https://grpc.io/docs/languages/java/basics/)

[google/protobuf-gradle-plugin: Protobuf Plugin for Gradle (github.com)](https://github.com/google/protobuf-gradle-plugin)

[Streaming with gRPC in Java | Baeldung](https://www.baeldung.com/java-grpc-streaming)

[[gRPC #8] Config Gradle to generate Java code from protobuf - YouTube](https://www.youtube.com/watch?v=nB7J6D_IXrA&list=PLy_6D98if3UJd5hxWNfAqKMr15HZqFnqf&index=8)

[[gRPC #9.2] Serialize protobuf message - Java - YouTube](https://www.youtube.com/watch?v=hS0sNC-sUzw&list=PLy_6D98if3UJd5hxWNfAqKMr15HZqFnqf&index=10)

[[gRPC #10.2] Implement unary gRPC API - Java - YouTube](https://www.youtube.com/watch?v=cweoyLhAbR0&list=PLy_6D98if3UJd5hxWNfAqKMr15HZqFnqf&index=12)

[[gRPC #11.2] Implement server-streaming gRPC API - Java - YouTube](https://www.youtube.com/watch?v=63vzrEsHfzg&list=PLy_6D98if3UJd5hxWNfAqKMr15HZqFnqf&index=14)

[[gRPC #12.2] Upload file in chunks with client-streaming gRPC - Java - YouTube](https://www.youtube.com/watch?v=rEmFMPv3TsA&list=PLy_6D98if3UJd5hxWNfAqKMr15HZqFnqf&index=16)

[[gRPC #13.2] Implement bidirectional-streaming gRPC - Java - YouTube](https://www.youtube.com/watch?v=2TOKvwoOpt4&list=PLy_6D98if3UJd5hxWNfAqKMr15HZqFnqf&index=25)