AI ENGINEERING

Docker + Ollama in 30 Minuten

Dein lokaler LLM-Stack, DSGVO-konform

Extrakt aus: **Der Lokale AI-Stack — Playbook**

ai-engineering.at

Seite 1: Deckblatt

```
AI ENGINEERING
_____________

Docker + Ollama in 30 Minuten
Dein lokaler LLM-Stack, DSGVO-konform


Extrakt aus: Der Lokale AI-Stack – Playbook
ai-engineering.at
_____________
```

Seite 2: Was du in 30 Minuten aufgebaut haben wirst

Ziel: Am Ende dieser Anleitung läuft auf deinem Rechner: - Docker Desktop (oder Docker Engine auf Linux/Mac) - Ollama in einem Docker-Container - Llama 3.1 8B — ein vollwertiger, lokaler LLM - Eine API-Schnittstelle die du sofort nutzen kannst

Voraussetzungen: - Windows 10/11, macOS 12+, oder Ubuntu 22.04+ - Mindestens 8 GB RAM (16 GB empfohlen) - Mindestens 10 GB freier Speicherplatz - GPU optional (Ollama läuft auch auf CPU, aber langsamer)

Warum lokal?

> *Jede Anfrage an ChatGPT/OpenAI verarbeitet deine Daten auf US-amerikanischen Servern. Art. 13 DSGVO verlangt, dass du weißt wo deine Daten sind. Lokale AI = Daten bleiben bei dir.*

Seite 3: Schritt 1 — Docker installieren (5 Min)

Windows/Mac: 1. Docker Desktop herunterladen: docs.docker.com/desktop 2. Installer ausführen, Standardoptionen beibehalten 3. Docker Desktop starten

Ubuntu/Debian:

```
curl -fsSL https://get.docker.com | sh
sudo usermod -aG docker $USER
newgrp docker
docker --version
```

Verifizierung:

```
docker run hello-world
# Erwartete Ausgabe: "Hello from Docker!"
```

NVIDIA GPU (optional): Falls du eine NVIDIA GPU hast und schnellere Inferenz willst:

```
# NVIDIA Container Toolkit installieren
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --
dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg
# (vollständige Anleitung: docs.nvidia.com/datacenter/cloud-native/container-
toolkit)
```

Seite 4: Schritt 2 — Ollama starten (3 Min)

docker-compose.yml erstellen:

```
version: '3.8'

services:
  ollama:
    image: ollama/ollama:latest
    container_name: ollama
    restart: unless-stopped
    ports:
      - "11434:11434"
    volumes:
      - ollama_data:/root/.ollama
    # GPU-Support (optional, entfernen wenn keine NVIDIA GPU):
    deploy:
      resources:
        reservations:
          devices:
            - driver: nvidia
              count: 1
              capabilities: [gpu]

volumes:
  ollama_data:
```

Starten:

```
docker compose up -d
docker logs ollama  # Logs prüfen
```

Verifizierung:

```
curl http://localhost:11434/api/health
# Erwartete Antwort: {"status":"ok"}
```

Seite 5: Schritt 3 — Modell laden (10 Min)

Llama 3.1 8B herunterladen:

```
docker exec ollama ollama pull llama3.1:8b
# Download: ~4.7 GB, dauert je nach Verbindung 5-15 Min
```

Schnellere Alternative (3B, nur ~2 GB):

```
docker exec ollama ollama pull llama3.2:3b
```

Deutsches Modell (sehr gut für DACH-Unternehmen):

```
docker exec ollama ollama pull qwen2.5:7b
# Qwen 2.5 = exzellentes Deutsch, 4.5 GB
```

Welches Modell für was?

Modell	Größe	Stärke	RAM nötig
llama3.2:3b	2 GB	Schnell, gut für einfache Aufgaben	8 GB
llama3.1:8b	4.7 GB	Ausgewogen, beste Wahl	16 GB
qwen2.5:7b	4.5 GB	Deutsch/Englisch exzellent	16 GB

Seite 6: Schritt 4 — Erster Test (2 Min)

Via Terminal:

```
docker exec -it ollama ollama run llama3.1:8b
>>> Erkläre DSGVO in 3 Sätzen auf Deutsch
# [LLM antwortet direkt im Terminal]
>>> /bye  # Beendet Chat-Session
```

Via API (JSON):

```
curl http://localhost:11434/api/generate \
  -H "Content-Type: application/json" \
  -d '{
    "model": "llama3.1:8b",
    "prompt": "Was ist die DSGVO?",
    "stream": false
  }'
```

Erwartete Antwort:

```
{
  "model": "llama3.1:8b",
  "response": "Die DSGVO (Datenschutz-Grundverordnung) ist...",
  "done": true,
  "total_duration": 3450000000
}
```

Seite 7: Schritt 5 — Python-Integration (5 Min)

Python-Client (einfachste Variante):

```
import requests

def ask_local_llm(question: str, model: str = "llama3.1:8b") -> str:
    response = requests.post(
        "http://localhost:11434/api/generate",
        json={
            "model": model,
            "prompt": question,
            "stream": False
        },
        timeout=60
    )
    return response.json()["response"]

# Verwendung:
answer = ask_local_llm("Fasse diesen Text zusammen: ...")
print(answer)
```

Mit dem offiziellen Ollama Python-Client:

```
pip install ollama
```

```
from ollama import Client

client = Client(host='http://localhost:11434')
response = client.generate(
    model='llama3.1:8b',
    prompt='Erkläre Docker in einem Satz.'
)
print(response['response'])
```

Seite 8: Was als Nächstes? (+ CTA)

Du hast jetzt: - Einen lokalen LLM der keine Daten nach extern sendet - Eine REST-API die alle KI-Anwendungen nutzen können - Die Basis für eine vollständige DSGVO-konforme AI-Infrastruktur

Im vollständigen Playbook findest du:

Kapitel	Inhalt
Kap. 1	Hardware-Auswahl: RTX 3090 vs. 4090 vs. CPU-only
Kap. 2	**Docker + Ollama (dieses Kapitel)**
Kap. 3	Docker Swarm: Multi-Node Cluster (3 Maschinen)
Kap. 4	Hybrid RAG: ChromaDB + Neo4j + BM25
Kap. 5	Voice-Pipeline: Whisper STT + Piper TTS
Kap. 6	Monitoring: Prometheus + Grafana (36 Metriken)
Kap. 7	DSGVO-Compliance: Art. 30 Template + DPIA
Kap. 8	Multi-Agent Orchestrierung via Mattermost

Jetzt das vollständige Playbook holen:

```
EUR 49 — Einmalig, kein Abo
PDF + EPUB + Zukunfts-Updates
DSGVO Art. 30 Template inklusive

→ ai-engineering.at
```

"Der Lokale AI-Stack" — AI Engineering, 2026 Fragen? kontakt@ai-engineering.at