1-1. Software installation on Windows WSL (optional)

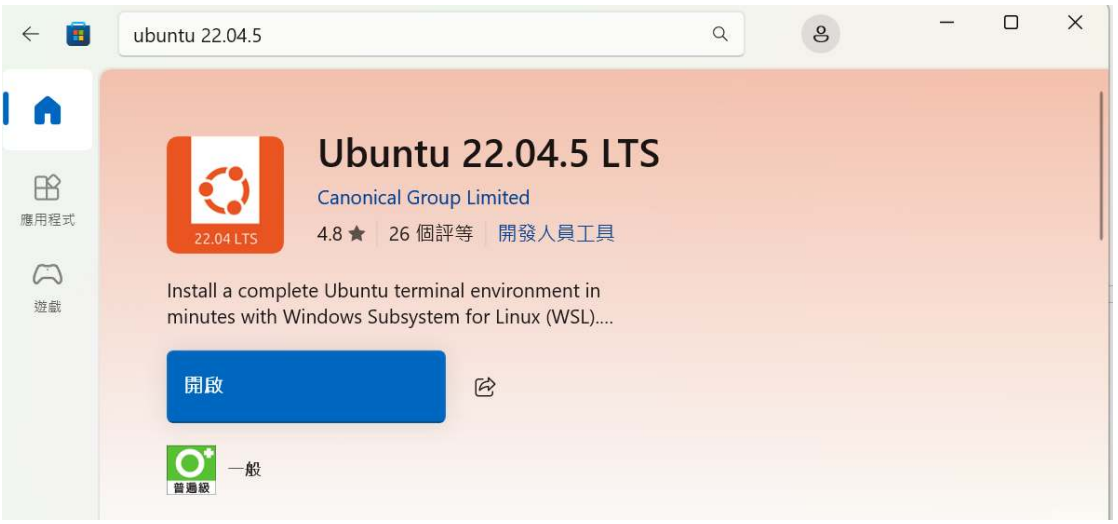
This section allows the user to install AI TOP Utility software on Windows 11 via Windows Subsystem for Linux (WSL)

1-1-1. Prerequisites:

First, please go to “Applications’ and uninstall any existing versions of Ubuntu (if already installed).
It is recommended that users install Windows 11 and WSL2 for this software.
WSL 2 (Windows Subsystem for Linux 2) is a powerful tool that allows you to run a Linux distribution alongside your Windows operating system. This section provides a step-by-step guide on how to install WSL 2 on your Windows system, enabling you to seamlessly work with Linux commands and applications within your Windows environment.

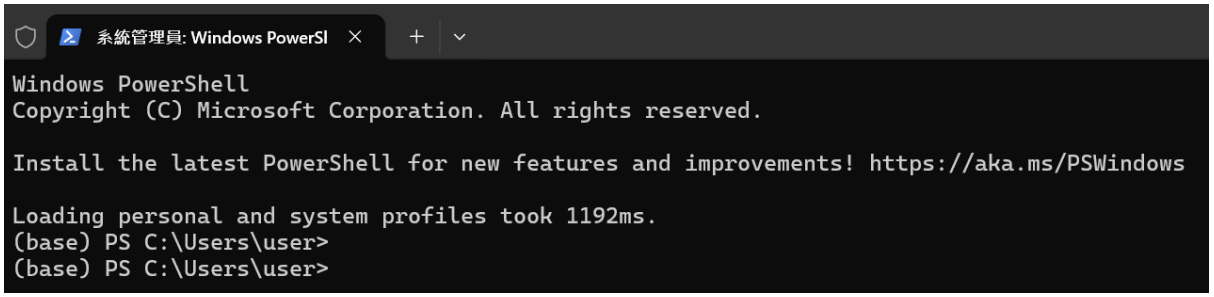
1-1-2. Download Ubuntu 22.04.5 LTS on Windows:

On Windows, search to open ‘Microsoft Store’ and find Ubuntu 22.04.5 LTS, then click ‘ Free’, then click ‘Get’ to install Ubuntu 22.04.5 LTS.



1-1-3. Update WSL:

On the Desktop, right-click on the “Start” menu > select “Windows Powershell (System Administrator)”



Update WSL by this command:

```
wsl --update
```

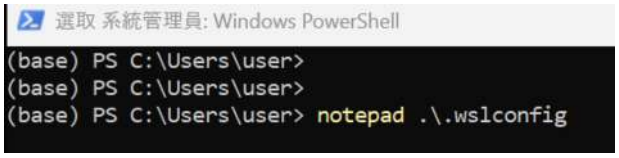
1-1-4. Setting WSL system memory

By default, **WSL2** automatically allocates up to **50% of your total system RAM** to the virtual machine (VM) where your Linux distribution runs. However, you can manually configure how much memory WSL should use by editing the .wslconfig file.

On the Desktop, right-click on the "Start" menu > select "Windows Powershell (System Administrator)"

Create the .wslconfig file and edit it using the following commands:

```
cd ~
notepad .\.wslconfig
```



`memory=192GB` → WSL will use a maximum of **192GB of RAM** instead of the default 50%. Adjust this based on your needs.

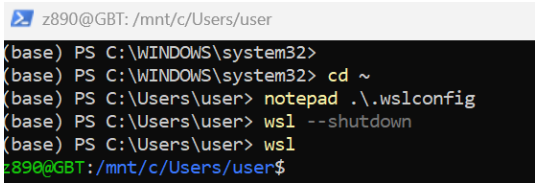
`swap=8GB` → Creates an **8GB swap file** in case RAM usage exceeds the limit (optional).

After saving the .wslconfig file, restart WSL for the settings to take effect:

```
wsl --shutdown
```

Then, restart WSL by launching it again:

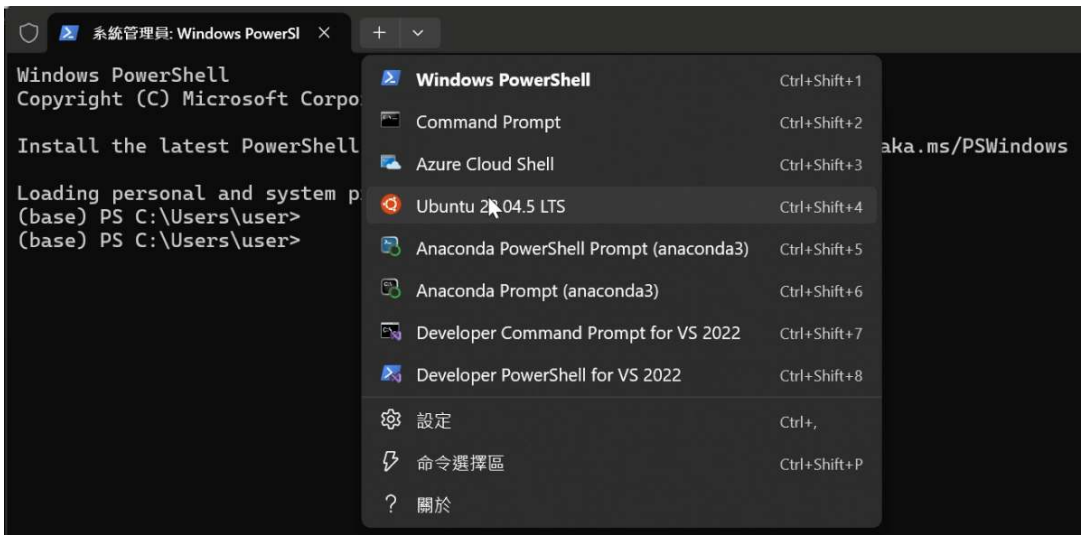
```
wsl
```



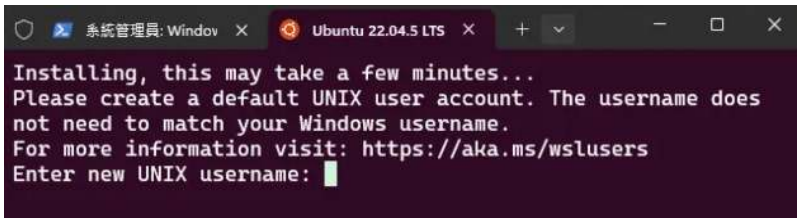
This should display the new memory allocation as per your .wslconfig settings.

1-1-5. Boot Ubuntu 22.04.5 LTS and Install missing dependencies:

Click to open the drop-down menu and select "Ubuntu 22.04.5 LTS"



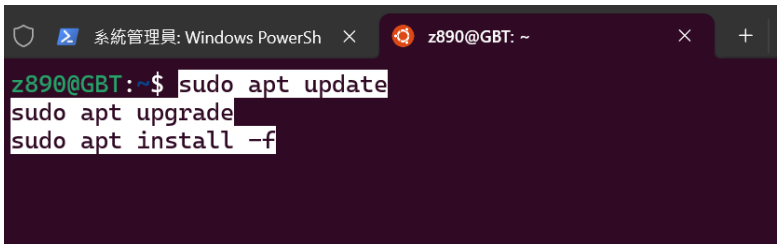
Follow the on-screen instructions to enter your username (any name) and password.



Continue to update the local package index in the Ubuntu environment:

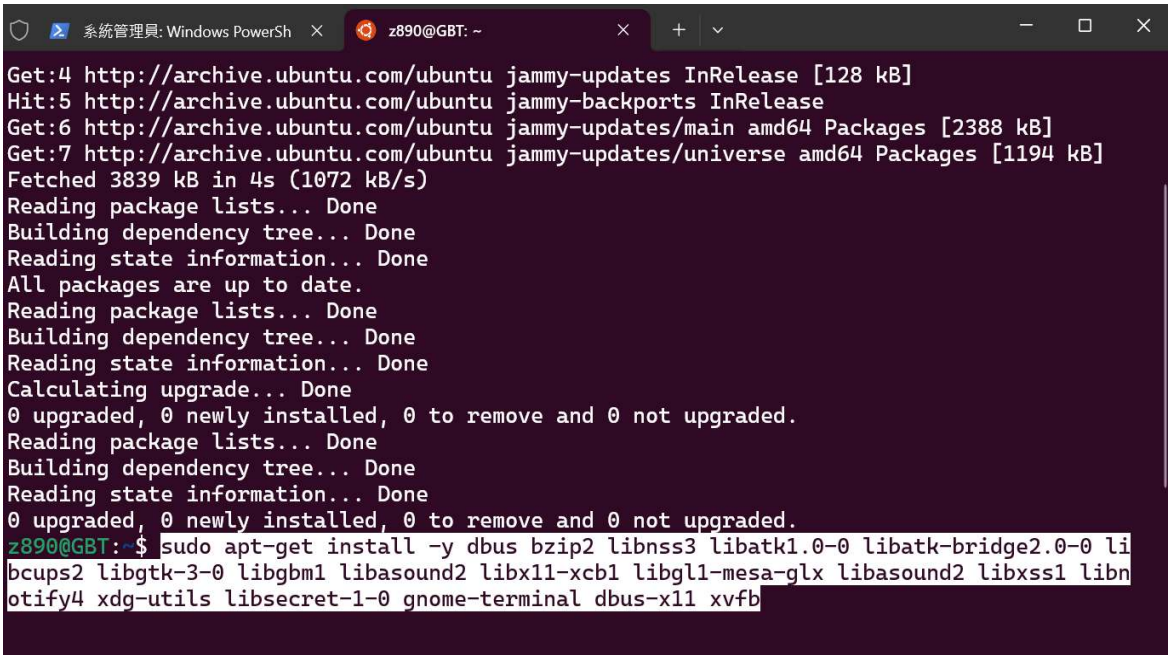
Please reset the time (hh:mm) on your OS if it does not match your watch/clock.

```
sudo apt update
sudo apt upgrade
sudo apt install -f
```



Install all missing dependencies:

```
sudo apt-get install -y dbus bzip2 libnss3 libatk1.0-0 --fix-missing
sudo apt-get install -y libatk-bridge2.0-0 libcups2 --fix-missing
sudo apt-get install -y libgtk-3-0 libgbm1 libasound2 --fix-missing
sudo apt-get install -y libx11-xcb1 libgl1-mesa-glx --fix-missing
sudo apt-get install -y libasound2 libxss1 libnotify4 --fix-missing
sudo apt-get install -y xdg-utils libsecret-1-0 --fix-missing
sudo apt-get install -y gnome-terminal dbus-x11 xvfb --fix-missing
```



1-1-6. Install VcXsrv (XLaunch.exe)

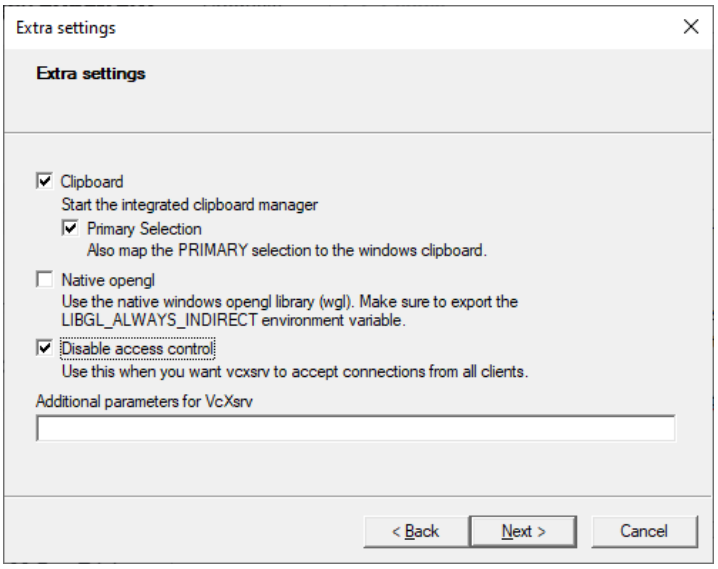
First, download the VcXsrv installer on Windows: VcXsrv

If you want WSL 2 to use VcXsrv:

- When running VcXsrv, select 'Disable access control.'
- Windows Firewall needs to allow VcXsrv (XLaunch.exe, VcXsrv.exe) to pass through, or you can turn off the firewall directly. Otherwise, WSL 2 will not be able to connect to VcXsrv, and the following error will occur: 'Authorization required, but no authorization protocol specified.'

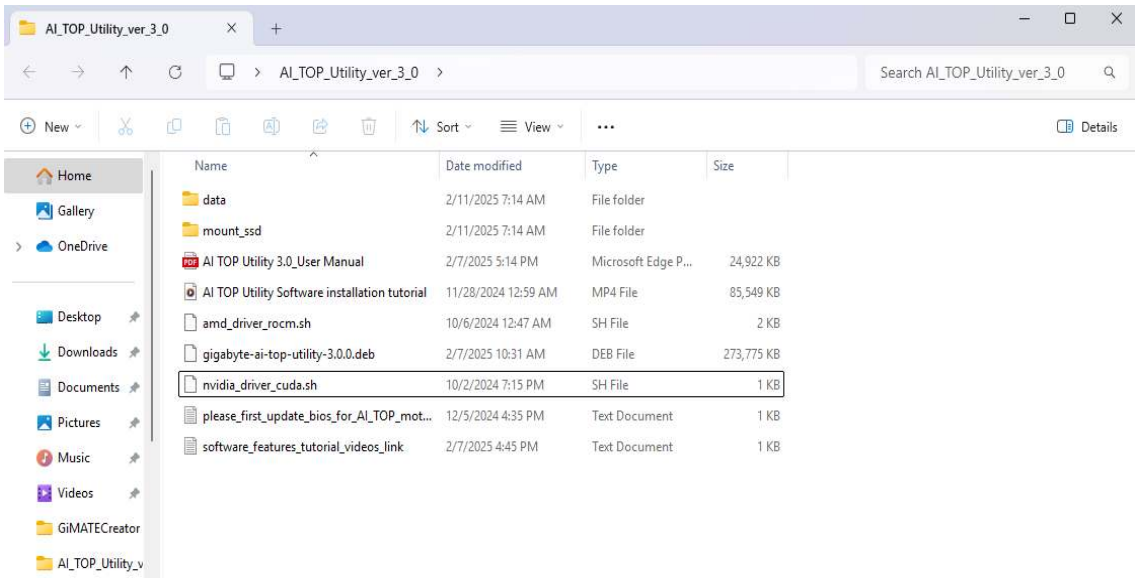
In Display settings, select Multiple windows.

In the Extra settings, select 'Disable access control.'

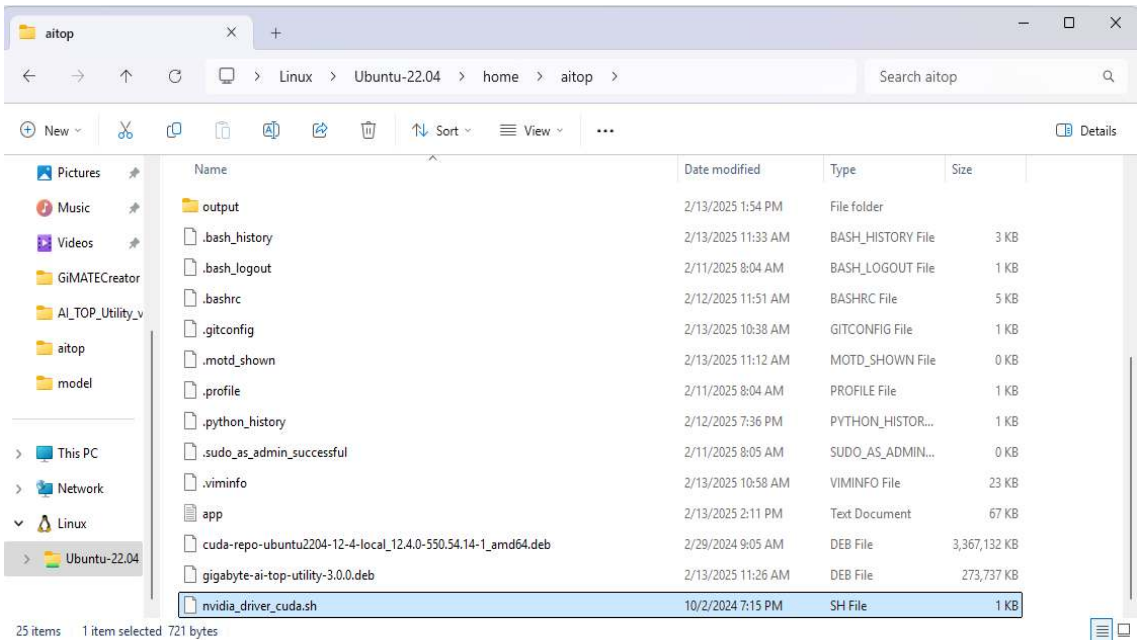


1-1-7. Install GPU driver and CUDA tool on WSL:

The **NVIDIA GPU driver file** is named `nvidia_driver_cuda.sh` (or the **AMD GPU driver file** is named `amd_driver_rocm.sh)`. Both files are located in the folder (**AI_TOP_Utility_ver_x_x**).



Please copy `nvidia_driver_cuda.sh` (or `amd_driver_rocm.sh)` based on your GPU type (NVIDIA or AMD) to \\wsl.localhost\Ubuntu-22.04\home\username. Make sure to check your username. For example, if $USER$ = z890, the destination path should be: \\wsl.localhost\Ubuntu-22.04\home\z890.



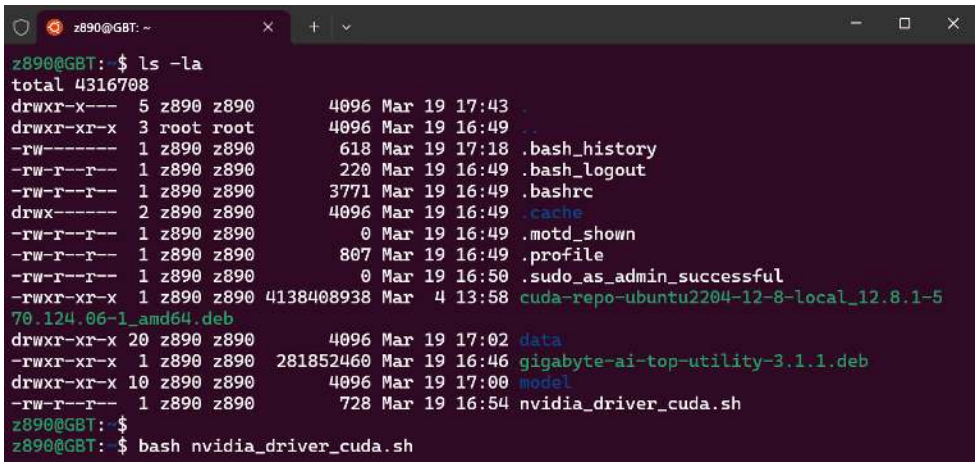
Open a terminal in another command-line window by clicking the **Ubuntu 22.04.5 LTS** desktop icon. Then, run the following command to install GPU driver and CUDA tool on WSL:

For **AMD GPUs**, install the ROCm driver:

```
cd ~
bash ./amd_driver_rocm.sh
```

For **NVIDIA GPUs**, install the NVIDIA driver and CUDA tools:

```
cd ~
bash ./nvidia_driver_cuda.sh
```



1-1-8. Install Google Chrome for Linux

Some features in the AI TOP Utility require a fully functional graphical browser within the WSL environment. To ensure complete compatibility and performance, you must install the Linux version of Google Chrome inside WSL.

> **Note: This is not the same as the Chrome browser installed on your Windows system. It must be installed within the Linux (Ubuntu) environment on WSL.**

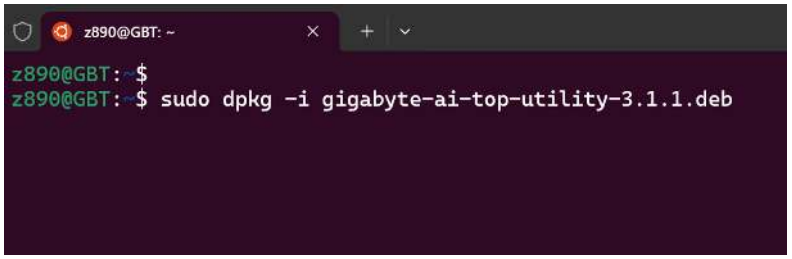
Follow these steps:

1. Open your Ubuntu terminal in WSL.
2. Change to the temporary directory:
   ```
   cd /tmp
   ```
3. Download the latest installation package:
   ```
   wget
   https://dl.google.com/linux/direct/google-chrome-stable_current_amd64
   .deb
   ```
4. Install the package with:
   ```
   sudo apt install --fix-missing
   ./google-chrome-stable_current_amd64.deb
   ```
   - **The `--fix-missing` option helps resolve any missing dependencies during installation.**
   - **The `./` prefix indicates the `.deb` file is located in the current directory. If the file is in a different location, adjust the path accordingly.**

1-5-8. Install AI TOP Utility software on WSL

Place the gigabyte-ai-top-utility-4.0.0.deb file under \\wsl.localhost\Ubuntu-22.04\home\username. Check the user name, for example: $USER$ = trx50. Open a terminal in another command line window by clicking the Ubuntu 22.04.5 LTS desktop icon, then run:

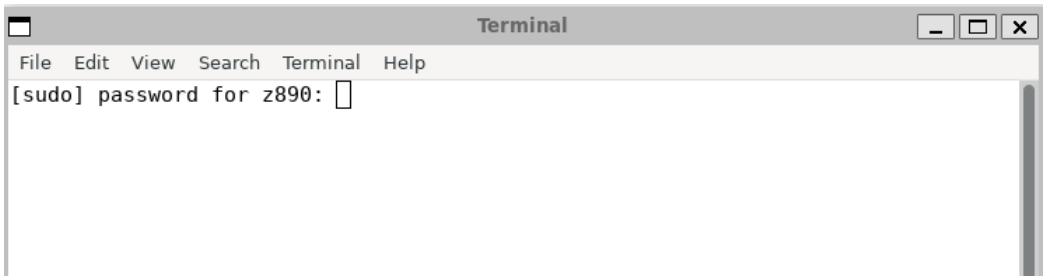
```
cd ~
sudo dpkg -i gigabyte-ai-top-utility-4.0.0.deb
```



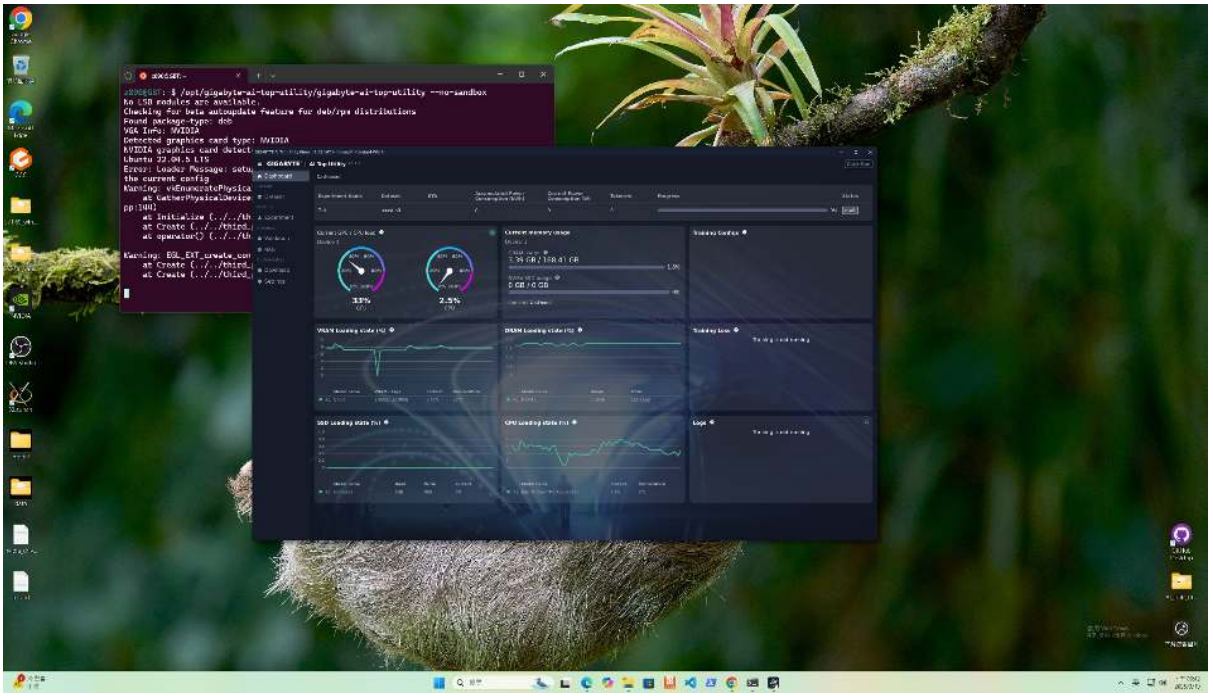
1-5-9. Run AI TOP Utility GUI

```
/opt/gigabyte-ai-top-utility/gigabyte-ai-top-utility --no-sandbox
```

When starting it for the first time, you'll need to install the necessary packages. Multiple windows may open during this process. Follow the prompts to enter your user password or press ‘Y’ when required. The installation may take around 30 minutes, depending on your network speed.



The software's GUI will pop up on your screen.



Now, you can start using the AI Top Utility software in the Windows WLS environment.

Please note that WSL is a virtualization layer, so system scheduling and resource allocation may not be as direct or efficient as on native Linux, which could affect computing speed.

1-2. Mount SSD for WSL (optional):

In cases where the model size is too large for VRAM and DRAM to handle during training, users need to offload training memory to an NVMe SSD for additional capacity.

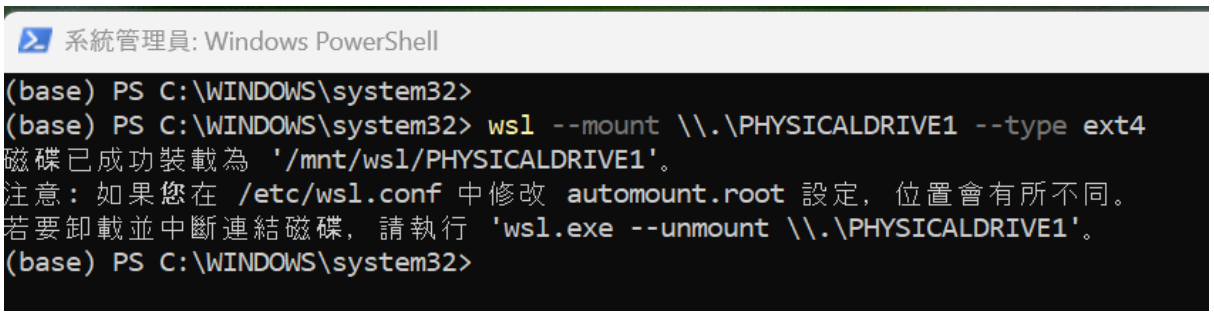
First, users must mount the SSD on WSL by Windows Powershell with System Administrator privileges.

On the Desktop, right-click the “Start” menu > select “Windows Powershell (System Administrator)”

Command for mounting SSD on WSL:

```
wsl --mount \\.\PHYSICALDRIVE1 --type ext4
```

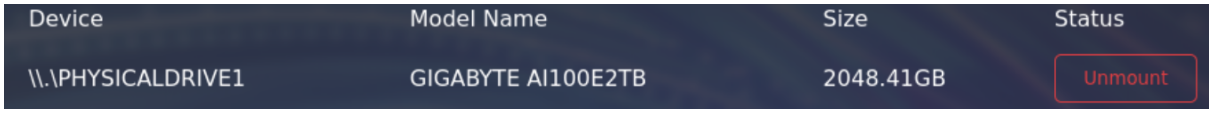
This command is used in Windows Subsystem for Linux (WSL) to mount a physical drive with an **EXT4** filesystem.



Then, it successfully mounts a physical drive (named PHYSICALDRIVE1) on WSL at (/mnt/wsl/PHYSICALDRIVE1).

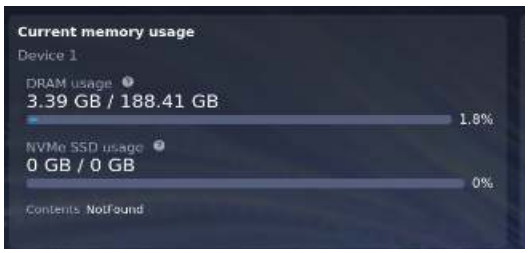
Second, users must enter their OS user password and click the **'Mount'** button in the **Settings** tab. This will mount the physical drive in WSL (`/mnt/wsl/PHYSICALDRIVE1`) to the Linux system's training path (`/media/user/nvme0`).

Then, the SSD is successfully mounted in WSL.



However, if you do not run the command `wsl --mount \\.\PHYSICALDRIVE1 --type ext4` in “Windows Powershell (System Administrator)” before clicking the **'Mount'** button, an error will appear: "Mount failed! 500: Command failed: mount: /media/user/nvme0: special device /mnt/wsl/PHYSICALDRIVE1 does not exist."

If mounting the SSD still fails after completing the **First** and **Second** steps, it is likely that the SSD is formatted as **NTFS** instead of the required **EXT4** format. In this case, you will need to reformat the SSD to **EXT4** before attempting to mount it again.

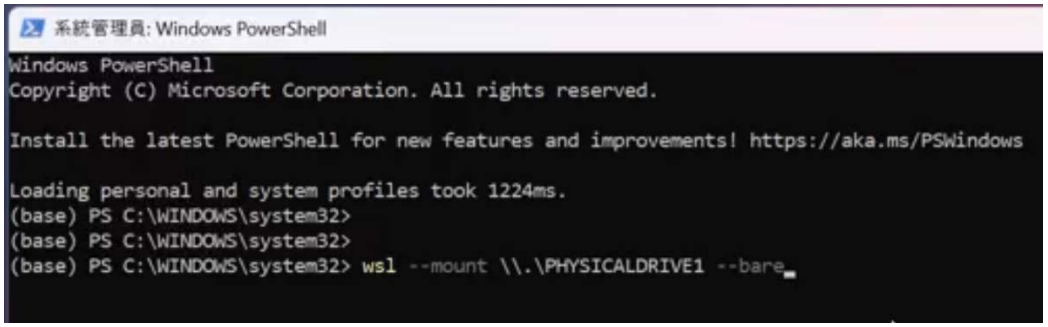


1. Open Windows Terminal as Administrator (PowerShell):

First, mount the SSD to WSL without specifying a filesystem. This step mounts the entire physical SSD directly into WSL.

```
wsl --mount \\.\PHYSICALDRIVE<disk_number> --bare
```

```
wsl --mount \\.\PHYSICALDRIVE1 --bare
```



2. Enter your WSL environment (e.g., Ubuntu) and verify the disk has been mounted:

```
sudo fdisk -l
```

Look for the mounted disk path, typically something like `/dev/sdb`.

3. Format the SSD to EXT4 (e.g., Ubuntu)

```
sudo mkfs.ext4 /dev/sdb
```

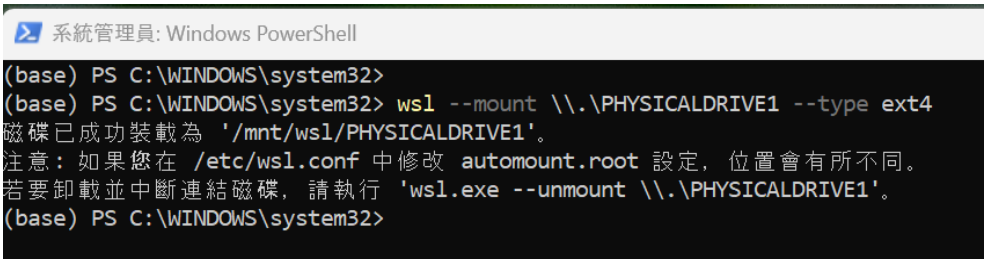
This step will erase all data on the SSD. This formats the SSD /dev/sdb to the EXT4 filesystem.

4. Unmount the SSD (PowerShell with Administrator):

```
wsl --unmount \\.\PHYSICALDRIVE1
```

5. Remount the SSD with EXT4 filesystem type (PowerShell with Administrator):

```
wsl --mount \\.\PHYSICALDRIVE1 --type ext4
```



Now your SSD is mounted in WSL with the EXT4 format.

6. Complete the mounting in the application settings:

As the **Second** step, enter your OS user password and click the "**Mount**" button in the Settings tab. This will mount the physical drive from WSL (/mnt/wsl/PHYSICALDRIVE1) to the Linux system's training path (/media/user/nvme0).

Then, the SSD is successfully mounted in WSL.