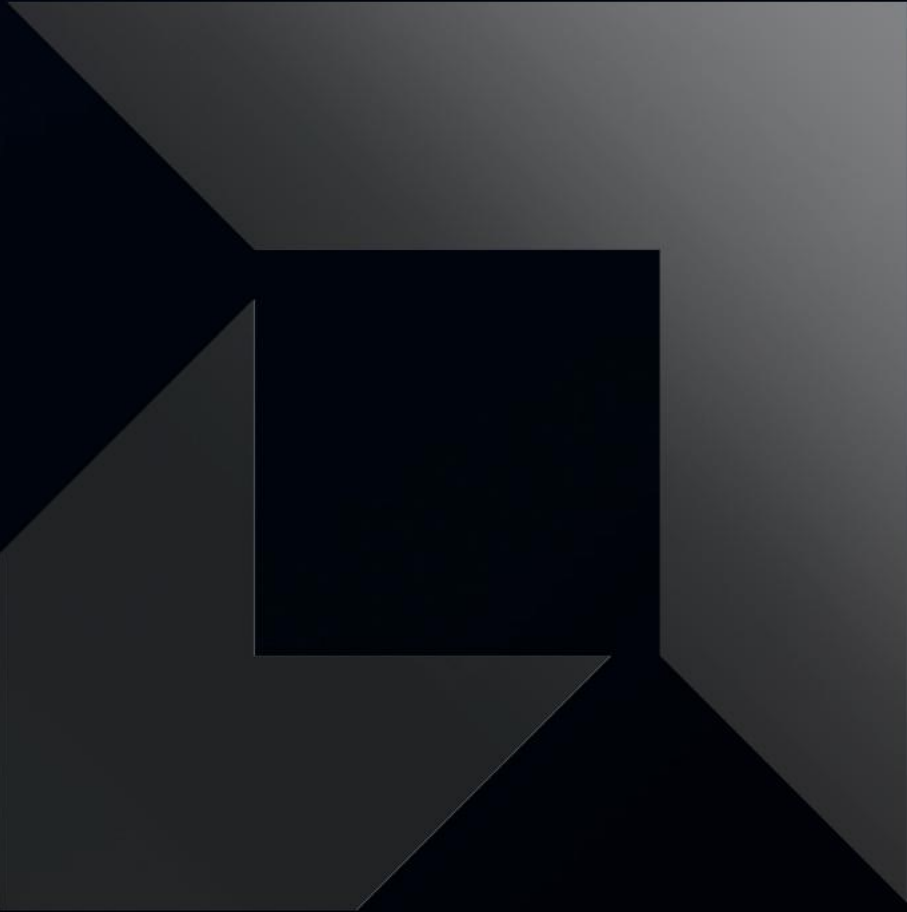




Build your AMD AI assistant using retrieval augmented generation (RAG)

Giacomo Capodaglio
AMD Datacenter Solutions Group
AMD @ CASTIEL

AMD 
together we advance_

What we cover in this presentation

We will discuss **several ways** to build and AMD AI assistant using RAG that you can interact with as you develop and run your code on AMD GPUs

Example questions might be high level like:

- ❖ *How do I get PyTorch to run on AMD GPUs using the latest ROCm enabled version?*
- ❖ *How do I tune my PyTorch workloads on AMD GPUs?*

Or something deeper like:

- *What are the profiling tools from AMD that I can use to profile my AI workloads?*
- *What is the difference between SPX and CPX mode and how do I know which one to set for my node?*

The approach you should select to setup the AMD AI assistant depends on several factors such as:

- ❖ How many users in your system (many or only you)?
- ❖ Are containers allowed in your system?
- ❖ Can a web user interface be used or should it work only from command line?

What is RAG?

Pre-trained neural language models learn a substantial amount of in-depth knowledge from data without any access to an external memory, as a parameterized implicit knowledge base

Cons:

- they cannot easily expand or revise their memory
- can't straightforwardly provide insight into their predictions
- may produce hallucinations

Hybrid models that combine parametric memory with non-parametric (i.e., **retrieval-based**) memories can address some of these issues because knowledge can be directly revised and expanded, and accessed knowledge can be inspected and interpreted.

Crucially, by using pre-trained access mechanisms, the ability to access knowledge is present **without additional training**

With **RAG**: pre-trained, parametric-memory generation models are endowed with a non-parametric memory through a general-purpose fine-tuning approach.

source: Lewis, Patrick, et al. "[Retrieval-augmented generation for knowledge-intensive nlp tasks.](#)"
Advances in neural information processing systems 33 (2020): 9459-9474.

Advantages Of RAG Over Traditional Generative AI models

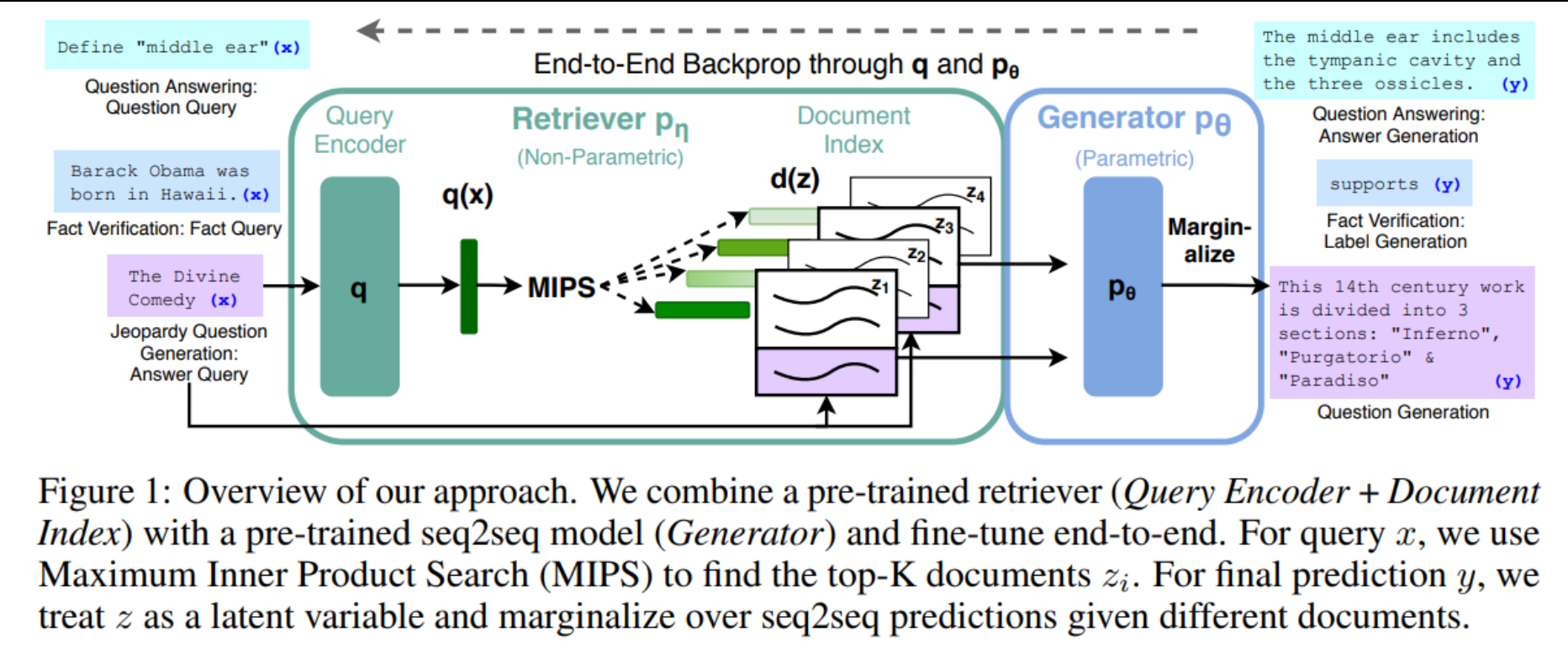
Feature	Traditional Gen AI	RAG
Factual Accuracy	May hallucinate	High accuracy via external data
Knowledge Updates	Needs retraining	Easy updates (~KB)
Explainability	Less transparent	Cites sources
Storage Efficiency	Large model sizes	Smaller model + big knowledge base
Personalization	Difficult	Easy with user-specific data

RAG vs Fine-Tuning vs Prompt Engineering

Let's compare three powerful AI enhancement strategies, each with unique advantages.

Feature	Prompt Engineering	Fine-Tuning	RAG
What it does	Optimizes input prompts to guide output	Changes model weights to learn new behavior	Feeds external data to model through retriever
External knowledge	✗	✗	✓
Model modification	✗	✓	✗ (uses base model)
Training needed	No	Yes (compute-intensive)	Some (for retriever/index)
Implementation speed	Fast	Slow	Medium
Cost	Low	High	Medium
Customization	Surface-level	Deep	Data-level
Best for	Quick tweaks, templates	Domain-specific language or tone	Dynamic, factual information

More details on RAG



The query encoder converts a user query (text) into a dense vector representation (a numerical embedding). This vector is used to search a (user provided) vector database to find the most relevant documents.

source: Lewis, Patrick, et al. "Retrieval-augmented generation for knowledge-intensive nlp tasks." Advances in neural information processing systems 33 (2020): 9459-9474.

What is Ollama?



Get up and running with large language models.

Run [Llama 3.3](#), [DeepSeek-R1](#), [Phi-4](#), [Mistral](#), [Gemma 3](#), and other models, locally.

Download ↓

Available for macOS, Linux,
and Windows

I asked ChatGPT:

“Ollama is a company and platform that provides an API for interacting with large language models (LLMs) and AI tools. The platform allows developers to integrate powerful AI capabilities into their applications, particularly focusing on advanced language models similar to GPT-like models.

Ollama aims to make AI tools more accessible by offering easy-to-use solutions for interacting with and deploying LLMs”

source: ollama.com

Ollama Installation

- **HPC System**

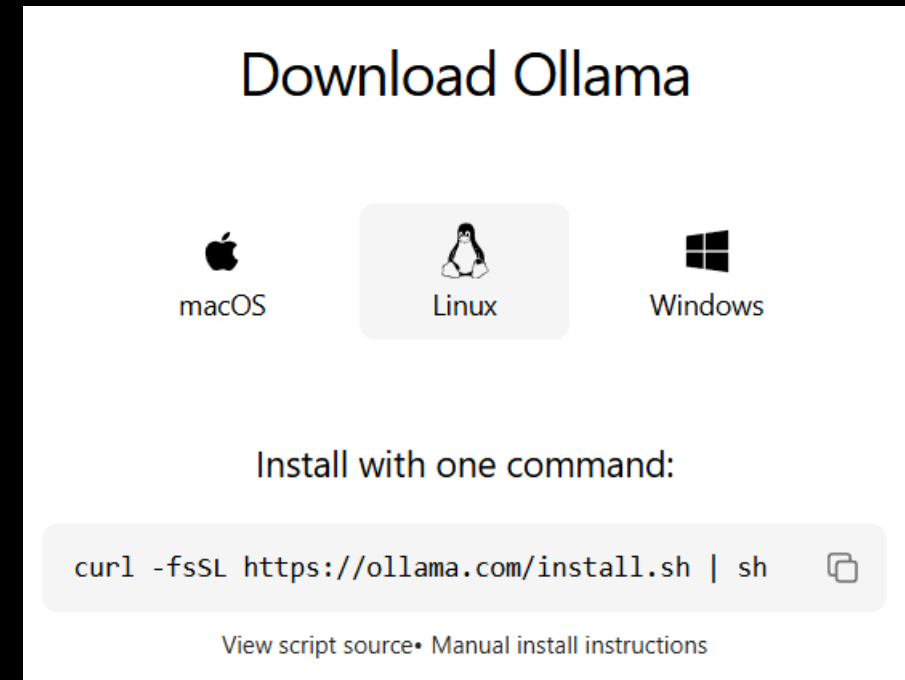
- `curl -fsSL https://ollama.com/install.sh | sh`
or maybe safer to check the commands that will be run with <https://ollama.com/install.sh>
- to run the script: `chmod +x install.sh && ./install.sh`
- manual installation also available on GitHub:
<https://github.com/ollama/ollama/blob/main/docs/linux.md>

The Ollama application will be installed in `/usr/local/bin`

- **Laptop/Workstation**

- Can follow the same procedure as above
- Models that are not too big can run on a laptop without GPUs, but will be slow

source: <https://ollama.com/download/linux>



How to get the Models

- Models can be found here: <https://ollama.com/library>

llama3.3

ollama run llama3.3

2.4M Downloads

Updated 8 months ago

New state of the art 70B model. Llama 3.3 70B offers similar performance compared to the Llama 3.1 405B model.

tools

70b

Models

[View all →](#)

Name	Size	Context	Input
llama3.3:latest	43GB	128K	Text
llama3.3:70b latest	43GB	128K	Text

models run **locally** on the cluster, workstation or laptop so no information leakage **should** be occurring

this command shows how to run it

this shows the number of parameters you can use

models themselves are open-source – no licensing, subscriptions or other fees to pay

Ollama supported AMD GPUs (as of August 27 2025)

AMD Radeon

Ollama supports the following AMD GPUs:

Linux Support

Family	Cards and accelerators
AMD Radeon RX	7900 XTX 7900 XT 7900 GRE 7800 XT 7700 XT 7600 XT 7600 6950 XT 6900 XTX 6900XT 6800 XT 6800 Vega 64 Vega 56
AMD Radeon PRO	W7900 W7800 W7700 W7600 W7500 W6900X W6800X Duo W6800X W6800 V620 V420 V340 V320 Vega II Duo Vega II VII SSG
AMD Instinct	MI300X MI300A MI300 MI250X MI250 MI210 MI200 MI100 MI60 MI50

Windows Support

With ROCm v6.1, the following GPUs are supported on Windows.

Family	Cards and accelerators
AMD Radeon RX	7900 XTX 7900 XT 7900 GRE 7800 XT 7700 XT 7600 XT 7600 6950 XT 6900 XTX 6900XT 6800 XT 6800
AMD Radeon PRO	W7900 W7800 W7700 W7600 W7500 W6900X W6800X Duo W6800X W6800 V620

Source: <https://github.com/ollama/ollama/blob/main/docs/gpu.md#amd-radeon>

Start Up Ollama: gfx90a (MI200 series)

- Start ollama daemon: `ollama serve &`
 - The & makes it run in the background
 - Might need `export OLLAMA_HOST=127.0.0.1:11435` if getting Error: listen tcp 127.0.0.1:11434: bind: address already in use but it should not happen if you killed the processes
- The GPU devices are recognized for gfx90a

working on gfx90a

```
gcapodag@5cbfd40d1aaa:~$ 2025/03/13 04:26:17 routes.go:1215: INFO server config env="map[CUDA_VISIBLE_DEVICES: GPU_DEVICE_ORDINAL: HIP_VISIBLE_DEVICES: HSA_OV
ERRIDE_GFX_VERSION: HTTPS_PROXY: HTTP_PROXY: NO_PROXY: OLLAMA_CONTEXT_LENGTH:2048 OLLAMA_DEBUG:false OLLAMA_FLASH_ATTENTION:false OLLAMA_GPU_OVERHEAD:0 OLLAMA
_HOST:http://127.0.0.1:11435 OLLAMA_INTEL_GPU:false OLLAMA_KEEP_ALIVE:5m0s OLLAMA_KV_CACHE_TYPE: OLLAMA_LLM_LIBRARY: OLLAMA_LOAD_TIMEOUT:5m0s OLLAMA_MAX_LOADE
D_MODELS:0 OLLAMA_MAX_QUEUE:512 OLLAMA_MODELS:/home/aac/shared/teams/dcgpu_training/amd/gcapodag/.ollama/models OLLAMA_MULTIUSER_CACHE:false OLLAMA_NEW_ENGINE
:false OLLAMA_NOHISTORY:false OLLAMA_NOPRUNE:false OLLAMA_NUM_PARALLEL:0 OLLAMA_ORIGINS:[http://localhost https://localhost http://localhost:* https://localho
st:* http://127.0.0.1 https://127.0.0.1 http://127.0.0.1:* https://127.0.0.1:* http://0.0.0.0 https://0.0.0.0 http://0.0.0.0:* https://0.0.0.0:* app://* file:
//* tauri://* vscode-webview://* vscode-file://*] OLLAMA_SCHED_SPREAD:false ROCR_VISIBLE_DEVICES: http_proxy: https_proxy: no_proxy:]"
time=2025-03-13T04:26:17.270Z level=INFO source=images.go:432 msg="total blobs: 0"
time=2025-03-13T04:26:17.271Z level=INFO source=images.go:439 msg="total unused blobs removed: 0"
time=2025-03-13T04:26:17.274Z level=INFO source=routes.go:1277 msg="Listening on 127.0.0.1:11435 (version 0.5.13)"
time=2025-03-13T04:26:17.274Z level=INFO source=gpu.go:217 msg="looking for compatible GPUs"
time=2025-03-13T04:26:17.308Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-5022e090808b5427 gpu_type=gfx90a
time=2025-03-13T04:26:17.309Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-eb8daca7c7f674a2 gpu_type=gfx90a
time=2025-03-13T04:26:17.309Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-d369dc9aab45520c gpu_type=gfx90a
time=2025-03-13T04:26:17.310Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-b598d638837782d7 gpu_type=gfx90a
time=2025-03-13T04:26:17.311Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-e32cbe3ec6f6a640 gpu_type=gfx90a
time=2025-03-13T04:26:17.312Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-ad0ed7fbc23d4c3 gpu_type=gfx90a
time=2025-03-13T04:26:17.313Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-31d0d92d2b0e6bbe gpu_type=gfx90a
time=2025-03-13T04:26:17.314Z level=INFO source=amd_linux.go:386 msg="amdgpu is supported" gpu=GPU-dfd218ad6f8e20b9 gpu_type=gfx90a
time=2025-03-13T04:26:17.314Z level=INFO source=types.go:130 msg="inference compute" id=GPU-5022e090808b5427 library=rocm variant="" compute=gfx90a driver=6.7
name=1002:740f total="64.0 GiB" available="64.0 GiB"
```

Start Up Ollama: gfx942 (MI300 series)

- Start ollama daemon: `ollama serve &`

currently issue on gfx942:

<https://github.com/ollama/ollama/issues/8735>

```
gcapodag@06a065014af1:~$ ollama serve &
[1] 3707250
gcapodag@06a065014af1:~$ 2025/03/12 03:58:04 routes.go:1215: INFO server config env="map[CUDA_VISIBLE_DEVICES: GPU_DEVICE_ORDINAL: HIP_VISIBLE_DEVICES: HSA_OV
ERRIDE_GFX_VERSION: HTTPS_PROXY: HTTP_PROXY: NO_PROXY: OLLAMA_CONTEXT_LENGTH:2048 OLLAMA_DEBUG:false OLLAMA_FLASH_ATTENTION:false OLLAMA_GPU_OVERHEAD:0 OLLAMA
_HOST:http://127.0.0.1:11435 OLLAMA_INTEL_GPU:false OLLAMA_KEEP_ALIVE:5m0s OLLAMA_KV_CACHE_TYPE: OLLAMA_LLM_LIBRARY: OLLAMA_LOAD_TIMEOUT:5m0s OLLAMA_MAX_LOADE
D_MODELS:0 OLLAMA_MAX_QUEUE:512 OLLAMA_MODELS:/home/aac/shared/teams/dcgpu_training/amd/gcapodag/.models OLLAMA_MULTIUSER_CACHE:false OLLAMA_NEW_ENGINE:false
OLLAMA_NOHISTORY:false OLLAMA_NOPRUNE:false OLLAMA_NUM_PARALLEL:0 OLLAMA_ORIGINS:[http://localhost https://localhost http://localhost:* https://localhost:* ht
tp://127.0.0.1 https://127.0.0.1 http://127.0.0.1:* https://127.0.0.1:* http://0.0.0.0 https://0.0.0.0 http://0.0.0.0:* https://0.0.0.0:* app://* file://* tau
ri://* vscode-webview://* vscode-file://*] OLLAMA_SCHED_SPREAD:false ROCR_VISIBLE_DEVICES: http_proxy: https_proxy: no_proxy:]"
time=2025-03-12T03:58:04.102Z level=INFO source=images.go:432 msg="total blobs: 0"
time=2025-03-12T03:58:04.103Z level=INFO source=images.go:439 msg="total unused blobs removed: 0"
time=2025-03-12T03:58:04.106Z level=INFO source=routes.go:1277 msg="Listening on 127.0.0.1:11435 (version 0.5.13)"
time=2025-03-12T03:58:04.107Z level=INFO source=gpu.go:217 msg="looking for compatible GPUs"
time=2025-03-12T03:58:04.167Z level=INFO source=amd_linux.go:296 msg="unsupported Radeon iGPU detected skipping" id=0 total="0 B"
time=2025-03-12T03:58:04.170Z level=INFO source=amd_linux.go:296 msg="unsupported Radeon iGPU detected skipping" id=1 total="0 B"
time=2025-03-12T03:58:04.173Z level=INFO source=amd_linux.go:296 msg="unsupported Radeon iGPU detected skipping" id=2 total="0 B"
time=2025-03-12T03:58:04.175Z level=INFO source=amd_linux.go:296 msg="unsupported Radeon iGPU detected skipping" id=3 total="0 B"
time=2025-03-12T03:58:04.175Z level=INFO source=amd_linux.go:402 msg="no compatible amdgpu devices detected"
time=2025-03-12T03:58:04.176Z level=INFO source=gpu.go:377 msg="no compatible GPUs were discovered"
time=2025-03-12T03:58:04.176Z level=INFO source=types.go:130 msg="inference compute" id=0 library=cpu variant="" compute="" driver=0.0 name="" total="502.2 Gi
B" available="443.5 GiB"
```

Pull a Model with Ollama

- ollama pull llama3.3:70b
will show something like this:

These are progress bars
that will fill in as you
download.

```
gcapodag@f617d5677fa9:~/HPCTrainingExamples/MLExamples/RAG_LangChainDemo$ ollama pull llama3.3:70b
[GIN] 2025/08/28 - 13:21:47 | 200 | 94.629µs | 127.0.0.1 | HEAD | "/"
pulling manifest : [GIN] 2025/08/28 - 13:21:47 | 200 | 371.988939ms | 127.0.0.1 | POST | "/api/pull"
pulling manifest
pulling 4824460d29f2: 100% ██████████ 42 GB
pulling 948af2743fc7: 100% ██████████ 1.5 KB
pulling bc371a43ce90: 100% ██████████ 7.6 KB
pulling 53a87df39647: 100% ██████████ 5.6 KB
pulling 56bb8bd477a5: 100% ██████████ 96 B
pulling c7091aa45e9b: 100% ██████████ 562 B
verifying sha256 digest
writing manifest
success
```

To test that ollama is working, you can also do `ollama run llama3.3:70b` in which case it will pull first, and then run the model.

AMD AI assistant using RAG: two ways of doing it

Depending on your needs and system you might want to use a different approach to create the AMD AI assistant

❖ Python script with command line interaction

Pros:

- Agile interaction from command line
- No need to use containers
- No need to port forwarding which can be tricky for big clusters

Cons:

- Every user will run an Ollama model: can run out of memory quickly
- Not so friendly user interface

❖ Container with Open WebUI (setup on your local machine or ask admin on cluster)

Pros:

- Better suited for systems with large number of users: ollama runs on host system and users interact with it from their local host
- User friendly interface
- Possibility to drag and drop content for RAG context

Cons:

- Difficult to setup with ssh tunnels and port management
- The official way is with containers (though an option with Python is available)

Python script with command line interaction 1/4

With conda

```
module load miniconda3  
conda create -y -n amd_ai_assistant  
conda activate amd_ai_assistant  
git clone https://github.com/amd/HPCTrainingExamples.git  
cd HPCTrainingExamples/MLEexamples/RAG_LangChainDemo  
pip3 install -r requirements.txt
```

Without conda (be mindful of your PYTHONPATH)

```
mkdir ai_assistant_deps  
cd ai_assistant_deps  
export AI_ASSISTANT_DEPS=$PWD  
cd ..  
git clone https://github.com/amd/HPCTrainingExamples.git  
cd HPCTrainingExamples/MLEexamples/RAG_LangChainDemo  
pip3 install -r requirements.txt --target=$AI_ASSISTANT_DEPS  
export PYTHONPATH=$AI_ASSISTANT_DEPS:$PYTHONPATH
```

Python script with command line interaction 2/4

Start up Ollama and pull LLama3.3:70b from Ollama:

```
ollama serve & (set OLLAMA_HOST if this command fails)
ollama pull llama3.3:70b
```

Then run with:

```
python3 amd_ai_assistant.py --rocm-version <rocm_version> --scrape
```

```
amd@sh5-pl1-s12-15:~/HPCTrainingExamples/MLExamples/RAG_LangChainDemo$ python3 amd_ai_assistant.py --rocm-version 6.4.1 --scrape
Using ROCm version: 6.4.1
Force scrape: True
Scraping (depth 0): https://rocm.docs.amd.com/en/docs-6.4.1/
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page2.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page3.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page4.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page5.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page6.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page7.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page8.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page9.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page10.html
Scraping (depth 0): https://rocm.blogs.amd.com/verticals-ai-page11.html
```

Python script with command line interaction 3/4

The first time you run the script it will perform the scraping with user defined depth:

```
# =====
# Configurable variables
# =====
TIMEOUT = 5 # seconds
MAX_DEPTH = 1
CRAWL_DELAY = 1 # seconds delay between requests to avoid overload
CONCURRENT_REQUESTS = 5 # Limit max concurrent requests for politeness
```

MAX_DEPTH=1 means that the script will scrape the urls in main_urls and also the links. MAX_DEPTH=2 means that it will also scrape the links at the linked pages from those in main_urls and so on.

Once the scraping is done it will save the documents so you don't have to scrape again and can run with just:

```
python3 amd_ai_assistant.py
```

```
async def scrape_all(rocm_version):
    rocm_docs_url=rocm_version
    if rocm_version == "latest":
        rocm_docs_url=rocm_version
    else:
        rocm_docs_url=f"docs-{rocm_version}"
    main_urls = [
        f"https://rocm.docs.amd.com/en/{rocm_docs_url}/",
        "https://rocm.blogs.amd.com/verticals-ai.html",
        "https://rocm.blogs.amd.com/verticals-ai-page2.html",
        "https://rocm.blogs.amd.com/verticals-ai-page3.html",
        "https://rocm.blogs.amd.com/verticals-ai-page4.html",
        "https://rocm.blogs.amd.com/verticals-ai-page5.html",
        "https://rocm.blogs.amd.com/verticals-ai-page6.html",
        "https://rocm.blogs.amd.com/verticals-ai-page7.html",
        "https://rocm.blogs.amd.com/verticals-ai-page8.html",
        "https://rocm.blogs.amd.com/verticals-ai-page9.html",
        "https://rocm.blogs.amd.com/verticals-ai-page10.html",
        "https://rocm.blogs.amd.com/verticals-ai-page11.html",
        "https://rocm.blogs.amd.com/verticals-ai-page12.html",
        "https://rocm.blogs.amd.com/verticals-ai-page13.html",
        "https://rocm.blogs.amd.com/verticals-ai-page14.html",
        "https://rocm.blogs.amd.com/verticals-hpc.html",
```

Python script with command line interaction 4/4

```
Scraped 303 documents.
Creating new vectorstore and persisting to disk...

AMD AI Assistant Ready! Type your questions. Type 'exit', 'quit' or 'bye' to stop.

Prompt: |
```

As it prepares the answer, it makes the source explicit

```
Begin by evaluating the performance of your workload in its current state. This involves running benchmarks and collecting performance data to establish a baseline. Understanding how your workload behaves under different conditions provides critical insights into where improvements are needed. Identify tuning requirements
#
source:https://rocm.docs.amd.com/en/docs-6.4.1/how-to/rocm-for-ai/inference-optimization/workload.html
```



Question: how can I optimize PyTorch to run on AMD GPUs?

Helpful Answer:

Answer to above question:

```
> Finished chain.
AMD AI Assistant: To optimize PyTorch on AMD GPUs, you can utilize the `torch._inductor.config` settings. Setting `freezing=True` or the `TORCHINDUCTOR_FREEZING=1` variable can help by in-lining weights as constants and enabling constant folding optimizations. Additionally, enabling `cpp_wrapper` might improve overhead by generating C++ code that launches Triton binaries directly. For convolution workloads, specifying `layout_optimization=True` can also provide a performance benefit by enforcing the `channel_last` memory format. Furthermore, using kernel-level profiling tools like ROCProfiler and ROCm Compute Profiler can help identify problematic GPU operations and guide targeted optimizations.
```

Container with Open WebUI

1. Ssh to host system (login node, for instance `ssh $ADMIN@aac6.amd.com`)
2. Ssh to compute node (compute node on which Ollama will run)
3. Add this line: `host: 0.0.0.0` to `.ollama/config.yaml`
4. `export OLLAMA_HOST=0.0.0.0:<port_number>` (for instance 11435)
5. Run: `ollama serve &`
6. Run: `ollama pull llama3.3:70b`
7. Run: `podman pull ghcr.io/open-webui/open-webui:ollama`
8. Run: `podman run -d -p 3000:8080 -e OLLAMA_BASE_URL=http://host.containers.internal:<port_number> --gpus all -v open-webui:/app/backend/data --name open-webui-ollama --restart always ghcr.io/open-webui/open-webui:ollama`
9. From your local machine run:
`ssh -L 3000:<compute_node>:3000 <host address>` (for instance `<host address>` could be `aac6.amd.com`)
10. Open in your browser: `localhost:3000`

Container with Open WebUI: troubleshooting

If you encounter unexpected behavior while setting up Open WebUI here is something you can do:

1. Kill ollama
 - `ps aux | grep 'ollama serve'`
 - `sudo pkill -f "ollama serve"`
2. Stop and remove the container and volume on Podman
 - `podman stop open-webui-ollama`
 - `podman rm open-webui-ollama`
 - `podman volume rm open-webui`
3. If you get 505:internal error when accessing localhost:3000
 - Keep refreshing the page and it should get you there
4. All the data is local but make sure to write down your admin password after creating the admin account

Container with Open WebUI: create admin account

Explore the cosmos
wherever you are


Get started



Get started with Open WebUI

① Open WebUI does not make any external connections, and your data stays securely on your locally hosted server.

Name
Enter Your Full Name

Email
Enter Your Email

Password
Enter Your Password 

Create Admin Account

make sure to take note of the admin password

Everything is local with the Open WebUI and store in the volume called `open-webui` on Podman

If you forget the admin password you will not be able to login anymore and you'll have to do:

```
podman volume rm open-webui
```

but then all other accounts you may have added (see *next slide*) will be lost

Container with Open WebUI: create users as admin

Users Evaluations Functions Settings

Overview

Users 2

Groups

Search

+

ROLE	NAME	EMAIL	LAST ACTIVE	CREATED AT ^	OAUTH ID
ADMIN	 Giacomo Capodaglio (AMD)	giacomo.capodaglio@amd.com	a few seconds ago	September 5, 2025	
USER	 Bob Robey (AMD)	brobey@amd.com	a minute ago	September 5, 2025	  

© Click on the user role button to change a user's role.

Add User

×

Form CSV Import

Role

User

▼

Name

Enter Your Full Name

Email

Enter Your Email

Password

Enter Your Password

Save

As admin, you will have to create users providing an email Address and password

There is also the option to add users from a csv file

Sign in to Open WebUI

Email

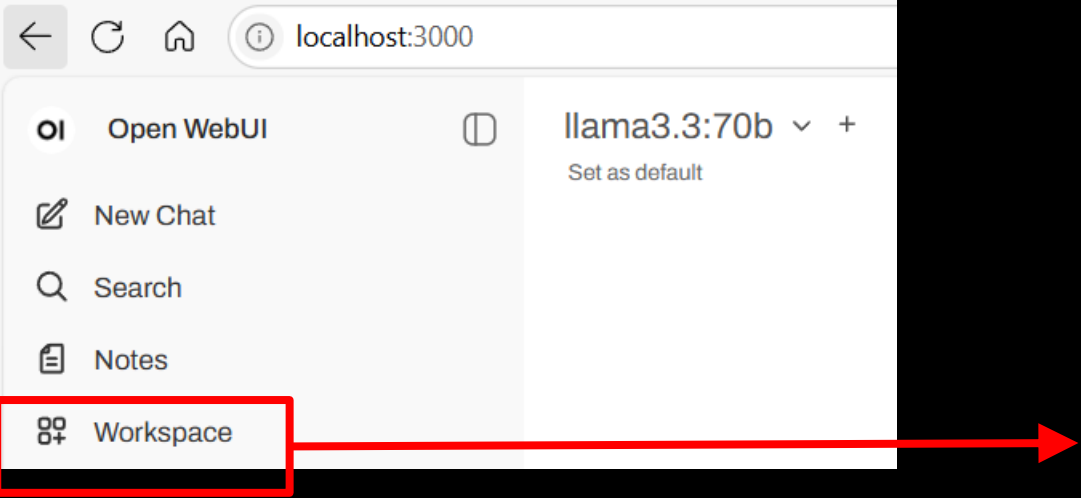
Enter Your Email

Password

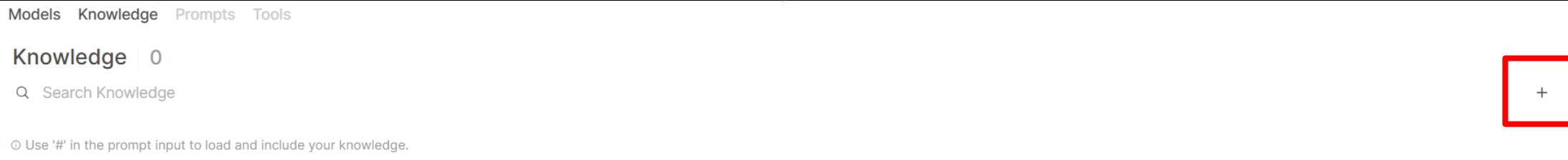
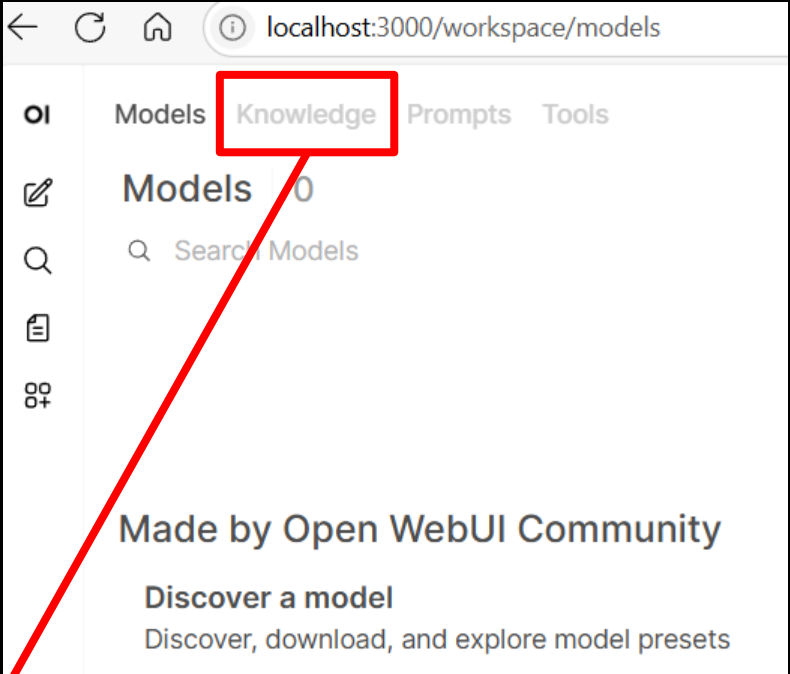
Enter Your Password

Sign in

Container with Open WebUI: adding a RAG database



Click on “Workspace” and then add a knowledge base to an existing model



Container with Open WebUI: adding a RAG database 1/2

Create a knowledge base

What are you working on?

AMD AI Assistant

What are you trying to achieve?

The knowledge base is a collection of materials about AMD GPU programming and software intended to help users with programming their AMD GPUs in a performant way.

Visibility

 **Public** 
Accessible to all users

Create Knowledge



Search Collection	+
07. ROCprof-sys.pdf	1.8 MB
04,. Transpose...	474.0 KB
10. Rocgdb_Debugger.pdf	1.6 MB
09. ROCprof Compute View...	1.3 MB
05. Reduction...	639.7 KB
06. ROCprofv3.pdf	859.3 KB
08. ROCprof-compute.pdf	1.5 MB
03. Register Pressure.pdf	331.5 KB
01. AMD Workshop...	282.4 KB

You can upload PDFs or drop directories directly as well as part of the “Knowledge”

- Drag and drop the `.md` and `.mdx` files from the extracted folder into the **Open WebUI Documentation** knowledge base.

Container with Open WebUI: creating the model

Models

Knowledge

Prompts

Tools



Reset Image

AMD AI Assistant

amd-ai-assistant

Base Model (From)

llama3.2:3b

Description

AMD AI Assistant fetching the latest info on AMD GPU software to users.

Add Tags

Visibility

Private

Only select users and groups with permission can access

Groups

Select a group

No groups with access, add a group to grant access

Model Params

System Prompt

You are an expert GPU programmer with knowledge of AMD GPU software and programming with experience in HPC and AI applications

Advanced Params

Prompt suggestions

Knowledge

To attach knowledge base here, add them to the "Knowledge" workspace first.

 AMD AI Assistant
Collection

Select Knowledge

Upload Files

Tools

To select toolkits here, add them to the "Tools" workspace first.

Filters

To select filters here, add them to the "Functions" workspace first.

Actions

To select actions here, add them to the "Functions" workspace first.

Capabilities

☒ Vision

☒ File Upload

☒ Web Search

☒ Image Generation

☒ Code Interpreter

☐ Usage

☒ Citations

Models

Knowledge

Prompts

Tools

Models

1

Search Models



AMD AI Assistant

AMD AI Assistant fetching the latest info on AMD GP...

By Giacomo Capodaglio (AMD)

...

The model is ready to be used

Advanced Parameters: token limits

Token Limits

Setting the number of tokens to generate in a response controls output length

Computational Impact

Generating more tokens requires more computation, leading to higher energy consumption and potentially slower response times

Cost Considerations

Longer outputs generally result in higher costs when using commercial LLM services

Advanced Parameters: top-k and top-p



Top-K Sampling

Selects from the top K most likely tokens from the model's predicted distribution. Higher values increase creativity; lower values produce more factual outputs



Top-P (Nucleus) Sampling

Selects from tokens whose cumulative probability doesn't exceed value P. Values range from 0 (greedy decoding) to 1 (all tokens considered)



Combined Approach

Often used together for optimal results. Tokens meeting both criteria become candidates for the next prediction

Feature	Top-k Sampling	Top-p Sampling
Selection Criteria	Fixed number of top tokens	Dynamic set based on probability mass
Parameter	k (e.g., 50)	p (e.g., 0.9)
Adaptability	Less adaptive	More adaptive
Diversity Control	Moderate	High
Risk of Truncation	May ignore low-probability but meaningful tokens	Less likely to ignore meaningful tokens
Use Case	When you want consistent diversity	When you want adaptive creativity

Advanced Parameters: temperature

Low Temperature (0-0.3)

More deterministic responses

Highest probability token is selected

Best for factual tasks

Medium Temperature (0.3-0.7)

Balance of creativity and coherence

Some variation in responses

Good for general content creation

High Temperature (0.7-1.0)

More random, diverse outputs

Wider range of token selection

Best for creative tasks

Container with Open WebUI: advanced parameters

Temperature

Reasoning Effort

logit_bias

max_tokens

top_k

top_p

Custom

0.4

Default

Default

Custom

1000

Custom

40

Custom

0.95

 WARNING

If you're using **Ollama**, note that it **defaults to a 2048-token context length**. This means that retrieved data may **not be used at all** because it doesn't fit within the available context window. To improve **Retrieval-Augmented Generation (RAG) performance**, you should **increase the context length to 8192+ tokens** in your Ollama model settings.

Source:
[Open WebUI docs](#)

num_ctx (Ollama)

Custom

8192

Chunking Text Strategies

Document chunking is essential for RAG systems to optimize retrieval accuracy and manage token limits. The strategy choice affects both precision and efficiency.

Strategy	Description	Pros/Cons
Simple Chunking	Fixed-size chunks with configurable overlap	Pros: Easy, predictable sizing Cons: Breaks semantic units
Semantic Chunking	Divides by meaning using document structures	Pros: Preserves context Cons: Variable sizes, more complex
LM-Based Chunking	Uses language models to create self-contained concepts	Pros: Highest coherence Cons: Resource-intensive
Hybrid Chunking	Combines fixed-size with semantic boundaries	Pros: Balanced precision/efficiency Cons: Requires tuning
Recursive Chunking	Hierarchical approach with progressively smaller units	Pros: Maintains hierarchy Cons: Complex implementation

Optimal chunk sizing depends on your use case. Larger chunks preserve context but may include irrelevant information; smaller chunks enable precise retrieval but may lose relationships. Most embedding models work best with 256-1024 tokens per chunk.

Container with Open WebUI: RAG document settings

The screenshot shows the Open WebUI Settings page. The 'Settings' tab is selected in the top navigation bar. On the left sidebar, the 'Documents' tab is highlighted with a red box. The main content area is divided into sections: General, Embedding, and Retrieval. In the General section, 'Chunk Size' is set to 1000 and 'Chunk Overlap' is 100. In the Retrieval section, 'Top K' is set to 40. The 'RAG Template' section contains a task description and guidelines. A red box highlights the first guideline: '- If you don't know the answer, clearly state that. Do not make-up an answer if you don't know it.' At the bottom right, a 'Save' button is highlighted with a red box and a red arrow points to it from the text 'Don't forget to hit save at the bottom right of the page'.

Users Evaluations Functions **Settings**

General

Content Extraction Engine Default

PDF Extract Images (OCR)

Bypass Embedding and Retrieval

Text Splitter

Chunk Size 1000

Chunk Overlap 100

Embedding

Embedding Model Engine Default (SentenceTransformers)

Embedding Model sentence-transformers/all-MiniLM-L6-v2

Warning: If you update or change your embedding model, you will need to re-import all documents.

Retrieval

Full Context Mode

Hybrid Search

Top K 40

RAG Template

Task:
Respond to the user query using the provided context, incorporating inline citations in the format [id] **only when the <source> tag includes an explicit id attribute** (e.g., <source id="1">).

Guidelines:

- If you don't know the answer, clearly state that. Do not make-up an answer if you don't know it.
- If uncertain, ask the user for clarification.
- Respond in the same language as the user's query.
- If the context is unreadable or of poor quality, inform the user and provide the best possible answer.
- If the answer isn't present in the context but you possess the knowledge, explain this to the user and provide the answer using your own understanding.
- **Only include inline citations using [id] (e.g., [1], [2]) when the <source> tag includes an id attribute.
- Do not cite if the <source> tag does not contain an id attribute.
- Do not use XML tags in your response.
- Ensure citations are concise and directly related to the information provided.

Example of Citation:
If the user asks about a specific topic and the information is found in a source with a provided id attribute, the response should include the citation like in the following example:
* "According to the study, the proposed method increases efficiency by 20% [1]."

Output:
Provide a clear and direct response to the user's query, including inline citations in the format [id] only when the <source> tag with id attribute is present in the context.

<context>
{CONTEXT}

Save

Edit also the Documents tab for the RAG retrieval

Reach it by clicking on The admin icon, then Admin Panel then Settings

Don't forget to hit save at the bottom right of the page

Container with Open WebUI: querying the assistant

Select the model from the drop down menu

The screenshot shows the Open WebUI interface. At the top left, a dropdown menu is highlighted with a red box, showing 'AMD AI Assistant' with a plus sign. To the right, a text input field contains the query 'what can you tell me about the HPCTrainingDock?'. Below the input, the assistant's response is displayed, starting with 'OI AMD AI Assistant'. The response text describes the HPCTrainingDock project and mentions several files: 'sys_admin_talk.pdf', 'main_setup.sh', and 'test_install.sh'. A red box highlights the text 'sys_admin_talk.pdf' in the response, with a red arrow pointing down from it. At the bottom of the response, there is a list of sources, with '1 sys_admin_talk.pdf' highlighted.

The assistant will also report what specific item from the knowledge based it used to answer the question. In this case it used a deck of slides we uploaded that has content on sys admin best practices

Container with Open WebUI: user friendly interface

deepseek-r1:1.5b ▾ +
llama3.3:70b ▾ -
phi:latest ▾ -

what does HSA_XNACK do for an MI300A AMD GPU?

deepseek-...
Thought for 16 seconds ▾
The HSA_XNACK for an MI300A AMD GPU is likely an enhancement within the AMD HSA series. This term typically stands for "AMD Enhance," indicating a specific upgrade or feature added to the GPU. The X in the model name, such as in the HSA models, suggests that this is an enhancement on top of the base model.

Given this context, it's probable that HSA_XNACK introduces improvements

llama3.3:70b
A very specific and technical question!

After digging through various documentation and forums, I found some information about **HSA_XNACK** in the context of AMD GPUs, including the MI300A.

What is HSA_XNACK?
HSA_XNACK stands for "Heterogeneous System Architecture eXtended Negative Acknowledgment". It's a feature introduced by AMD as part of their Heterogeneous System

phi:latest
I don't have access to the specific details of the mipmap cross-network adapter xnack system, so i cannot provide you with its exact functionality or compatibility with the mi300a amd gpu. however, typically, hsa_xnack is a hardware management utility that allows for remote access and monitoring of network devices such as routers, switches, and firewalls. this can be useful for managing an entire network from a central location, allowing

Send a Message

+ Code Interpreter

Compare
model
answers
side by side

Modify
model
parameters
with
interface

Chat Controls ×

Valves ▾

System Prompt ^
Enter system prompt

Advanced Params ^

Stream Chat Response	Default
Stream Delta Chunk Size	Default
Function Calling	Default
Reasoning Tags	Default
Seed	Default
Stop Sequence	Default
Temperature	Default
Reasoning Effort	Default
logit_bias	Default
max_tokens	Default
top_k	Default
top_p	Default
min_p	Default
frequency_penalty	Default
presence_penalty	Default
mirostat	Default
mirostat_eta	Default
mirostat_tau	Default
repeat_last_n	Default

Disclaimer

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS.' AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Third-party content is licensed to you directly by the third party that owns the content and is not licensed to you by AMD. ALL LINKED THIRD-PARTY CONTENT IS PROVIDED "AS IS" WITHOUT A WARRANTY OF ANY KIND. USE OF SUCH THIRD-PARTY CONTENT IS DONE AT YOUR SOLE DISCRETION AND UNDER NO CIRCUMSTANCES WILL AMD BE LIABLE TO YOU FOR ANY THIRD-PARTY CONTENT. YOU ASSUME ALL RISK AND ARE SOLELY RESPONSIBLE FOR ANY DAMAGES THAT MAY ARISE FROM YOUR USE OF THIRD-PARTY CONTENT.

© 2025 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo, AMD CDNA, AMD ROCm, AMD Instinct, and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and/or other jurisdictions. Other names are for informational purposes only and may be trademarks of their respective owners.

