□ MANUAL: Como trocar a IA da Groq para OpenAI no Agente Solen Whtas

---

□ O que vamos fazer?

O **Agente Solen Whtas** hoje usa a **Groq API** para gerar respostas inteligentes.

Se você quiser usar a **OpenAI** (ChatGPT oficial da OpenAI, como o modelo gpt-3.5-turbo ou gpt-4), é totalmente possível!

Basta mudar algumas configurações simples:

De	Para
Groq API	OpenAI API
groq biblioteca	openai biblioteca

🛠 Passo a Passo: Mudando da Groq para a OpenAI

---

1. 📦 Instalar a biblioteca da OpenAI

No terminal, instale o pacote oficial da OpenAI:

bash

CopiarEditar

pip install openai

ou se preferir direto no seu projeto:

bash

CopiarEditar

pip install --upgrade openai

---

2. □ Atualizar seu .env

Agora o sistema vai precisar da **API Key da OpenAI**.

- Pegue sua chave aqui na OpenAI.
- Atualize o arquivo .env:

env

CopiarEditar

OPENAI_API_KEY=sk-xxxxxxxxxxxxxxxxxxxxxxxxxxx

 *Não misture! Você usará a chave da OpenAI, e **não** a da Groq agora.*

---

3. Atualizar o código do bot

Abra o arquivo **bot/ai_bot.py**.

Você vai fazer 3 mudanças principais:

A) Trocar a importação da Groq pela OpenAI

ANTES:

python

CopiarEditar

from groq import Groq

DEPOIS:

python

CopiarEditar

import openai

B) Alterar como conecta à API

ANTES no __init__:

python

CopiarEditar

self.groq_api_key = "sua-chave-da-groq"

DEPOIS:

python

CopiarEditar

self.openai_api_key = os.getenv("OPENAI_API_KEY")

openai.api_key = self.openai_api_key

(e pode apagar a linha do Groq completamente.)

C) □ Alterar como chama o modelo

ANTES, chamando a Groq:

python

CopiarEditar

client = Groq(api_key=self.groq_api_key)

resposta = client.chat.completions.create(

    model=self.model_name,

    messages=[

        {"role": "system", "content": system_message},

        {"role": "user", "content": pergunta}

    ],

    temperature=0.1

)

return resposta.choices[0].message.content

DEPOIS, chamando a OpenAI:

python

CopiarEditar

```
resposta = openai.ChatCompletion.create(
    model="gpt-3.5-turbo",  # ou "gpt-4" se você tiver acesso
    messages=[
        {"role": "system", "content": system_message},
        {"role": "user", "content": pergunta}
    ],
    temperature=0.1
)
return resposta['choices'][0]['message']['content']
```

---

4. ✅ Resumindo: Como seu ai_bot.py vai ficar

Aqui está o resumo do que muda:

```
python
CopiarEditar
import os
import openai
from vectorstore_manager import carregar_vectorstore


class AIBot:
    def __init__(self):
        self.model_name = "gpt-3.5-turbo"  # Modelo da OpenAI
        self.vectorstore = carregar_vectorstore()
        if self.vectorstore is None:
            raise FileNotFoundError("Arquivos do vectorstore incompletos ou falha ao carregar. Use o Gerenciador para processar os documentos.")
```

```
        self.openai_api_key = os.getenv("OPENAI_API_KEY")
        openai.api_key = self.openai_api_key

    def responder(self, pergunta: str) -> str:
        try:
            docs_with_scores = self.vectorstore.similarity_search_with_score(pergunta, k=10)
            contexto = "\n\n".join([
                f"**{doc.metadata.get('fonte', 'Fonte desconhecida')} (Score: {score:.2f}):**\n{doc.page_content}"
                for doc, score in docs_with_scores
            ])

            system_message = f"""Você é um assistente técnico. Baseie-se nestas informações:
{contexto}

Regras:
1. Seja conciso e técnico.
2. Cite a fonte quando relevante.
3. Se não souber, diga "Não encontrado nos documentos".
4. Use markdown para formatação."""

            resposta = openai.ChatCompletion.create(
                model=self.model_name,
                messages=[
                    {"role": "system", "content": system_message},
```

```
            {"role": "user", "content": pergunta}
        ],
        temperature=0.1
    )
    return resposta['choices'][0]['message']['content']

except Exception as e:
    return f"⚠ Erro: {str(e)}"
```

---

5. 🔁 Depois disso

- Reinicie seu sistema.
- Agora as respostas serão geradas usando **a OpenAI** (ChatGPT original), **não mais a Groq**.
- Pode usar tanto no WhatsApp quanto no WebApp como antes.

---

6. 🎁 Bônus: Mudança no WebApp webapp.py

Se quiser mudar o WebApp (webapp.py) também, siga **as mesmas mudanças**:

- Importar openai
- Configurar a chave
- Substituir onde usa Groq pelo openai.ChatCompletion.create

Exatamente igual ao ai_bot.py.

🚀 Resumo Final

Item	Antes	Depois
Biblioteca	groq	openai

Item	Antes	Depois
Chave no .env	GROQ_API_KEY	OPENAI_API_KEY
Modelo	"qwen-2.5-coder-32b"	"gpt-3.5-turbo" ou "gpt-4"
Modo de chamada	client.chat.completions.create()	openai.ChatCompletion.create()

 Observações Importantes

- **Custo**: A OpenAI cobra por tokens usados. Confira aqui.
- **Velocidade**: A Groq pode ser um pouco mais rápida e barata dependendo do caso.
- **Limites**: A OpenAI tem limites de uso (tokens/mês), principalmente em planos gratuitos.

 Pronto!

Agora seu **Agente Solen Whtas** está usando a **OpenAI** como cérebro!