Lab 03: Middleware Orientado a Mensagens

Aluno: Arthur Gums Willrich
Matrícula: 18204648

Passo a passo:

Passo 1: Atualização e Instalação de Pacotes:

sudo apt-get update
sudo apt-get install erlang rabbitmq-server

Passo 2: Execução do serviço

sudo systemctl enable rabbitmq-server
sudo systemctl start rabbitmq-server

Passo 3: Plugin de gerenciamento via navegador

sudo rabbitmq-plugins enable rabbitmq_management
sudo systemctl status rabbitmq-server

Passo 4: Console de gerenciamento

Acessar a url: http://localhost:15672/

Usuário: guest
Password: guest

Passo 5: Execução do código Python 3.8

python produtor.py
python consumidor.py

O produtor.py estabelece uma conexão com o RabbitMQ, cria a fila "minha_fila" (se ainda não existir) e define uma função enviar_mensagem que publica mensagens na fila. Ele também possui um loop onde solicita ao usuário mensagens para enviar e chama a função enviar_mensagem para publicar cada mensagem na fila.

O consumidor.py também estabelece uma conexão com o RabbitMQ e cria a fila "minha_fila" (se necessário). Ele define uma função de callback, que é executada sempre que uma mensagem é recebida da fila. A função callback simplesmente imprime o conteúdo da mensagem. Em seguida, o código inicia o consumo de mensagens da fila "minha_fila" e permanece aguardando novas mensagens, printando quando recebidas.

Essa aplicação poderia funcionar em um sistema de um aplicativo de entrega de comida, sempre que um novo pedido for realizado, é enviada uma notificação para um serviço de entrega que irá processar e entregar o pedido.

Nesse caso, o código implementa a comunicação assíncrona entre o aplicativo de pedidos e o serviço de entrega. O aplicativo de pedidos atuaria como o produtor, enviando informações sobre cada novo pedido para a fila, e o serviço de entrega atuaria como o consumidor, consumindo as mensagens da fila e processando cada pedido.

O código está anexado no ZIP da entrega em que se encontra este PDF