Documentação do Projeto: BioAssist - Interface de Chatbot

Data: 16 de Maio de 2025 - **Versão:** 1.0
Autor: Maria Clara Cavalcante

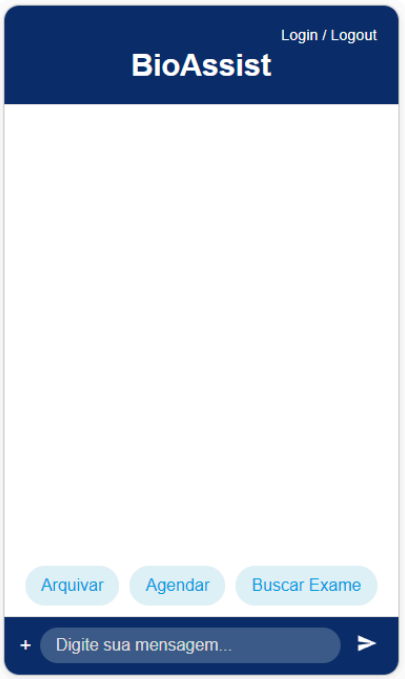
1. Introdução

Este documento detalha o desenvolvimento do chatbot BioAssist, uma solução integrada ao Google Workspace para simplificar a organização de informações de saúde dos usuários, oferecendo as seguintes funcionalidades:

- **Arquivar Exames:** Salva dados relevantes de exames médicos no Google Sheets.
- **Agendar Compromissos:** Cria eventos de saúde no Google Agenda.
- **Buscar Exames:** Recupera dados de exames arquivados no Google Sheets.
- **Buscar Compromissos:** Lista eventos de saúde agendados no Google Agenda.

A interação inicial do usuário ocorre através de uma interface web construída com HTML e CSS.

Layout da tela:



2. Arquitetura da Solução

A arquitetura compreende:

- **Frontend (Interface Web):** Página HTML (`index.html`) estilizada com CSS (`styles.css` ou `BioAssist.css`), fornecendo campos de entrada e botões para interação do usuário. A comunicação com o backend é realizada via JavaScript (AJAX).
- **Backend (Lógica do Chatbot):** Aplicação responsável por receber e processar as mensagens do frontend, interpretar as intenções do usuário e orquestrar a interação com as APIs do Google. Implementado em Python utilizando um framework web (Flask ou Django).
- **APIs do Google:**
    - **Google Sheets API:** Para leitura e escrita de dados em planilhas do Google Sheets.
    - **Google Calendar API:** Para criação e consulta de eventos no Google Calendar.
- **Autenticação e Autorização:** Implementação do protocolo OAuth 2.0 no backend para garantir acesso seguro e autorizado aos dados do Google Workspace do usuário.

3. Detalhamento das Funcionalidades e Implementação

3.1. Arquivar Exames Médicos

- **Fluxo:** O usuário insere os detalhes do exame na interface web e os envia. O JavaScript captura a mensagem e a envia ao backend via AJAX. O backend processa a mensagem, extrai as informações necessárias e utiliza a Google Sheets API para adicionar uma nova linha com os dados do exame na planilha configurada. O status da operação é retornado ao frontend para feedback ao usuário.
- **Implementação Backend (Google Sheets API):** Necessária a configuração da autenticação OAuth 2.0, a especificação da planilha de destino e a utilização da função de "append" da API para inserir os dados. (Consultar exemplo de código Python anterior para detalhes).
- **Interface Web:** O botão "Arquivar" pode direcionar o foco para o campo de entrada com uma instrução clara. A área `.white-space` pode exibir um histórico de conversas para melhor interação.

3.2. Agendar Compromissos de Saúde

- **Fluxo:** O usuário informa os detalhes do agendamento (data, hora, descrição, local) na interface web, que são enviados ao backend. O backend processa a mensagem e utiliza a Google Calendar API para criar um novo evento com essas informações no calendário do usuário. O frontend recebe e exibe a confirmação ou erro.
- **Implementação Backend (Google Calendar API):** Requer autenticação OAuth 2.0 e a construção de um objeto JSON no formato esperado pela API para a criação de eventos. (Consultar exemplo de código Python anterior).
- **Interface Web:** O botão "Agendar" pode focar o input e sugerir o formato da entrada.

3.3. Buscar por Exame Arquivado

- **Fluxo:** O usuário digita um termo de busca na interface, enviado ao backend. O backend utiliza a Google Sheets API para ler o conteúdo da planilha de exames e implementa a lógica de filtragem para encontrar as entradas correspondentes ao termo de busca. Os resultados são enviados e exibidos na interface web.
- **Implementação Backend (Google Sheets API):** Envolve a leitura dos dados da planilha e a implementação de algoritmos de busca e filtragem no backend. (Consultar exemplo de código Python anterior).
- **Interface Web:** O botão "Buscar" pode apresentar uma instrução para a busca de exames. A área `.white-space` deve formatar os resultados de forma legível.

3.4. Buscar Compromissos Agendados

- **Fluxo:** O usuário solicita a busca por compromissos, possivelmente especificando um período. O backend processa a solicitação e utiliza a Google Calendar API para consultar os eventos dentro do intervalo de tempo definido. Os compromissos encontrados são enviados para exibição na interface.
- **Implementação Backend (Google Calendar API):** Utilização da função de listagem de eventos da API, com parâmetros para definir o período da busca. (Consultar exemplo de código Python anterior).
- **Interface Web:** O botão "Buscar" pode incluir a funcionalidade de busca de agendamentos ou um botão dedicado pode ser considerado. A exibição dos compromissos deve incluir detalhes relevantes.

4. Tecnologias Envolvidas

- **Frontend:** HTML, CSS, JavaScript.
- **Backend:** Python, Flask/Django, `google-api-python-client`, `google-auth`.
- **Infraestrutura:** Google Cloud Platform (GCP).

5. Próximos Passos

- Desenvolvimento completo do backend e definição dos endpoints da API.
- Implementação da lógica de interpretação da linguagem do usuário no backend.
- Integração das chamadas às APIs do Google.
- Desenvolvimento da comunicação entre frontend e backend.
- Melhorias na interface web para histórico e exibição de resultados.
- Implementação da funcionalidade de login/logout.
- Tratamento de erros e feedback ao usuário.
- Testes e garantia de qualidade.
- Considerações de segurança.

6. Conclusão

O projeto BioAssist, com sua interface web inicial, visa proporcionar uma maneira intuitiva de gerenciar informações de saúde através da integração com os serviços do Google Workspace. O desenvolvimento do backend é fundamental para conectar a interface com as funcionalidades desejadas, oferecendo uma solução prática e eficiente para os usuários.