## Documentação Técnica do Projeto

Esta documentação descreve a arquitetura, os fluxos de dados e as instruções de integração entre o front-end React e o back-end orquestrado no n8n para envio à banca avaliadora.

### 1. Visão Geral

O projeto consiste em uma aplicação web front-end em React/TypeScript que envia requisições a um fluxo de automação configurado no n8n. Este fluxo processa diferentes tipos de entrada (texto, áudio, imagem, PDF), interage com APIs da OpenAI e consulta dados em um Google Sheets da Universidade X, retornando respostas formatadas para o cliente.

### 2. Componentes

1. **Front-End** (React/TS)
   * Componentes de chat, captura de áudio e upload de imagem
   * Serviço de comunicação HTTP com o webhook do n8n
2. **Back-End** (n8n)
   * **Webhook**: Ponto de entrada único
   * **Switch/If**: Roteamento por tipo de mensagem
   * **Transcrição de Áudio**: OpenAI Audio API
   * **Análise de Imagem**: OpenAI Vision
   * **Consulta de Dados**: Google Sheets Tool
   * **AI Agent**: LangChain com memória simples e formato de resposta
   * **Parser Chain**: Formatação final em Markdown para WhatsApp

### 3. Endpoint

POST https://n8n.grupohernandes.com/webhook/5234fb23-85c7-4ef4-91d1-0f28bf6ddd26  
Headers:  
 Content-Type: application/json  
Body (exemplo):  
{  
 "message": "Olá, como posso ajudar?",  
 "user\_id": "usuario123",  
 "audio": "data:audio/webm;base64,...", // opcional  
 "image": "data:image/png;base64,...", // opcional  
 "pdf": "data:application/pdf;base64,..." // opcional  
}

* **message**: texto puro (ou string especial como [AUDIO])
* **user\_id**: identificador único do usuário
* **audio**, **image**, **pdf**: Data URIs em Base64 para arquivos multimídia

### 4. Fluxo de Processamento

1. **Recepção** do payload no node Webhook.
2. **Roteamento** (If/Switch) conforme tipo:
   * Texto → envio direto ao AI Agent
   * Áudio → decodificação → transcrição → texto → AI Agent
   * Imagem → análise → descrição textual → AI Agent
   * PDF → (em configuração futura)
3. **AI Agent** (LangChain): gera resposta usando GPT-4‑o-mini com contexto de memória (últimas 20 interações).
4. **Consulta de Dados** (opcional): chama node consultardados em Google Sheets para informações acadêmicas.
5. **Parser Chain**: formata resposta em Markdown para WhatsApp (negrito, itálico, links em backticks).
6. **Respond to Webhook**: devolve JSON com a resposta final.

### 5. Integração no Front-End

// src/services/n8n.ts  
export interface EngagePayload { message: string; user\_id: string; audio?: string; image?: string; pdf?: string; }  
export async function sendToN8n(payload: EngagePayload) {  
 const res = await fetch(  
 "https://n8n.grupohernandes.com/webhook/5234fb23-85c7-4ef4-91d1-0f28bf6ddd26",  
 { method: "POST", headers: { "Content-Type": "application/json" }, body: JSON.stringify(payload) }  
 );  
 if (!res.ok) throw new Error(`Erro ${res.status}`);  
 return res.json();  
}

No componente de chat, basta chamar sendToN8n({ message, user\_id }) e tratar a resposta para exibir na interface.

### 6. Requisitos de Ambiente

* **Front-End**:
  + Node.js v18+
  + gerenciador de pacotes (npm/yarn)
* **n8n**:
  + Hospedagem (ex: Docker, serviço em nuvem)
  + Credenciais:
    - OpenAI API key
    - Google Sheets OAuth2

### 7. Como Executar

1. **Front-End**:

* cd frontend  
  npm install  
  npm run dev

1. **n8n** (se local):

* docker run -it --rm -p 5678:5678 \  
   -e N8N\_BASIC\_AUTH\_ACTIVE=true \  
   -e N8N\_BASIC\_AUTH\_USER=<user> \  
   -e N8N\_BASIC\_AUTH\_PASSWORD=<password> \  
   -v ~/.n8n:/home/node/.n8n n8nio/n8n

### 8. Considerações Finais

* O webhook pode ser protegido com tokens ou autenticação básica.
* O fluxo de PDF ainda precisa de implementação para leitura e análise.
* Ajustar configurações de modelo e memória conforme volume de uso.

*Essa documentação foi gerada para apresentação à banca avaliadora do hackathon.*