Hướng Dẫn Chi Tiết Về Các Hàm Quan Trọng và Kết Nối Client Với RabbitMQ

Thành viên Nhóm:

20127272 – Trần Thịnh phát

20127055 - Lê Minh Nhân

1653118 - Phạm Quốc Bảo

1. Khái Quát

Ứng dụng chat này sử dụng RabbitMQ để truyền tin nhắn và hình ảnh giữa các client. RabbitMQ là một hệ thống message broker giúp các ứng dụng gửi và nhận tin nhắn thông qua các hàng đợi (queues). Trong ứng dụng này, chúng ta sử dụng RabbitMQ để gửi và nhận tin nhắn văn bản cũng như hình ảnh.

2. Các Hàm Quan Trọng

2.1. update_chat_window(message, image=None)

- **Mô Tả:** Cập nhật giao diện cửa sổ chat để hiển thị tin nhắn và hình ảnh.
- **Tham Số:**
  - message (str): Tin nhắn văn bản để hiển thị.
  - image (PIL.ImageTk.PhotoImage, tùy chọn): Hình ảnh để hiển thị.
- **Chức Năng:**
  - Kích hoạt chế độ chỉnh sửa của widget chat_window.
  - Chèn tin nhắn vào cuối widget chat_window.
  - Nếu có hình ảnh, chèn hình ảnh vào cuối widget chat_window sau tin nhắn.
  - Đặt chế độ của widget chat_window trở lại không chỉnh sửa.
  - Cuộn đến cuối của widget chat_window để đảm bảo tin nhắn mới nhất luôn hiển thị.

2.2. save_to_history(user, message, image_path=None)

- **Mô Tả:** Lưu tin nhắn hoặc hình ảnh vào file lịch sử.

- **Tham Số:**
    - user (str): Tên người gửi tin nhắn hoặc hình ảnh.
    - message (str): Nội dung tin nhắn.
    - image_path (str, tùy chọn): Đường dẫn đến hình ảnh nếu có.
- **Chức Năng:**
    - Mở file "History.txt" với chế độ append.
    - Ghi nội dung tin nhắn hoặc đường dẫn hình ảnh vào file.
    - Đóng file sau khi ghi xong.

2.3. send_message()

- **Mô Tả:** Xử lý việc gửi tin nhắn văn bản qua RabbitMQ.
- **Chức Năng:**
    - Lấy nội dung tin nhắn từ widget message_entry.
    - Tạo chuỗi tin nhắn với định dạng đặc biệt gồm nội dung tin nhắn, số thứ tự và tên người gửi.
    - Gửi tin nhắn qua RabbitMQ đến exchange chat.
    - Cập nhật giao diện để hiển thị tin nhắn vừa gửi.
    - Lưu tin nhắn vào lịch sử.

2.4. send_image()

- **Mô Tả:** Xử lý việc gửi hình ảnh qua RabbitMQ.
- **Chức Năng:**
    - Mở hộp thoại để người dùng chọn hình ảnh.
    - Đọc và mã hóa hình ảnh thành định dạng base64.
    - Tạo chuỗi tin nhắn với định dạng đặc biệt gồm thông tin hình ảnh, số thứ tự, tên người gửi, tên file và dữ liệu hình ảnh mã hóa.
    - Gửi tin nhắn qua RabbitMQ đến exchange chat.
    - Hiển thị hình ảnh vừa gửi trên GUI.
    - Lưu hình ảnh vào thư mục image và vào lịch sử.

2.5. repost_callback(ch, method, properties, body)

- **Mô Tả:** Xử lý tin nhắn nhận được từ RabbitMQ.
- **Tham Số:**
    - ch: Kênh RabbitMQ.
    - method: Thông tin về tin nhắn.
    - properties: Các thuộc tính của tin nhắn.
    - body: Nội dung tin nhắn.
- **Chức Năng:**
    - Giải mã nội dung tin nhắn từ định dạng base64 nếu là hình ảnh.
    - Nếu tin nhắn là hình ảnh:
        - Lưu hình ảnh vào thư mục image.
        - Hiển thị hình ảnh trên GUI.
    - Nếu tin nhắn là văn bản:
        - Hiển thị nội dung tin nhắn trên GUI.
    - Lưu tin nhắn vào lịch sử.

2.6. display_messenger_receive(user, message, image=None)

- **Mô Tả:** Hiển thị tin nhắn hoặc hình ảnh nhận được trên GUI.
- **Tham Số:**
    - user (str): Tên người gửi tin nhắn hoặc hình ảnh.
    - message (str): Nội dung tin nhắn.
    - image (PIL.ImageTk.PhotoImage, tùy chọn): Hình ảnh cần hiển thị.
- **Chức Năng:**
    - Định dạng tin nhắn với tên người gửi và nội dung tin nhắn.
    - Gọi hàm update_chat_window để hiển thị tin nhắn và hình ảnh nếu có.

2.7. start_consuming()

- **Mô Tả:** Bắt đầu tiêu thụ tin nhắn từ RabbitMQ.

- **Chức Năng:**
    - Gọi start_consuming trên kênh RabbitMQ để bắt đầu nhận tin nhắn liên tục.
    - Tin nhắn nhận được sẽ được xử lý bởi hàm repost_callback.

2.8. start_session() và end_session()

- **Mô Tả:** Quản lý việc ghi lịch sử phiên làm việc vào file "History.txt".
- **Chức Năng:**
    - start_session(): Ghi dấu mở phiên { vào file khi bắt đầu phiên làm việc.
    - end_session(): Ghi dấu đóng phiên } vào file khi kết thúc phiên làm việc.

3. Kết Nối Client Với RabbitMQ

3.1. Tạo Kết Nối

- **Hàm:** pika.BlockingConnection(pika.ConnectionParameters(host='localhost'))
- **Mô Tả:** Tạo kết nối đến RabbitMQ chạy trên localhost. Đây là nơi mà các client kết nối để gửi và nhận tin nhắn.

3.2. Tạo Kênh

- **Hàm:** connection.channel()
- **Mô Tả:** Tạo một kênh để gửi và nhận tin nhắn qua RabbitMQ. Một kết nối có thể chứa nhiều kênh.

3.3. Đăng Ký Exchange và Queue

- **Hàm:**
    - channel.exchange_declare(exchange='chat', exchange_type='fanout'): Đăng ký một exchange kiểu fanout có tên là chat. Exchange kiểu fanout phát đi tin nhắn đến tất cả các queue đã liên kết.
    - channel.queue_declare('', exclusive=True): Tạo một queue tạm thời với tên tự động sinh ra để nhận tin nhắn.
    - channel.queue_bind(exchange='chat', queue=queue_name): Ràng buộc queue với exchange chat để nhận các tin nhắn được gửi đến exchange này.

3.4. Gửi Tin Nhắn

- **Hàm:** channel.basic_publish(exchange='chat', routing_key='', body=message)

- **Mô Tả:** Gửi tin nhắn đến exchange chat. routing_key không được sử dụng với exchange kiểu fanout, vì tất cả các tin nhắn được gửi đến tất cả các queue liên kết với exchange này.

3.5. Nhận Tin Nhắn

- **Hàm:** channel.basic_consume(queue=queue_name, on_message_callback=repost_callback, auto_ack=True)
- **Mô Tả:** Đăng ký hàm repost_callback để xử lý tin nhắn nhận được từ queue. auto_ack=True có nghĩa là tin nhắn sẽ được tự động xác nhận khi được nhận.

Hy vọng tài liệu chi tiết này sẽ giúp bạn hiểu rõ hơn về cách các hàm hoạt động và cách kết nối giữa client và RabbitMQ trong ứng dụng chat. Nếu có bất kỳ câu hỏi nào khác, đừng ngần ngại hỏi nhé!