ChatBot
Aulas 11 e 12

Prof. Me Daniel Vieira

SENAI

Agenda

1 - Criando API com LLM integrada
2 - FastAPI
3 - Estrutura do projeto
4 - Testes com Insomnia

Estrutura do projeto API com Chatbot

```
chatbot_api/
│
├── main.py
├── serenatto_bot.py
├── documentos/
│   └── serenatto.pdf
├── chroma_db/   ← será criado ao rodar o código
└── requirements.txt
```

Código serenatto_bot.py

```
import os
from typing import List
from llama_index.core import SimpleDirectoryReader, StorageContext,
VectorStoreIndex
from llama_index.core.node_parser import SentenceSplitter
from llama_index.embeddings.huggingface import HuggingFaceEmbedding
from llama_index.vector_stores.chroma import ChromaVectorStore
from llama_index.llms.groq import Groq
from llama_index.core.memory import ChatSummaryMemoryBuffer
import chromadb
from tempfile import TemporaryDirectory
from PyPDF2 import PdfReader
```

Código serenatto_bot.py

```
class ChromaEmbeddingWrapper:
    def __init__(self, model_name: str):
        self.model = HuggingFaceEmbedding(model_name=model_name)

    def __call__(self, input: List[str]) -> List[List[float]]:
        return self.model.embed_documents(input)
```

Código serenatto_bot.py

```
class SerenattoBot:
    def __init__(self):
        self.embed_model =
HuggingFaceEmbedding(model_name='intfloat/multilingual-e5-large')
        self.embed_model_chroma =
ChromaEmbeddingWrapper(model_name='intfloat/multilingual-e5-large')

        chroma_client = chromadb.PersistentClient(path='./chroma_db')
        collection_name = 'documentos_serenatto'
        chroma_collection = chroma_client.get_or_create_collection(
            name=collection_name,
            embedding_function=self.embed_model_chroma
        )
```

Código serenatto_bot.py

```
self.vector_store = ChromaVectorStore(chroma_collection=chroma_collection)
        self.storage_context =
StorageContext.from_defaults(vector_store=self.vector_store)

        self.llms = Groq(model='llama3-70b-8192',
api_key='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')

        self.document_index = None
        self.chat_engine = None

        self.carregar_pdf()
```

Código serenatto_bot.py

```
def carregar_pdf(self):
        with TemporaryDirectory() as tmpdir:
            pdf_path = "documentos/serenatto.pdf"
            text = ""
            reader = PdfReader(pdf_path)
            for page in reader.pages:
                text += page.extract_text() or ""
```

Código serenatto_bot.py

```
with open(os.path.join(tmpdir, "temp.txt"), "w", encoding="utf-8") as f:
                f.write(text)

            documentos = SimpleDirectoryReader(input_dir=tmpdir)
            docs = documentos.load_data()

            node_parser = SentenceSplitter(chunk_size=1200)
            nodes = node_parser.get_nodes_from_documents(docs)

            self.document_index = VectorStoreIndex(nodes,
storage_context=self.storage_context, embed_model=self.embed_model)
```

Código serenatto_bot.py

```
memory = ChatSummaryMemoryBuffer(llm=self.llms, token_limit=256)

            self.chat_engine = self.document_index.as_chat_engine(
                chat_mode='context',
                llm=self.llms,
                memory=memory,
                system_prompt='''Você é especialista em cafés da loja Serenatto.
Responda de forma simpática e natural sobre os grãos disponíveis.'''
            )
```

Código serenatto_bot.py

```
def responder(self, mensagem: str) -> str:
        if self.chat_engine is None:
            return "Erro: o bot ainda não está pronto."
        response = self.chat_engine.chat(mensagem)
        return response.response

    def resetar(self):
        if self.chat_engine:
            self.chat_engine.reset()
```

main.py

```python
from fastapi import FastAPI, Body
from pydantic import BaseModel
from serenatto_bot import SerenattoBot

app = FastAPI(title="Serenatto Chatbot API")

# Instancia o bot na inicialização
bot = SerenattoBot()
```

main.py

```python
class MensagemRequest(BaseModel):
    mensagem: str

class MensagemResponse(BaseModel):
    resposta: str
@app.post("/conversar", response_model=MensagemResponse)
def conversar(request: MensagemRequest):
    resposta = bot.responder(request.mensagem)
    return MensagemResponse(resposta=resposta)

@app.post("/resetar")
def resetar():
    bot.resetar()
    return {"status": "Chat resetado com sucesso"}
```

main.py

```python
class MensagemRequest(BaseModel):
    mensagem: str

class MensagemResponse(BaseModel):
    resposta: str
@app.post("/conversar", response_model=MensagemResponse)
def conversar(request: MensagemRequest):
    resposta = bot.responder(request.mensagem)
    return MensagemResponse(resposta=resposta)

@app.post("/resetar")
def resetar():
    bot.resetar()
    return {"status": "Chat resetado com sucesso"}
```

Requirements.txt

```
fastapi
uvicorn
llama-index
PyPDF2
chromadb
transformers
huggingface_hub
```

Comando para executar a api

uvicorn main:app –reload

```
PS D:\SENAI\2025-1\ChatBot\chatbot_api> uvicorn main:app --reload
>>
INFO:     Will watch for changes in these directories: ['D:\\SENAI\\2025-1\\ChatBot\\chatbot_api']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [720] using WatchFiles
INFO:     Started server process [17540]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     127.0.0.1:58451 - "POST /conversar HTTP/1.1" 200 OK
INFO:     127.0.0.1:58457 - "GET /docs HTTP/1.1" 200 OK
INFO:     127.0.0.1:58457 - "GET /openapi.json HTTP/1.1" 200 OK
INFO:     127.0.0.1:58477 - "POST /conversar HTTP/1.1" 200 OK
INFO:     127.0.0.1:58526 - "POST /conversar HTTP/1.1" 200 OK
INFO:     127.0.0.1:58733 - "POST /conversar HTTP/1.1" 200 OK
INFO:     127.0.0.1:58763 - "POST /conversar HTTP/1.1" 200 OK
INFO:     127.0.0.1:58771 - "POST /conversar HTTP/1.1" 200 OK
```

FastApi

Serenatto Chatbot API 0.1.0 OAS 3.1

/openapi.json

default

POST /conversar Conversar

Parameters

Try it out

No parameters

Request body required

application/json

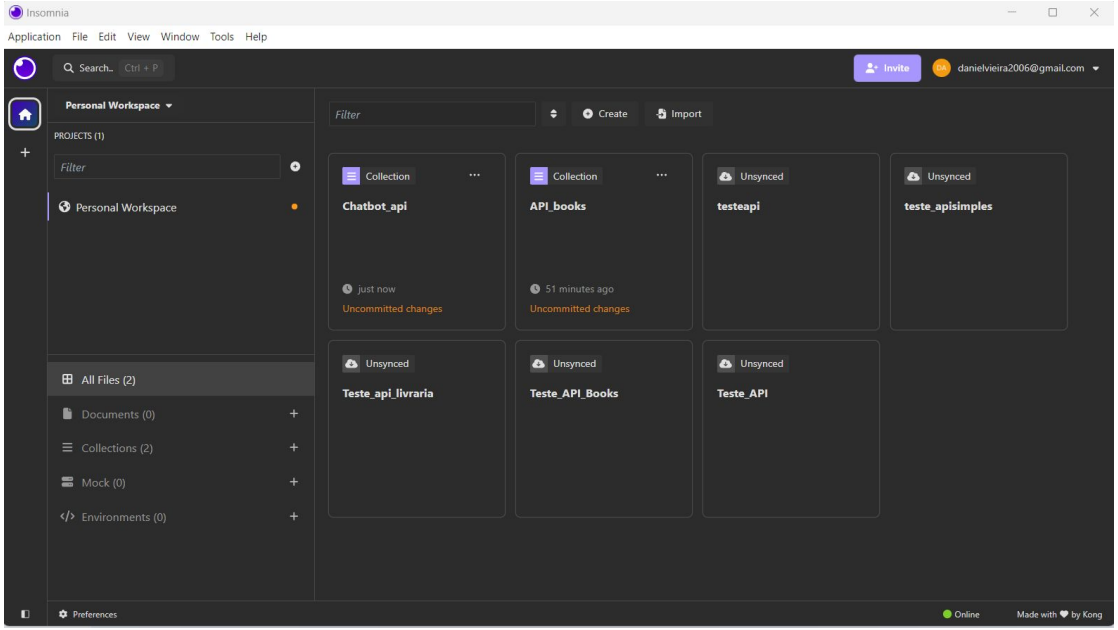
Example Value | Schema

```
{
  "mensagem": "string"
}
```

Responses

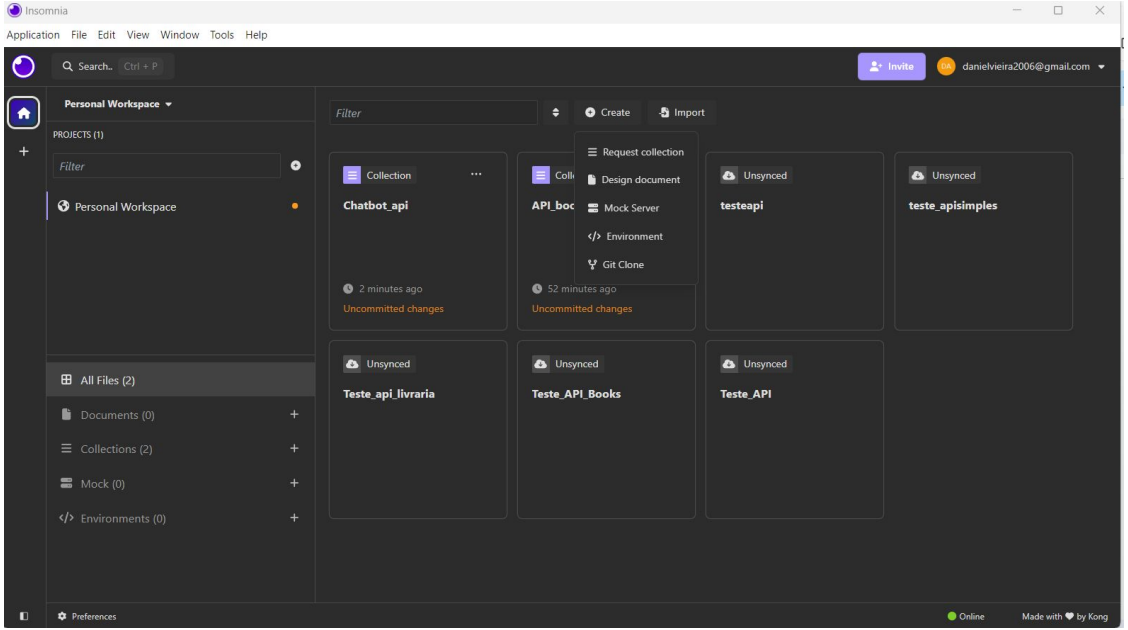
Code	Description	Links

Para testar a API será utilizado o Insomnia



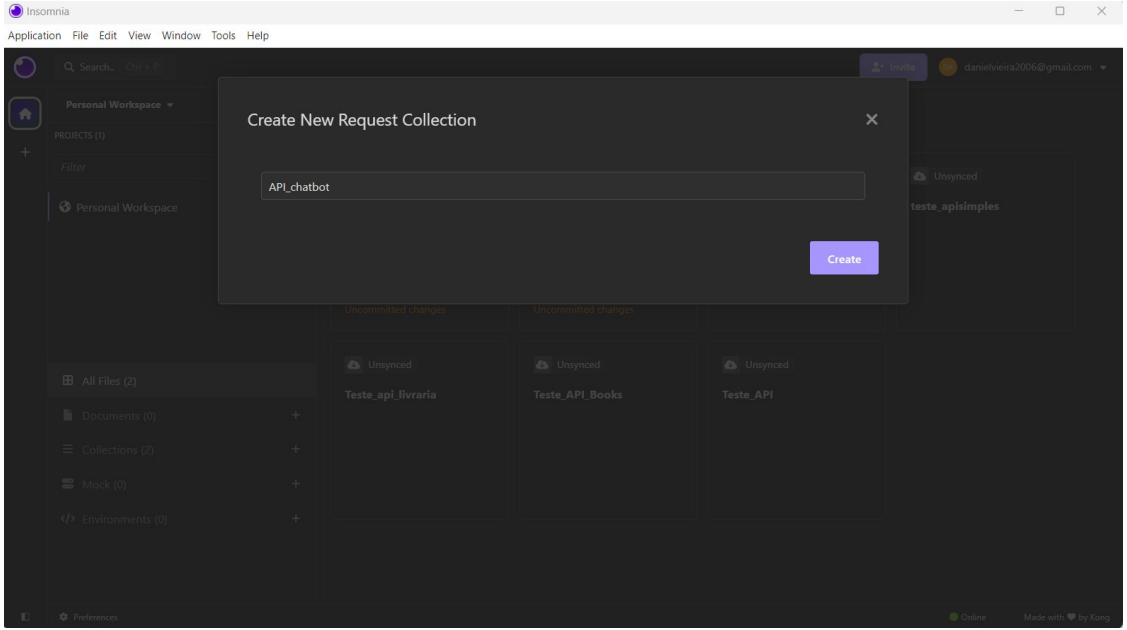
Para testar a API será utilizado o Insomnia

Clicar em create a request collection



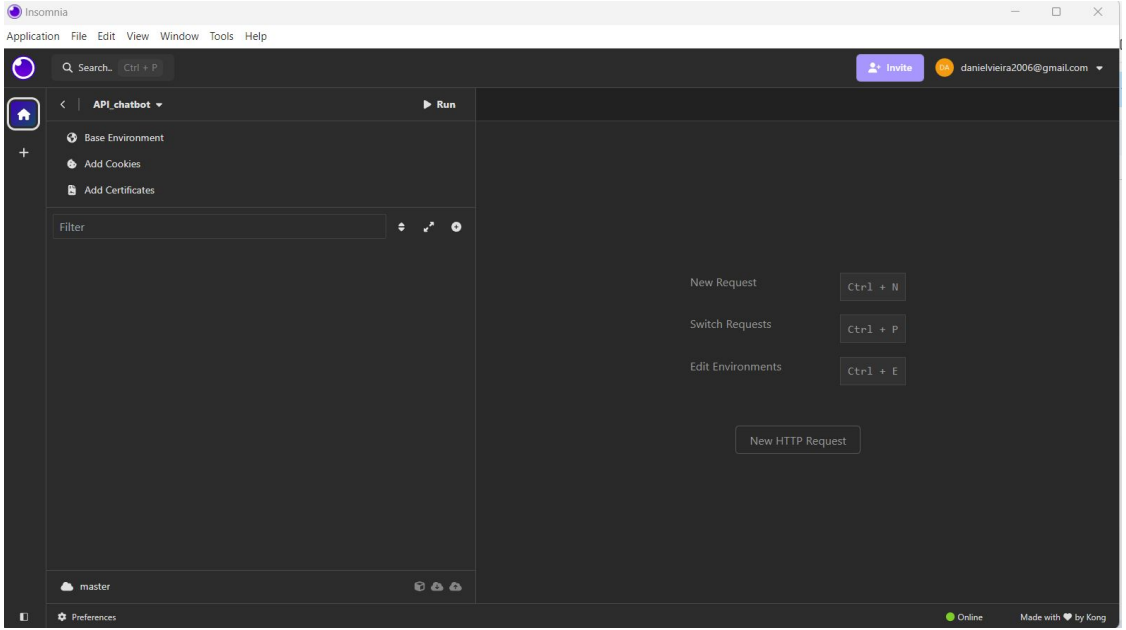
Para testar a API será utilizado o Insomnia

Clicar em create a request collection



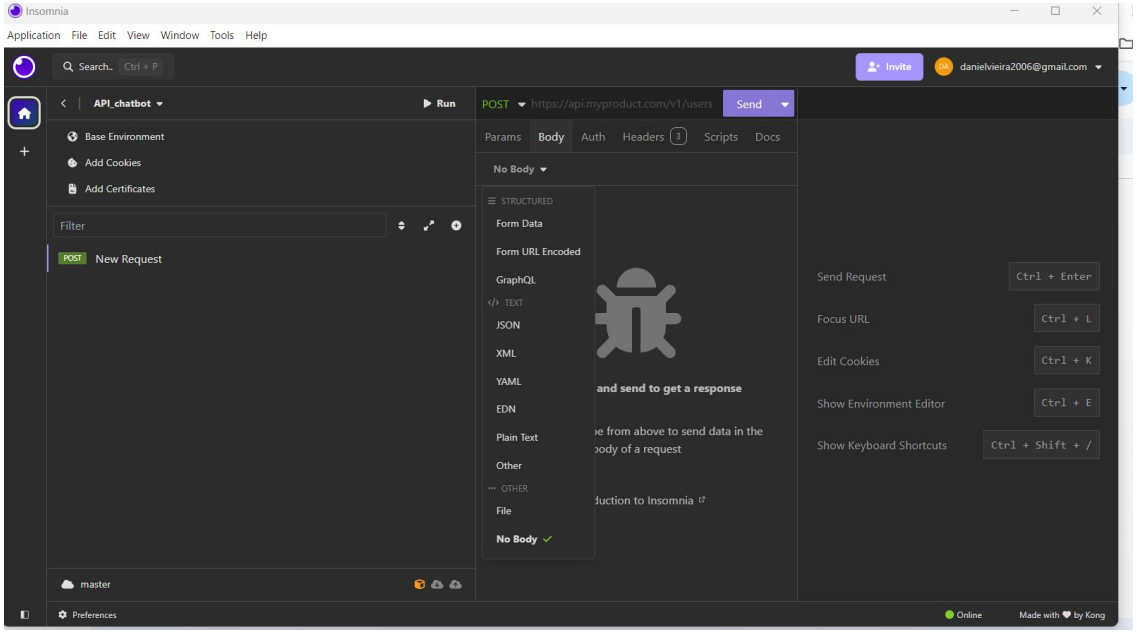
Para testar a API será utilizado o Insomnia

Clicar em New HTTP request



Para testar a API será utilizado o Insomnia

O método http escolher o método post e utilizar o ip http://127.0.0.1:8000/conversar



Para testar a API será utilizado o Insomnia

O método http escolher o método post e utilizar o ip
http://127.0.0.1:8000/conversar

Para testar a API será utilizado o Insomnia

Insomnia

Application File Edit View Window Tools Help

Search.. Ctrl + P

Invite

danielvieira2006@gmail.com

API_chatbot

Run

Base Environment

Add Cookies

Add Certificates

Filter

POST New Request

POST http://127.0.0.1:8000/conversar

Send

200 OK 2.41 s 138 B Just Now

Params Body Auth Headers 4 Scripts Doc

JSON

```
1 {
2   "mensagem":"ola"
3 }
```

Preview Headers 4 Cookies Tests 0 / 0 → Mock

Preview

```
1 {
2   "resposta": "Olá! Você está procurando por informações sobre nossos cafés ou gostaria de fazer uma compra? Estou aqui para ajudar!"
3 }
```

master

Beautify JSON

$.store.books[*].author

Preferences

Online

Made with ♥ by Kong

Para testar a API será utilizado o Insomnia

Insomnia

Application File Edit View Window Tools Help

Search.. Ctrl + P

Invite

danielvieira2006@gmail.com

API_chatbot

Run

Base Environment

Add Cookies

Add Certificates

Filter

POST New Request

POST http://127.0.0.1:8000/conversar

Send

200 OK 2.72 s 295 B Just Now

Params Body Auth Headers 4 Scripts Doc

JSON

```
1 {
2   "mensagem":"Quais produtos são vendidos no site da Serenatto?"
3 }
```

Preview Headers 4 Cookies Tests 0 / 0 → Mock

Preview

```
1 {
2   "resposta": "A Serenatto é uma loja online especializada na venda de grãos de cafés especiais, oferecendo grãos torrados, provenientes de fazendas selecionadas de Minas Gerais. Todos os nossos cafés possuem nota SCA acima de 80, garantindo alta qualidade e perfis sensoriais excepcionais."
3 }
```

master

Beautify JSON

$.store.books[*].author

Preferences

Online

Made with ♥ by Kong

Para testar a API será utilizado o Insomnia

Insomnia

Application File Edit View Window Tools Help

Search.. Ctrl + P

Invite

danielvieira2006@gmail.com

API_chatbot

Run

Base Environment

Add Cookies

Add Certificates

Filter

POST New Request

POST http://127.0.0.1:8000/conversar

Send

200 OK 1.93 s 263 B Just Now

Params Body Auth Headers 4 Scripts Doc

JSON

```
1 {
2   "mensagem":"De onde vêm os grãos de café da Serenatto?"
3 }
```

Preview Headers 4 Cookies Tests 0 / 0 → Mock

Preview

```
1 {
2   "resposta": "Nossos grãos são cuidadosamente selecionados de fazendas localizadas em Minas Gerais, conhecidas pela alta qualidade de seus cafés especiais. Cada fazenda é escolhida por suas práticas sustentáveis e pelo cuidado excepcional com a produção."
3 }
```

master

Beautify JSON

$.store.books[*].author

Preferences

Online

Made with ♥ by Kong

Para testar a API será utilizado o Insomnia

Insomnia

Application File Edit View Window Tools Help

Search.. Ctrl + P

Invite danielvieira2006@gmail.com

API_chatbot Run

Base Environment

Add Cookies

Add Certificates

Filter

POST New Request

POST http://127.0.0.1:8000/conversar Send

Params Body Auth Headers 4 Scripts Doc

JSON

```
1 {
2   "mensagem":"Quais são as variedades de café disponíveis?"
3 }
```

200 OK 3.21 s 774 B Just Now

Preview Headers 4 Cookies Tests 0 / 0 → Mock

Preview

```
1 {
2   "resposta": "Excelente pergunta! Nós da Serenatto oferecemos variedades de café especiais, cada uma com seu perfil sensorial único. Atualmente, temos as seguintes variedades disponíveis:\n\n* Bourbon Vermelho: Doçura intensa, corpo encorpado, notas de chocolate.\n* Bourbon Amarelo: Acidez vibrante, notas cítricas de maracujá, ameixa e nêspera.\n* Blend Especial: Mistura do Bourbon Vermelho e Bourbon Amarelo, gerando equilíbrio entre doçura e acidez, com notas de caramelo e chocolate.\n* Catuaí Amarelo: Perfil exótico, que carrega uma expressiva acidez e notas de acerola e pitanga.\n* Geisha: Aroma floral, notas de jasmim e mamão papaya.\n\nEssas variedades são cuidadosamente selecionadas para oferecer uma experiência sensorial única e agradável."
3 }
```

master

Beautify JSON

$.store.books[*].author

Preferences

Online

Made with ♥ by Kong

Para testar a API será utilizado o Insomnia

Insomnia

Application File Edit View Window Tools Help

Search.. Ctrl + P

Invite

danielvieira2006@gmail.com

API_chatbot

Run

Base Environment

Add Cookies

Add Certificates

Filter

POST New Request

POST http://127.0.0.1:8000/conversar

Send

200 OK 2.82 s 635 B Just Now

Params Body Auth Headers 4 Scripts Doc

JSON

```
1 {
2   "mensagem":"qual o preço deles?"
3 }
```

Preview Headers 4 Cookies Tests 0 / 0 → Mock

Preview

```
1 {
2   "resposta": "Excelente pergunta! Os preços
das nossas variedades de café são os
seguintes:\n\n* Bourbon Vermelho: R$ 41,00\n*
Bourbon Amarelo: R$ 43,00\n* Blend Especial:
R$ 37,50\n* Catuaí Amarelo: R$ 55,00\n*
Geisha: R$ 105,00\n* Yirgachef fe: R$
110,00\n\nÉ importante notar que os preços são
por pacote de 250g de café torrado. Nós
acreditamos que a qualidade excepcional dos
nossos cafés justifica os preços competitivos
que oferecemos. Além disso, estamos sempre
trabalhando para melhorar a experiência do
nosso cliente, então não hesite em nos
contatar se tiver alguma dúvida ou precisar de
mais informações!"
3 }
```

master

Beautify JSON

$.store.books[*].author

Preferences

Online

Made with ❤ by Kong

Para testar a API será utilizado o Insomnia

Insomnia

Application File Edit View Window Tools Help

Search.. Ctrl + P

Invite

danielvieira2006@gmail.com

API_chatbot

Run

Base Environment

Add Cookies

Add Certificates

Filter

POST New Request

POST http://127.0.0.1:8000/conversar

Send

200 OK 2.82 s 635 B Just Now

Params Body Auth Headers 4 Scripts Doc

JSON

```
1 {
2   "mensagem":"qual o preço deles?"
3 }
```

Preview Headers 4 Cookies Tests 0 / 0 → Mock

Preview

```
1 {
2   "resposta": "Excelente pergunta! Os preços
das nossas variedades de café são os
seguintes:\n\n* Bourbon Vermelho: R$ 41,00\n*
Bourbon Amarelo: R$ 43,00\n* Blend Especial:
R$ 37,50\n* Catuaí Amarelo: R$ 55,00\n*
Geisha: R$ 105,00\n* Yirgachef fe: R$
110,00\n\nÉ importante notar que os preços são
por pacote de 250g de café torrado. Nós
acreditamos que a qualidade excepcional dos
nossos cafés justifica os preços competitivos
que oferecemos. Além disso, estamos sempre
trabalhando para melhorar a experiência do
nosso cliente, então não hesite em nos
contatar se tiver alguma dúvida ou precisar de
mais informações!"
3 }
```

master

Beautify JSON

$.store.books[*].author

Preferences

Online

Made with ❤ by Kong

Para testar a API será utilizado o Insomnia

2. Você está testando em um celular ou emulador?

- **Emulador Android:** use `http://10.0.2.2:8000`

- **Celular físico:** use o **IP local da sua máquina**, exemplo: `http://192.168.0.108:8000`
  💡 Você pode descobrir seu IP local com:

```
ipconfig (Windows)
ifconfig (Linux/Mac)
```

3. Você permitiu conexões externas no FastAPI?

Por padrão, o `uvicorn` escuta **apenas `127.0.0.1`**.

👉 Para permitir conexões do celular ou emulador, use:

```
uvicorn main:app --host 0.0.0.0 --port 8000 --reload
```

Para testar a API será utilizado o Insomnia

2. Assim que terminar, rode o comando:

bash Copiar Editar

```bash
docker run -p 8000:8000 -e GROQ_API_KEY=sua-chave chatbot-agro-full
```

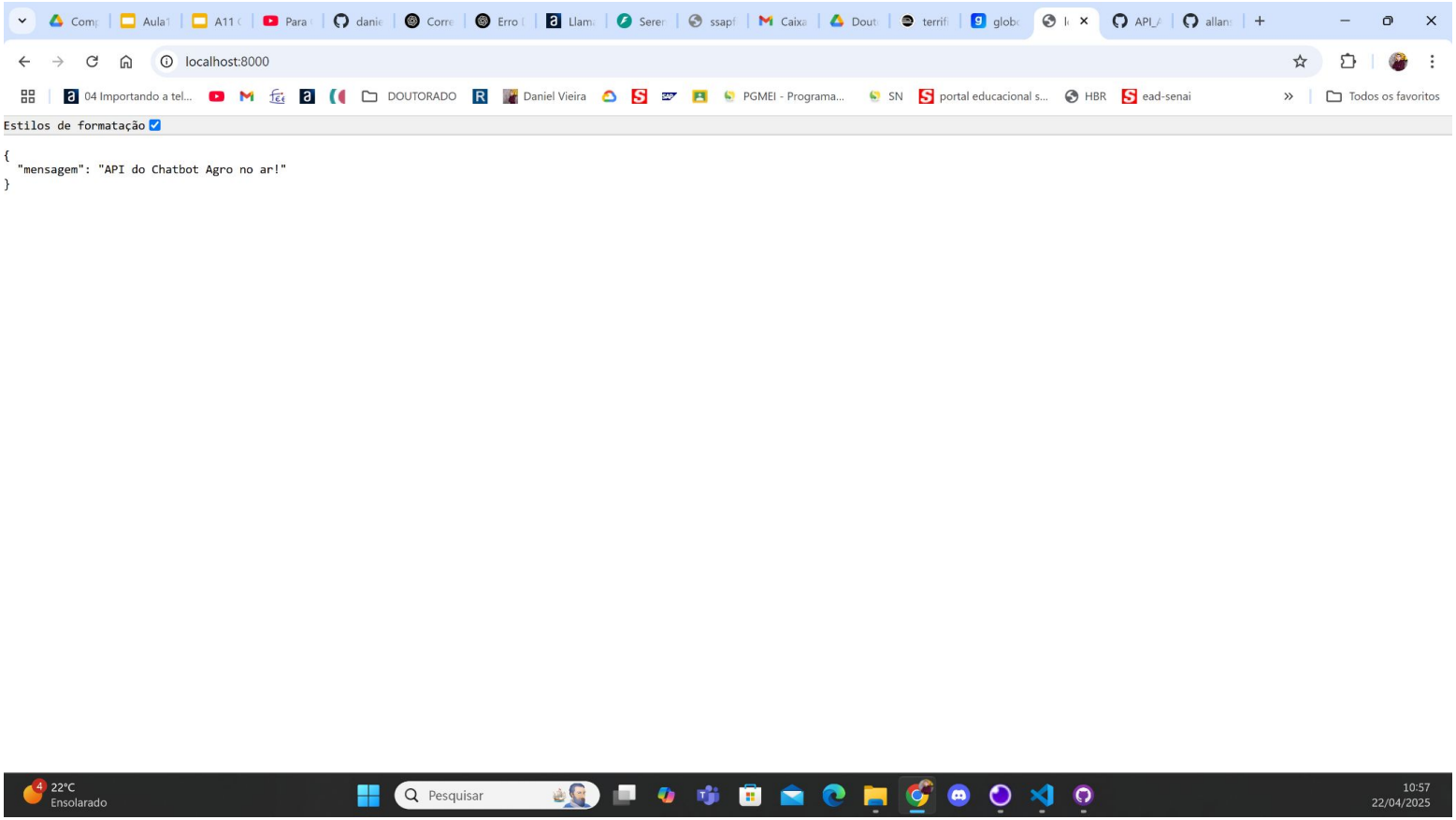
3. Testa no Insomnia ou no navegador em `http://localhost:8000/docs`.

Para testar a API será utilizado o Insomnia

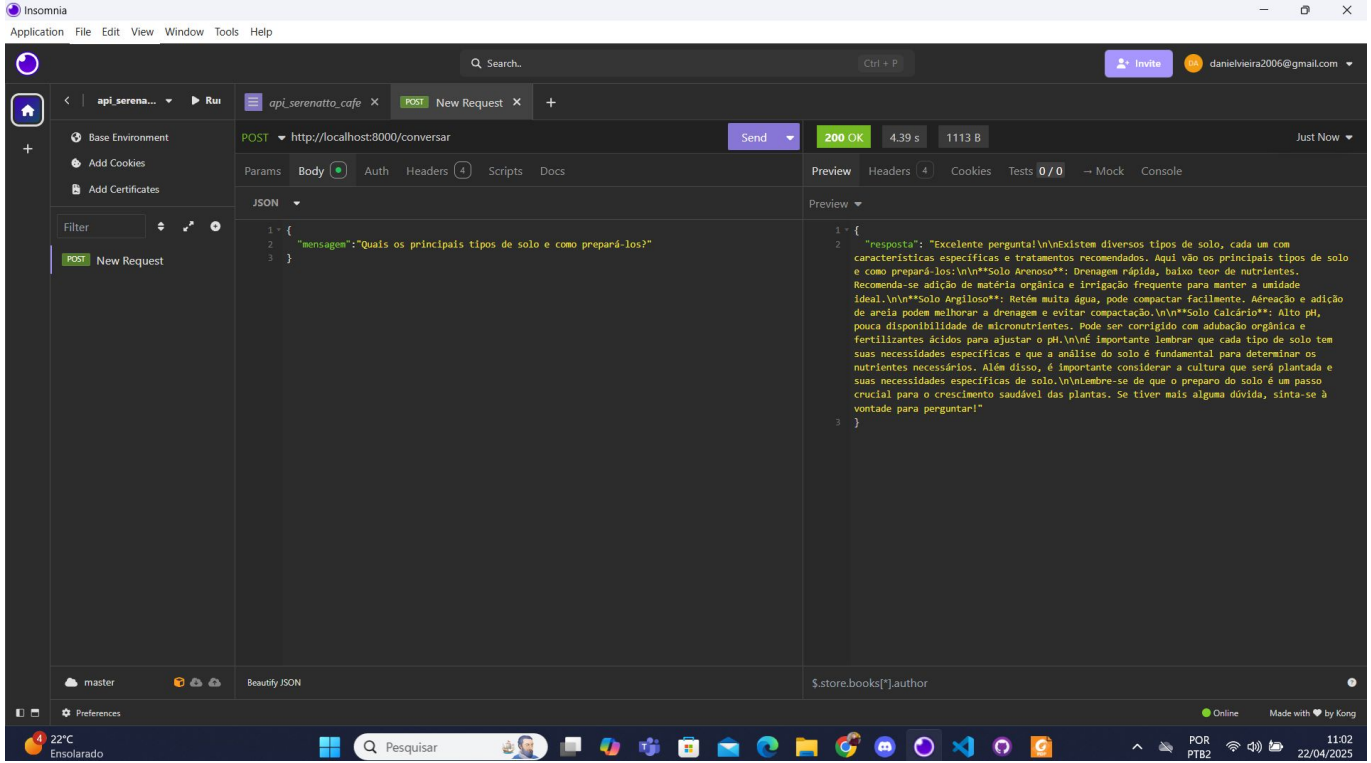
PROBLEMS 1 OUTPUT TERMINAL PORTS ESP-IDF MEMORY XRTOS SERIAL MONITOR DEBUG CONSOLE

```
<none>                                  <none>    6ec1ee5b654b   24 hours ago     17GB
<none>                                  <none>    074649510875   34 hours ago     17GB
<none>                                  <none>    0a94be34fc70   35 hours ago     1.96GB
<none>                                  <none>    1246a9fc57f3   35 hours ago     2.12GB
<none>                                  <none>    b83c4da03a05   35 hours ago     2.12GB
<none>                                  <none>    78d3dd107d39   36 hours ago     2.12GB
<none>                                  <none>    42e326facd59   36 hours ago     2.07GB
<none>                                  <none>    e8042c868182   36 hours ago     2.04GB
<none>                                  <none>    c1277f8fad24   36 hours ago     2.12GB
serenatto-chatbot                       latest    06fff8904d08   36 hours ago     2.09GB
chatbot-bitdoglab_v2                    latest    6889e315381a   43 hours ago     35.3GB
chatbot_bitdog_huggingface-chatbot      latest    8b7d63303b19   2 days ago       20.4GB
flask-app                               latest    2d4ceec7bea9   2 weeks ago      1.49GB
ml-env                                  latest    7d89f3587381   2 weeks ago      1.64GB
hello-world                             latest    7e1a4e2d11e2   3 months ago     20.4kB
PS D:\SENAI\2025-1\ChatBot\chatbot_api-agricola> docker run -p 8000:8000 -e GROQ_API_KEY=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 api_agro
```

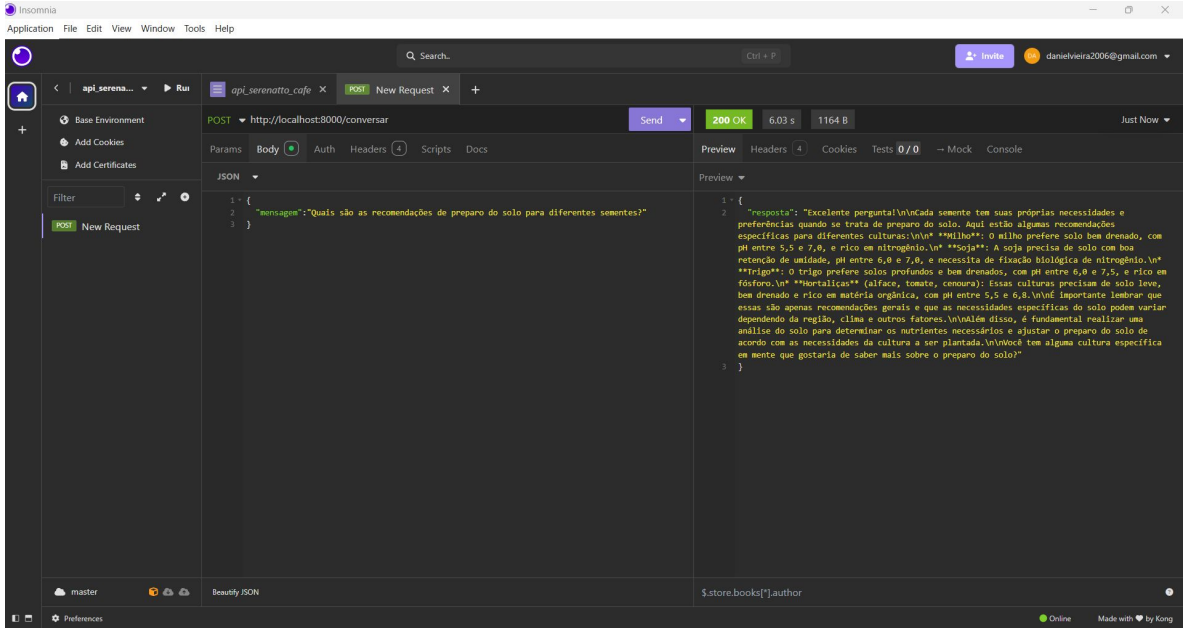
Para testar a API será utilizado o Insomnia



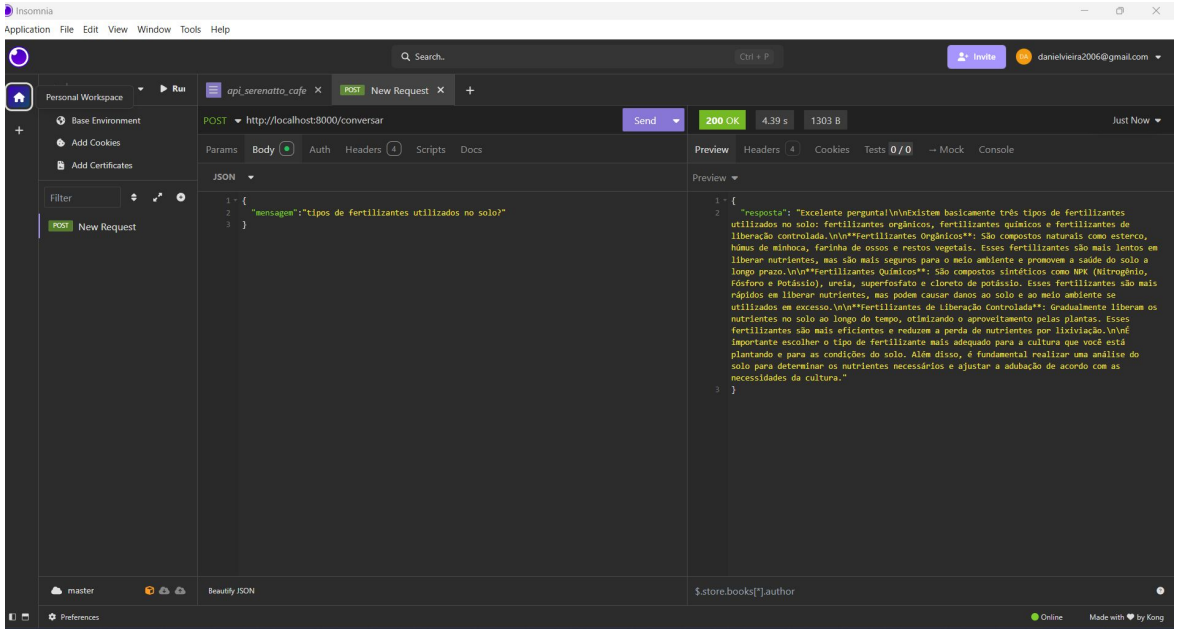
Para testar a API será utilizado o Insomnia



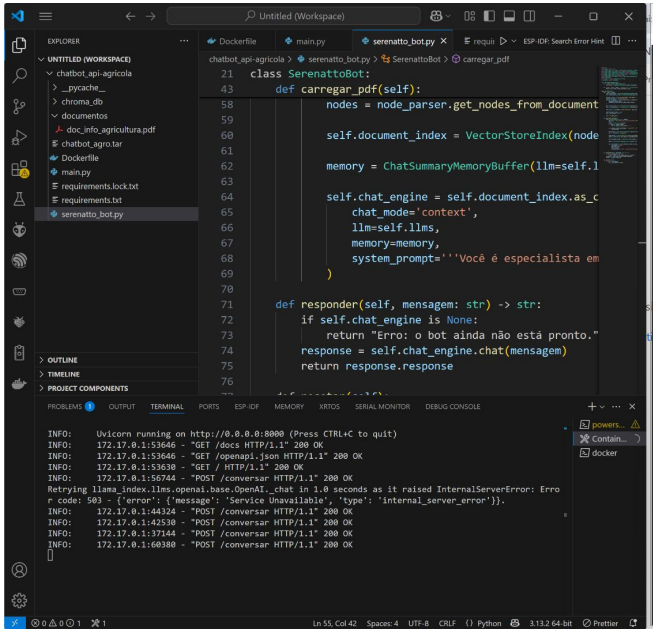
Para testar a API será utilizado o Insomnia

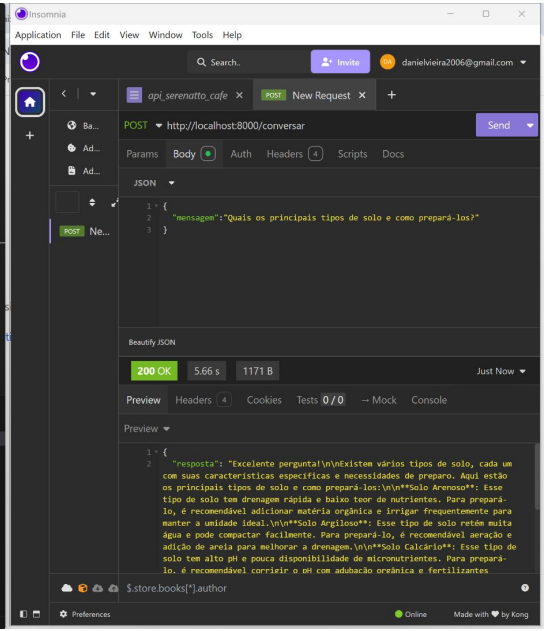


Para testar a API será utilizado o Insomnia



Para testar a API será utilizado o Insomnia





Para testar a API será utilizado o Insomnia

Comando para exportar a imagem.tar
docker save -o api_agro.tar api_agro

```
PROBLEMS 1   OUTPUT   TERMINAL   PORTS   ESP-IDF   MEMORY   XRTOS   SERIAL MONITOR   DEBUG CONSOLE

INFO:     172.17.0.1:54882 - "POST /conversar HTTP/1.1" 200 OK
INFO:     172.17.0.1:35736 - "POST /conversar HTTP/1.1" 200 OK
INFO:     172.17.0.1:50252 - "POST /conversar HTTP/1.1" 200 OK
INFO:     172.17.0.1:60902 - "POST /conversar HTTP/1.1" 200 OK
INFO:     Shutting down
INFO:     Waiting for application shutdown.
INFO:     Application shutdown complete.
INFO:     Finished server process [1]
PS D:\SENAI\2025-1\ChatBot\chatbot_api-agricola> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS D:\SENAI\2025-1\ChatBot\chatbot_api-agricola> docker save -o api_agro.tar api_agro
```

Obrigado!

Prof. Me Daniel Vieira

Email: danielvieira2006@gmail.com

Linkedin: Daniel Vieira

Instagram: Prof daniel.vieira95