Okay, starting fresh with a new Ubuntu installation is often the cleanest way to resolve persistent driver or configuration issues. Let's get your NVIDIA GPU working with Docker correctly this time, including verification steps.

We will follow this order:

1. Install NVIDIA Host Drivers (Using NVIDIA's official repository).
2. Verify Host Driver Installation (nvidia-smi).
3. Install Docker Engine (Using Docker's official repository).
4. Verify Docker Installation (hello-world).
5. Install NVIDIA Container Toolkit.
6. Verify GPU Access within Docker (nvidia-smi inside container).

**Step 1: Install NVIDIA Host Drivers**

Using NVIDIA's official repositories is generally recommended for compatibility with their toolkits.

1. **Update System:** Open a terminal and update your package list and upgrade existing packages.  
   Bash  
   sudo apt update  
   sudo apt upgrade -y
2. **Install Prerequisite:** Needed for building kernel modules.  
   Bash  
   sudo apt install -y build-essential
3. **Add NVIDIA CUDA Repository:**
   * Go to the official NVIDIA CUDA Toolkit Downloads page: <https://developer.nvidia.com/cuda-downloads>
   * Select: Linux -> x86\_64 -> Ubuntu -> Your Ubuntu Version (e.g., 22.04) -> deb (network)
   * The page will generate a sequence of commands specific to your selections. **Execute these commands precisely** in your terminal. They will typically involve:
     + Downloading a .pin file.
     + Downloading a repository setup .deb file.
     + Installing the .deb file (sudo dpkg -i ...).
     + Adding the CUDA public GPG key.
     + Updating the apt package list (sudo apt update).
   * *Follow the instructions on the NVIDIA website carefully for this step.*
4. **Install NVIDIA Driver:** Once the repository is set up and sudo apt update has run, install the driver metapackage. This usually installs the latest recommended driver from that repository.  
   Bash  
   sudo apt install -y cuda-drivers  
     
   *(Alternatively, if you know a specific driver branch you need, like 550, you could use sudo apt install nvidia-driver-550, but cuda-drivers is generally preferred when using the CUDA repo).*
5. **Reboot:** A reboot is essential for the new driver to load correctly.  
   Bash  
   sudo reboot

**Step 2: Verify Host Driver Installation**

1. After rebooting, log back in and open a terminal.
2. Run the NVIDIA System Management Interface tool:  
   Bash  
   nvidia-smi
3. **Expected Output:** You should see a table detailing your NVIDIA GPU(s), the installed Driver Version, and the compatible CUDA Version. **If this command works correctly**, your host driver is installed properly. This confirms the base library (libnvidia-ml.so.1) is accessible, addressing the root cause of your previous issue.

**Step 3: Install Docker Engine**

We'll use Docker's official repository for the latest stable version.

1. **Uninstall Old Versions (Precautionary):**  
   Bash  
   sudo apt remove docker docker-engine docker.io containerd runc -y
2. **Set up Docker's Repository:**  
   Bash  
   # Add Docker's official GPG key:  
   sudo apt update  
   sudo apt install ca-certificates curl -y  
   sudo install -m 0755 -d /etc/apt/keyrings  
   sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc  
   sudo chmod a+r /etc/apt/keyrings/docker.asc  
     
   # Add the repository to Apt sources:  
   echo \  
    "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \  
    $(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \  
    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null  
   sudo apt update
3. **Install Docker Packages:**  
   Bash  
   sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

**Step 4: Verify Docker Installation**

1. Run the hello-world container to ensure Docker is working:  
   Bash  
   sudo docker run hello-world
2. **Expected Output:** You should see a message starting with "Hello from Docker!". This confirms Docker itself is installed and running.
3. **(Optional but Recommended) Manage Docker as a non-root user:** Add your user to the docker group to avoid typing sudo for every docker command.  
   Bash  
   sudo usermod -aG docker $USER  
   * **Important:** You need to **log out and log back in** (or run newgrp docker in your current shell) for this group change to take effect.

**Step 5: Install NVIDIA Container Toolkit**

This toolkit bridges Docker and your NVIDIA driver.

1. **Set up NVIDIA Container Toolkit Repository:**  
   Bash  
   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 > /dev/null  
   sudo apt update
2. **Install the Toolkit Package:**  
   Bash  
   sudo apt install -y nvidia-container-toolkit
3. **Restart Docker Daemon:** The toolkit installation should configure Docker, but you need to restart Docker to apply the changes.  
   Bash  
   sudo systemctl restart docker

**Step 6: Final Verification - GPU Access in Docker**

This is the ultimate test to ensure everything is working together and the previous issues are resolved.

1. Run nvidia-smi *inside* a CUDA container, telling Docker to use the GPU:  
   Bash  
   # If you added your user to the docker group and logged out/in, you can omit sudo  
   # Using a CUDA 12.4 base image for Ubuntu 22.04 as an example.  
   # Adjust the tag if your nvidia-smi showed a different major CUDA version capability  
   # or if you are on a different Ubuntu base (e.g., 24.04).  
   docker run --rm --gpus all nvidia/cuda:12.4.1-base-ubuntu22.04 nvidia-smi
2. **Expected Output:** You should see the **same nvidia-smi output table** as you saw in Step 2 (when run directly on the host), but this time it's being generated from *within* the Docker container.

**Conclusion:**

If the final docker run ... nvidia-smi command executes successfully and displays your GPU information, you have correctly installed the NVIDIA driver, Docker, and the NVIDIA Container Toolkit. The previous issues with library paths (libnvidia-ml.so.1) should be resolved by this clean, repository-based installation process. You can now run GPU-accelerated Docker containers.

**Sources**

1. <https://github.com/Samiii777/AMD_MachineLearning>

2. <https://forums.docker.com/t/an-error-while-setting-up-container/140117>

3. <https://www.bujarra.com/ollama-empezando-con-la-ia-local/?lang=ja>

4. <https://github.com/AbdelrahmanGaberMohamed/Terraform-aws-dev-ec2>

5. <https://github.com/leizhenyu-lzy/Blog>

6. <https://github.com/21Kdev/IaC-lib>