



Computação em  
Nuvem  
Aula 13

Prof. Me Daniel Vieira



# Agenda





- 1 - ChatBot Indústria 4.0
- 2 - Comandos Docker
- 3 - Estrutura do projeto
- 4 - Atividade

# Atividade

- **Criando um ambiente Dockerizado para um Chatbot de Indústria 4.0 IA**

- Você foi contratado por uma startup do setor industrial chamada **IndTech Solutions**, que busca implementar soluções baseadas em inteligência artificial para a transformação digital das indústrias.

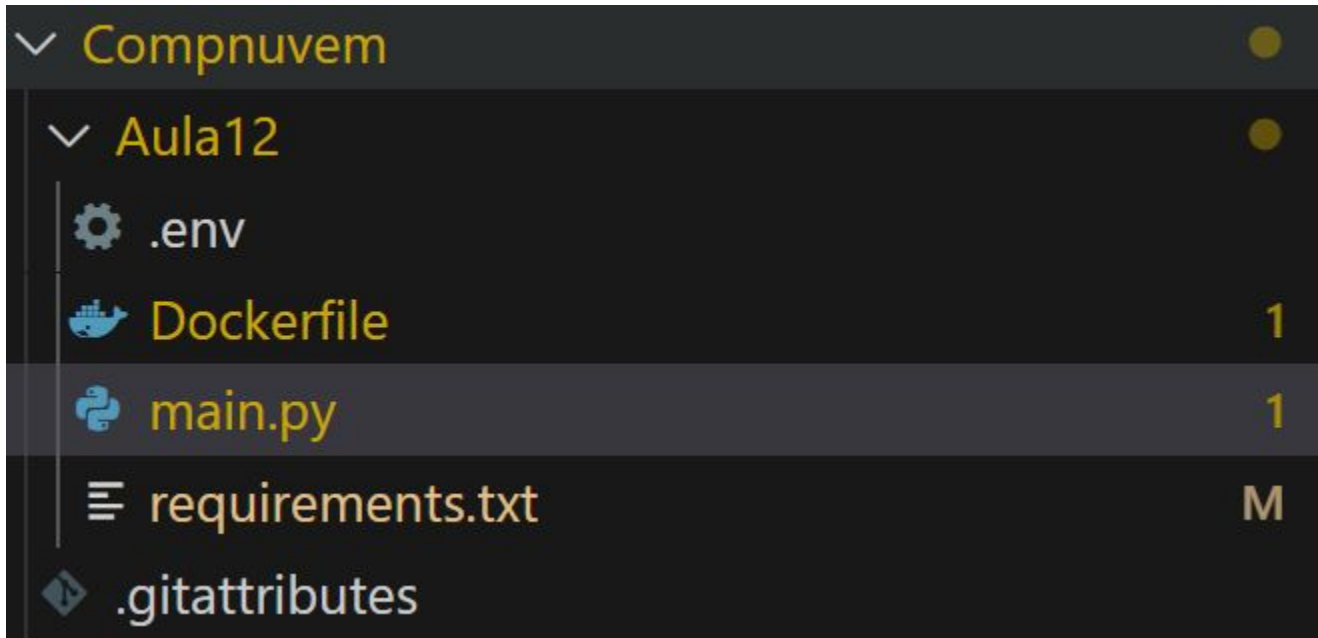
Uma das iniciativas da empresa é disponibilizar um **chatbot especializado em Indústria 4.0**, capaz de responder dúvidas comuns sobre automação, IoT, manutenção preditiva, integração de sistemas, segurança cibernética e outras práticas modernas do setor industrial.

- O objetivo da equipe de desenvolvimento é **empacotar esse chatbot em um contêiner Docker** para facilitar a implantação em diferentes ambientes — seja localmente, em servidores de nuvem, ou até mesmo em infraestruturas industriais privadas.
- Sua missão é:
-  **Criar um ambiente Docker para esse chatbot**, garantindo que ele possa ser executado com facilidade em qualquer máquina que tenha Docker instalado.
-  **Configurar as dependências** (como Gradio, Groq, Pydantic, LlamaIndex-LLMs) para que o chatbot rode perfeitamente.
-  **Utilizar variáveis de ambiente** para manter segura a chave de API utilizada na comunicação com o modelo de linguagem.
-  **Expor a interface web do Gradio**, de forma que qualquer usuário da rede possa acessá-la e interagir com o assistente de Indústria 4.0 IA.

# Perguntas para o chatbot

- 1) Diferença entre a comunicação RS232 e RS485 ?
- 2) Qual protocolo de comunicação é utilizado no ambiente industrial para comunicação entre máquinas menos suscetível a EMC e EMI?
- 3) Defina protocolo Hart
- 4) O que é OPC-UA ?
- 5) Características do protocolo IO-Link ?
- 6) Defina o protocolo CAN ?
- 7) Quais as características do protocolo Modbus ?

# Estrutura do projeto



# Comandos Docker

## 1. Carregar imagem a partir do arquivo `.tar`

bash

 Copiar


 Editar

```
docker load -i nome_da_imagem.tar
```

### Exemplo:

bash

 Copiar

 Editar

```
docker load -i chatbot_bitdog_huggingface-chatbot.tar
```

# Comandos Docker

## 2. Verificar se a imagem foi carregada com sucesso

bash

 Copiar

 Editar

```
docker images
```

## 3. Rodar um container a partir da imagem carregada

bash

 Copiar

 Editar

```
docker run -it --rm -p 7860:7860 nome_da_imagem
```

# Comandos Docker

Se quiser exportar de outro computador, use:

bash

 Copiar

 Editar

```
docker save -o chatbot_bitdog_huggingface-chatbot.tar chatbot_bitdog_huggingface-chatbot
```



# Comandos Docker

Se quiser rodar em background, com nome fixo para o container:

bash

 Copiar

 Editar

```
docker run -d --name chatbot_container -p 7860:7860 chatbot_bitdog_huggingface-chatbot
```

Se quiser que ela reinicie sempre:

bash

 Copiar

 Editar

```
docker run -d --restart always --name chatbot_container -p 7860:7860 chatbot_bitdog_huggingface-cl
```

# Comandos Docker

## Terminal

```
Run 'docker run --help' for more information
PS C:\Users\dsadm\Desktop\compnuvem-b> docker run -p 7860:7860 chat_agro_docker
Traceback (most recent call last):
  File "/app/main.py", line 10, in <module>
    raise ValueError("❌ variável de ambiente GROQ_API_KEY não foi definida!")
ValueError: ❌ variável de ambiente GROQ_API_KEY não foi definida!
PS C:\Users\dsadm\Desktop\compnuvem-b> docker run -p 7860:7860 GROQ_API_KEY=gsk_D6qheWgXIaQ5jl3Pu8LNWGdyb3FYJXU0RvNNNoIpEKV1NreqLAFnf chat_agro_docker
docker: invalid reference format: repository name (library/GROQ_API_KEY=gsk_D6qheWgXIaQ5jl3Pu8LNWGdyb3FYJXU0RvNNNoIpEKV1NreqLAFnf) must be lowercase

Run 'docker run --help' for more information
PS C:\Users\dsadm\Desktop\compnuvem-b> docker run -p 7860:7860 -e GROQ_API_KEY=gsk_D6qheWgXIaQ5jl3Pu8LNWGdyb3FYJXU0RvNNNoIpEKV1NreqLAFnf chat_agro_docker
```

comando para executar o container `docker run -p 7860:7860 -e GROQ_API_KEY = gsk_D6qheWgXIaQ5jl3Pu8LNWGdyb3FYJXU0RvNNNoIpEKV1NreqLAFnf chat_agro_docker`

# Comandos Docker

docker ps - mostrar os containers ativos

```
PS D:\SENAI\2025-1\Compnuvem> docker ps
CONTAINER ID   IMAGE             COMMAND                  CREATED        STATUS        PORTS                    NAMES
bda01b113185   chat_agro_docker  "python main.py"        13 minutes ago Up 13 minutes  0.0.0.0:7860->7860/tcp   nice_matsumoto
PS D:\SENAI\2025-1\Compnuvem> docker stop bda01b113185
```

# Comandos Docker

`docker ps -a` mostra todos os containers que existem na máquina

```
PS D:\SENAI\2025-1\Compuvem> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
bda01b113185	chat_agro_docker	"python main.py"	17 minutes ago	Exited (137) 2 minutes ago		nice_matsumoto
7a1d09ddc1b4	7f63d9775708	"python main.py"	31 minutes ago	Exited (1) 30 minutes ago		brave_golick
b73e9469eccc	a9af19b75edf	"python main.py"	About an hour ago	Exited (0) 56 minutes ago		compassionate_faraday
ba2e174fe941	4724d598756e	"python app.py"	About an hour ago	Exited (2) About an hour ago		clever_robinson
995e2fafedae	4724d598756e	"python app.py"	About an hour ago	Exited (2) About an hour ago		determined_hodgkin
840932d05b8f	4724d598756e	"python app.py"	About an hour ago	Exited (2) About an hour ago		recursing_golick
94ed14da79ac	api_agro	"uvicorn main:app --..."	2 hours ago	Exited (0) 2 hours ago		bold_darwin
4253f830ebee	api_agro	"uvicorn main:app --..."	2 hours ago	Created		peaceful_heyrovsky
d5ac44497f61	api_agro	"uvicorn main:app --..."	2 hours ago	Created		focused_hoover
2b898306b214	api_agro	"uvicorn main:app --..."	7 hours ago	Exited (137) 2 hours ago		stoic_driscoll
be0a9814480b	ac445792b0af	"uvicorn main:app --..."	16 hours ago	Exited (0) 16 hours ago		magical_murdock
cac1945a94df	39aa5290c050	"uvicorn main:app --..."	21 hours ago	Exited (1) 21 hours ago		dazzling_lamarr
6b14b38c4d51	chatbot_api-docker-serenatto-api	"uvicorn main:app --..."	23 hours ago	Exited (137) 22 hours ago		serenatto-api_vteste
25d6980ad59f	serenatto-api v14	"uvicorn main:app --..."	28 hours ago	Exited (137) 27 hours ago		thirsty_solomon

# Comandos Docker

docker images - mostra a imagem de todos os containers existentes

```
PS D:\SENAI\2025-1\Compuvem> docker images
```

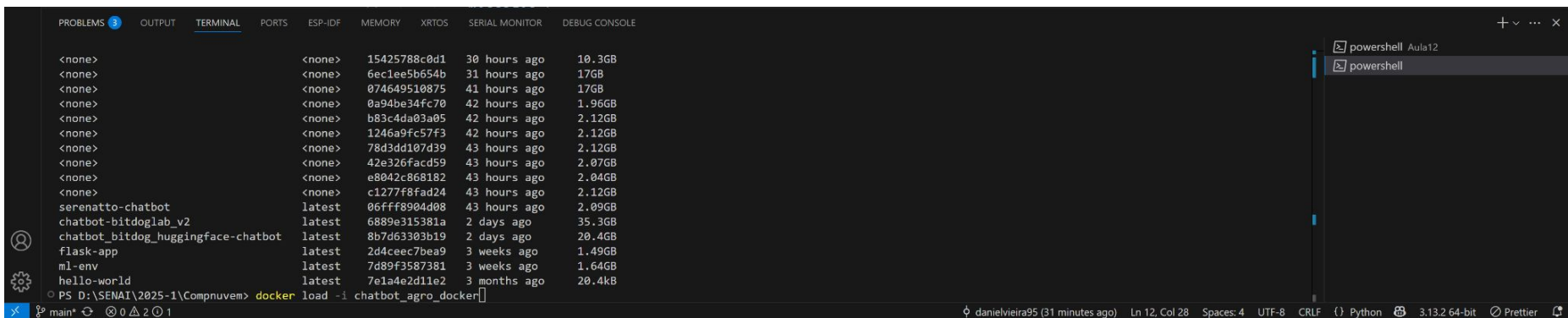
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
chat_agro_docker	latest	2405b49c1651	27 minutes ago	1.08GB
api_agro	latest	06675ec2bda2	7 hours ago	34GB
chatbot_api-docker-serenatto-api	latest	17d73a9f06b6	23 hours ago	10GB
serenatto-api_v14	latest	d47a1feb5ba1	29 hours ago	9.76GB
serenatto-api_v13	latest	d612f274547b	29 hours ago	9.02GB
<none>	<none>	15425788c0d1	30 hours ago	10.3GB
<none>	<none>	6ec1ee5b654b	31 hours ago	17GB
<none>	<none>	074649510875	41 hours ago	17GB
<none>	<none>	0a94be34fc70	42 hours ago	1.96GB
<none>	<none>	b83c4da03a05	42 hours ago	2.12GB
<none>	<none>	1246a9fc57f3	42 hours ago	2.12GB
<none>	<none>	78d3dd107d39	43 hours ago	2.12GB
<none>	<none>	42e326facd59	43 hours ago	2.07GB
<none>	<none>	e8042c868182	43 hours ago	2.04GB
<none>	<none>	c1277f8fad24	43 hours ago	2.12GB

main\* 0 0 2 1

danielvieira95 (31 minutes ago) Ln 12, Col 28 Spaces: 4 UTF-8 CRLF {} Python 3.13.2 64-bit Prettier

# Comandos Docker

`docker load -i chatbot_agro_docker` - carrega a imagem .tar criada, utilizado para carregar uma imagem de outro computador



The screenshot shows the Visual Studio Code interface with the Docker extension. The 'TERMINAL' tab is active, displaying a list of Docker images. The 'OUTPUT' tab is also visible, showing the command `docker load -i chatbot_agro_docker` being executed. On the right, a PowerShell terminal window is open, titled 'powershell Aula12'.

Image Name	Architecture	Size	Created
<none>	<none>	15425788c0d1	30 hours ago
<none>	<none>	6ec1ee5b654b	31 hours ago
<none>	<none>	074649510875	41 hours ago
<none>	<none>	0a94be34fc70	42 hours ago
<none>	<none>	b83c4da03a05	42 hours ago
<none>	<none>	1246a9fc57f3	42 hours ago
<none>	<none>	78d3dd107d39	43 hours ago
<none>	<none>	42e326facd59	43 hours ago
<none>	<none>	e8042c868182	43 hours ago
<none>	<none>	c1277f8fad24	43 hours ago
serenatto-chatbot	latest	06fff8904d08	43 hours ago
chatbot-bitdoglab_v2	latest	6889e315381a	2 days ago
chatbot_bitdog_huggingface-chatbot	latest	8b7d63303b19	2 days ago
flask-app	latest	2d4ceec7bea9	3 weeks ago
ml-env	latest	7d89f3587381	3 weeks ago
hello-world	latest	7e1a4e2d11e2	3 months ago

```
PS D:\SENAI\2025-1\Compuvum> docker load -i chatbot_agro_docker
```

# Atividade

Requirements.txt

gradio==5.22.0

llama-index-llms-groq==0.3.1

pydantic==2.10.6

groq

# Atividade

## Arquivo Dockerfile

# Usa imagem base oficial do Python

FROM python:3.11-slim

# Evita prompts interativos

ENV DEBIAN\_FRONTEND=noninteractive

# Define o diretório de trabalho

WORKDIR /app

# Copia os arquivos da aplicação

COPY . .

# Atualiza pip e instala dependências

RUN pip install --upgrade pip && pip install --no-cache-dir -r requirements.txt

# Define a variável para o Gradio escutar em todas as interfaces

ENV GRADIO\_SERVER\_NAME=0.0.0.0

# Expõe a porta do Gradio

EXPOSE 7860

# Comando para rodar o app

CMD ["python", "main.py"]



# Atividade - Código app.py

```
import os # Biblioteca para interagir com o sistema operacional
import gradio as gr # Interface web simples
from groq import Groq # Cliente da API Groq

# Carrega a chave da API Groq da variável de ambiente
GROQ_API_KEY = os.getenv('GROQ_API_KEY')

# Valida se a chave foi fornecida
if not GROQ_API_KEY:
    raise ValueError("❌ A variável de ambiente GROQ_API_KEY não foi definida!")

# Inicializa o cliente Groq
client = Groq(api_key=GROQ_API_KEY)
```

# Atividade

```
def assistente_industria(user_prompt):  
    if user_prompt.strip() == "15":  
        return "Encerrando assistente Indústria 4.0! Até mais! 🏭"  
    completion = client.chat.completions.create(  
        model="llama3-8b-8192",  
        messages=[  
            {"role": "system", "content": "Você é um assistente especializado em Indústria 4.0. Responda perguntas relacionadas a automação,  
IoT industrial, manutenção preditiva, integração de sistemas, robótica avançada, análise de dados industriais e boas práticas de  
digitalização no setor industrial."},  
            {"role": "user", "content": user_prompt}  
        ],  
        temperature=0,  
        max_tokens=1024,  
        top_p=1,  
        stream=False  
    )  
  
    return completion.choices[0].message.content
```

# Atividade

# Interface Gradio

```
iface = gr.Interface(  
    fn=assistente_Industria 4.0,  
    inputs=gr.Textbox(lines=2, placeholder="Digite sua pergunta sobre agricultura..."),  
    outputs="text",  
    title= " 🤖 Assistente Industria 4.0",  
    description="Digite sua pergunta sobre industria 4.0 e receba respostas de IA especializadas! ,  
    live=True  
)
```

# Executa a interface web

```
if __name__ == "__main__":  
    iface.launch()
```

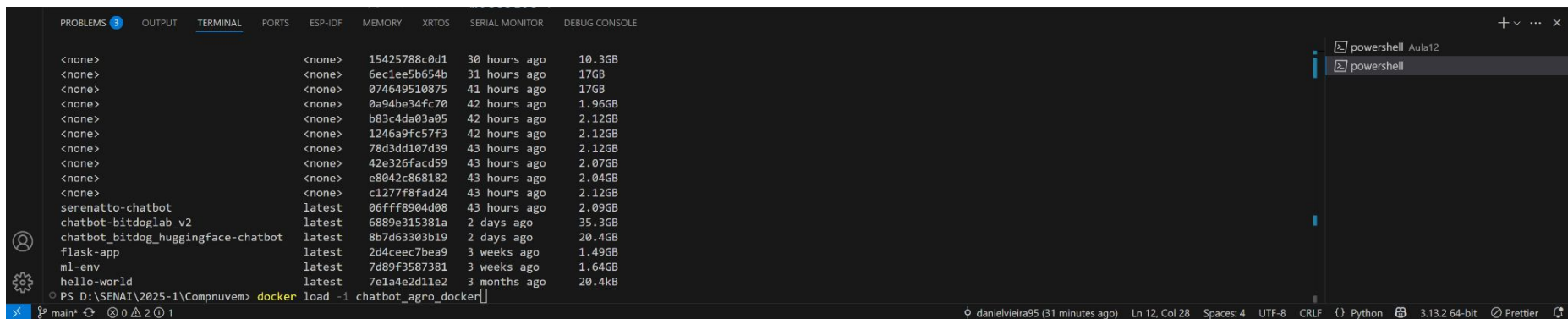
# Atividade

Criando o container chamado chat\_ind4\_docker

# Atividade

Criando a image.tar do container do chat agro

`docker save -o chat_agro_docker.tar chat_agro_docker`



The screenshot shows the Visual Studio Code interface. The 'TERMINAL' tab is active, displaying a list of Docker images. The list includes various images like 'serenatto-chatbot', 'chatbot-bitdoglab\_v2', 'chatbot\_bitdog\_huggingface-chatbot', 'flask-app', 'ml-env', and 'hello-world'. The 'chatbot\_agro\_docker' image is highlighted. To the right, a PowerShell terminal window titled 'powershell Aula12' is open.

Image Name	Architecture	Size	Created
<none>	<none>	15425788c0d1	30 hours ago
<none>	<none>	6ec1ee5b654b	31 hours ago
<none>	<none>	074649510875	41 hours ago
<none>	<none>	0a94be34fc70	42 hours ago
<none>	<none>	b83c4da03a05	42 hours ago
<none>	<none>	1246a9fc57f3	42 hours ago
<none>	<none>	78d3dd107d39	43 hours ago
<none>	<none>	42e326facd59	43 hours ago
<none>	<none>	e8042c868182	43 hours ago
<none>	<none>	c1277f8fad24	43 hours ago
serenatto-chatbot	latest	06fff8904d08	43 hours ago
chatbot-bitdoglab_v2	latest	6889e315381a	2 days ago
chatbot_bitdog_huggingface-chatbot	latest	8b7d63303b19	2 days ago
flask-app	latest	2d4ceec7bea9	3 weeks ago
ml-env	latest	7d89f3587381	3 weeks ago
hello-world	latest	7e1a4e2d11e2	3 months ago

PS D:\SENAI\2025-1\Compnuvem> docker load -i chatbot\_agro\_docker

Carregando a imagem gerada

`docker load -i chat_agro_docker.tar chat_agro_docker`

# Atividade

Criando a image.tar do container do chat agro

```
docker save -o chat_agro_docker.tar chat_agro_docker
```

```
docker save -o chat_agro_docker.tar chat_agro_docker
```

- `-o` = **output** (nome do arquivo .tar que será criado)
- `chat_agro_docker` = nome da imagem no seu Docker local

# Atividade

Carregando a image.tar do container do chat agro

`docker load -i chat_agro_docker.tar`

```
docker load -i chat_agro_docker.tar
```

- `docker load` : carrega uma imagem Docker salva anteriormente em um arquivo `.tar`
- `-i` : é o shorthand para `--input` , ou seja, o caminho do arquivo `.tar` contendo a imagem
- `chat_agro_docker.tar` : é o nome do arquivo da imagem que você exportou com `docker save`

# Atividade

Comandos para criar a imagem e executar o Docker

# Acesse a pasta do projeto

```
cd Compnuvem
```

Aula13

# Build da imagem

```
docker build -t chat_ind4_docker .
```

# Executa o container (com a API Key como variável de ambiente)

```
docker run -p 7860:7860 --env-file .env chat_ind4_docker
```



# Atividade

Criar um relatório em Word documentando o passo a passo da atividade, explicando e tirar print das telas.

- 1) Porque o Docker consegue simplificar o desenvolvimento de aplicações ?
- 2) Qual o comando utilizado para criar a imagem do Docker e qual comando é utilizado para executar a imagem do Docker ?
- 3) Cite a principal vantagem de se utilizar Docker nas aplicações desenvolvidas ?

# Obrigado!

Prof. Me Daniel Vieira

Email: [danielvieira2006@gmail.com](mailto:danielvieira2006@gmail.com)

Linkedin: Daniel Vieira

Instagram: Prof daniel.vieira95

