exterro®
Data Risk Management

vLLM DEPLOYMENT & OPERATIONS GUIDE

JULY 2026

Table of Contents

About Exterro

Exterro was founded with the simple vision that applying the concepts of process optimization and data science to how companies manage digital information and respond to litigation would drive more successful outcomes at a lower cost. We remain committed to this vision today. We deliver a fully integrated Data Risk Management platform that enables our clients to address their privacy, regulatory, compliance, digital forensics, and litigation risks more effectively and at lower costs. We provide software solutions that help some of the world's largest organizations, law enforcement and government agencies work smarter, more efficiently, and support the Rule of Law.

Purpose of the Document

This document provides a comprehensive guide for deploying, managing, and decommissioning the **Exterro vLLM Server**.

Infrastructure Requirements

Before proceeding with the deployment, you are recommended to refer to the '*Exterro FTKC - AI Infrastructure Requirements'* guide for LLM Server deployment to ensure your host meets all hardware and GPU specifications.

Prerequisites

- docker and docker compose must be pre-installed on the target system.
- Root (sudo) privileges are required.

vLLM Deployment & Operations

Part 1: Host Prerequisites (NVIDIA Drivers & Container Toolkit)

Ensure your host operating system (Ubuntu) has the correct NVIDIA display drivers, CUDA Toolkit, and NVIDIA Container Toolkit installed so Docker containers can access GPU resources.

1 **Uninstall Any Existing/Broken NVIDIA Drivers:**

```
sudo apt --purge remove nvidia-* cuda-*
sudo apt autoremove -y
```

2 **Add NVIDIA's Official Driver Repository:**

```
sudo add-apt-repository ppa:graphics-drivers/ppa
sudo apt update
```

3 **Check Recommended Driver Version:**

```
ubuntu-drivers devices
You will see output suggesting something like:
driver : nvidia-driver-570
```

4 **Install NVIDIA Driver:**

```
sudo apt install nvidia-driver-570 nvidia-dkms-570
sudo reboot

nvidia-smi
```

If below error occurs:

(NVIDIA-SMI has failed because it couldn't communicate with the NVIDIA driver. Make sure that the latest NVIDIA driver is installed and running.

Fix:

i. **Install the headers for your *exact* current kernel:**

```
sudo apt install linux-headers-$(uname -r)
```

ii. **Trigger a rebuild of the driver:**

```
sudo apt install --reinstall nvidia-driver-570
```

iii. **Check nvidia-driver version:**

```
nvidia-smi
```

5 **Install CUDA Toolkit (Without Overwriting Drivers):**

```
sudo apt install cuda-toolkit-12-8
```

6 **Add NVIDIA Container Toolkit Repository:**

```
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | \
 sudo gpg --dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg

curl -s -L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-toolkit.list | \
 sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-toolkit-keyring.gpg] https://#g' |
\
 sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
```

7 **Update Package Index:**

```
sudo apt-get update
```

8 **Install NVIDIA Container Toolkit:**

```
sudo apt-get install -y nvidia-container-toolkit
```

9 **Configure Docker to Use the NVIDIA Runtime:**

```
sudo nvidia-ctk runtime configure --runtime=docker
```

10 **Restart Docker Service:**

```
sudo systemctl restart docker
```

11 **Verify GPU Access Inside Docker Container:**

```
docker exec -it <container name> bash
nvidia-smi
```

***Note**: After successfully deploying the vLLM container by the steps given below, this verification can be executed to ensure GPU & NVIDIA dependencies are properly mounted to the running container from the host machine.*

Part 2: Deployment & Configuration

Step 1: Prepare Working Directory & Download SFX Archive

1. Create and navigate to the deployment directory, then elevate to root privileges:

```
mkdir -p deployments/sfx_deploy
cd deployments/sfx_deploy
sudo su
```

2. Download the deployment SFX file using the S3 link provided by the Exterro team:
3. Confirm that the downloaded file is present in the directory using 'ls' command:

Step 2: Configure Environment Variables

Configure the deployment environment parameters based on your target model and hardware setup. Run these sequentially:

```
export MODEL_NAME=Qwen3-30B-A3B-Instruct-2507
export VLLM_TENSOR_PARALLEL_SIZE=1
export VLLM_MAX_MODEL_LEN=64000
export S3_URL='<ADD_S3_BUCKET_URL_HERE>'
```

Notes:

- **S3 Bucket URL:** Contact the Exterro Implementation Team to obtain a pre-signed URL for downloading the model.

  Note: *Pre-signed S3 links expire after **12 hours**. Please download the model immediately upon receiving it. If the link expires before execution, you will need to request a new pre-signed URL from the Implementation Team and update the configuration token.*

- ***GPU Scaling:*** *Run nvidia-smi to verify available hardware.*
  - *If your server has **2 GPUs**, update the parallel size: export VLLM_TENSOR_PARALLEL_SIZE=2*
  - *Depending on your GPU memory capacity, VLLM_MAX_MODEL_LEN can be increased (e.g., 128000, 256000).*

To verify that environment variables are correctly loaded:

```
set | grep -E 'MODEL_NAME|VLLM|S3_URL'
```

```
UID=0
USER=root
VLLM_MAX_MODEL_LEN=128000
VLLM_TENSOR_PARALLEL_SIZE=1
```

```
MODEL_NAME=Qwen3-4B-Instruct-2507
OLDPWD=/deployments
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
PIPESTATUS=([0]="0")
PPID=128068
PS1='\[\e]0;\u@\h: \w\a\]${debian_chroot:+($debian_chroot)}\u@\h:\w\$ '
PS2='> '
PS4='+ '
PWD=/deployments/sfx_deploy
S3_URL='https://forensics-ai-models-version-controller.s3.us-west-1.amazonaws.com/LLM_Models/Qwen3-4B-Instruct-2507/Qwen3-4B-Instruct-2507.zip?response-content-disposition=inline&X-Amz-Content-Sha256=UNSIGNED-PAYLOAD&X-Amz-Security-Token=IQoJb3JpZ2luX2VjEHkaCXVzLXdlc3QtMSJHMEUCIQDTQPXO9mmCMk394dy61ruzmYeiRxFx5rsbSlW5ijSqOAIgJrB7zJC6npPzTlx26kI6990Ovse9QFotSRXA99IKbFgq0AMIQhAAGgwxNzg3MzYwMDg1NjYiDJjqWbcZLw25ARUksCqtA7f1402NwB8Ob81%2BijvDocZHq6aDhFv4aLvgcFVQkIyT69sD7CSW%2B%2Bm6UxiooQEkt3knvO27mj4a5D4ZO%2BWZ%2FDnEG1qFGX12V1IF91iayK4VvxbYO1FJz8YU14eITEd6s0Cs%2Bvgzgs5Piuw0A9nWL8fzyeePxvS3j6Qhjugc5eSp8y9q3fPaJXZ5RFpjWlll6SnO%2BBLtqd3X1Cj4ZIR0Jyd3So5vwb0j1QpedqAxlMa7WSvuWeoBBYgewVvuz0lO%2FAxL2cRnQadw04%2BCrAtF%2Fkhb6RtUAW1zvyeYSH%2B8S0HktIOpdR7yWwI%2BbT8ZZ0MWR5788tvKGkKcsyVy%2FeFnGyV1bJroLC%2Flm1Lsl7SLkjuYnxk9k0zEyaB%2Bmf4CzL2S%2Fmfz%2BzEd3U2LlEvm7Ku1orzq%2FlE%2BaOl2%2BgkG86FSs9Q98BnB%2F7voNOfkhbdhcqBj0CGERBf2DR5cl8ZLC%2BNUzMVeuEgu5EJS6xrjqG1g5xwlOEI6uhh6hhUbz4PTl6lNORaoRux3nuIU3v6p9OgpJwX5iExwI%2BANi1qtLgTUzSD5qJbB8%2F1%2FBPnDsfw7bTDPqeLSBjreAhhIGGFktOUAdaD4cB%2BnN%2BtHgCiqUcO5SwIVRZ8c8gFbsWysSvlct%2BPzd9FtPCFRKek7kExuCtbhRcxQO6r6ou2B524tVVc21Xdxerm506j%2BKGtR9L%2BiM1glwf4ceD%2BX0JHNJH6vvMZEJLr5gAhFwTqLIE5jlwkl8%2B3mhKd4QdpjR5tUYBsI5ZW3DRBufuYkDmVCyggSyxkRGTEUczPaxX3AZ4NgHFllIVBjkka5esrx8whdUONyfo0bF0zgoZUgyqkCBPl5VgZi5P058ULYeR5O7SPv5807G0UhifXAZ8yhudsgEwURVK7DgIEVuOmg2uG9AkPDZ0mK1e9ukLyKN8Lw8Z%2Fs%2BK0rR8KFavNhhbTcnEUmytDdOj8anPd5ElQ3byOMa8MmB3RBSfkTxaHjj3YSeuDYCBPCiDhYVcOm5KIYIEYaJGCtk%2BL91P6KLdcjDmzV6clWtkBC1HbHKTgP&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=ASIASTHL7BV3NZGPJSSG%2F20260716%2Fus-west-1%2Fs3%2Faws4_request&X-Amz-Date=20260716T082307Z&X-Amz-Expires=43200&X-Amz-SignedHeaders=host&X-Amz-Signature=56a05ab70e4ee0119baa099a13d2d5a24c6276ea1eb1fb23d4ab326186dbdfcb'
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:history:interactive-comments:monitor
```

Step 3: Execute Deployment

1. Run the self-extracting deployment script (this will automatically download and extract model weights from S3):

```
sh exterro-vllm-manylinux.sfx
```

```
root@ubuntu-LLM:/deployments/sfx_deploy# sh exterro-vllm-manylinux.sfx
Verifying archive integrity...  100%   MD5 checksums are OK. All good.
Uncompressing SFX archive for installing the Divebell AI sensor...
  100%
Press enter to continue with these values, or C-c to abort
```

2. Press **Enter** when prompted to begin deployment.

```
Length: 6397396412 (6.0G) [application/zip]
Saving to: '/exterro/Qwen3-4B-Instruct-2507.zip'

/exterro/Qwen3-4B-Instruct-2507   1%[                                                  ] 106.55M  36.9MB/s
```

```
  inflating: /exterro/models/Qwen3-4B-Instruct-2507/config.json
  inflating: /exterro/models/Qwen3-4B-Instruct-2507/tokenizer.json
  inflating: /exterro/models/Qwen3-4B-Instruct-2507/vocab.json
  inflating: /exterro/models/Qwen3-4B-Instruct-2507/model-00003-of-00003.safetensors
  inflating: /exterro/models/Qwen3-4B-Instruct-2507/merges.txt
  inflating: /exterro/models/Qwen3-4B-Instruct-2507/README.md
+ validate_model_download
+ local model_dir=/exterro/models/Qwen3-4B-Instruct-2507
+ '[' '!' -d /exterro/models/Qwen3-4B-Instruct-2507 ']'
+ info 'Model verified at /exterro/models/Qwen3-4B-Instruct-2507'
+ echo 'Model verified at /exterro/models/Qwen3-4B-Instruct-2507'
Model verified at /exterro/models/Qwen3-4B-Instruct-2507
+ cleanup_model_zip
+ local zip_path=/exterro/Qwen3-4B-Instruct-2507.zip
+ '[' -f /exterro/Qwen3-4B-Instruct-2507.zip ']'
+ rm -f /exterro/Qwen3-4B-Instruct-2507.zip
+ info 'Cleaned up /exterro/Qwen3-4B-Instruct-2507.zip'
+ echo 'Cleaned up /exterro/Qwen3-4B-Instruct-2507.zip'
Cleaned up /exterro/Qwen3-4B-Instruct-2507.zip
+ VLLM_MODEL=/exterro/services/models/Qwen3-4B-Instruct-2507
+ info 'Wrote resolved environment to /opt/exterro/vllm/etc/vllm.env'
+ echo 'Wrote resolved environment to /opt/exterro/vllm/etc/vllm.env'
Wrote resolved environment to /opt/exterro/vllm/etc/vllm.env
+ info 'Model '\''Qwen3-4B-Instruct-2507'\'' ready at /exterro/models/Qwen3-4B-Instruct-2507'
+ echo 'Model '\''Qwen3-4B-Instruct-2507'\'' ready at /exterro/models/Qwen3-4B-Instruct-2507'
Model 'Qwen3-4B-Instruct-2507' ready at /exterro/models/Qwen3-4B-Instruct-2507
+ _systemctl start docker.service
+ '[' -z '' ']'
+ systemctl start docker.service
+ sudo -u exterro sg docker -c 'docker pull vllm/vllm-openai:v0.13.0'
root@ubuntu-LLM:/deployments/sfx_deploy#
```

```
+ export DAEMON_HOME=/opt/exterro/vllm
+ export VLLM_CONTAINER VLLM_IMAGE
+ export MODELS_VOLUME MODELS_MOUNT VLLM_MODEL
+ export VLLM_GPU_MEMORY_UTILIZATION VLLM_TENSOR_PARALLEL_SIZE VLLM_MAX_MODEL_LEN
+ export VLLM_PORT NETWORK_NAME
+ docker-compose -f /opt/exterro/vllm/conf/service.yaml up --no-build
[+] up 24/24
 ✔ Image vllm/vllm-openai:v0.13.0 Pulled
 ✔ Network llm-network            Created
 ✔ Container vllm_server          Created
Attaching to vllm_server
```

```
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /v1/audio/translations, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /ping, Methods: GET
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /ping, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /invocations, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /classify, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /v1/embeddings, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /score, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /v1/score, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /rerank, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /v1/rerank, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /v2/rerank, Methods: POST
vllm_server  | (APIServer pid=1) INFO 07-16 08:23:55 [launcher.py:46] Route: /pooling, Methods: POST
vllm_server  | (APIServer pid=1) INFO:     Started server process [1]
vllm_server  | (APIServer pid=1) INFO:     Waiting for application startup.
vllm_server  | (APIServer pid=1) INFO:     Application startup complete.
```

3. Successful deployment is indicated when the logs display:

```
Application startup complete
```

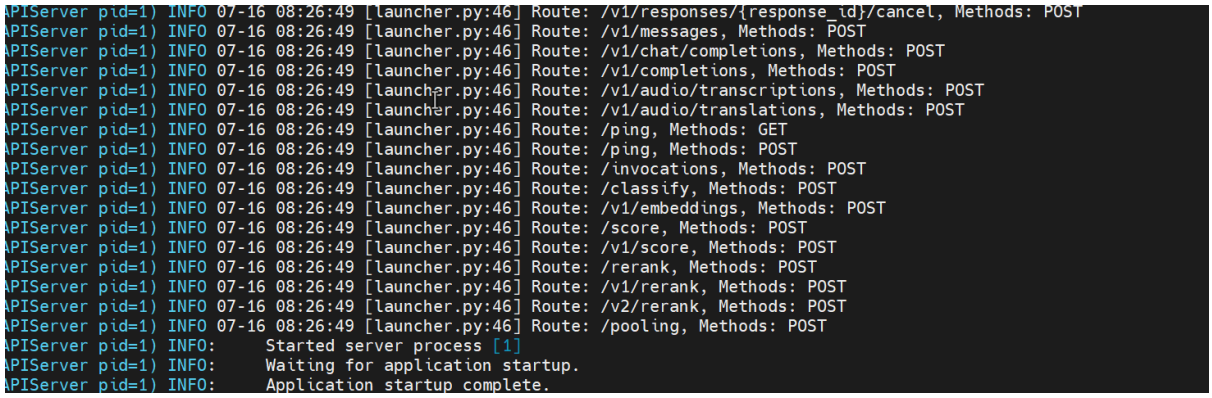
4. Verify Container Status: Ensure the Docker container is running properly:

```
docker ps
```

```
(APIServer pid=1) INFO:     Application startup complete.
root@ubuntu-LLM:/opt/exterro/vllm/bin# docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATED          STATUS         PORTS
                NAMES
c8851c5d5f26   vllm/vllm-openai:v0.13.0   "vllm serve --model …"   14 minutes ago   Up 9 minutes   0.0.0.0:443->8000/tcp, [::]
:443->8000/tcp   vllm_server
root@ubuntu-LLM:/opt/exterro/vllm/bin#
```

5. **Start Service & Check Logs:**

```
cd /opt/exterro/vllm/bin
./vllm-start
docker logs vllm_server
```

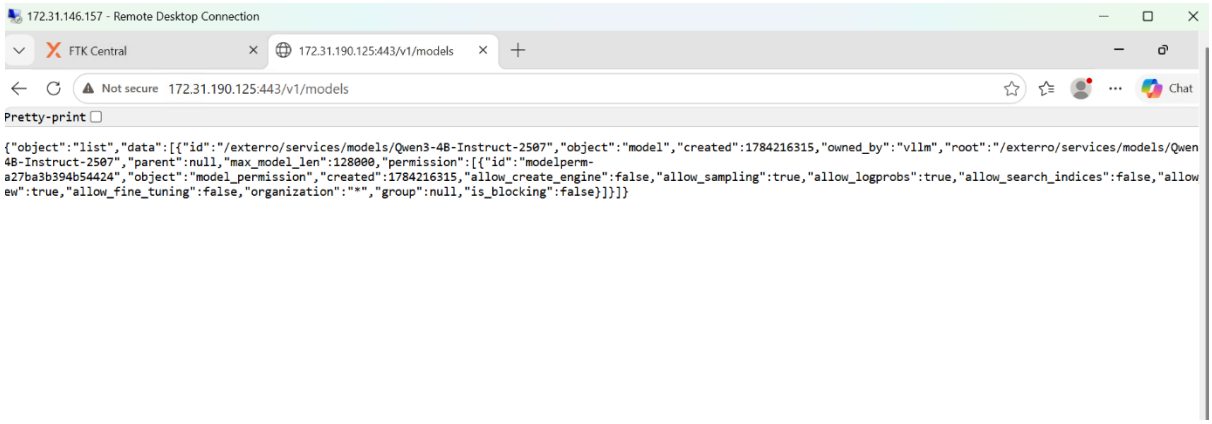


Part 3: Monitoring & Verification

Step 1: Validate Service Health & Container GPU Access

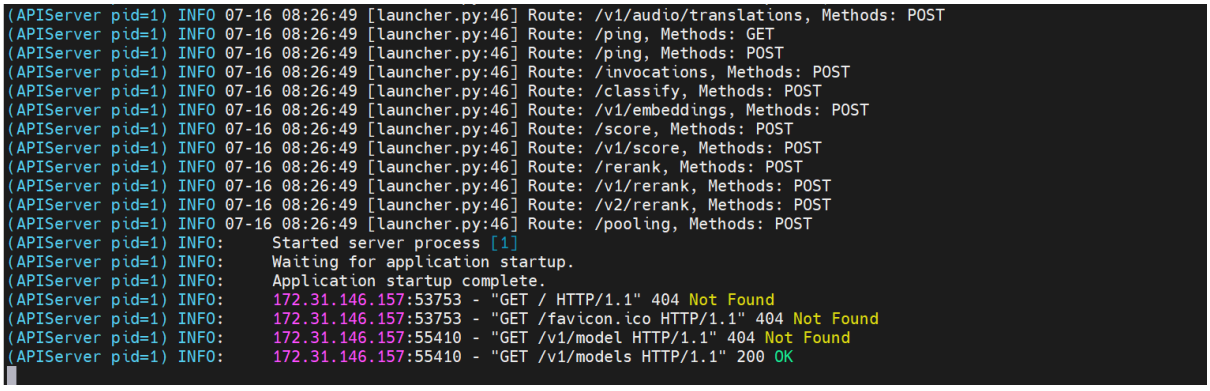
- **API Health Check:** Open a browser or run a curl request to verify the model metadata endpoint:

```
HTTP
http://<SERVER_IPv4>:443/v1/models
```



- **Real-time Container Logs:** Monitor operational logs or diagnose failures:

```
docker logs vllm_server -f
```



Configure LLM server in FTKC:

Once you have verified that the model is available in the model metadata endpoint, configure the LLM Server in FTK Central.

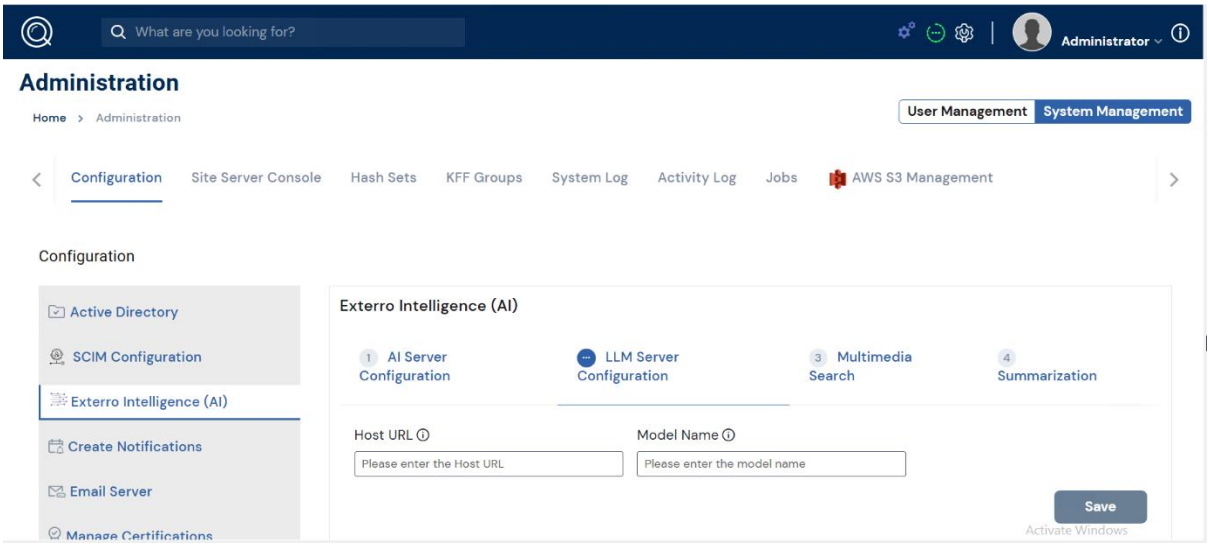
i. In the application, go to 'Administration' > 'System Management'.

ii. Under 'Configuration', go to 'Exterro Intelligence (AI)' and select 'LLM server configuration'.

iii. Enter the LLM Server endpoint (http://<SERVER_IPv4>:443) and the model name.

iv. Click 'Save'.

Now the LLM server will be configured with FTKC and the model will be used for inference jobs.



- **Verify GPU Access Inside Container:**

```
docker exec -it vllm_server nvidia-smi
```

Step 2: Service Management

Manage the vLLM background process using standard systemd commands:

- **Stop the Service:**

```
systemctl stop exterro-vllm.service
```

- **Start the Service:**

```
systemctl start exterro-vllm.service
```

Reference File Mapping:

Description	Path Location
Downloaded Model Weights	/exterro/models/Qwen3-30B-A3B-Instruct-2507
Service Configurations	/opt/exterro/vllm/conf *(Default Port: 8000)*
vLLM Runtime Environment Config	/opt/exterro/vllm/etc

To inspect active environment flags from system configs:

```
cd /opt/exterro/vllm/etc
cat vllm.env
```

Part 4: Decommissioning & Uninstallation

Choose **Method A** (recommended) or **Method B** (manual cleanup).

Method A: Automated Uninstaller Script

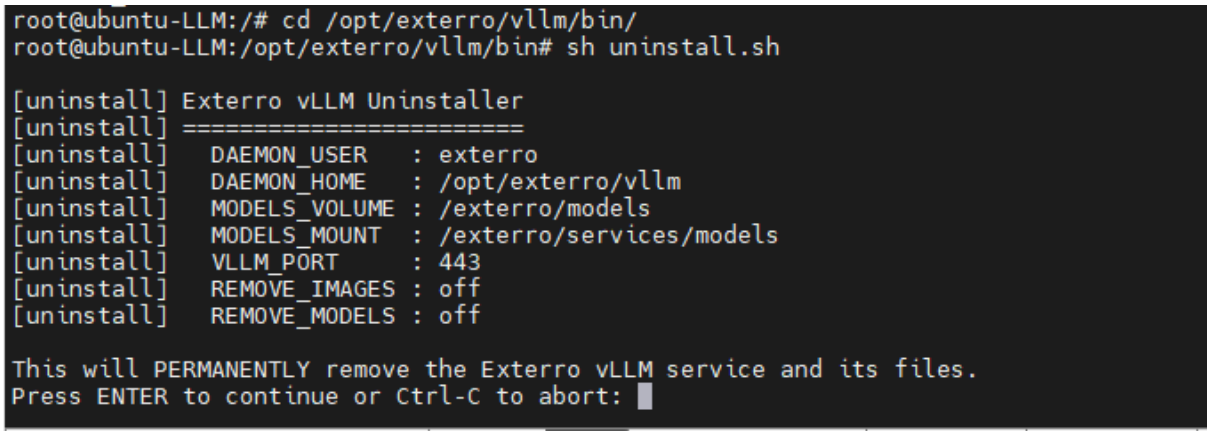
1. **Stop Active Service & Verify Containers:**

```
sudo su
systemctl stop exterro-vllm.service
docker ps
# Verify no containers remain active
```

2. *(Optional)* **Set Purge Flags:**

By default, docker images and model weights are preserved. To permanently purge them, set these variables before running the script:

Purge Target	Command	Action
Docker Images	export REMOVE_IMAGES=on	Purges vLLM Docker base layers
Model Files	export REMOVE_MODELS=on	Permanently deletes local model weights



3. **Execute Uninstallation:**

```
cd /opt/exterro/vllm/bin/
sh uninstall.sh
```

*Press **ENTER** to confirm, or **Ctrl+C** to abort.*

4. **Post-Cleanup Housekeeping:** Remove initial installation assets:

```
sudo su
rm -f /home/exterro/deployments/sfx_deploy/exterro-vllm-manylinux.sfx
```

Method B: Manual Hard Decommissioning

Use this fallback procedure if the automated script encounters errors:

1. **Elevate privileges and stop service:**

```
systemctl stop exterro-vllm.service
```

2. **Delete local model weight binaries:**

```
cd /exterro/models/
rm -rf Qwen3-30B-A3B-Instruct-2507
```

3. **Purge installation directories and system configs:**

```
cd /opt/exterro/vllm/
rm -rf /opt/exterro/vllm/
cd /opt/
rm -rf exterro/
```

4. **Remove installation SFX artifacts:**

```
cd /home/exterro/deployments/sfx_deploy/
rm -f exterro-vllm-manylinux.sfx
```

Contact Exterro

If you have any questions, please refer to this document, or any other related materials provided to you by Exterro. For usage questions, please check with your organization's internal application administrator. Alternatively, you may contact your Exterro Training Manager or other Exterro account contact directly.

For technical difficulties, support is available through support@exterro.com.

Contact:

Exterro, Inc.

2175 NW Raleigh St., Suite 110

Portland, OR 97210.

Telephone: 503-501-5100

Toll Free: 1-877-EXTERRO (1-877-398-3776)

Fax: 1-866-408-7310

General E-mail:info@exterro.com

Website: www.exterro.com