Monitor Whatsap

Visão Geral Arquitetural

Este aplicativo é um **Gateway de Mensagens do WhatsApp**, projetado para operar como um serviço de backend robusto e autônomo. Ele transforma um banco de dados Firebird em uma poderosa fila de envio, permitindo que outros sistemas deleguem completamente a tarefa de enviar mensagens de forma segura e rastreável.

Principais Características:

- **Fila Assíncrona:** O sistema lê tarefas da tabela WHATS_ENVIADO e as processa em segundo plano.
- **Rastreamento Granular:** Cada mensagem individual (texto e cada anexo) é registrada na tabela WHATS_MENSAGENS, e seu status de entrega (ENVIADO, ENTREGUE, VISUALIZADO) é atualizado em tempo real.
- **Idempotência:** O sistema é projetado para ser "idempotente", ou seja, se uma tarefa for interrompida e reiniciada, ele não reenviará mensagens que já foram despachadas com sucesso.
- **Técnicas Anti-Bloqueio:** Utiliza pausas aleatórias e a criação de conteúdo único (textos e arquivos com hashes diferentes) para simular um comportamento mais humano e reduzir o risco de bloqueio pelo WhatsApp.
- **Configuração Externa:** Todas as configurações sensíveis (banco de dados, pausas) são gerenciadas em um arquivo .env, tornando a aplicação portátil e segura.

O Ciclo de Vida de um Envio

A confiabilidade do sistema é garantida por um fluxo de trabalho em múltiplas etapas, onde cada passo importante é registrado no banco de dados.

1. A Solicitação (INSERT em WHATS_ENVIADO)

- Um sistema externo cria uma "tarefa de envio" ao inserir um registro na tabela WHATS_ENVIADO com SITUACAO_TAREFA = 'AGUARDANDO'.

2. O Início do Processamento (UPDATE para 'PROCESSANDO')

- A função processarFilaDoBanco busca a tarefa. A primeira coisa que ela faz é atualizar a SITUACAO_TAREFA para 'PROCESSANDO'.
- Isso funciona como um "lock". Garante que, mesmo que o ciclo de verificação rode novamente, ele não pegará a mesma tarefa duas vezes, evitando envios duplicados.

3. O Despacho da Mensagem (sendMessageAndCapture)

- Para cada texto ou arquivo, a função sendMessageAndCapture é chamada. Esta função é o coração da "certeza do despacho".
- **Como funciona:**
    1. Ela tenta enviar a mensagem via client.sendMessage().
    2. Simultaneamente, ela "escuta" o evento message_create, que é a confirmação da biblioteca de que uma mensagem foi criada com sucesso.
    3. ***Ela só retorna com sucesso quando captura este evento e extrai o ID único da mensagem (msg.id._serialized).***
- Não confiamos apenas que o comando de envio foi "disparado". Esperamos ativamente pela confirmação da própria biblioteca de que a mensagem foi gerada e tem um ID rastreável. Se essa confirmação não chegar em 15 segundos, a operação falha com um "Timeout", impedindo um estado incerto.

4. A Prova do Envio (INSERT em WHATS_MENSAGENS)

- Imediatamente após sendMessageAndCapture retornar com sucesso, a função registrarMensagemEnviada é chamada.
- Ela insere um novo registro na tabela WHATS_MENSAGENS, armazenando o ID_MSG_WHATSAPP que acabamos de obter.
- ***Este INSERT é a nossa prova irrefutável de que a aplicação cumpriu sua parte: a mensagem foi enviada e agora temos um ID para rastreá-la.***

5. A Confirmação do Destinatário (UPDATE do ACK)

- O handler setupAckHandler ouve os eventos message_ack do WhatsApp.
- Quando o status de uma mensagem muda (para entregue ou visto), a função atualizarStatusAck faz um UPDATE na tabela WHATS_MENSAGENS, encontrando a mensagem pelo ID_MSG_WHATSAPP e atualizando sua coluna STATUS_ACK.
- ***Esta é a confirmação do "outro lado". É a prova de que a mensagem não apenas saiu do nosso sistema, mas chegou e <u>foi interagida</u> no dispositivo do destinatário.***

6. A Finalização da Tarefa (UPDATE para 'CONCLUIDO' ou 'ERRO')

- Após tentar enviar todos os itens da tarefa, o código analisa se houve erros.
- Ele faz um último UPDATE na tabela WHATS_ENVIADO, marcando a tarefa como CONCLUIDO, ERRO_PARCIAL ou ERRO, completando o ciclo de vida.

Análise Detalhada das Funções do Banco

A interação com o banco é o pilar do sistema.

async function atualizarStatusTarefa(id, status, observacao)

- **Propósito:** Gerenciar o estado macro da tarefa de envio.

- **Como funciona:** Executa um UPDATE na tabela WHATS_ENVIADO. É usado para marcar uma tarefa como 'PROCESSANDO' (para evitar reprocessamento) e, no final, como 'CONCLUIDO' ou 'ERRO', fornecendo uma visão rápida e de alto nível do resultado.

async function registrarMensagemEnviada(idEnvio, msgId, tipo, conteudo)

- **Propósito:** Criar um registro histórico e rastreável para cada mensagem individual enviada.
- **Como funciona:** Executa um INSERT na tabela WHATS_MENSAGENS. Esta é a função mais crítica para a auditoria. Ela só é chamada após a confirmação de que uma mensagem foi despachada com sucesso e um ID foi obtido. Ela armazena este ID (msgId) que será a chave para futuras atualizações de status.

async function atualizarStatusAck(msgSerializedId, status)

- **Propósito:** Rastrear o status de entrega da mensagem no lado do destinatário.
- **Como funciona:** Executa um UPDATE na tabela WHATS_MENSAGENS, usando o ID_MSG_WHATSAPP (que foi salvo por registrarMensagemEnviada) para encontrar o registro correto. Atualiza a coluna STATUS_ACK com informações fornecidas diretamente pelos servidores do WhatsApp (ENTREGUE, VISUALIZADO).

Recursos Avançados Implementados

1. **Idempotência (processarFilaDoBanco):** Antes de enviar um texto ou arquivo, a função faz um SELECT em WHATS_MENSAGENS para ver se aquele item específico já foi registrado. Se sim, ele o pula. Isso torna o sistema seguro contra reinicializações e falhas, prevenindo envios duplicados e erros de "chave duplicada" no banco.
2. **Técnicas Anti-Bloqueio (generateInvisibleSuffix, createUniqueFileCopy):**
    - **Texto Único:** Adiciona caracteres Unicode invisíveis ao final de cada mensagem de texto. Para um ser humano, o texto parece igual; para um computador, a string é diferente, evitando filtros de "conteúdo duplicado".
    - **Arquivo Único:** Cria uma cópia temporária de cada anexo e adiciona um "carimbo" (UUID) no final do arquivo. Isso altera o *hash* (impressão digital) do arquivo, fazendo com que cada envio de anexo seja, para o WhatsApp, um arquivo completamente novo.
3. **Resiliência e Recuperação de Erros:**
    - O try...catch...finally em enviarArquivos garante que os arquivos temporários sejam sempre excluídos, mesmo em caso de erro.
    - O setTimeout na função sendMessageAndCapture previne que o programa fique "preso" indefinidamente se um evento não for recebido.
    - O handler do evento disconnected tenta reconectar automaticamente, mantendo o serviço online.