**AIfinity Research**

*Your GenAI Launchpad !*

**Amazon EC2 Instance Setup for Hosting WAN 2.1 Models**

**Purpose** : Step-by-Step guide to configure AWS EC2 g5.2xlarge CPU with NVIDIA AMI *Deep Learning OSS Nvidia Driver AMI GPUPyTorch 2.7 (Ubuntu 22.04)*

**About WAN Models** : WAN, particularly the Wan2.1 video generation model, was developed and open-sourced by Alibaba in early 20252. It’s part of their broader push into generative AI, and they’ve released it under an open-source license to encourage community use and innovation.

* **Wan2.1-1.3B**: A lightweight model that runs on ~8 GB VRAM, ideal for consumer-grade GPUs like RTX 4090. Great for real-time prototyping and experimentation.
* **Wan2.1-7B**: Mid-tier model offering a balance between performance and resource usage.
* **Wan2.1-14B**: The flagship model with state-of-the-art performance in motion realism, instruction following, and multilingual video generation.

WAN Models are available at Huggingface

<https://huggingface.co/Wan-AI>

<https://huggingface.co/Wan-AI/Wan2.1-VACE-1.3B>

These models support a wide range of tasks:

* Text-to-video
* Image-to-video
* Video editing
* Text-to-image
* Video-to-audio
* Multilingual captioning (e.g., English and Chinese)

They’re all open-source, which makes them especially appealing for developers and researchers looking to build or fine-tune their own video generation pipelines.

**Real World Applications**

* AI-generated short films: Artists and studios can generate entire video sequences from text prompts or storyboards.
* Video editing & enhancement: Automatically upscale, color-correct, or modify scenes using text or image guidance.
* Multilingual video content: Generate videos with embedded captions in multiple languages—great for global audiences.

🛍️ E-commerce & Marketing

* Product showcase videos: Turn product images into dynamic 360° or lifestyle videos.
* Ad creatives: Generate personalized video ads based on user behavior or regional preferences.

🏫 Education & Training

* Interactive learning modules: Create visual explanations of complex topics (e.g., physics simulations or historical reenactments).
* Language learning: Generate scenario-based videos for immersive language practice.

🏗️ Engineering & Design (this one’s right up your alley!)

* CAD-to-video visualization: Animate product designs or tool mechanisms for client presentations or internal reviews.
* Simulation of assembly processes: Generate instructional videos from CAD models to train technicians or customers.

🧠 AI Coaching & Content Creation

* Tutorial generation: Produce engaging video tutorials for your AI coaching program.
* Dynamic course content: Auto-generate visual examples or explainer videos based on lesson plans or user queries.

📰 Journalism & Social Media

* News summarization: Turn headlines or articles into short video summaries.
* Social storytelling: Creators can generate stylized video content from a single image or phrase.

**Deploying WAN Models to AWS EC2 – Why ?**

Deploying WAN models like Wan2.1 on private AWS EC2 instances offers companies a powerful blend of performance, control, and scalability—especially when working with compute-heavy tasks like video generation. Here's how it helps:

🔐 1. Data Privacy & Security

Running models on private EC2 instances ensures sensitive data never leaves your environment. This is crucial for industries like healthcare, finance, or defense where compliance and confidentiality are non-negotiable.

⚙️ 2. Customization & Fine-Tuning

You can fine-tune WAN models on proprietary datasets or integrate them into custom pipelines—something that’s often limited or restricted on public platforms.

🚀 3. Scalable Performance

Using GPU-optimized EC2 instances (like g5.xlarge with NVIDIA A10G) allows for high-performance video generation without investing in physical infrastructure. You can scale up or down based on demand.

💸 4. Cost Efficiency

With pay-as-you-go pricing, companies avoid the upfront costs of on-prem hardware. Spot instances or reserved pricing can further reduce expenses.

🧩 5. Seamless Integration

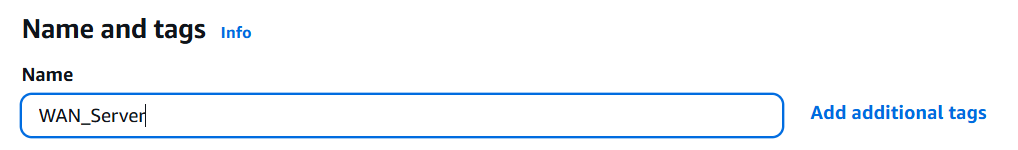
Deploying on AWS makes it easier to integrate with other cloud-native services—like S3 for storage, Lambda for automation, or SageMaker for orchestration.

🛠️ 6. DevOps & Automation

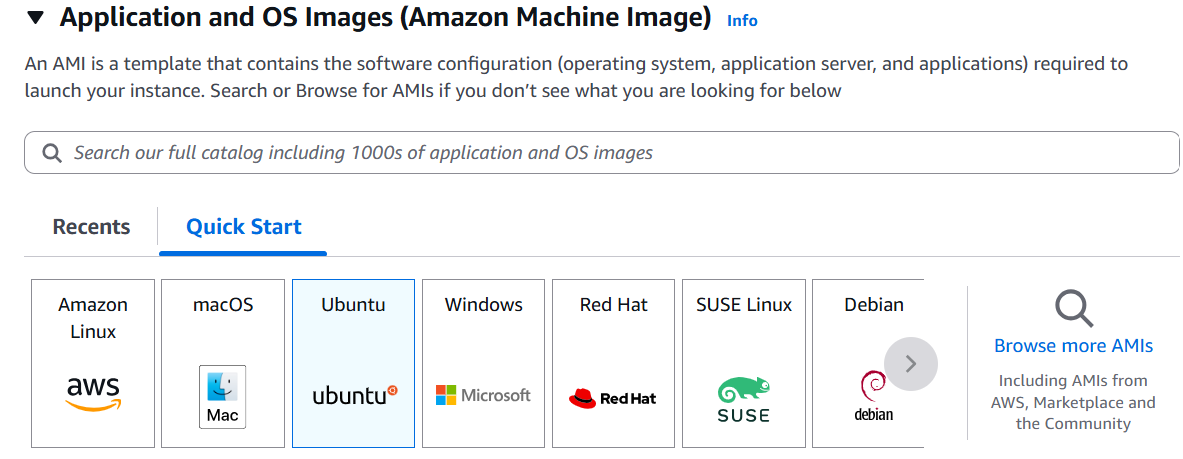
You can automate deployment, monitoring, and updates using tools like Terraform, CloudWatch, and CI/CD pipelines, making WAN models production-ready.

**AWS EC2 Creation & Installation Steps**

* 1. Login/Sign up to AWS Console <https://aws.amazon.com/console/>
  2. Go to EC2
  3. Launch Instance --> Give it a name “WAN\_Server”

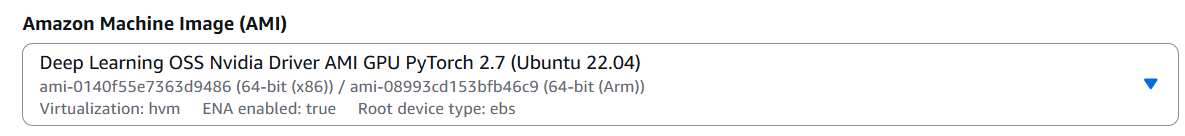


* 1. Application and OS Image . Choose Ubuntu

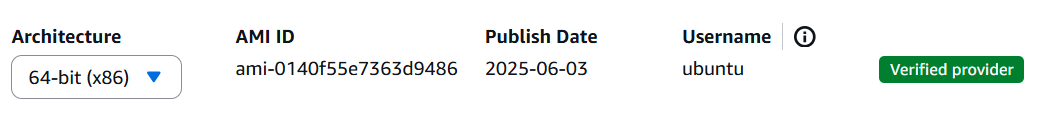
****

* 1. Choose Amazon Machine Image ( AMI ) – Deep Learning OSS Nvidia Driver AMI GPU

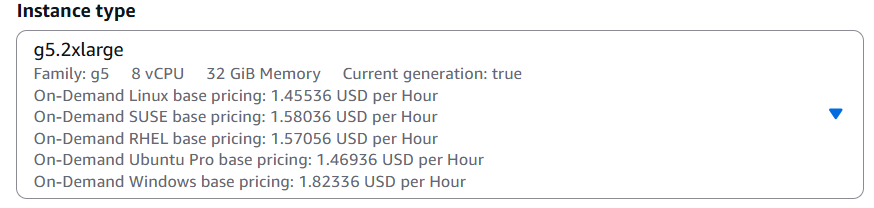
PyTorch 2.7 (Ubuntu 22.04)

****

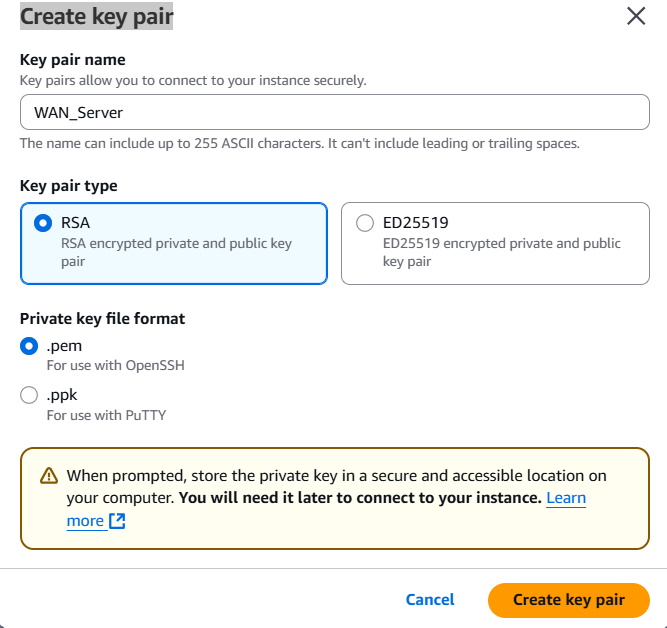
* 1. Architecture – Keep 64-bit ( x86)



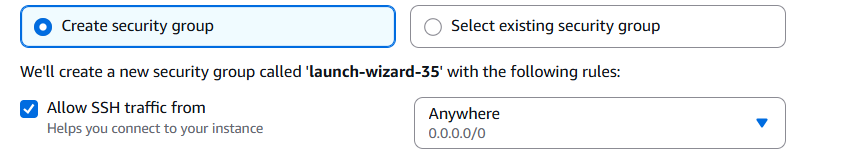
* 1. Instance Type – g5.2xlarge

****

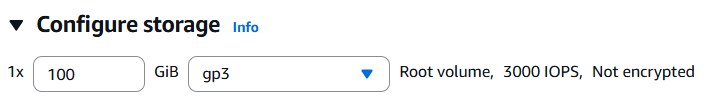
8 . Create new key pair – “WAN\_Server” , type .pem ( Don’t forget to download , save it in a safe location )



9 . Network Settings . Choose “Create Security Group” , “Allow SSH traffic from Anywhere 0.0.0.0“



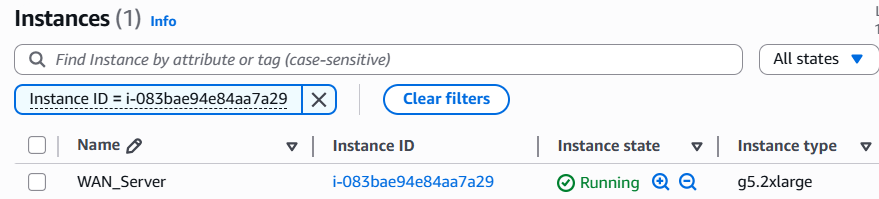
10 . Configure Storage – make it 100 GB ( SSD)



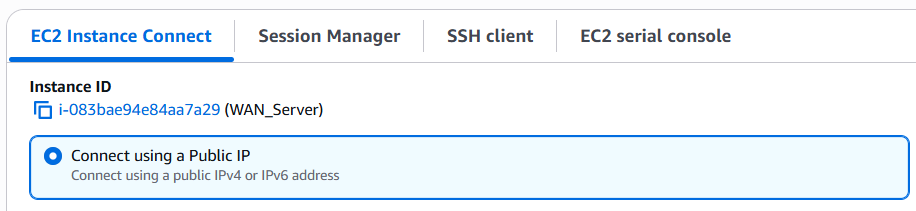
*Note : to deploy larger models like WAN 2.1 14B , you have to choose higher SSD size as per the model size*

11. Click on Launch Instance. Wait for a minute for the instance to spin up

12. Go to Instances tab -> Click on the instance



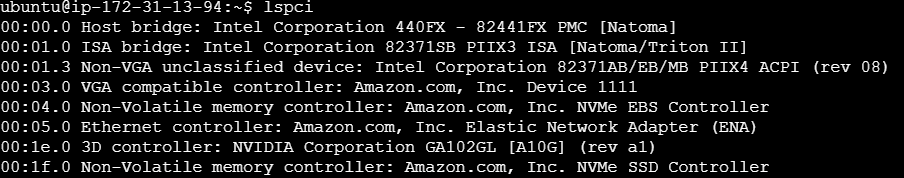
13 . Choose “EC2 Instance Connect” and “Connect using a public IP”



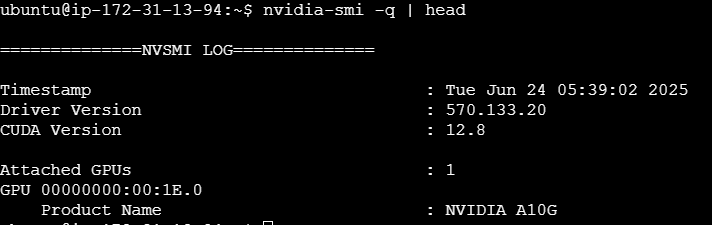
14. Wait for prompt to open up

15 . Execute commands

CMD:/>lspci # to check if GPU is running



CMD:/>nvidia-smi -q | head # to check if CUDA drivers are installed

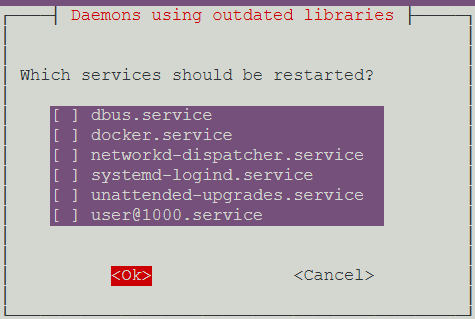


16. Execute commands

*sudo apt update && sudo apt upgrade -y && sudo apt install -y python3-pip git*

*sudo apt install -y python3.10 python3.10-venv python3.10-dev*

*When prompted with below dialog box, use tab to select “Ok” and Press Enter*

**

17. Create a new user aifinity , add it to sudo group to grant permissions to execute command with superuser rights

*sudo -i*

*useradd aifinity*

*passwd aifinity*

*mkdir /home/aifinity*

*chown aifinity:aifinity /home/aifinity*

*usermod -aG sudo aifinity*

*su - aifinity*

*bash*

18 . Create a python virtual environment

*python3 -m venv myenv*

*source myenv/bin/activate*

*pip install --upgrade pip*

*pip install torch torchvision torchaudio packaging wheel*

19 . Download Wan2.1 code base from Github

*git clone https://github.com/Wan-Video/Wan2.1.git*

*cd Wan2.1*

*pip install -r requirements.txt*

20. Add bin to path

*echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.bashrc*

*source ~/.bashrc*

21. Re-install flash\_attn with NVIDIA’s custom packages for CUDA where prebuilt wheels for flash-attn are hosted. These wheels are optimized for specific GPU architectures and CUDA versions

*pip install flash-attn --no-build-isolation --extra-index-url https://pypi.nvidia.com*

22. Install Huggingface’s CLI to download models

*pip install huggingface\_hub[cli]*

23. Login to Huggingface, generate your HF token , store it in a notepad

24. huggingface-cli login – paste the HK\_TOKEN value

25. Add token as git credential? (Y/n) n -> Say No

26. Download the required WAN 2.1 model

*huggingface-cli download Wan-AI/Wan2.1-T2V-1.3B --local-dir ./models/Wan2.1-T2V-1.3B*

27. Download Aifinity’s demo code for text2video

*git clone* [*https://github.com/aifinityresearch/demo\_text2video.git*](https://github.com/aifinityresearch/demo_text2video.git)

28 . Cd /demo\_text2video/server

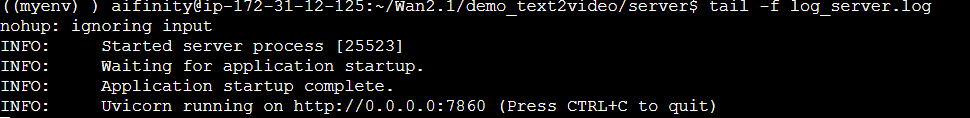
28a . pip install -r requirements.txt

29 . Start the uvicorn server

*nohup uvicorn server:app --host 0.0.0.0 --port 7860 > server.log 2>&1 &*

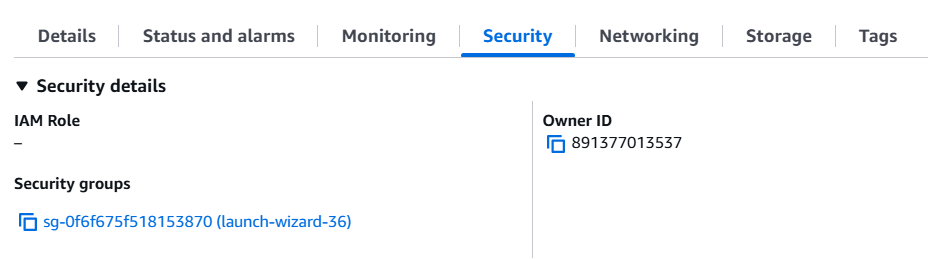
*20. Check the log\_server.log*

*tail -f log\_server.log*

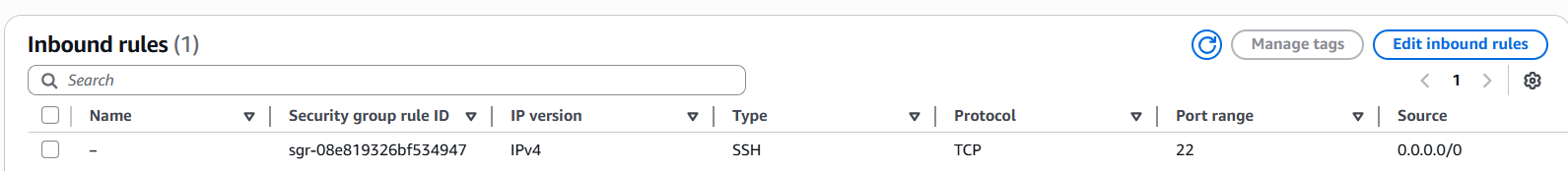
**

*Ctrl + C*

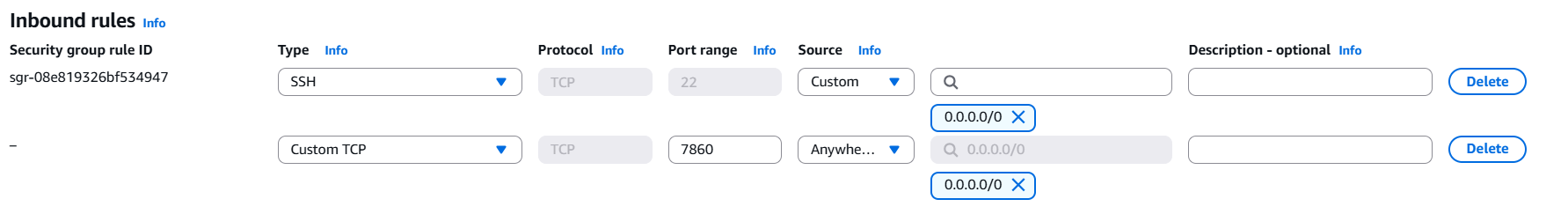
21. Go to EC2 in AWS Console 🡪 Select your ec2 instance 🡪 Security Tab 🡪Click on Security Group “sg-0f6..”



22. Click on “Edit Inbound Rules”

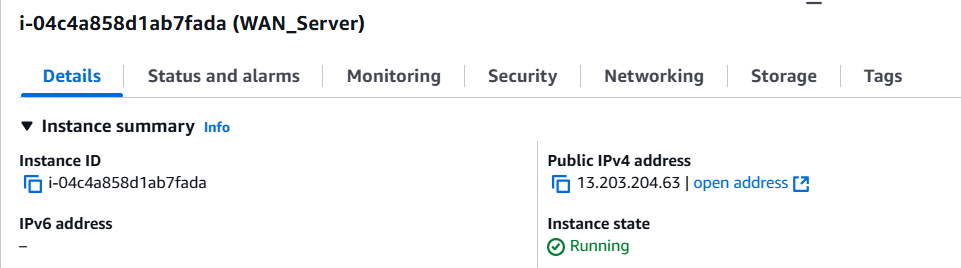


23. Add New Rule “Custom TCP” , port 7860 , Source -> From Anywhere ( 0.0.0.0/0)



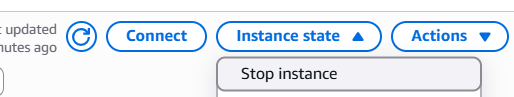
N*ote : In production systems , its not recommended to expose your EC2 instance to public . The EC2 instances in production are always deployed in a private subnet with private IP only* . For demo purposes only EC2 is being exposed via public IP

24. Copy the Public IP address ( ex 13.203.204.63) of the ec2 and use it in the Client Code



25. Warning : Beware of keeping the EC2 running which will attract cost . Remember g5.2xlarge CPU’s cost ~1.5 $ an hour . Always stop the ec2 when not in use . Always remember to keep it STOPPED

Go to EC2 Instance -> Instance State -> Stop Instance



Optional : Stopping an instance will also kill the server.py from running . So when you restart the instance , you have to use the command in step 29 to restart the uvicorn server OR you may configure a systemd service file ( refer APPENDIA A below )

**APPENDIX A**

**Step-by-step: Create a systemd service**

1. Open the service file

sudo nano /etc/systemd/system/wan\_server.service

1. Paste this configuration

[Unit]

Description=Start WAN2.1 FastAPI server with uvicorn

After=network.target

[Service]

WorkingDirectory=/home/aifinity/Wan2.1/demo\_text2video/server

ExecStart=/usr/bin/env nohup uvicorn server:app --host 0.0.0.0 --port 7860

StandardOutput=append:/home/aifinity/Wan2.1/demo\_text2video/server/server.log

StandardError=append:/home/aifinity/Wan2.1/demo\_text2video/server/server.log

Restart=always

User=aifinity

[Install]

WantedBy=multi-user.target

1. Reload and Restart

sudo systemctl daemon-reexec

sudo systemctl enable wan\_server.service

sudo systemctl start wan\_server.service

1. (Optional) Check that it's running:

sudo systemctl status wan\_server.service