```
from fastapi import FastAPI, HTTPException, BackgroundTasks
from pydantic import BaseModel
from typing import Optional, List
import time
import uvicorn
import os

# Mock import for vLLM client (assuming OpenAI compatible API)
from openai import AsyncOpenAI

app = FastAPI(
    title="Neuro-Doc Inference API",
    description="Gateway for RAG Agent and vLLM Inference",
    version="1.0.0"
)

# Configuration
VLLM_URL = os.getenv("VLLM_URL", "http://localhost:8000/v1")
client = AsyncOpenAI(api_key="EMPTY", base_url=VLLM_URL)

class QueryRequest(BaseModel):
    query: str
    temperature: float = 0.7
    use_rag: bool = True

class InferenceResponse(BaseModel):
    response: str
    latency_ms: float
    sources: List[str] = []

@app.get("/health")
async def health():
    return {"status": "healthy", "gpu_backend": "vLLM", "version":
"0.4.1"}

@app.post("/generate", response_model=InferenceResponse)
async def generate(request: QueryRequest):
    start_time = time.time()

    # 1. RAG Retrieval Step (Mocked for generic repo)
    context = ""
    sources = []
    if request.use_rag:
        # connect_to_qdrant(request.query)
        context = "Context: The hydraulic pressure standard is
ISO-1219.\n"
        sources = ["manual_v4.pdf", "iso_standard.docx"]
```

```
    # 2. Construct Prompt
    system_prompt = "You are Neuro-Doc, an expert engineering
assistant. Use the context provided."
    full_prompt = f"{system_prompt}\n{context}\nUser: {request.query}"

    try:
        # 3. Call vLLM Inference Engine
        response = await client.completions.create(
            model="neuro-doc-model",
            prompt=full_prompt,
            max_tokens=512,
            temperature=request.temperature
        )

        output_text = response.choices[0].text

        # Calculate Latency
        latency = (time.time() - start_time) * 1000

        return InferenceResponse(
            response=output_text.strip(),
            latency_ms=round(latency, 2),
            sources=sources
        )

    except Exception as e:
        raise HTTPException(status_code=500, detail=str(e))

if __name__ == "__main__":
    uvicorn.run(app, host="0.0.0.0", port=8080)
```