mQTT

Tài liệu mô tả giao thức mQTT của nhóm gồm 3 thành viên :
Đặng Quang Huy _ 18020653 _ K63N
Phạm Quang Bình _ 18020217 _ K63N
Ngô Quang Phong _ 18021000 _ K63N

I.Mô tả chung:

Giao thức thiết kể để truyền tải thông tin từ Publisher đên Subcriber qua một Broker ở giữa. Các broker có thể nhận nhiều kết nối, các Subcriber có thể subcribe nhiều topic và nhận dữ liệu cùng lúc, các Publisher có thể tạo ra topic và tự động gửi data đến 01 topic.

II.Phân chia công việc của từng thành viên:

1. Ngô Quang Phong : code Broker
2. Phạm Quang Bình: code Publisher và Subcriber
3. Đặng Quang Huy: thiết kế giao thức, viết tài liệu, code phần Publisher và Subcriber cùng Bình.

III.Cấu trúc các file trong project:

1.Trên Broker:

a. Core:
   + ActionType: code tương ứng với các yêu cầu gia tiếp.
   + Broker: File khởi chạy Broker chứa thông tin về giao diện broker và các event click button.
   + ResultCode: Chứa các giá trị phản hồi: thành công hoặc lỗi.
   + ServerManager: Chứa các hàm để xử lí thông tin đến broker
   + Topic : Chứa các thông tin mô tả về topic.
   + User: chứa các thông tin mô tả về client (cả publisher và subcribe)

b. N/A

2. Trên Publisher và subscriber:

Các file trên 2 thành phần này có chức năng giống nhau tuy nhiên có tinh chỉnh để phù hợp với cách hoạt động của từng loại user.(Mô tả chi tiết ở phần sau)

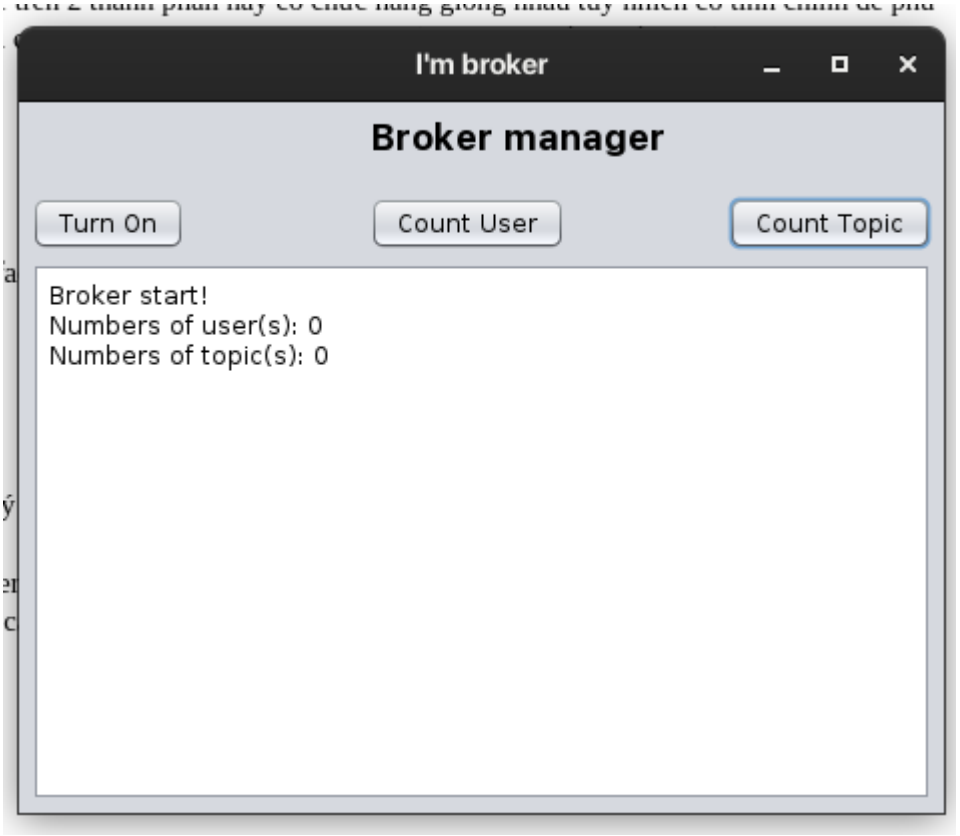
a. Core:
   + ActionType: code tương ứng với các yêu cầu gia tiếp.
   + ClientManager: Chứa các hàm để xử lí thông tin đến user.
   + Result: có các biến chứa biến kết quả.
   + ResultCode: Chứa các giá trị phản hồi: thành công hoặc lỗi.

b. Interface:
   + Login : Giao diện ban đầu và là file khởi chạy hệ thống.
   + TopicConservation: Giao diện hộp nhận thông tin(đôi với Subcriber) và gửi thông tin(đối với Publisher)

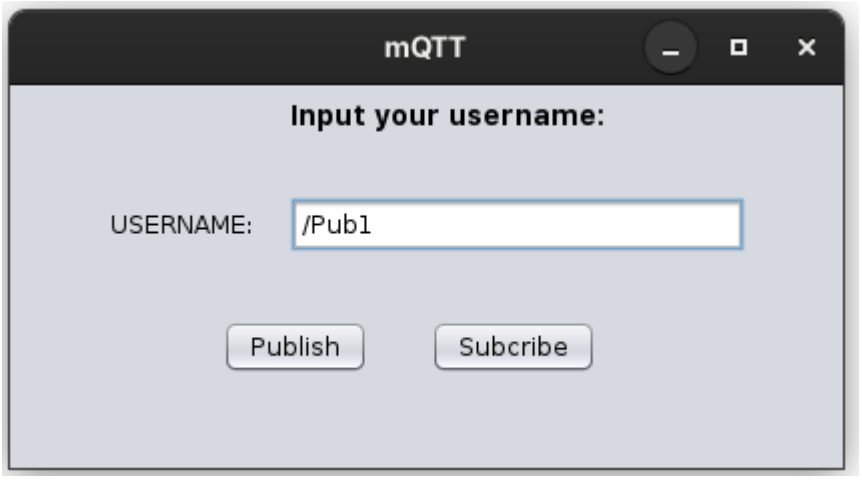
+ TopicList: Giao diện chứa thông tin về danh sách topic đang hoạt động và các nút chức năng.

IV. Chu trình hoạt động:

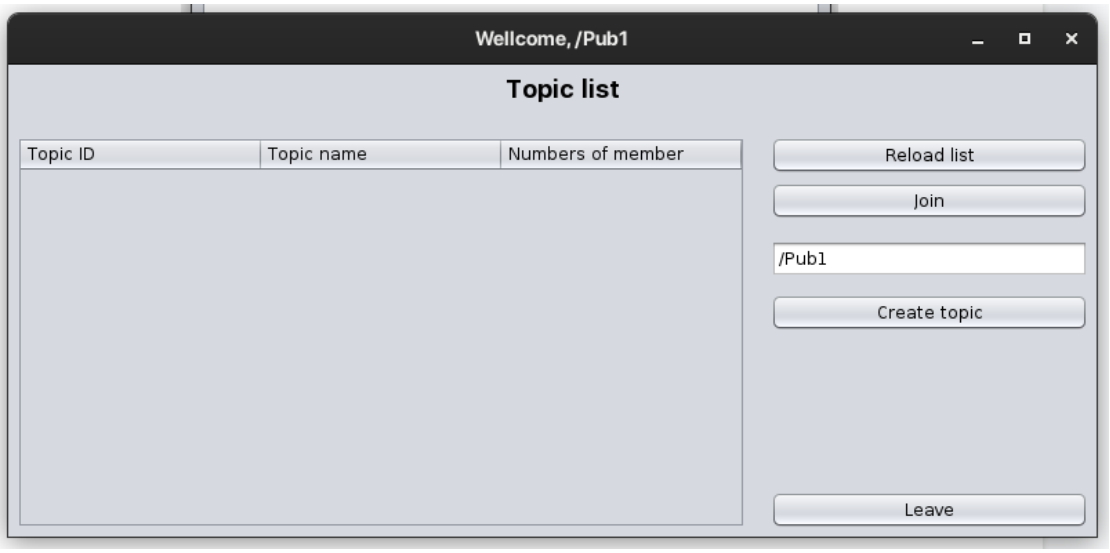
+ Broker bật lên, khởi động server bằng button. khi khởi động xong bắt đầu broker nhận các yêu cầu đến của client với form:



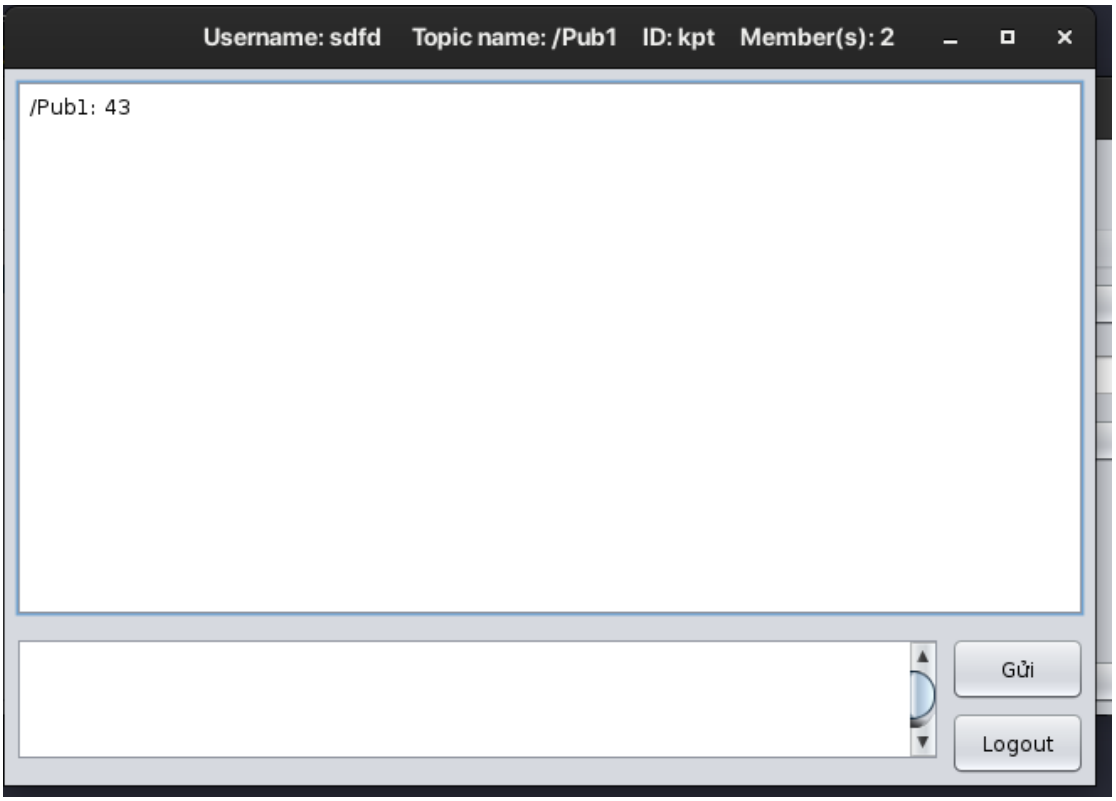
+ Khởi động các pulisher để publish với tên bắt buộc với "/" ở đầu tiên.



+ Publisher hiển thị các topic sẵn có và có thể tạo được topic mới để gửi dữ liệu. Khi topic yêu cầu thì broker sẽ tạo topic với topc id sinh tự động, topic name là tên của publisher.



+ Khởi động subcriber lên, nhập tên không được bắt đầu bằng "/". Subcriber chỉ được join vào 1 hoặc nhiều topic sẵn có mà không được tạo ra topic mới. Sau khi join vào phòng Subcriber bắt đầu thấy được dữ liệu hiển thị trên hệ thống.



+ Subcriber có thể chọn nhiều topic để join và nhận về nhiều dữ liệu cùng lúc.
+ Kết thúc phiên bằng cách tất cả mọi user đầu rời topic và topic sẽ tự động xóa.

V. Phân tích và giải thích các biến, hàm trong code:

1. Broker:
    a. Broker:
        + btn… : Sự kiện xảy ra khi ấn nút trên giao diện người dùng.
        + formWindowClosing: sự kiện tắt cửa sổ.

+ Main() : khởi động cửa sổ.
+ update() : khi có sự kiện xảy ra update chạy và đưa vào text area

b. ServerManager:
+ Khai báo các biến trong broker: cổng 6789,biến Socket, danh sách người dùng, luồng, danh sách logout,...
+ Ở đây sử dụng kế thừa Observable để các hàm khác nhau có thể theo dõi nhau và update thông tin.
+ Dispose(): Dùng server
+ StartServer(): bật server
+ StartThreadAccept(): bắt đầu chấp nhận các yêu cầu từ server
+ StartThreadProcess(): khi client vào một luồng mới, bắt đầu xử lý các request ở đây.
+ notifyObservers() : thông baó sự kiện cho các hàm quan sát khác
+ CheckRequest(): kiểm tra request đến.
+ CheckTimeConnect(): kiểm tra thời gian kết nối: Bình thường kết nối với login sẽ đến cùng lúc. Tuy nhiên có trường hợp login không đến được kết nối thì sau 5s kết nối sẽ ngắt.
+ RemoveUserLoggedOut() : Xóa người dùng
+ ProcessRequest(User user, String request): xử lý các request đến.
+ ActionType. …: mỗi một ActionTpye ứng với một sự kiện khác nhau
+ CheckUserName: kiểm tra username nên tồn tại
+ GeneralTopic(String topicName): tạo Topic
+ GeneralTopicID(): tạo topic ID
+ CheckTopicID: kiểm tra topic ID tồn tại
+ RandomString(int length): tự động tạo chuỗi
+ int CountUser() : đếm người dùng.
+ int CountTopic(): đếm topic

c. Topic: tạo ra các biến topic:
d. User: định nghĩa user.

2. Publiser/Subcriber:
a. Core:
- ClientManagerment: Quảnlí user
+ Khởi tạo kết nối vs broker
+ Dispose(): ngắt kết nối
+ StartConnect(): Bắt đầu kết nối, gửi nhân dữ liệu
+ StartThreadWaitResult() : luồng chờ kết quả của Broker
+ notifyObservers: thông báo cho các hàm có update
+ SendMess() gửi request đi
+ Login() tạo kết nối với tên

+ Pubsub(): Gửi yêu cầu để broker phân biệt publisher và subcriber
+ GetTopicList() Lấy danh sách topic
+ CreateTopic(String topicName): yêu cầu tạo topic
+ JoinTopic(String topicID) : Tham gia topic
+ LeaveTopic(): rời topic
+ Logout(): ngắt kết nối vs server

- Result: form cho kết quả trả về

b. Interface:
- Login: form login
- TopicConsevation: form gửi nhận kết quả.
- TopicList: form nhận danh sách topic và các button chức năng

VI. Form nhập:

Cổng: 6789

Server cố định trong code: localhost

Form trao đổi thông tin: `actionType + ";" + resultCode + ";" + content`

Code:

```
CHECK_ONLINE = "100";
LOGIN = "101";
LEAVE_TOPIC = "300";
JOIN_TOPIC = "302";
SEND_MESSAGE = "303";
GET_LIST_TOPIC = "304";
CREATE_TOPIC = "305";
UPDATE_NUMBER_USER = "202";
NOTIFY_JUST_JOIN_TOPIC = "203";
NOTIFY_JUST_LEAVE_TOPIC = "204";
LOGOUT = "102";
PUBSUB = "205";
OK = "400";
ERROR = "401";
```

V. Cách triển khai hệ thống:

Cho 3 folder MQTTBroker MQTTPub MQTTSub thành 3 project khác nhau(Nhóm sử dụng intellij)

Chạy MQTTBroker với file khởi động là Broker.java

Chạy MQTTPub và MQTTSub với file khởi động là Login.java