# Self-Hosting Models IS for Nerds: A Practical Guide to Local AI Dev

**A developer's take on Ollama, Llama.cpp, and building smart with your own gear.**

by: Miriah Peterson

https://linktr.ee/soypete_tech

## Who Am I?

- Engineer, tinkerer, and homelab builder
- Have been working on Data Engineering for several years
- I built IAM_pedro — a locally hosted AI discord and twitch bot
- Have been streaming all my work with Pedro on Twitch and it makes for some great content

https://linktr.ee/soypete_tech

## 👋 Quick poll for the room:

- Who is self-hosting models?
- Who is using APIs (OpenAI, Claude)?
- Who has used Ollama or Llama.cpp?

https://linktr.ee/soypete_tech

## Why Local?

- AI is **experimental** — do it takes a lot of iteration to get right
- Local dev = faster feedback, higher throughput, and more calls
- **Ownership** of your workflow and data

https://linktr.ee/soypete_tech

## The Local AI Hosting

- **Ollama** — Local-first LLM runner for open source models
- **llama.cpp** — Lean inference engine for open source models
- **LMStudio** - UI based model runner for open source models

https://linktr.ee/soypete_tech

## What is Ollama?

- **Ollama** is a local-first LLM runner
- Run models with a single command
- It is written in Go and uses llama.cpp under the hood

https://linktr.ee/soypete_tech

# Ollama

- Run models like LLaMA3, Mistral, Qwen locally
- Works on CPU or GPU
- Perfect for:
  - Prompt testing
  - Embedding generation
  - POC for AI projects

https://linktr.ee/soypete_tech

## Demo: Ollama

```
ollama run llama3
```

> Fast, easy, portable. Great for solo developers.

https://linktr.ee/soypete_tech

## Connecting Ollama to Your Workflow

Code:

- Ollama can be used in any programming language
- Use Ollama's REST API or CLI
- Lanchain also supports Ollama so you can just use it in your existing LLM workflows

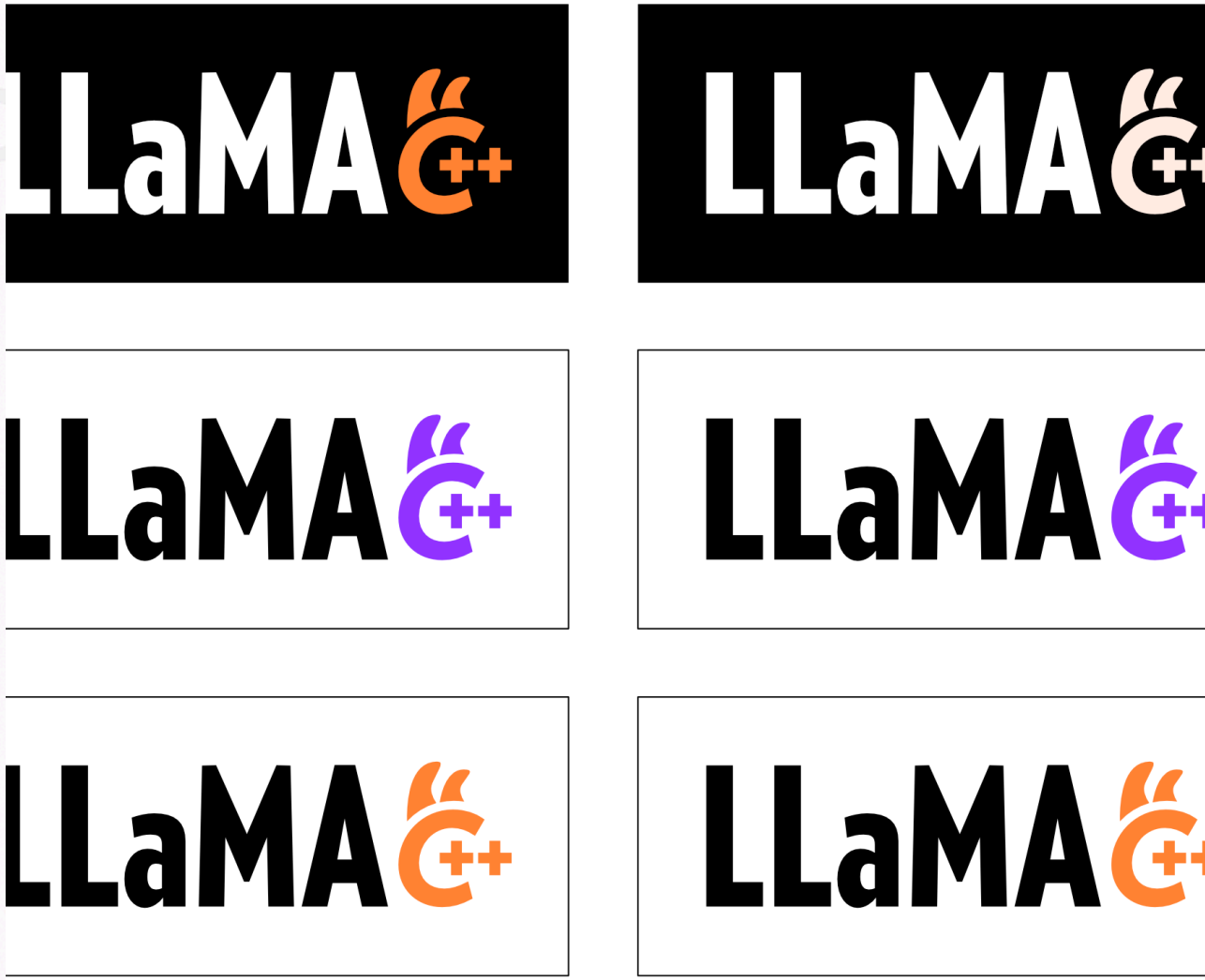
## Drawbacks of Ollama

- **Limited to Ollama's model repo** — you can’t run any model you want.
- **Not as lightweight as llama.cpp** — Ollama is a wrapper around llama.cpp, so it has some overhead.
- **Not as flexible** — Ollama is designed for ease of use, not flexibility.

> There just aren't as many nobs to turn.

## What is Llama.cpp?

- **Llama.cpp** is a lean inference engine for open source models
- It allows you to run models locally with minimal dependencies
- Supports GGUF files for any model.



## Demo: Llama.cpp

```
llama-server --hf-repo TheBloke/Mistral-7B-Instruct-v0.2-GGUF \
--hf-file mistral-7b-instruct-v0.2.Q3_K_S.gguf
```

## Yes, It can use GPUs

- Supports NVIDIA and AMD GPUs

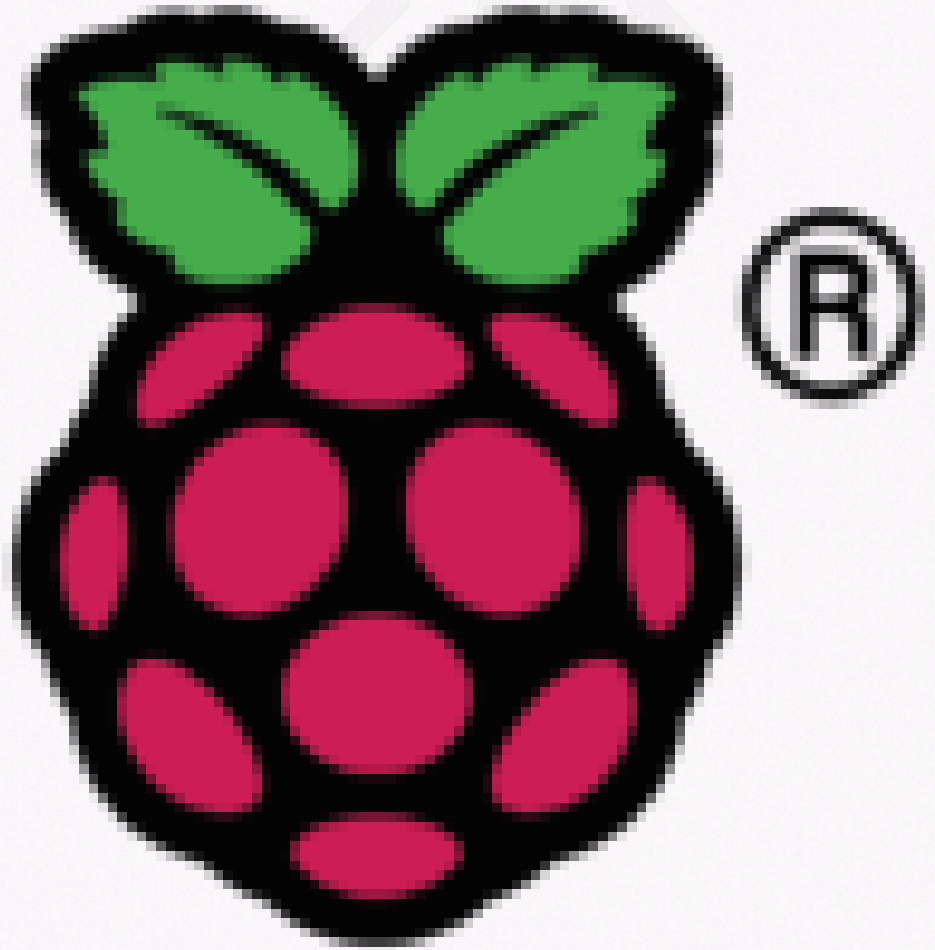
| Backend | Target devices |
| --- | --- |
| Metal | Apple Silicon |
| BLAS | All |
| BLIS | All |
| SYCL | Intel and Nvidia GPU |
| MUSA | Moore Threads GPU |
| CUDA | Nvidia GPU |
| HIP | AMD GPU |
| Vulkan | GPU |
| CANN | Ascend NPU |
| OpenCL | Adreno GPU |
| RPC | All |

## AI homelab stack

If you want an AI for your homelab, you need:

1 raspberry pi or old laptop with 8GB RAM

raspberry pi setup video



## My Homelab:

- MSI tower with 64GB RAM and NVIDIA RTX 5090
- WSL 2 with Ubuntu
- llama.cpp for model inference
- Tailscale for secure local network access
- Golang bot integration for Discord and Twitch on kubernetes(soon)


https://linktr.ee/soypete_tech

## Vector Stores and Embeddings

- **Vector stores** are databases for storing embeddings
- Postgres is a great option for local vector stores
    - pgvector is a Postgres extension for vector search

# Vector Stores and Embeddings

How to get embeddings:

```
curl https://api.openai.com/v1/embeddings \
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -H "Content-Type: application/json" \
  -d '{
    "input": "The food was delicious and the waiter...",
    "model": "text-embedding-ada-002",
    "encoding_format": "float"
  }
```

# Vector Stores and Embeddings

Tips:

- Use **Ollama** or **Llama.cpp** to generate embeddings locally using the models you have running
- Use **pgvector** to store and query embeddings in Postgres and do vector search/comparisons
- You can manage vector stores with code, but I rarely send vectors directly to an llm. I use them for search and retrieval.

## Local AI coding tools

- llama.vim - LLM-powered Vim plugin
- llama.vscode - LLM-powered VSCode extension

https://linktr.ee/soypete_tech

## Future For PedroGPT

- Github Actions for CI/CD:
  - use the action to trigger a call to local runners that run on my homelab and call the Ollama or llama.cpp API. will send code diffs to the bot for analysis. Write summaries, comments, and changes to the codebase in comments.
- Local-first AI agents:
  build a cli that like claude code uses prompts/regex to create agents that generate code, run tests, and deploy changes.

## Future For PedroGPT

- Vector store integration:
  - use pgvector to store code embeddings and do vector search for code snippets, similar to how you would use a code search engine like Sourcegraph or OpenAI's code search.
  - use vector store for twitch quick links so that the bot can quickly find and respond to user queries about past streams, code snippets, or commands.

# Final Thoughts

> “Don’t wait for GPU credits—your machine is already powerful.”

- I ran pedro on a $200 refurbished desktop with 32GB RAM for a year. It worked great and did about 11 tokens per second.
- Local-first AI = speed, control, creativity without waiting for cloud credits.
- Encourage reproducible, offline-first experimentation

https://linktr.ee/soypete_tech

## Downside of Local AI

- Limited to your hardware
- No cool tools for experimentation like Langfuse
- Requires more setup and maintenance

https://linktr.ee/soypete_tech

## Q&A + Discussion

🤔 What's your local setup?

📦 Who's experimenting with vector stores, streaming, or edge agents?

⚙️ What tools do you want to see built?

https://linktr.ee/soypete_tech

# Thank You

Follow me at @soypetetech

Questions, feedback, or collabs? Let’s connect.

https://linktr.ee/soypete_tech