

 **DAY 3 — AWS FUNDAMENTALS + IAM + EC2 + HANDS-ON DEPLOYMENT****★ PART 1 — AWS Core Concepts (You Must Know These)****✓ 1. Region & Availability Zones (AZs)**

- Region → geographical area (ap-south-1)
  - AZs → multiple data centers inside a region
  - High availability = deploy across multiple AZs
- 

**✓ 2. IAM (Identity & Access Management)**

IAM manages identities:

- Users
- Groups
- Roles
- Policies

IAM is the MOST important AWS security concept.

---

**✓ 3. EC2 (Elastic Compute Cloud)**

Provides virtual machines for:

- Running applications
  - Hosting backend services
  - Hosting Docker containers
  - CI/CD runners
- 

**✓ 4. Security Groups**

Firewall for EC2 instances.

Rules:

- Inbound → who can reach the server
  - Outbound → who server can reach
- 

## ✓ 5. VPC (Virtual Private Cloud)

Private network inside AWS.

Subnets:

- Public subnet → accessible from internet
- Private subnet → no direct internet access

## ★ PART 2 — HANDS-ON: IAM USER + PERMISSIONS

### 🔥 Step 1: Create IAM Group

Go to AWS Console → IAM → User Groups

- Create User groups  
**Example:** devops-group
- Attach policies:
  - AmazonEC2FullAccess
  - AmazonS3FullAccess
  - IAMReadOnlyAccess
  - CloudWatchFullAccess

The screenshot shows the AWS IAM User Groups page. The left sidebar is collapsed. The main area shows the 'devops-group' info. The 'Summary' section includes the user group name 'devops-group', creation time 'December 12, 2025, 16:01 (UTC+05:30)', and ARN 'arn:aws:iam::247562657840:group/devops-group'. Below this, the 'Permissions' tab is selected, showing 'Permissions policies (4)'. A table lists four AWS managed policies: AmazonEC2FullAccess, AmazonSSFFullAccess, CloudWatchFullAccess, and IAMReadOnlyAccess, each with its type (AWS managed) and attached entities count (2, 3, 1, 1 respectively). Buttons for 'Simulate', 'Remove', and 'Add permissions' are visible.

## 🔥 Step 2: Add your IAM user to this group

Go to: IAM → Users → your-user → Add to group

The screenshot shows the AWS IAM Users page. The left sidebar is collapsed. The main area shows the 'devops-user' info. The 'Summary' section includes the ARN 'arn:aws:iam::247562657840:user/devops-user', console access status 'Disabled', and last console sign-in. It also shows two access keys: 'Access key 1' (AKIATTI7CMAYC25O4AZN - Active, used 2 days ago, 2 days old) and 'Access key 2' (Create access key). The 'Groups' tab is selected, showing the user's membership in 'Administrator' and 'devops-group' groups. Attached policies for 'Administrator' include 'AdministratorAccess', 'AmazonEC2FullAccess', 'IAMReadOnlyAccess', and 2 more. A button for 'Add user to groups' is visible.

## 🔥 Step 3: Create a Role

- IAM → Roles → Create role
- Trusted entity → AWS Service
- Use case → EC2
- Attach policies:
  - AmazonS3ReadOnlyAccess
  - CloudWatchAgentServerPolicy
- Role Names:
  - **DevOps-EC2-Role**

The screenshot shows the IAM Roles page for the 'DevOps-EC2-Role'. The 'Permissions' tab is selected, displaying two managed policies: 'AmazonS3ReadOnlyAccess' and 'CloudWatchAgentServerPolicy', both of which are AWS-managed policies.

This role is used by EC2 instances to access:

- S3
- CloudWatch

## ★ PART 3 — HANDS-ON: Launch an EC2 Instance (Step-by-Step)

### 🔥 Step 1: Go to EC2 Dashboard

- AWS Console → EC2 → Instances → Launch Instance

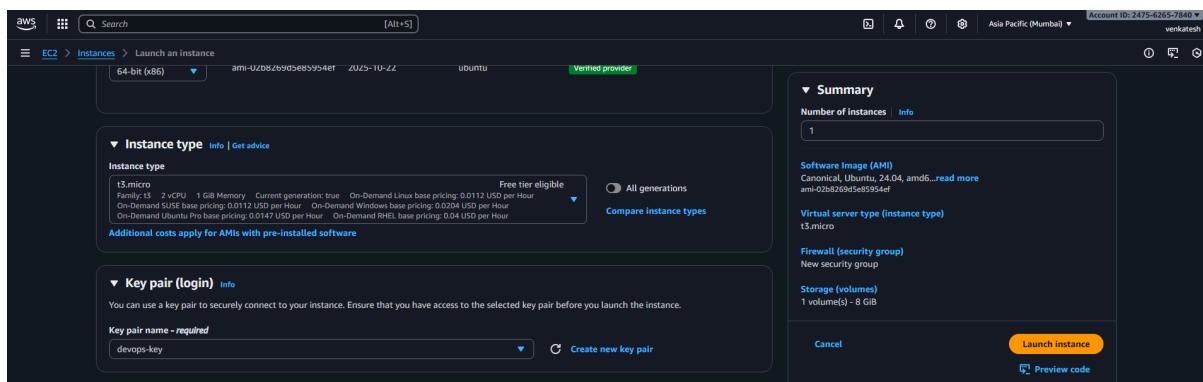
### 🔥 Step 2: Select AMI

- Choose: Ubuntu AMI

The screenshot shows the 'Launch an instance' page in the EC2 console. In the 'Name and tags' section, 'My Web Server' is entered. In the 'Application and OS Images (Amazon Machine Image)' section, 'Ubuntu' is selected from the 'Recent' list. On the right, the 'Summary' panel shows 1 instance and provides options to 'Launch instance' or 'Preview code'.

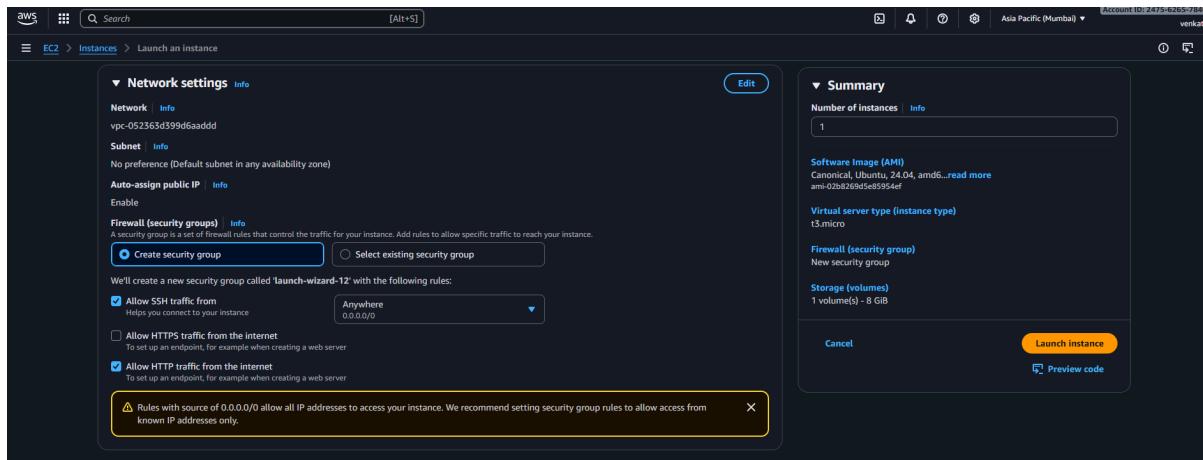
### 🔥 Step 3: Choose Instance Type and Key pair

- Choose: t3.micro (Free tier) and create new key pair (Store safely)



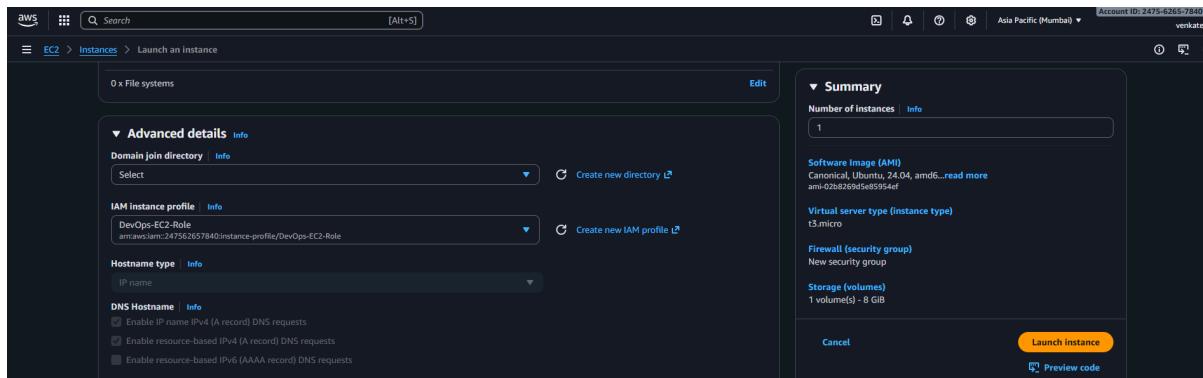
## 🔥 Step 4: Networking

- VPC: default
- Subnet: choose any AZ
- Auto-assign IP: ENABLED



## 🔥 Step 5: Attach IAM Role

- Under "Advanced details" → IAM Instance Profile → Choose:



## 🔥 Step 6: Launch

- Launch Instance - Instance state -> Running

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with 'EC2' selected under 'Instances'. The main area has a table titled 'Instances (1/1) Info'. The table includes columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4 IP, and Elastic IP. One row is visible for an instance named 'My Web Server' with the following details: i-04412747220d64500, Running, t3.micro, Initializing, View alarms, ap-south-1b, ec2-65-0-124-201.ap-s..., 65.0.124.201, and -.

## ★ PART 4 — HANDS-ON: SSH INTO THE EC2 INSTANCE

- Open terminal:

```
$ chmod 400 devops-key.pem      #Key-pair permission change
```

```
ASUS@Venky MINGW64 /f/devops-learning-journey
$ chmod 400 ./devops-key.pem
```

- SSH command (replace with your IP):

```
$ ssh -i devops-key.pem ubuntu@IP-Address
```

```
ASUS@Venky MINGW64 /f/devops-learning-journey
$ ssh -i ./devops-key.pem ubuntu@65.0.124.201
System load: 0.0          Temperature:      -273.1 C
Usage of /: 25.8% of 6.71GB Processes:        111
Memory usage: 23%         Users logged in:   0
Swap usage: 0%            IPv4 address for ens5: 172.31.3.118

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-3-118:~$
```

## ★ PART 5 — Deploy a Live Web Application (Real DevOps Task)

- Inside EC2 Terminal:

```
$ sudo apt update -y
```

```
$ sudo apt install nginx -y
```

- Start Nginx:

```
$ sudo systemctl enable nginx
```

```
$ sudo systemctl start nginx
```

- Verify:

```
$ sudo systemctl status nginx
```

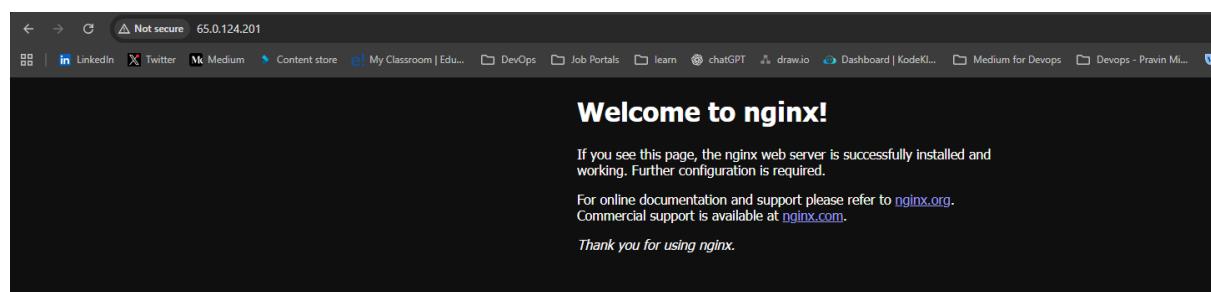
```

ubuntu@ip-172-31-3-118:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
  Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
  Active: active (running) since Fri 2025-12-12 17:04:12 UTC; 13s ago
    Docs: man:nginx(8)
 Process: 1727 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Process: 1729 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Main PID: 1759 (nginx)
   Tasks: 3 (limit: 1008)
  Memory: 2.4M (peak: 5.3M)
     CPU: 26ms
    CGroup: /system.slice/nginx.service
            ├─1759 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
            ├─1761 "nginx: worker process"
            ├─1762 "nginx: worker process"

Dec 12 17:04:12 ip-172-31-3-118 systemd[1]: Starting nginx.service - A high performance web server and a rev>
Dec 12 17:04:12 ip-172-31-3-118 systemd[1]: Started nginx.service - A high performance web server and a rev>
ubuntu@ip-172-31-3-118:~$
```

- Now, Open browser -> Visit

<http://<your-ec2-public-ip>>



## ★ PART 6 — Deploy Your Own HTML Page

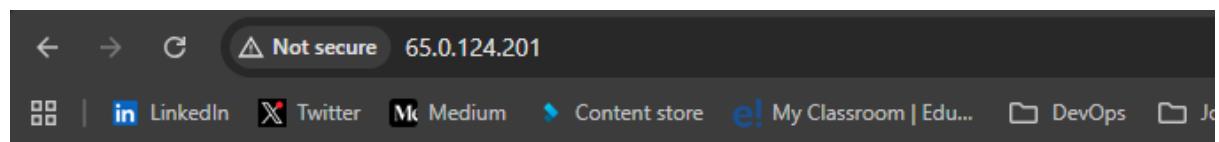
- Inside EC2 Instance:

```
$ echo "<h1>Hello from DevOps Day-3</h1>" | sudo tee /var/www/html/index.html
```

```

ubuntu@ip-172-31-3-118:~$ echo "<h1>Hello from DevOps Day-3</h1>" | sudo tee /var/www/html/index.html
<h1>Hello from DevOps Day-3</h1>
ubuntu@ip-172-31-3-118:~$
```

- Refresh browser → Should display:



**Hello from DevOps Day-3**

## ★ PART 7 — Enable CloudWatch Logs (Production Skill)

- Install CloudWatch agent:

```
$ sudo apt install amazon-cloudwatch-agent -y
```

It will fail in Ubuntu no worries buddy — this is a **very common issue** on Ubuntu EC2 instances.

The package **amazon-cloudwatch-agent** is **NOT available in the default Ubuntu repositories**, so `apt install` will always fail.

- To fix this, you must **install the CloudWatch Agent using the official AWS package download link**, not `apt`.

## ★ STEP 1 — Download the CloudWatch Agent Package

```
$ wget  
https://s3.amazonaws.com/amazoncloudwatch-agent/ubuntu/amd64/latest/amazon-cl  
oudwatch-agent.deb
```

```
ubuntu@ip-172-31-3-118:~$ wget https://s3.amazonaws.com/amazoncloudwatch-agent/ubuntu/amd64/latest/amazon-cl  
oudwatch-agent.deb  
--2025-12-12 17:12:57-- https://s3.amazonaws.com/amazoncloudwatch-agent/ubuntu/amd64/latest/amazon-cloudwat  
ch-agent.deb  
Resolving s3.amazonaws.com (s3.amazonaws.com)... 52.217.163.232, 16.15.180.204, 16.15.218.4, ...  
Connecting to s3.amazonaws.com (s3.amazonaws.com)|52.217.163.232|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 66799118 (64M) [application/octet-stream]  
Saving to: ‘amazon-cloudwatch-agent.deb’  
  
amazon-cloudwatch-agent.deb 100%[=====] 63.70M 15.4MB/s in 5.4s  
  
2025-12-12 17:13:03 (11.9 MB/s) - ‘amazon-cloudwatch-agent.deb’ saved [66799118/66799118]
```

## ★ STEP 2 — Install the Package

```
$ sudo dpkg -i -E ./amazon-cloudwatch-agent.deb
```

```
ubuntu@ip-172-31-3-118:~$ sudo dpkg -i -E ./amazon-cloudwatch-agent.deb  
Selecting previously unselected package amazon-cloudwatch-agent.  
(Reading database ... 71783 files and directories currently installed.)  
Preparing to unpack ./amazon-cloudwatch-agent.deb ...  
create group cwagent, result: 0  
create user cwagent, result: 0  
Unpacking amazon-cloudwatch-agent (1.300062.0b1304-1) ...  
Setting up amazon-cloudwatch-agent (1.300062.0b1304-1) ...  
ubuntu@ip-172-31-3-118:~$ █
```

## ★ STEP 3 — Verify Installation

```
$ sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a  
status
```

```
ubuntu@ip-172-31-3-118:~$ sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a status
{
    "status": "stopped",
    "starttime": "",
    "configstatus": "not configured",
    "version": "1.300062.0b1304"
}
ubuntu@ip-172-31-3-118:~$
```

## ★ STEP 4 — Start the CloudWatch Agent Using Default Config

- AWS provides a default config file.

```
$ sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl \
-a fetch-config \
-m ec2 \
-c default \
-s
```

```
ubuntu@ip-172-31-3-118:~$ sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl \
-a fetch-config \
-m ec2 \
-c default \
-s
***** processing amazon-cloudwatch-agent *****
Starting config-downloader, this will map back to a call to amazon-cloudwatch-agent
Executing /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent with arguments: [config-downloader -ode ec2 -download-source default -output-dir /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d -config /opt/aws/amazon-cloudwatch-agent/etc/common-config.toml -multi-config default]I! Trying to detect region from ec2
D! [EC2] Found active network interface
I! imds retry client will retry 1 times
Start configuration validation...
2025/12/12 17:15:59 Reading json config file path: /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d/default.tmp ...
2025/12/12 17:15:59 I! Valid Json input schema.
2025/12/12 17:15:59 D! ec2tagger processor required because append_dimensions is set
2025/12/12 17:15:59 Configuration validation first phase succeeded
Starting config-translator, this will map back to a call to amazon-cloudwatch-agent
Executing /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent with arguments: [config-translator -input /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.json -input-dir /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.d -output /opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.toml -mode ec2 -config /opt/aws/amazon-cloudwatch-agent/etc/common-config.toml -multi-config default]I! Tryin to detect region from ec2
D! [EC2] Found active network interface
```

## ★ STEP 5 — Check Service Status

```
$ sudo systemctl status amazon-cloudwatch-agent
```

```

ubuntu@ip-172-31-3-118:~$ sudo systemctl status amazon-cloudwatch-agent
● amazon-cloudwatch-agent.service - Amazon CloudWatch Agent
   Loaded: loaded (/etc/systemd/system/amazon-cloudwatch-agent.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-12-12 17:16:00 UTC; 1min 5s ago
     Main PID: 2065 (amazon-cloudwat)
        Tasks: 8 (limit: 1008)
       Memory: 25.9M (peak: 26.3M)
          CPU: 475ms
        CGroup: /system.slice/amazon-cloudwatch-agent.service
                  └─2065 /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent -config /opt/aws/amazon-cl

Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: D! [EC2] Found active network interface
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: I! imds retry client will retry 1 time>
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: 2025/12/12 17:16:00 Reading json config
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: /opt/aws/amazon-cloudwatch-agent/etc/a
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: 2025/12/12 17:16:00 Reading json config
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: 2025/12/12 17:16:00 I! Valid Json input
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: I! Trying to detect region from ec2
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: 2025/12/12 17:16:00 D! ec2tagger process
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2075]: 2025/12/12 17:16:00 Configuration valid
Dec 12 17:16:00 ip-172-31-3-118 start-amazon-cloudwatch-agent[2065]: I! Detecting run_as_user...
ubuntu@ip-172-31-3-118:~$
```

## ★ STEP 6 — Now Check Cloudwatch log groups in AWS Console

- Go to the AWS Console:  
CloudWatch → Logs → Log groups
- You should see log groups like:
  - ✓ `/aws/amazon-cloudwatch-agent/instance-id`
  - ✓ `/aws/amazon-cloudwatch-agent/logs`
  - ✓ `/var/log/syslog` (if enabled)
  - ✓ `/var/log/nginx/access.log` (if configured later)

