Local LLMs & Gemma 2026

Run AI on your own Mac. What hardware runs what, and when it's worth it vs the cloud.

32 GB	$0	3	10 min
SWEET-SPOT RAM	PER-INFERENCE COST	TOOLS TO KNOW	SETUP TIME

Hardware tier table (Apple Silicon)

8 GB RAM
2-3B models · Gemma 2B, Phi-3 Mini, Llama 3.2 3B

Light

16 GB
~8B comfortably · Llama 3.3 8B, Gemma 9B, Qwen 7B

Practical

SWEET SPOT

32 GB
14B-24B · Mistral Small 24B, Qwen 14B, Gemma 27B

64 GB+
70B at Q4 · Llama 3.3 70B, DeepSeek-V3 (sharded)

Power

The 3 tools

START HERE

Ollama
CLI, one command to install, models pull on demand. `ollama run gemma2:9b`

LM Studio
Friendly GUI desktop app. Browse + chat. Best for non-CLI users.

llama.cpp
The C++ engine under Ollama + LM Studio. Use directly for max control.

PRO TIP

Ollama exposes an OpenAI-compatible API at `localhost:11434`. Code that calls OpenAI works against Ollama with one URL change.

HONEST LIMITS

Best local model trails frontier (Opus / GPT-5 / Gemini Ultra) on hard reasoning. Speed is 5-15 tok/sec vs 100+ in cloud. Context maxes at 32-128K vs 1M.

WHEN LOCAL ACTUALLY WINS

1. **Privacy-sensitive material**
   Drafts that shouldn't be logged
2. **Offline use**
   Plane, café with bad Wi-Fi
3. **Batch / automation loops**
   Per-call cost would add up
4. **Embeddings + RAG**
   Small embedding models = blazing fast
5. **Learning**
   Running locally teaches you the most

10-MINUTE OLLAMA SETUP

1. **brew install ollama**
   macOS
2. **ollama serve**
   Auto-starts background
3. **ollama pull gemma2:9b**
   Downloads ~5GB
4. **ollama run gemma2:9b**
   Interactive chat
5. **POST to /api/chat**
   Use from code

Full guide: **djenterprises.ai/blog/local-llms-gemma**