Sumário

[Documentação do Programa: app 2](#_Toc184403766)

[Documento de Requisitos 2](#_Toc184403767)

[Manual do Usuário final do programa 7](#_Toc184403768)

[Manual do Usuário Final - Chatbot "Mestre dos Alimentos" 7](#_Toc184403769)

[Manual do Desenvolvedor 11](#_Toc184403770)

[Manual do Desenvolvedor 11](#_Toc184403771)

[Caso de Uso 16](#_Toc184403772)

[Diagrama de Sequência 21](#_Toc184403773)

[Diagrama de Caso de Uso 21](#_Toc184403774)

[Diagrama de Atividades 22](#_Toc184403775)

[Diagrama de Fluxograma 22](#_Toc184403776)

[Mapa Mental 23](#_Toc184403777)

# Documentação do Programa: app

Esta documentação descreve a estrutura e o funcionamento do Programa app, incluindo a definição das funções extraídas do código e os diagramas associados para visualizar o fluxo e as relações do Programa.

## Documento de Requisitos

Documento de Requisitos do Sistema: Chatbot Assistente de Nutrição - Mestre dos Alimentos  
1. Introdução ao Programa  
1.1 Contexto

O **Mestre dos Alimentos** é um chatbot interativo desenvolvido para fornecer informações e orientações sobre nutrição, alimentação saudável e dietas. Ele utiliza inteligência artificial (IA) para responder perguntas de forma personalizada e precisa, com validações para garantir que as respostas estejam alinhadas a tópicos específicos relacionados à nutrição.  
1.2 Objetivo

O objetivo do programa é criar um assistente digital confiável e acessível para ajudar usuários a esclarecer dúvidas sobre nutrição, oferecendo informações baseadas em boas práticas e ciência.  
1.3 Problema Resolvido

O sistema aborda a crescente demanda por informações confiáveis sobre nutrição e dietas, evitando a disseminação de desinformação. Ele também restringe as respostas a tópicos relevantes, garantindo foco e precisão.  
2. Arquitetura do Programa  
 2.1 Componentes Principais  
**Interface do Usuário (Streamlit):** - Permite que os usuários interajam com o chatbot por meio de uma interface web simples e amigável.  
 - Exibe imagens e mensagens de aviso ou erro.  
**Modelo de IA (OpenAI):**   
 - Utiliza a API da OpenAI para gerar respostas baseadas no contexto fornecido e nas perguntas do usuário.  
**Validação de Conteúdo (Guardrails):**  
 - Garante que as respostas estejam alinhadas aos tópicos permitidos (nutrição e alimentação) e bloqueia respostas irrelevantes ou inadequadas.  
**Gerenciamento de Recursos:** - Utiliza a biblioteca `dotenv` para carregar variáveis de ambiente, como a chave da API da OpenAI.  
 - Gerencia imagens carregadas localmente para enriquecer a interface do usuário.  
2.2 Tecnologias Utilizadas  
- \*\*Linguagem de Programação:\*\* Python  
- \*\*Framework de Interface:\*\* Streamlit  
- \*\*Modelo de IA:\*\* OpenAI GPT (via API)  
- \*\*Validação de Respostas:\*\* Guardrails AI  
- \*\*Gerenciamento de Variáveis de Ambiente:\*\* `python-dotenv`  
- \*\*Manipulação de Imagens:\*\* PIL (Python Imaging Library)  
2.3 Comunicação entre Componentes  
1. O usuário interage com a interface Streamlit.  
2. A entrada do usuário é enviada para o modelo de IA da OpenAI.  
3. A resposta gerada pelo modelo é validada pelo Guardrails.  
4. A resposta validada é exibida ao usuário na interface.  
 3. Requisitos Funcionais e Não Funcionais  
3.1 Requisitos Funcionais  
- Permitir que o usuário insira perguntas sobre nutrição e dietas.  
- Garantir que as respostas sejam geradas pelo modelo GPT da OpenAI.  
- Validar as respostas para garantir que estejam dentro dos tópicos permitidos.  
- Exibir mensagens de erro claras em caso de falhas na API ou validação.  
- Carregar e exibir imagens na interface, como logotipos e banners.  
3.2 Requisitos Não Funcionais  
**Performance:** O sistema deve responder às perguntas do usuário em até 5 segundos.  
**Segurança:** As chaves da API devem ser armazenadas de forma segura usando variáveis de ambiente.  
**Escalabilidade:** O sistema deve ser capaz de atender a múltiplos usuários simultaneamente.  
**Usabilidade:** A interface deve ser intuitiva e acessível para usuários leigos.  
**Manutenibilidade:** O código deve ser modular e bem documentado para facilitar futuras atualizações.  
4. Diagrama de Fluxo de Trabalho  
```plaintext  
[Usuário] --> [Interface Streamlit] --> [Entrada do Usuário]  
 ↓ ↓  
[Validação de Entrada] [Envio para OpenAI API]  
 ↓ ↓  
[Resposta da API] <-- [Validação com Guardrails] <-- [Restrição de Tópicos]  
 ↓  
[Exibição da Resposta na Interface]  
```  
5. Manual do Usuário Final  
 5.1 Instruções de Uso  
1. Acesse a interface do chatbot no navegador.  
2. Insira uma pergunta sobre nutrição ou dietas no campo de entrada.  
3. Clique no botão "Enviar".  
4. Aguarde enquanto o sistema processa sua pergunta.  
5. Visualize a resposta gerada pelo chatbot na tela.  
 5.2 Mensagens de Feedback  
**Avisos:**  
 - "Por favor, insira uma pergunta." (quando o campo de entrada está vazio)  
 - "Logo não encontrado." (quando a imagem do logotipo não está disponível)  
**Erros:**   
 - "Erro na API da OpenAI: [detalhes do erro]" (em caso de falha na comunicação com a API)  
 - "Erro ao aplicar Guardrails: [detalhes do erro]" (em caso de falha na validação da resposta)  
 6. Manual do Desenvolvedor  
 6.1 Configuração do Ambiente  
1. Clone o repositório do projeto.  
2. Instale as dependências:  
 ```bash  
 pip install -r requirements.txt  
 ```  
3. Crie um arquivo `.env` no diretório raiz e adicione sua chave da API da OpenAI:  
 ```  
 OPENAI\_API\_KEY=your\_api\_key\_here  
 ```  
4. Certifique-se de que as imagens necessárias estejam no diretório `./assets`.  
 6.2 Estrutura do Código  
**main.py:** Contém a lógica principal do programa.  
**assets/:** Diretório para imagens usadas na interface.  
**.env:** Arquivo para armazenar variáveis de ambiente.  
6.3 Pontos de Extensão  
**Adicionar novos tópicos permitidos:** Modifique a lista `valid\_topics` em `RestrictToTopic`.  
**Alterar o modelo de IA:** Substitua o valor de `model` na chamada da API da OpenAI.  
**Customizar o backstory:** Edite o texto da variável `backstory` para ajustar o contexto do chatbot.  
6.4 Boas Práticas  
- Sempre teste o sistema após alterações no código.  
- Use mensagens de log para depuração.  
- Mantenha as dependências atualizadas.  
7. Exceções e Tratamento de Erros  
7.1 Possíveis Exceções  
**Erro de API da OpenAI:**  
 - Causa: Chave da API inválida, falta de conectividade, ou limite de uso excedido.  
 - Tratamento: Exibir mensagem de erro detalhada ao usuário.  
**Erro de Validação com Guardrails:**  
 - Causa: Resposta gerada fora dos tópicos permitidos.  
 - Tratamento: Informar ao usuário que a resposta não foi validada.  
**Erro ao Carregar Imagens:**  
 - Causa: Caminho de arquivo incorreto ou imagem ausente.  
 - Tratamento: Exibir uma mensagem de aviso.  
7.2 Logs de Erros  
- Adicione logs para registrar exceções no servidor para facilitar a depuração.  
8. Atualizações Futuras  
- Implementar suporte a múltiplos idiomas.  
- Permitir upload de imagens para análise nutricional (ex.: rótulos de alimentos).  
- Adicionar integração com APIs de bancos de dados nutricionais.  
Este documento pode ser atualizado conforme o sistema evolui.

# Manual do Usuário final do programa

Manual do Usuário Final - Chatbot "Mestre dos Alimentos"  
 Introdução

Bem-vindo ao \*\*Mestre dos Alimentos\*\*, seu assistente digital de nutrição! Este chatbot foi projetado para responder perguntas relacionadas à dieta, alimentação saudável, nutrientes e outros tópicos de nutrição. Ele é baseado em inteligência artificial e utiliza tecnologias avançadas para fornecer respostas personalizadas e de alta qualidade.

Este manual irá guiá-lo sobre como utilizar o chatbot de maneira eficiente, desde o acesso à interface até a realização de perguntas e interpretação das respostas.  
Como Acessar o Chatbot

O chatbot "Mestre dos Alimentos" é acessado por meio de uma interface web criada com Streamlit. Certifique-se de que você tenha acesso à página onde o chatbot está hospedado. O endereço será fornecido pelo administrador do sistema ou pela equipe que disponibilizou o serviço.  
Funcionalidades Principais  
1. **Realizar Perguntas sobre Nutrição:** Pergunte sobre dietas, nutrientes, alimentos saudáveis, benefícios de alimentos específicos, entre outros tópicos relacionados à nutrição.  
**2. Interface Simples e Intuitiva:** A interface permite que você insira perguntas de forma prática e obtenha respostas instantâneas.  
**3. Restrições de Tópicos:** O chatbot é configurado para responder apenas perguntas relacionadas à nutrição, garantindo que as respostas sejam relevantes e confiáveis.  
**4. Validação de Respostas:** O sistema utiliza validações avançadas para garantir que as respostas sejam apropriadas e dentro do escopo permitido.  
Passo a Passo para Utilizar o Chatbot  
1. Acessando a Página  
- Ao acessar a página do chatbot, você será recebido com o título \*\*"Chatbot Assistente de Nutrição - Mestre dos Alimentos"\*\* e uma breve descrição do serviço.  
2. Inserindo uma Pergunta  
- Localize o campo de texto com o rótulo **"Faça uma pergunta sobre dieta ou nutrição:".**  
**Digite sua pergunta no campo de texto. Por exemplo:** - "Quais são os benefícios do consumo de abacate?"  
 - "Como montar uma dieta equilibrada para ganho de massa muscular?"  
3. Enviando a Pergunta  
- Após digitar sua pergunta, clique no botão \*\*"Enviar"\*\*.  
- O sistema processará sua pergunta e exibirá uma mensagem de carregamento: \*\*"Pensando..."\*\*.  
4. Visualizando a Resposta  
- Após alguns segundos, o chatbot exibirá uma resposta:  
 - Se a pergunta for válida e processada corretamente, a resposta aparecerá em um quadro de sucesso.  
 - Se houver algum erro, como uma pergunta fora do escopo ou problemas técnicos, uma mensagem de erro será exibida.  
5. Mensagens de Erro  
**Pergunta vazia:** Se você clicar em "Enviar" sem inserir uma pergunta, o sistema exibirá uma mensagem de aviso: \*\*"Por favor, insira uma pergunta."\*\*  
**Erro técnico:** Se ocorrer um problema com o processamento da pergunta, uma mensagem de erro será exibida, como: \*\*"Erro na API da OpenAI"\*\* ou \*\*"Erro ao aplicar Guardrails"\*\*.  
Exemplos de Perguntas  
Aqui estão alguns exemplos de perguntas que você pode fazer ao chatbot:  
1. "Quais alimentos são ricos em ferro?"  
2. "Como posso melhorar minha dieta para perder peso?"  
3. "O que é uma dieta cetogênica e como funciona?"  
4. "Quais são os benefícios do consumo de frutas cítricas?"  
**Nota:** Certifique-se de manter suas perguntas relacionadas à nutrição. Perguntas fora do escopo, como política ou esportes, não serão respondidas.  
Elementos Visuais  
**Imagens:** A interface pode exibir imagens, como um logotipo ou ilustrações relacionadas à nutrição. Caso a imagem não seja exibida, uma mensagem de aviso será mostrada.  
**Mensagens de Status:**  
 **Sucesso:** Respostas válidas são exibidas em um quadro de sucesso.  
 **Aviso:** Mensagens de erro ou avisos aparecem em quadros de aviso ou erro.  
Requisitos Técnicos para o Usuário Final  
- Um navegador web moderno (Google Chrome, Mozilla Firefox, Microsoft Edge, etc.).  
- Conexão estável com a internet.  
- Acesso à página do chatbot.  
*Dicas para Melhor Experiência***1. Seja Específico:** Quanto mais clara e específica for sua pergunta, mais relevante será a resposta.  
 - Exemplo ruim: "Fale sobre comida."  
 - Exemplo bom: "Quais alimentos ajudam a melhorar a saúde do coração?"  
**2. Evite Tópicos Não Relacionados:** O chatbot é configurado para não responder perguntas fora do escopo de nutrição.  
**3. Atualize a Página:** Se você encontrar algum erro, tente atualizar a página antes de fazer uma nova pergunta.  
*Mensagens de Feedback*

Se você tiver sugestões ou encontrar problemas, entre em contato com o administrador ou suporte técnico responsável pelo sistema.

Com essas instruções, você está pronto para utilizar o \*\*Mestre dos Alimentos\*\* e aproveitar ao máximo as orientações nutricionais fornecidas pelo chatbot!

# Manual do Desenvolvedor

## Manual do Desenvolvedor

Este manual foi criado para orientar desenvolvedores na compreensão, configuração, manutenção e extensão do sistema "Mestre dos Alimentos", um chatbot assistente de nutrição baseado em \*\*Streamlit\*\* e \*\*OpenAI GPT\*\*. Ele inclui informações sobre a configuração do ambiente, estrutura do código, pontos de extensão e boas práticas.  
*1. Introdução*O "Mestre dos Alimentos" é um chatbot interativo especializado em responder a perguntas relacionadas à nutrição e dieta. Ele utiliza a API da OpenAI para gerar respostas e aplica validações usando a biblioteca \*\*Guardrails\*\* para restringir tópicos e garantir a conformidade do conteúdo.  
Principais Tecnologias Utilizadas  
- \*\*Streamlit\*\*: Para a interface do usuário.  
- \*\*OpenAI API\*\*: Para geração de respostas baseadas em IA.  
- \*\*Guardrails\*\*: Para validação e restrição de tópicos.  
- \*\*Pillow (PIL)\*\*: Para manipulação de imagens.  
- \*\*dotenv\*\*: Para gerenciar variáveis de ambiente.  
*2. Configuração do Ambiente*2.1. Pré-requisitos  
**Certifique-se de ter instalado:**  
- Python 3.8 ou superior  
- Pip (gerenciador de pacotes do Python)  
2.2. Instalação  
**1. Clone o repositório do projeto**:  
 ```bash  
 git clone <URL\_DO\_REPOSITORIO>  
 cd <NOME\_DO\_DIRETORIO>  
 ```  
  
2. Crie um ambiente virtual e ative-o:  
 ```bash  
 python -m venv venv  
 source venv/bin/activate # No Windows: venv\Scripts\activate  
 ```  
3. Instale as dependências:  
 ```bash  
 pip install -r requirements.txt  
 ```  
4. Configure as variáveis de ambiente:  
 - Crie um arquivo `.env` na raiz do projeto.  
 - Adicione a chave da API da OpenAI:  
 ```  
 OPENAI\_API\_KEY=your\_openai\_api\_key\_here  
 ```  
5. Inicie o aplicativo:  
 ```bash  
 streamlit run app.py  
 ```  
3. Estrutura do Código3.1. Organização dos Arquivos  
- `app.py`: Arquivo principal que contém o código do chatbot e a interface Streamlit.  
- `assets/`: Diretório para imagens e outros recursos estáticos.  
- `.env`: Arquivo para variáveis de ambiente (não deve ser versionado).  
- `requirements.txt`: Lista de dependências do projeto.  
  
### \*\*3.2. Fluxo Principal\*\*  
1. \*\*Carregamento de imagens\*\*: Função `load\_image` verifica se as imagens existem antes de carregá-las.  
2. \*\*Entrada do usuário\*\*: O usuário insere uma pergunta no campo de texto.  
3. \*\*API da OpenAI\*\*: Função `get\_response` envia a entrada do usuário para a API da OpenAI e obtém uma resposta.  
4. \*\*Validação com Guardrails\*\*: A resposta é validada para garantir que está dentro dos tópicos permitidos.  
5. \*\*Exibição da resposta\*\*: A resposta validada é exibida na interface.  
4. Pontos de Extensão  
4.1. Adicionar Novos Tópicos Permitidos  
Os tópicos permitidos são definidos no objeto `RestrictToTopic`. Para adicionar novos tópicos, edite a lista `valid\_topics`:  
```python  
valid\_topics=["nutrition", "diet", "food", "health", "exercise"]  
```  
4.2. Alterar o Modelo Utilizado

O modelo padrão é `gpt-4o-mini`. Para usar outro modelo, altere o parâmetro `model` na função `get\_response`:  
```python  
model="gpt-3.5-turbo"  
4.3. Personalizar o Backstory

O backstory define o comportamento do chatbot. Você pode personalizá-lo ajustando o texto na variável `backstory`:  
```python  
backstory = """  
Você é um especialista em nutrição reconhecido mundialmente...  
"""  
4.4. Melhorar a Interface  
Você pode adicionar mais elementos visuais ou funcionalidades à interface Streamlit. Por exemplo:  
- Adicionar gráficos ou tabelas.  
- Incluir um seletor de idioma para internacionalização.  
5. Boas Práticas  
5.1. Gerenciamento de Dependências  
- Sempre atualize o arquivo `requirements.txt` ao adicionar novas dependências:  
 ```bash  
 pip freeze > requirements.txt  
 ```  
5.2. Segurança  
- Nunca exponha a chave da API da OpenAI no código. Use o arquivo `.env` para armazenar informações sensíveis.  
- Certifique-se de que o arquivo `.env` está listado no `.gitignore`.  
5.3. Tratamento de Erros  
- Certifique-se de que todas as exceções são tratadas adequadamente para evitar falhas inesperadas.  
- Adicione mensagens de erro amigáveis ao usuário final.  
6. Tratamento de Erros e Exceções  
6.1. Erros da API OpenAI  
Caso a API da OpenAI retorne um erro, a função `get\_response` captura a exceção e exibe uma mensagem descritiva:  
```python  
except Exception as e:  
 return f"Erro na API da OpenAI: {str(e)}"  
```  
6.2. Falha na Validação com Guardrails  
Se a validação falhar, o sistema exibe uma mensagem de erro indicando o motivo:  
```python  
except Exception as e:  
 return f"Erro ao aplicar Guardrails: {str(e)}"  
```  
6.3. Imagem Não Encontrada  
Se uma imagem não for encontrada no diretório especificado, uma mensagem de aviso será exibida:  
```python  
st.warning("Adicione uma imagem de laboratório de nutrientes para enriquecer a interface.")  
```  
7. Contribuições Futuras  
7.1. Como Contribuir  
1. Faça um fork do repositório.  
2. Crie um branch para sua contribuição:  
 ```bash  
 git checkout -b minha-contribuicao  
 ```  
3. Faça suas alterações e envie um pull request.  
7.2. Testes  
- Adicione testes para novas funcionalidades antes de enviá-las.  
- Utilize ferramentas como `pytest` para automatizar os testes.  
8. Referências  
- [Streamlit Documentação](<https://docs.streamlit.io/>)  
- [OpenAI API Documentação](https://platform.openai.com/docs/)  
- [Guardrails AI Documentação](<https://docs.guardrailsai.com/>)

Com este manual, você estará apto a configurar, manter e expandir o sistema "Mestre dos Alimentos". Para dúvidas ou problemas, entre em contato com o mantenedor do projeto.

# Caso de Uso

Caso de Uso Completo: Chatbot Assistente de Nutrição - "Mestre dos Alimentos"  
1. Introdução ao Programa

O \*\*Mestre dos Alimentos\*\* é um chatbot interativo baseado em inteligência artificial, desenvolvido para fornecer informações e orientações sobre nutrição e dietas. O sistema é projetado para atender usuários que buscam respostas confiáveis sobre alimentação saudável, nutrientes e planejamento de dietas. Ele utiliza a API da OpenAI e a biblioteca Guardrails para garantir que as respostas sejam relevantes e restritas ao domínio de nutrição, evitando tópicos irrelevantes ou sensíveis.  
Objetivo do Programa:   
- Oferecer informações confiáveis e personalizadas sobre nutrição e dietas.  
- Restringir a interação a tópicos relacionados à saúde e alimentação, garantindo segurança e relevância para o usuário.  
Problema que Resolve:   
- Dificuldade de encontrar informações precisas e confiáveis sobre nutrição em fontes online.  
- Necessidade de um assistente digital que possa responder perguntas de forma interativa e personalizada, sem desviar para tópicos fora do escopo de nutrição.  
2. Arquitetura do Programa  
**Componentes Principais:**1. \*\*Interface do Usuário:\*\*  
 - Desenvolvida com Streamlit, oferece uma interface simples e intuitiva para interação com o chatbot.  
 - Permite ao usuário inserir perguntas e visualizar respostas.  
2. \*\*Backend de IA:\*\*  
 - Utiliza a API da OpenAI (modelo `gpt-4o-mini`) para gerar respostas baseadas em prompts personalizados.  
 - Inclui um "backstory" para definir o comportamento e tom do chatbot como um especialista em nutrição.  
3. Validação com Guardrails:   
 - Integra a biblioteca Guardrails para validar as respostas geradas pela IA, garantindo que elas estejam dentro do escopo de nutrição e dietas.  
 - Restringe tópicos inadequados ou irrelevantes, como política, violência, ou spam.  
4. Gerenciamento de Recursos:   
 - Carrega imagens (como logotipos ou banners) para enriquecer a interface do usuário, utilizando a biblioteca PIL.  
\*\*Tecnologias Utilizadas:\*\*  
- \*\*Streamlit\*\*: Para a interface do usuário.  
- \*\*OpenAI API\*\*: Para geração de respostas.  
- \*\*Guardrails\*\*: Para validação de respostas.  
- \*\*PIL (Pillow)\*\*: Para manipulação de imagens.  
- \*\*dotenv\*\*: Para gerenciamento de variáveis de ambiente (como a chave da API).  
\*\*Comunicação entre Componentes:\*\*  
- O usuário interage com a interface Streamlit.  
- As perguntas são enviadas para o backend, que utiliza a API da OpenAI para gerar uma resposta.  
- A resposta é validada pelo Guardrails antes de ser exibida na interface.  
3. Requisitos Funcionais e Não Funcionais  
**Requisitos Funcionais:**   
1. O sistema deve permitir que o usuário insira perguntas relacionadas à nutrição e dietas.  
2. O sistema deve retornar respostas geradas pela API da OpenAI.  
3. O sistema deve validar as respostas utilizando a biblioteca Guardrails.  
4. O sistema deve exibir mensagens de erro claras em caso de falhas, como problemas na API ou validação.  
5. O sistema deve carregar e exibir imagens na interface, se disponíveis.  
**Requisitos Não Funcionais:**1. O sistema deve ser responsivo e rápido, com tempo de resposta inferior a 5 segundos na maioria dos casos.  
2. O sistema deve ser seguro, utilizando variáveis de ambiente para proteger a chave da API.  
3. O sistema deve ser modular e fácil de manter, com código bem organizado e documentado.  
4. O sistema deve ser escalável para suportar múltiplos usuários simultaneamente.  
4. Diagrama de Fluxo de Trabalho  
```mermaid  
flowchart TD  
 A[Usuário] -->|Insere Pergunta| B[Interface Streamlit]  
 B -->|Envia Pergunta| C[Backend]  
 C -->|Chama OpenAI API| D[Modelo GPT]  
 D -->|Retorna Resposta| C  
 C -->|Valida Resposta| E[Guardrails]  
 E -->|Resposta Válida| F[Exibe Resposta na Interface]  
 E -->|Resposta Inválida| G[Exibe Mensagem de Erro]  
```  
5. Manual do Usuário Final  
\*\*Passos para Utilizar o Programa:\*\*  
1. Acesse a interface do chatbot através do navegador.  
2. Insira uma pergunta sobre nutrição ou dietas no campo de texto.  
3. Clique no botão "Enviar".  
4. Aguarde enquanto o sistema processa sua pergunta e retorna uma resposta.  
5. Leia a resposta exibida na interface. Caso ocorra um erro, siga as instruções exibidas.  
\*\*Exemplo de Uso:\*\*  
- Pergunta: "Quais são os benefícios da dieta mediterrânea?"  
- Resposta: "A dieta mediterrânea é rica em frutas, vegetais, grãos integrais, azeite de oliva e peixes, e está associada à redução do risco de doenças cardiovasculares, diabetes e obesidade."  
 6. Manual do Desenvolvedor  
\*\*Configuração do Ambiente:\*\*  
1. Clone o repositório do projeto.  
2. Instale as dependências utilizando `pip install -r requirements.txt`.  
3. Crie um arquivo `.env` na raiz do projeto e adicione a chave da API da OpenAI:  
 ```  
 OPENAI\_API\_KEY=your\_openai\_api\_key  
 ```  
4. Certifique-se de que as imagens necessárias (como logotipos) estão no diretório `./assets`.  
\*\*Estrutura do Código:\*\*  
- \*\*`load\_image`\*\*: Função para carregar imagens.  
- \*\*`get\_response`\*\*: Função principal para gerar e validar respostas.  
- \*\*`guard`\*\*: Configuração do Guardrails para validação.  
- \*\*Interface Streamlit\*\*: Configuração da interface do usuário.  
\*\*Boas Práticas:\*\*  
- Sempre valide respostas geradas pela IA utilizando o Guardrails.  
- Utilize variáveis de ambiente para armazenar informações sensíveis.  
- Adicione novos tópicos válidos ou inválidos ao Guardrails conforme necessário.  
7. Exceções e Tratamento de Erros  
Erros Comuns:  
1. \*\*Erro na API da OpenAI:\*\*  
 - Mensagem exibida: "Erro na API da OpenAI: [detalhes do erro]"  
 - Solução: Verifique sua conexão com a internet e a chave da API.  
2. \*\*Erro ao Aplicar Guardrails:\*\*  
 - Mensagem exibida: "Erro ao aplicar Guardrails: [detalhes do erro]"  
 - Solução: Verifique a configuração do Guardrails e os tópicos válidos/invalidos.  
3. \*\*Imagem Não Encontrada:\*\*  
 - Mensagem exibida: "Adicione uma imagem de laboratório de nutrientes para enriquecer a interface."  
 - Solução: Certifique-se de que a imagem está no diretório correto.  
4. \*\*Pergunta Inválida:\*\*  
 - Mensagem exibida: "Por favor, insira uma pergunta."  
 - Solução: Insira uma pergunta válida no campo de texto.  
8. Atualizações Futuras  
- Adicionar suporte a múltiplos idiomas.  
- Melhorar a interface com mais opções visuais.  
- Expandir a base de conhecimento do chatbot para incluir tópicos complementares, como exercícios físicos.

Essa documentação fornece uma visão completa do sistema, permitindo que usuários e desenvolvedores compreendam e utilizem o programa de maneira eficiente.

# Diagrama de Sequência

O diagrama abaixo descreve o Diagrama de Sequência.

Tabela

Descrição gerada automaticamente com confiança baixa

## Diagrama de Caso de Uso

O diagrama abaixo descreve o Diagrama de Caso de Uso.

Diagrama

Descrição gerada automaticamente

# Diagrama de Atividades

O diagrama abaixo descreve o Diagrama de Atividades.

Uma imagem contendo Gráfico

Descrição gerada automaticamente

# Diagrama de Fluxograma

O fluxograma abaixo descreve o processo executado pelo programa.

Diagrama, Texto

Descrição gerada automaticamente

# Mapa Mental

O mapa mental abaixo descreve as funcionalidades principais do programa.

Diagrama

Descrição gerada automaticamente