📄 Documentação Técnica do Bot de Automação (N8N Self-Host)

1. Introdução e Escopo do Projeto

1.1. Propósito e Objetivos do Bot

Este documento tem como objetivo detalhar a arquitetura, a implementação técnica e os fluxos de trabalho (workflows) do **AutoDocs**, um assistente de automação corporativa implementado no Telegram.

1.2. Visão Geral do Bot e Funcionalidades

O **AutoDocs** atua como um Ponto Central de Automação e Formalização de Documentos, oferecendo as seguintes funcionalidades chave:

1. **Formalização de Documentos Corporativos:** Geração de comunicações em Papel Timbrado, com suporte para múltiplos modelos de template (`/newdoc modeloX`).
2. **Criação de Relatórios Visuais:** Elaboração de Relatórios Financeiros Simplificados em PDF, que injetam dados e **imagens de gráficos** (`/newrelatorio`).
3. **Busca de Notícias:** Coleta e entrega de manchetes e links recentes da capa do G1 (`/news`).
4. **Transcrição de Áudio:** Conversão automática de mensagens de voz e arquivos de áudio em texto (basta enviar o áudio).
5. **Agente de Atendimento:** Um agente de IA que utiliza memória estática para responder a dúvidas e direcionar o usuário para o comando correto.

O bot está hospedado em uma arquitetura **Self-Hosted** (GCP) para garantir estabilidade e total controle sobre o ambiente de execução.

1.3. Tecnologias principais

O projeto utiliza o **N8N** orquestrador central, ele é responsável por estruturar e executar todos os *workflows*. A interface gráfica é o **Telegram**, que recebe os comandos, envia as mídias e faz a entrega final dos documentos.

Para a gestão e manipulação de documentos, o projeto utiliza o **Google Drive**, responsável pelo armazenamento dos *templates*, arquivos gerados e imagens, e o **Google Docs** é utilizado para a edição de documentos e a conversão final para o formato PDF. O **Gemini** (Google AI) é integrado para fornecer as funcionalidades de Inteligência Artificial, sendo o motor de linguagem natural para respostas do Agente e para a transcrição de áudios (Speech-to-Text).

Do lado da Hospedagem, a automação utiliza o serviço de Máquina Virtual (VM) do **Google Cloud Platform** (GCP) em um ambiente Self-Hosted. Dentro da própria VM, o **Docker** roda os contêineres do N8N e do Cloudflare. O **Cloudflare Tunnel** é responsável por estabelecer um túnel (HTTPS) para um domínio próprio, garantindo a comunicação do Webhook com o Telegram.

2. Detalhamento dos Workflows (Casos de Uso)

2.1. Workflow Principal: Bot de Atendimento e Decisão

2.1.1. Trigger (Gatilho): Telegram Webhook

O *workflow* é ativado pelo nó **Telegram Trigger** (Webhook) sempre que uma nova mensagem (texto, comando, ou áudio) é recebida no canal do bot.

2.1.2. Lógica de Roteamento de Comandos (Switch/IF)

A mensagem recebida passa por uma sequência de nós **IF** para determinar o fluxo correto, seguindo a ordem de prioridade:

1. **Prioridade 1: Comando Reconhecido?**
    - Um nó **IF** verifica se o texto da mensagem contém um comando. Se for **TRUE**, o fluxo é direcionado para a lógica de **Switch/Case** para identificar o comando exato e iniciar o *workflow* específico. Se o comando não for reconhecido ou digitado incorretamente, o fluxo entra em um *fallback*.
2. **Prioridade 2: Transcrição de Áudio?**
    - Se a mensagem não for um comando, um segundo nó **IF** verifica a existência do *payload* `voice` (`{{ json.message.voice }}`). Se for **TRUE**, a mensagem é enviada ao fluxo de **Transcrição de Áudio** (Caso 4).
3. **Prioridade 3: Agente de Atendimento:**
    - Se a mensagem não for um comando nem um áudio, o **Agente de IA** entra em cena. O Agente de Linguagem Natural (LLM) responde à mensagem, utilizando o *System Prompt* (Memória Estática) para tirar dúvidas simples e **direcionar** o usuário aos comandos existentes, mantendo-se dentro do escopo do projeto.

2.2. Caso de Uso 1: Geração de Documentos Corporativos

2.2.1. Introdução à funcionalidade

Esta funcionalidade visa a formalização rápida de comunicações corporativas utilizando um papel timbrado padrão. Para oferecer flexibilidade, foram disponibilizados **três modelos** de *template* com variações propositais:

- **Modelo 1 (Default/Básico):** Ideal para comunicações rápidas e objetivas, com campos de informação mínimos.
- **Modelo 2 e Modelo 3:** Modelos mais **completos** que introduzem campos adicionais (ex: telefone, endereço, website) e oferecem layouts visuais distintos.

Essa variação permite ao usuário selecionar o formato mais adequado, seja para um documento mais enxuto ou para uma comunicação mais completa.

2.2.2. Fluxo de Criação (Template Copy e Replace Text)

Ao receber o comando `/newdoc` (seguido ou não pela especificação do modelo), o *workflow* executa a seguinte sequência:

1. **Identificação do Modelo:** A mensagem é pré-processada (Nó Code/Function) para extrair o modelo desejado (`modelo1`, `modelo2`, `modelo3`).

2. **Direcionamento (Switch):** Um nó **Switch** utiliza a variável do modelo para direcionar o fluxo ao ramo de cópia de *template* específico.
    - *(Justificativa da redundância: A redundância entre alguns nós foi mantida para garantir a legibilidade e fácil manutenção dos IDs de templates, evitando expressões complexas.)*
3. **Coleta de Dados:** O sistema envia um Formulário Web, garantindo que os dados (remetente, destinatário e conteúdo) sejam recebidos de forma estruturada.
4. **Processamento:**
    - O nó Google Drive cria uma cópia do *template* (`.gdoc` nativo).
    - O nó Google Docs: Replace Text injeta os dados do formulário na cópia, utilizando expressões ternárias para preencher campos opcionais (ex: `telefone`, `endereço`) com texto vazio, caso não sejam fornecidos.
5. **Entrega Final:** A cópia é configurada para compartilhamento público (link), e o nó Google Drive: Export converte o documento para PDF. O *workflow* conclui enviando uma mensagem de sucesso ao usuário com o link de acesso e o arquivo PDF anexado.

2.3. Caso de Uso 2: Relatório Visual e Exportação

2.3.1. Introdução à funcionalidade

Esta funcionalidade implementa a criação de um Relatório Financeiro Simplificado, alinhando o bot a um escopo de utilidade corporativa. A principal diferença dessa funcionalidade, é permitir que o usuário insira tanto dados estruturados (texto e valores) quanto uma evidência visual (o gráfico de receita/despesa).

O *workflow* exige o *upload* de uma imagem de gráfico ou prova visual, que é integrada ao layout do relatório.

2.3.2. Inserção de Gráficos e Imagens (Google Docs API)

O fluxo de injeção de conteúdo é dividido em três fases críticas:

1. **Coleta e Armazenamento:** O *workflow* se inicia com a coleta dos dados de texto via form (inclusive a imagem). A imagem é, então, enviada e salva em uma pasta do Google Drive e recebe permissão de leitura pública . Este passo é obrigatório, pois a API do Google Docs exige uma URI pública para buscar o arquivo.
2. **Injeção de Texto:** O nó Google Docs injeta todos os dados textuais (Resumo Executivo, Receita, Despesa) nos marcadores definidos do *template*.
3. **Injeção da Imagem:** O sistema localiza o índice do marcador `[IMAGEM]` no documento e utiliza uma requisição **HTTP Request** direta à API do Docs. A requisição envia o comando `insertInlineImage`, utilizando o link público do Drive (URI) como fonte. O texto do marcador é removido e substituído pela imagem binária salva no Drive, garantindo o posicionamento visual.

2.3.3. Conversão e Entrega (PDF via Telegram)

O nó Telegram envia uma mensagem de sucesso ao usuário, anexando o arquivo PDF e fornecendo o link público do documento.

2.4. Caso de Uso 3: Notícias.

2.4.1. Introdução à funcionalidade

A funcionalidade é ativada via comando (`/news`) e tem como objetivo recuperar as três manchetes mais recentes da capa do portal G1, entregando o título e o link correspondente diretamente no chat do Telegram.

2.4.2. Fluxo de Execução

O *workflow* executa um processo de *web scraping* estruturado:

1. **Requisição HTTP:** Um nó **HTTP Request (GET)** é utilizado para acessar o URL principal do portal G1 e obter o código-fonte HTML completo da página.
2. **Extração de Dados:** O conteúdo HTML é processado por um nó de manipulação de dados **HTML Extract** que utiliza seletores CSS para filtrar e isolar os elementos das notícias (manchetes e URLs) da capa do site.
3. **Filtragem e Formatação:** Os dados brutos são separados em uma lista estruturada de notícias. O *workflow* então limita o conjunto de resultados às **três primeiras notícias** e formata a saída como uma mensagem clara com o título e o *link*.
4. **Entrega:** O nó **Telegram** envia o resumo das notícias formatado ao usuário.

2.5. Caso de Uso 4: Transcrição de áudio.

2.5.1. Introdução à funcionalidade

Esta funcionalidade permite que o usuário converta qualquer conteúdo de voz (mensagens de voz ou arquivos de áudio) em texto. Basta que o usuário envie o arquivo de áudio para o bot.

2.5.2. Fluxo de Execução

A execução do *workflow* é dependente da natureza do arquivo multimídia:

1. **Gatilho Condicional:** O *workflow* é ativado apenas se o nó **IF** identificar a presença do *payload* `voice` (`{{ json.message.voice }}`) na mensagem do Telegram.
2. **Download e Binário:** O arquivo de áudio é baixado por um nó HTTP Request. O conteúdo é salvo como dado binário na execução do N8N.
3. **Transcrição de IA:** O dado binário do áudio é enviado ao nó **Google Gemini** (transcribe áudio). O modelo de linguagem de IA processa o áudio e retorna a transcrição textual.
4. **Entrega:** O nó **Telegram** envia o texto transcrito de volta para o *chat* do usuário.

2.6. Caso de Uso 5: Agente IA.

2.6.1. Introdução à funcionalidade

Esta funcionalidade opera como o mecanismo de *fallback* do *workflow* principal. O Agente de IA é ativado sempre que a mensagem do usuário não corresponde a um comando formal ou a um áudio.
Seu propósito é:

1. **Direcionamento:** Guia o usuário a utilizar as funcionalidades programadas do bot, reforçando os comandos existentes.

2. **Atendimento:** Responde a dúvidas simples sobre a operação do bot e suas capacidades.

2.6.2. Fluxo de Execução

Estratégia de Prompt (Memória Estática): A LLM do Gemini recebe um System Prompt detalhado, que atua como sua Memória Estática. Este *prompt* define a personalidade do agente e específica, de forma explícita, todas as funcionalidades e comandos do bot.

Restrição: O *System Prompt* instrui o Agente a nunca executar tarefas diretamente e sempre redirecionar o usuário para o comando apropriado, garantindo o controle e a eficiência do *workflow* no N8N.

Resposta: O agente processa a dúvida em contexto e devolve a resposta ao usuário via Telegram.

4. Gestão de Erros e Logs

4.1. Workflow de Erro Dedicado (Error Handler)

A confiabilidade do sistema é garantida pela implementação de um Workflow de Erro Dedicado (Error Handler). Este workflow é ativado automaticamente pelo sistema interno do N8N sempre que uma execução no workflow principal falha.

4.1.1 Configuração e Notificação

Gatilho de Erro: O *workflow* é iniciado pelo nó Error Trigger, que capta o contexto da falha.
Notificação: A mensagem de alerta é enviada ao Telegram pessoal do desenvolvedor.
Conteúdo do Log: A notificação é formatada para incluir dados críticos para o *debugging*: o nome do *workflow* que falhou, o tipo de erro (`error.name`), a mensagem completa do erro (`error.message`) e o *timestamp* (dia e hora da ocorrência).

5. Fluxograma

5.1 Mapeamento do Processo Geral