**🚀 Using vLLM with Local Models (Offline)**

**1. Copy model files from Hugging Face**

When you first download a model, Hugging Face stores it under:

~/.cache/huggingface/hub/

Each model repo has symlinks and blob files.  
To copy everything properly (including symlinks → actual files), use:

# Example: Copy Qwen/Qwen3-4B-AWQ to ~/models/Qwen3-4B-AWQ

mkdir -p ~/models/Qwen3-4B-AWQ

cp -rL ~/.cache/huggingface/hub/models--Qwen--Qwen3-4B-AWQ/snapshots/\*/\* ~/models/Qwen3-4B-AWQ

🔑 The -L flag ensures symlinks are followed so you get real files, not broken links.

Now your folder should contain files like:

~/models/Qwen3-4B-AWQ/

├── config.json

├── generation\_config.json

├── merges.txt

├── model.safetensors

├── tokenizer.json

├── tokenizer\_config.json

├── vocab.json

**2. Launch vLLM server**

Point vLLM directly to your local folder:

python -m vllm.entrypoints.openai.api\_server \

--host 0.0.0.0 \

--port 8002 \

--model ~/models/Qwen3-4B-AWQ \

--served-model-name Qwen3-4B-AWQ \

--dtype auto \

--quantization awq \

--max-model-len 16384 \

--gpu-memory-utilization 0.80

✅ If successful, logs will show:

INFO: Uvicorn running on http://0.0.0.0:8002 (Press CTRL+C to quit)

**3. Verify server**

Run:

curl http://127.0.0.1:8002/v1/models

You should see something like:

{

"object": "list",

"data": [

{"id": "Qwen3-4B-AWQ", "object": "model"}

]

}

**4. Use in Python**

When sending requests, use the **served model tag** — the folder name you used when launching (Qwen3-4B-AWQ):

import requests

import json

url = "http://127.0.0.1:8002/v1/chat/completions"

payload = {

"model": "Qwen3-4B-AWQ", # same as folder name

"messages": [{"role": "user", "content": "Hello, world!"}],

"temperature": 0.7,

"max\_tokens": 256,

}

resp = requests.post(url, json=payload)

print(json.dumps(resp.json(), indent=2))

✅ **Key Takeaways**

* Always copy Hugging Face models with cp -rL so blobs aren’t lost.
* Use the **local folder path** in --model when starting vLLM.
* Use the same folder name (e.g., Qwen3-4B-AWQ) as the model in Python requests.