AMI

> ### AMI = Amazon Machine Image

- It’s like a **template** for launching a virtual computer (EC2 instance) on AWS.
- **Snapshot** or **template** of an EC2 instance

> 📦 **A snapshot of a full machine** — including the operating system, installed software, settings, and sometimes even preloaded data or scripts.

💡 Every time you launch an EC2 instance, it **must be based on an AMI**.

It includes:

- The **operating system** (e.g., Ubuntu, Windows).
- **Pre-installed software** (e.g., Python, Docker, LLM models).
- **Configuration settings** (users, permissions, security).

You can use AMIs to **launch identical EC2 instances** quickly.

🎒 Real-World Analogy

Imagine you're setting up 100 computers in a lab.

Would you install Windows and all the tools manually on each one? No. You’d:

- Configure one “master” PC with OS + software
- Take an image (copy) of it
- Use that image to clone the rest

That “image” is like an **AMI in AWS**.

✅ What Does an AMI Contain?

An AMI typically includes:

Component	Meaning
✅ OS	Ubuntu, Amazon Linux, Windows, etc.
✅ Software	Python, Docker, Apache, Jupyter, etc.
✅ Config files	Environment variables, shell scripts
✅ EBS snapshot	A snapshot of the root volume

> 💡 **So when you launch a new EC2 instance from an AMI, you instantly get everything preinstalled.**

💻 Example Use Case: Custom AMI

Let’s say:

- You install Ubuntu EC2
- Set up Python, Conda, Apache, model files
- You want to reuse this setup later

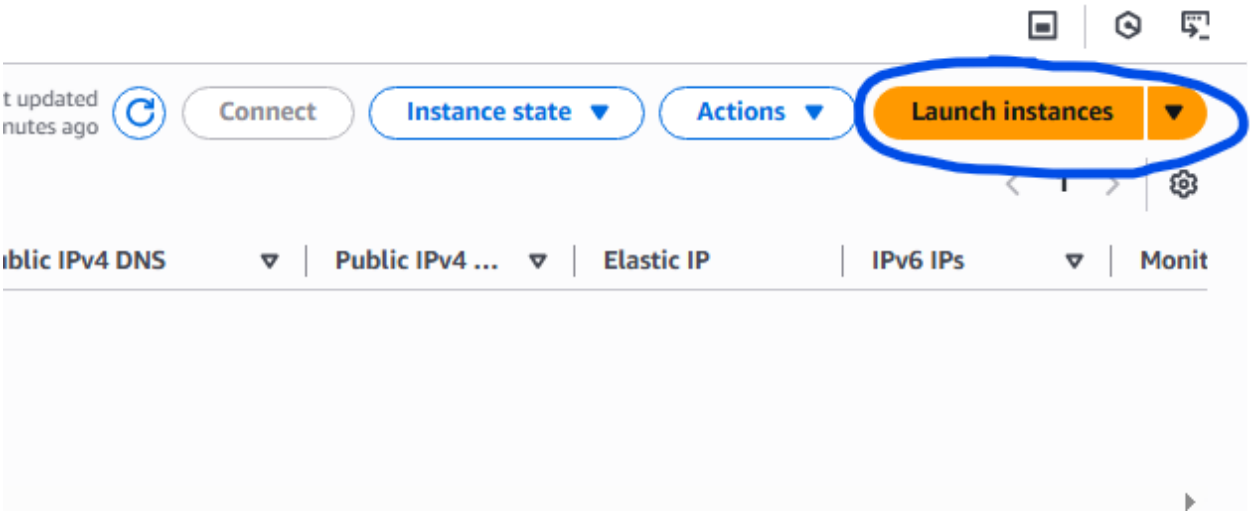
👉 You can create a **Custom AMI** of that EC2 instance.

Then anytime you launch a new instance from that AMI:

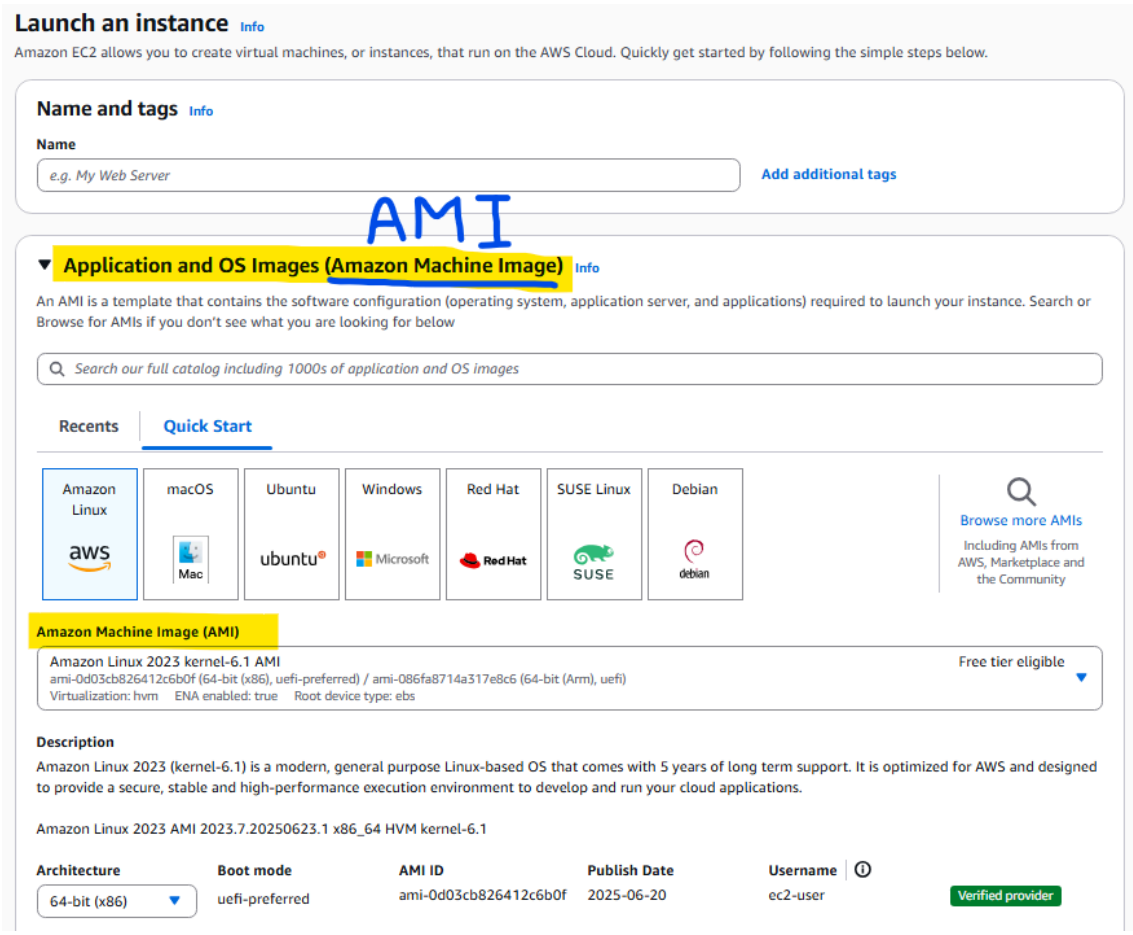
- **You save hours of setup time**
- You get consistent results

🚀 How to Launch an EC2 Using an AMI (AWS Console)?

1. Go to EC2 → **Launch Instance**



2. **Choose an AMI**
   - Pick from:
     - ✅ Amazon Linux 2023
     - ✅ Ubuntu Server 22.04
     - ✅ Deep Learning AMI (comes with PyTorch, TensorFlow)
     - ✅ Your Custom AMIs (if you've created one)



📸 How to Create a Custom AMI from EC2?

Step-by-Step:

1. Go to EC2 → Instances
2. Select your running instance
3. Click **Actions → Image → Create Image**
4. Give it:
   - A name (e.g. `llm-inference-server-v1`)
   - Description
   - Choose "No reboot" if you want a hot snapshot
5. Click **Create Image**

🎉 Done! You've created your own AMI.

It will appear under:

> EC2 → AMIs → Owned by me

🧪 Advanced Use Cases

Use Case	Description
Scale identical servers	Launch 5–100 EC2s with same AMI
Backup a machine	AMI acts like a full backup
Share setup with others	AMIs can be shared with specific AWS accounts
CI/CD pipelines	Create versioned AMIs for software releases

🧠 Common Beginner Mistakes

Mistake	Fix
❌ Assuming AMI = EC2 instance	EC2 is the **running computer**, AMI is the **template**
❌ Creating AMIs without cleaning up	Always stop services and delete temp files before AMI
❌ Forgetting to update AMI after installing new things	If you install updates, make a new AMI
❌ Not checking AMI region	AMIs are **region-specific** — copy them if needed elsewhere

🔑 Key Concepts

Term	What It Means	Example
AMI ID	Unique identifier for an AMI (e.g., `ami-0abcdef1234567890`).	`ami-0c55b159cbfafe1f0` (Amazon Linux).

Term	What It Means	Example
Root Volume	The primary disk (usually 8GB–100GB) containing the OS.	An Ubuntu AMI with 20GB root volume.
Instance Type	The hardware specs (CPU, RAM) the AMI supports.	`t2.micro`, `g5.xlarge`.

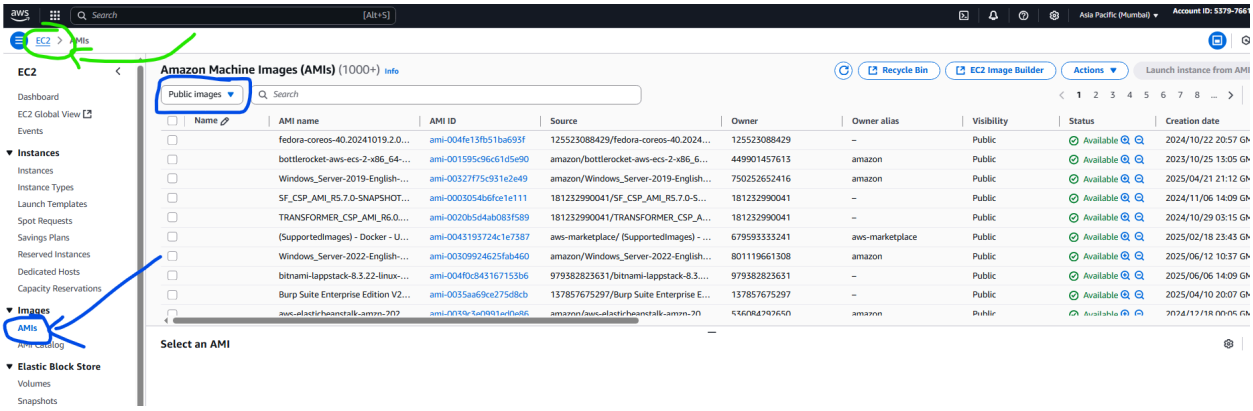
🚀 How to Use AMIs?

1. Find a Public AMI

- **AWS Marketplace**: Pre-configured AMIs (e.g., Deep Learning AMI).
- **Community AMIs**: Free images (Ubuntu, CentOS, etc.).

Steps:

1. Go to **EC2 Dashboard** → **AMIs** → **Public images**.
2. Search for keywords like:
    - `ubuntu` → Official Ubuntu AMI.
    - `deep learning` → Pre-installed TensorFlow/PyTorch.



2. Launch an EC2 Instance from an AMI

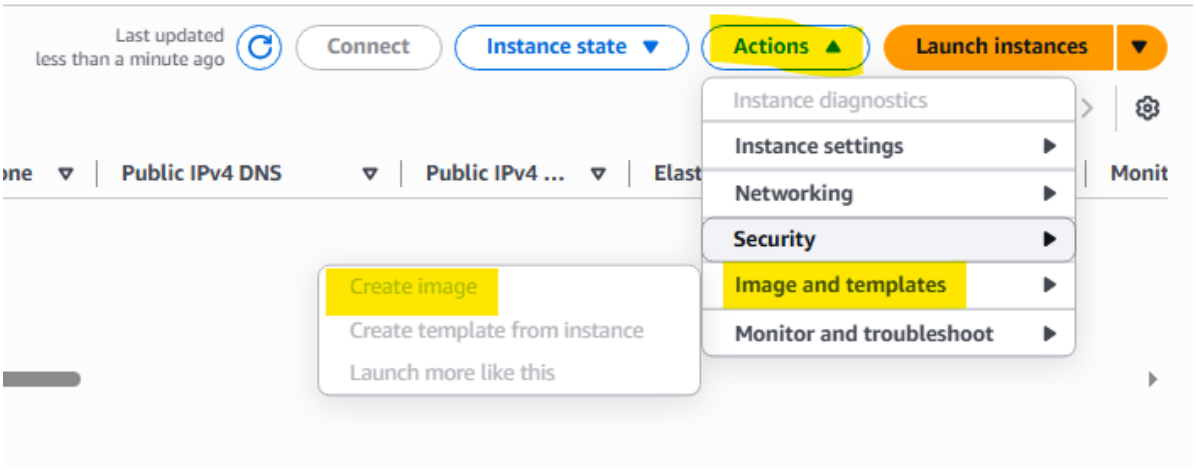
1. Select an AMI → Click **Launch Instance**.
2. Choose an **instance type** (e.g., `t2.micro` for testing).
3. Configure storage, security groups, and keys.

4. Click **Launch**.

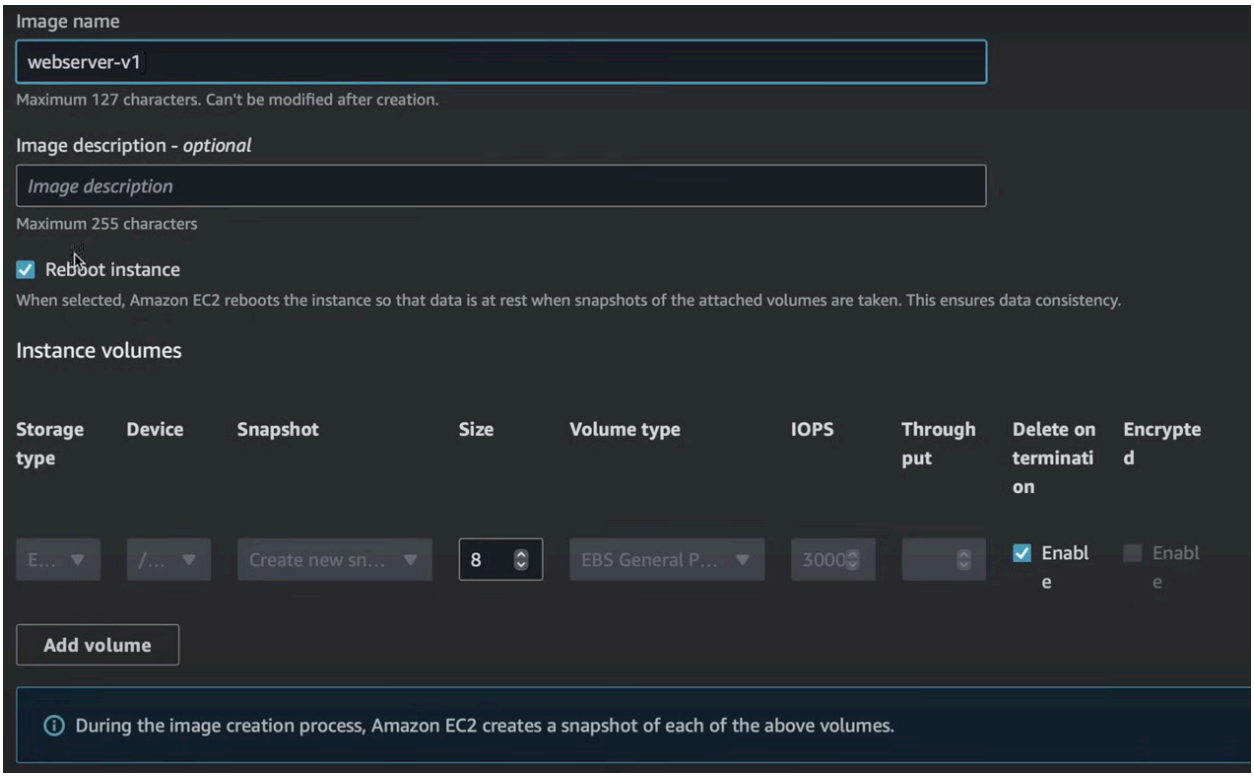
3. Create Your Own AMI

If you customize an EC2 instance (e.g., install an LLM model), save it as an AMI:

1. Go to **EC2** → **Instances** → Select your instance.
2. Click **Actions** → **Image and templates** → **Create Image**.



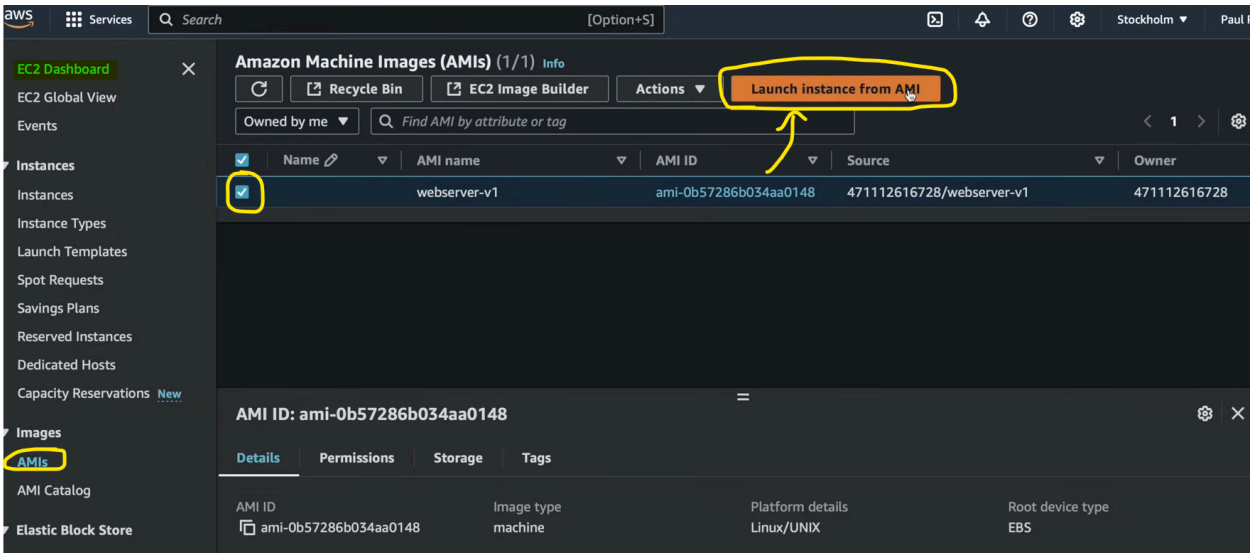
3. Name it (e.g., `my-llm-server-ami`) → **Create**.



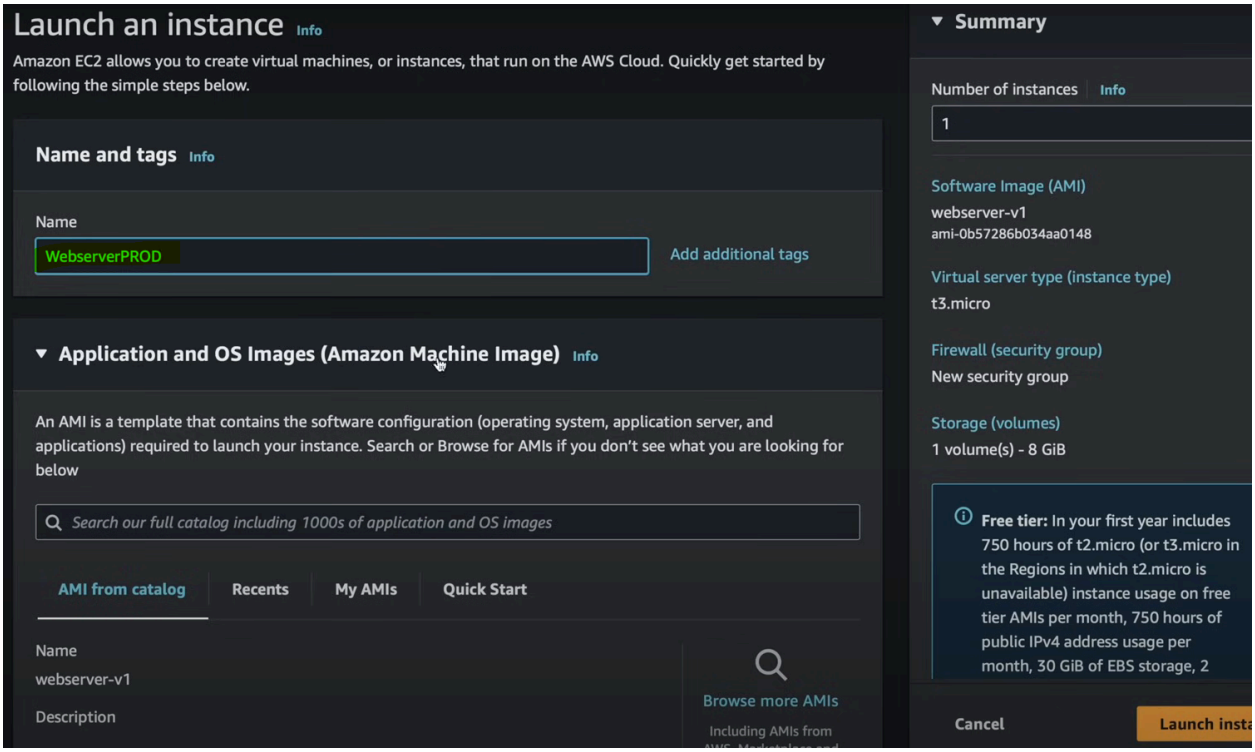
✅ Now, you can launch **identical copies** of this instance anytime!

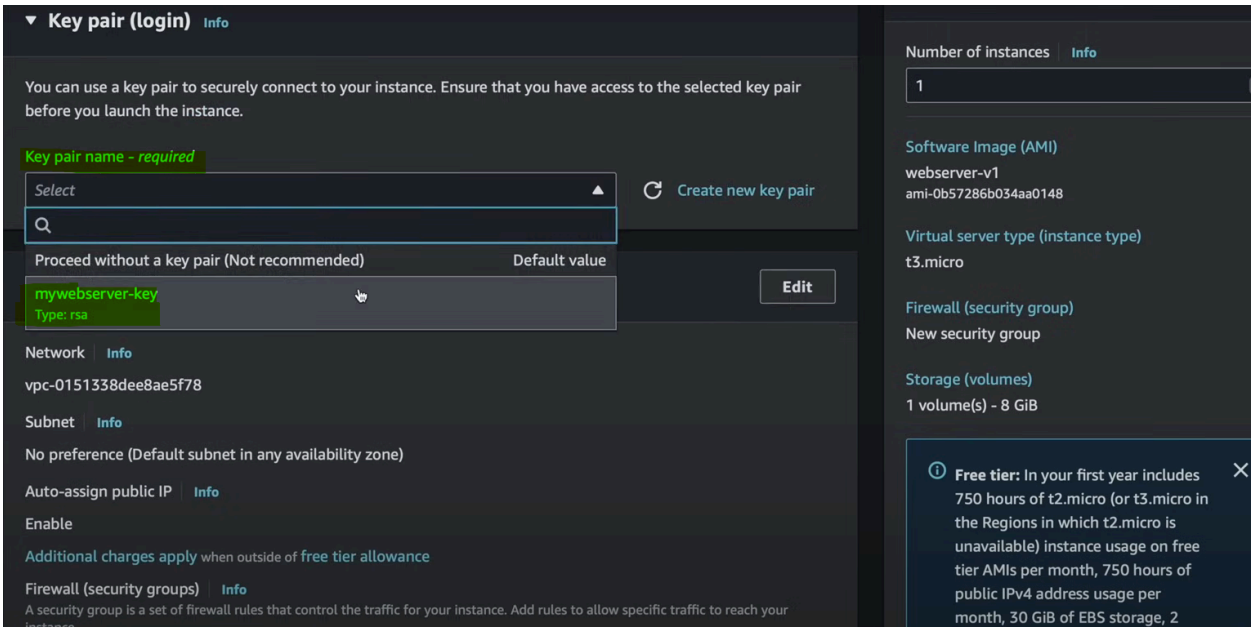
Create an EC2 Instance from AMI

- EC2 → Images → AMI
- Select the AMI
- Click **Launch instance from AMI**

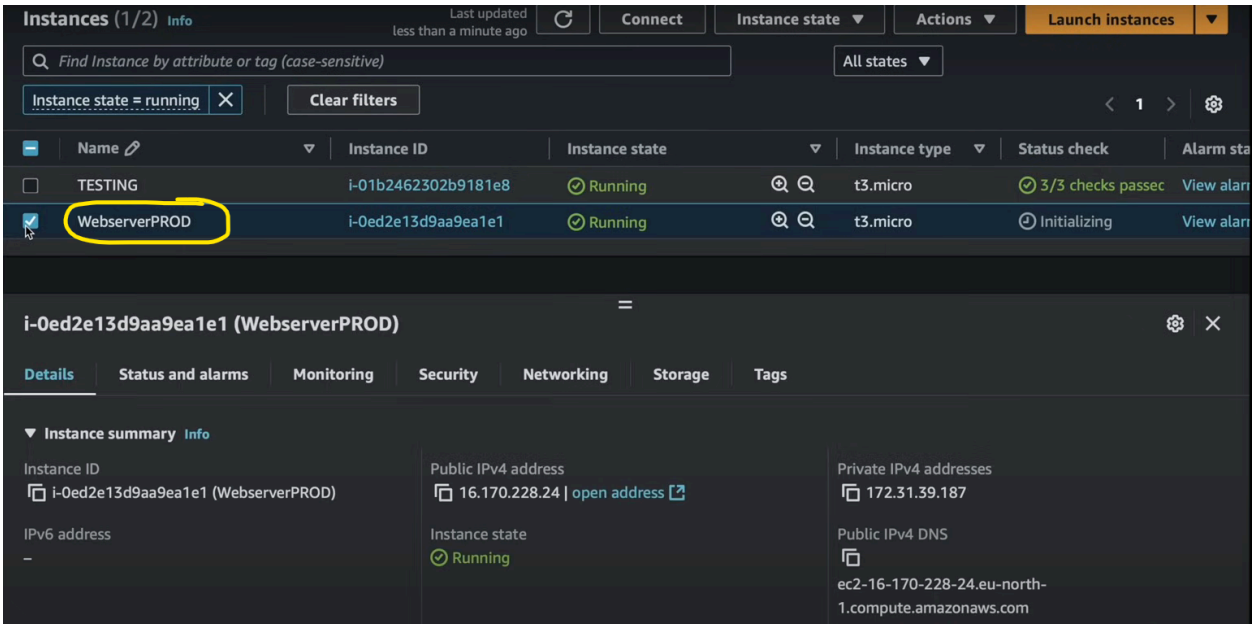


- Select the configurations





- **New instance created**



- If you Copy & open the public IP, the website deployed in the old EC2 instance will be available from the public IP of the new instance.

💰 AMI Costs

- **Storing AMIs**:
  - You pay for **EBS snapshot storage** (~₹2.30/GB/month).
  - Example: A 20GB AMI costs ~₹46/month.
- **Public AMIs**:
  - Some are free (e.g., Ubuntu), others charge hourly (e.g., Windows).

Use Cases

1. **Pre-installed AI Tools**
   - Use the **AWS Deep Learning AMI** (has PyTorch, TensorFlow preloaded).
2. **Disaster Recovery**
   - Backup your server as an AMI → Restore if crashes.
3. **Scaling**
   - Launch 10 identical servers from one AMI for load balancing.

⚠️ Pitfalls to Avoid

- **Don't leave unused AMIs** → They still cost money!
- **Region-locked**: AMIs work only in the region they're created.
- **Licensing costs**: Windows/paid AMIs bill hourly.

🔧 AWS CLI Commands

Task	Command
List AMIs	`aws ec2 describe-images --owners self` (for your custom AMIs).
Create AMI	`aws ec2 create-image --instance-id i-123abc --name "My-LLM-AMI"`
Launch from AMI	`aws ec2 run-instances --image-id ami-0abc123 --instance-type t2.micro`

❓ FAQ

Q: Can I share my AMI with others?

A: Yes! Go to **AMIs** → Select AMI → **Actions** → **Modify Image Permissions**.

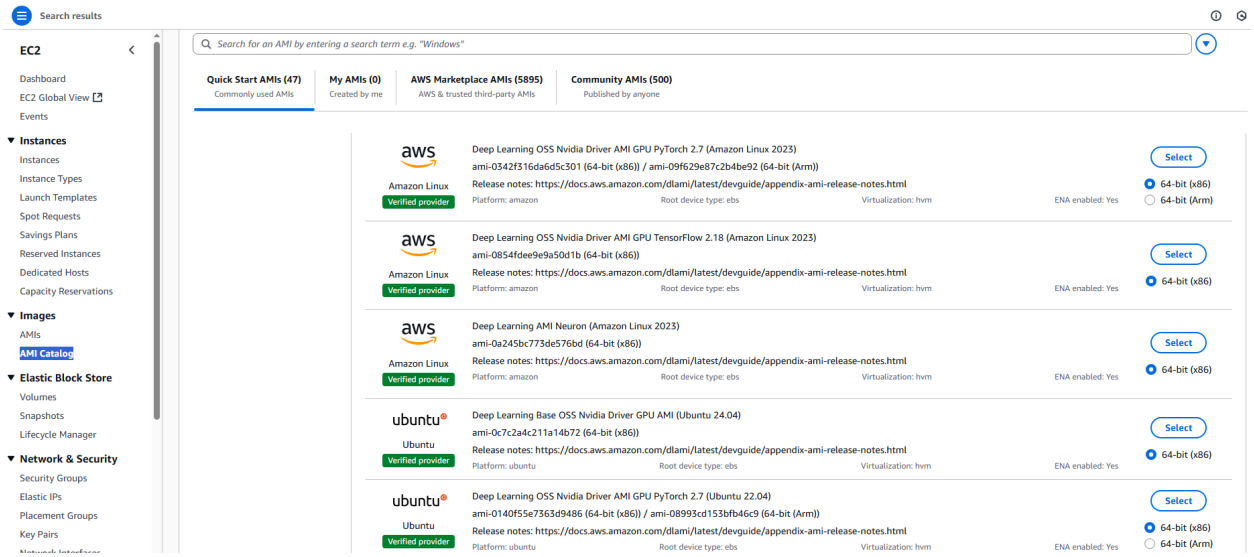
Q: How do I delete an AMI?

A: **Deregister it** (EC2 → AMIs → Select → **Actions** → **Deregister**). Then delete its **snapshot** (EC2 → Snapshots).

Q: Can I move an AMI to another region?

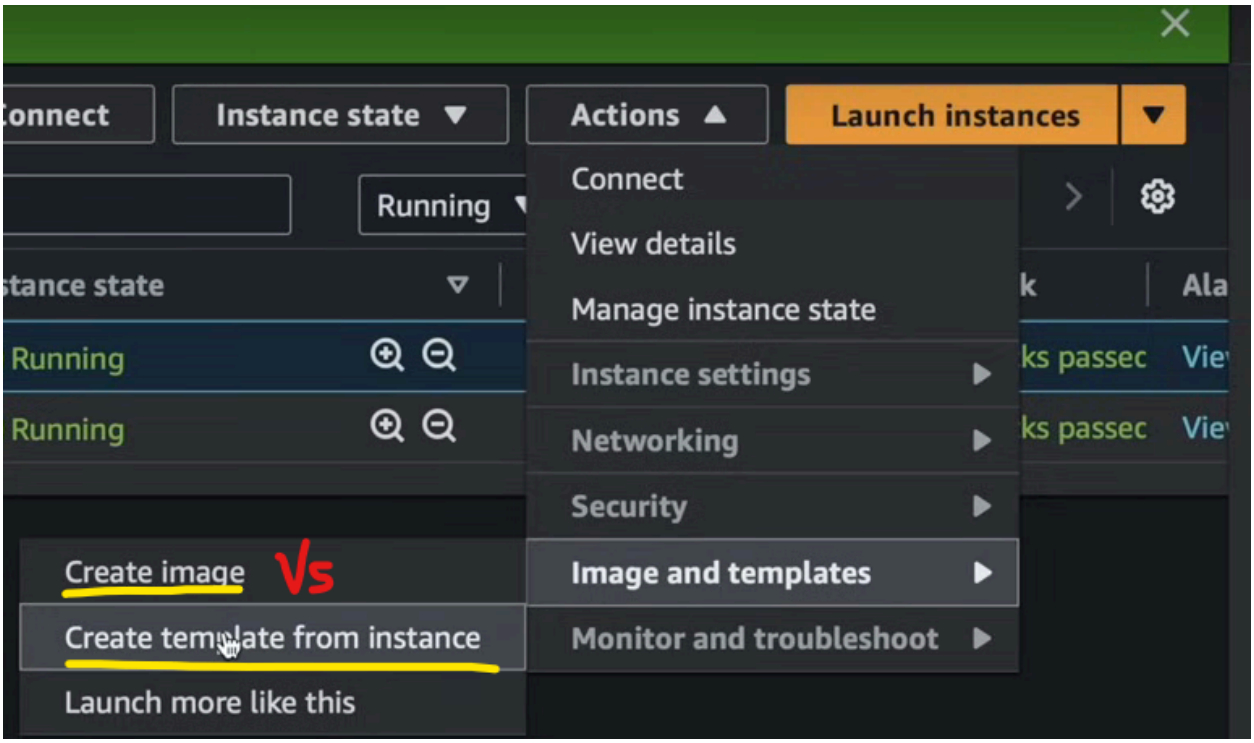
A: Yes! Use **EC2** → **AMIs** → **Actions** → **Copy AMI**.

AMI Catalog



✅

Create Image vs Launch Template from Instance



Feature	**Create Image (AMI)**	**Launch Template**
Saves what?	A **snapshot** of the EC2 instance's **disk** (OS, software, data)	A **blueprint** of how to launch new EC2 instances (AMI, instance type, key, etc.) **without storing the data**
What is it?	A **copy of the machine's disk**	A **recipe to launch machines** with certain settings
Includes?	OS, app code, files, packages ✅	AMI ID, instance type, VPC, key pair, security group, user data, etc. ✅
Used for?	Backup, cloning, portability	Consistent, repeatable EC2 launches
Replaces EC2?	Yes — restores the entire machine state	No — launches a new EC2 based on saved settings
Cost?	Costs to store the image (in S3)	No cost — just saved metadata

🚢 Think of It Like:

Real World Analogy	Description
Create Image	You take a **snapshot of your laptop**: full OS, installed apps, files, settings
Launch Template	You fill a **form** that says: “Next time I build a laptop, use Ubuntu, 16GB RAM, preinstall Python, use this password, etc.”

Concept	What It’s Like
Create Image (AMI)	🔁 **Ctrl+C** & **Ctrl+V** of your full EC2 instance → OS, apps, config files, everything copied exactly
Launch Template	📄 Blueprint or form that says: “***Next time I want a server like this, use this AMI,*** ***this size, this startup script*** ”

- **AMI = actual snapshot (disk copy)**
- **Launch Template = how to launch a new instance using that snapshot and other configs**

📦 What Is `Create Image` (AMI)?

✅ It creates a **complete image (AMI)** of your EC2 instance:

- OS (e.g., Ubuntu, Amazon Linux)
- Software installed (Python, Nginx, etc.)
- Files and folders
- Even user config (if included)

Use it when:

- You’ve finished setting up an EC2 and want to **reuse the same environment later**
- You want a **backup** (like a restore point)
- You want to launch **clones** in the future

How to use:

- Go to EC2 → Instances → Select → `Actions → Image → Create Image`

It becomes available in **EC2 → AMIs**

You can now launch new EC2 instances from this AMI.

📄 What Is `Launch Template` ?

✅ It is a **saved configuration** for launching EC2 instances.

Includes settings like:

- AMI ID
- Instance type (e.g., `t2.micro`, `g4dn.xlarge`)
- Key pair
- Security groups
- IAM role
- EBS volume size
- User data script (startup commands)

Use it when:

- You want to **automate EC2 creation**
- You want to always launch instances with the same settings
- You use **Auto Scaling Groups** or **Spot Fleets**
- You want to launch LLM or app servers consistently with custom startup scripts

How to create:

- EC2 → Instances → Select → `Actions → Create launch template from instance`

It becomes available under:

EC2 → Launch Templates

You can now launch new EC2s using that template **instantly with 1 click**.

Feature	Comes from AMI?	Comes from Launch Template?
Ubuntu + Python + Apache	✅ Yes	❌ No
Your model/code files	✅ Yes	❌ No
EC2 instance type (e.g. t2.micro)	❌ No	✅ Yes
IAM role, key pair, VPC	❌ No	✅ Yes
Auto-run script on boot	❌ No	✅ Yes
Launch policy (Spot vs On-Demand)	❌ No	✅ Yes