

MENSAGENS RABBITMQ

Listener da queue: *dispositivos-state-queue*

Essa listener está presente no microserviço “main-api” e captura as mensagens enviadas do microserviço “disp-monitor” sobre o estado atual do monitoramento do dispositivo. Isto é, envia informações como: id, status e latência média. Essas informações são mostradas em tempo real na página de dispositivos.

Fluxo principal:

- 1) Busca o dispositivo pelo ID e carrega nele os dados recebidos na mensagem
- 2) Salva os dados do dispositivo no banco de dados
- 3) Envia os dados como mensagem via websocket para todos os usuários conectados e que estão vinculados a mesma empresa do dispositivo ou que são usuários admin.
- 4) Envia notificação de dispositivos inativos via telegram e e-mail se necessário
 - a) Verifica se a data/hora atual é posterior a data/hora da última notificação (registrada na empresa) mais o tempo mínimo para próxima notificação
 - 1) Se sim, envia uma mensagem, tanto por telegram, quanto por e-mail, registrados para a empresa e registra no banco de dados para a empresa a data/hora atuais como data/hora de última notificação. A mensagem a ser enviada contém a data/hora atuais e todos os dispositivos inativos.
 - A) Se não houver dispositivos inativos, a mensagem não é enviada
 - B) Se o chat id não tiver sido informado para a empresa, a mensagem de telegram não é enviada
 - C) Se o e-mail de notificação para a empresa não tiver sido informado, o e-mail não é enviado
 - D) Se tentar e não conseguir enviar a mensagem via telegram, um e-mail é enviado para o e-mail de notificação, caso tenha sido informado, informando que não foi possível enviar a mensagem via telegram
 - E) Se um RestClientException for lançado com a tentativa de enviar a mensagem via telegram, um log é registrado informando o ocorrido.

Listener da queue: *eventos-queue*

Esse listener está presente no microserviço “main-api” e captura as mensagens enviadas do microserviço “disp-monitor” com dados do evento conforme a configuração de frequência de envio de eventos para registro no banco de dados. Esses eventos incluem informações do monitoramento como: quantidade de sucessos no período, quantidade de falhas no período, quantidade de quedas, entre outras informações.

Fluxo principal:

1. Busca o dispositivo pelo ID recebido
2. Carrega o objeto evento com os dados da mensagem e o dispositivo buscado
3. Registra o evento no banco de dados

Listener da queue: *configs-queue*

Esse listener está presente no microserviço “disp-monitor” e recebe mensagens do microserviço “main-api”. A mensagem recebida tem dados da configuração que foi alterada no banco de dados.

Fluxo principal:

- 1) O novo config é setado em todas as threads de monitoramento. Há uma sincronização!
 - a) Se o delay de monitoramento tiver sido alterado, então a execução da task de cada thread é cancelada e são iniciadas novas threads com o delay entre cada execução passando a ser o novo delay de monitoramento. Isto é, as thread são executadas várias vezes, isso até o monitoramento ser parado e, entre uma execução e outra, há um delay de monitoramento que geralmente é de 1ms. O mais curto possível (quase zero).

Listener da queue: *dispositivos-queue*

Esse listener está presente no microserviço “disp-monitor” e recebe mensagens do microserviço “main-api”. A mensagem recebida tem dados do dispositivo que foi alterado no banco de dados.

Fluxo principal:

1. O sistema busca na lista (em memória) o objeto com a thread de monitoramento pelo ID do dispositivo
2. Se o encontrado, então o objeto dispositivo é setado na thread de monitoramento correspondente, caso contrário, nada acontece