

School of Computer Science, Engineering and Applications(SCSEA)
B.C.A. TY (CCSA)
Subject: Containers & Orchestration

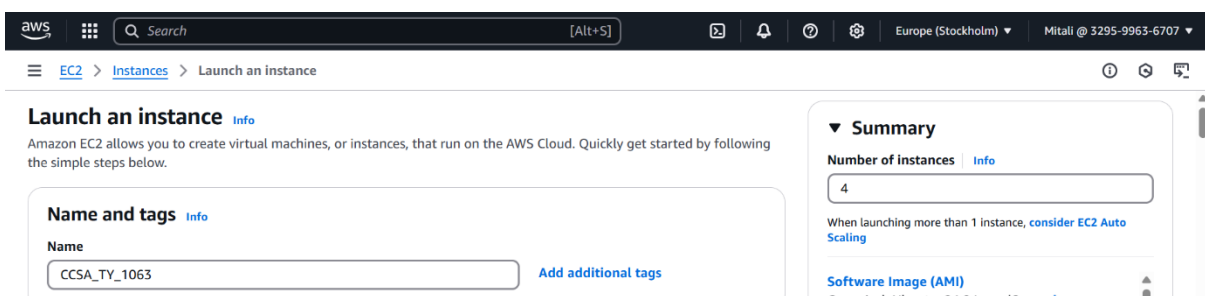
Name of the Student: Mitali Bhattad

PRN: 20220801063

Title of Practical: High Availability Container Orchestration with Docker
Swarm Master-Worker Setup on AWS EC2

Step1: Launch 4 EC2 INSTANCES

- Name the Instance



aws Search [ALT+S] Europe (Stockholm) Mitali @ 3295-9963-6707

EC2 > Instances > Launch an instance

Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name
CCSA_TY_1063

Add additional tags

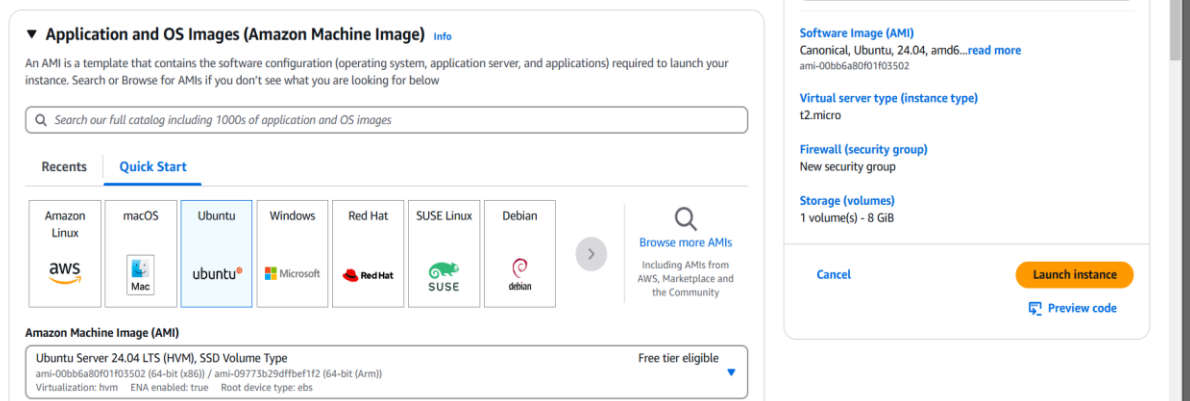
Summary

Number of instances 4

When launching more than 1 instance, consider EC2 Auto Scaling

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd64, read more

- Choose AMI: Ubuntu



Application and OS Images (Amazon Machine Image)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents Quick Start

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux Debian

aws Mac ubuntu Microsoft Red Hat SUSE debian

Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
ami-00bb6a80f01f03502 (64-bit (x86)) / ami-09773b29dfbf1f2 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Summary

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd64, read more
ami-00bb6a80f01f03502

Virtual server type (instance type)
t2.micro

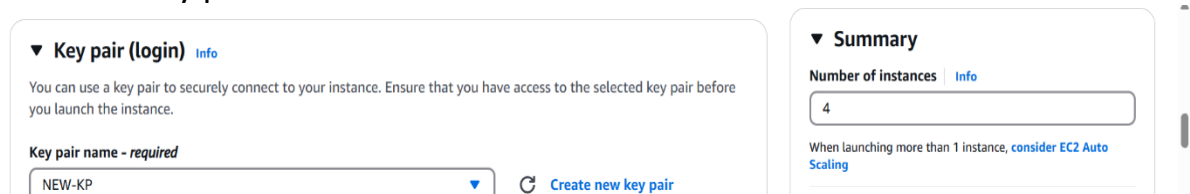
Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Cancel Launch instance Preview code

- Select the instance type: t2 micro

- Create a Key pair



Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required
NEW-KP

Create new key pair

Summary

Number of instances 4

When launching more than 1 instance, consider EC2 Auto Scaling

School of Computer Science, Engineering and Applications(SCSEA)
B.C.A. TY (CCSA)
Subject: Containers & Orchestration

Name of the Student: Mitali Bhattad

PRN: 20220801063

Title of Practical: High Availability Container Orchestration with Docker
Swarm Master-Worker Setup on AWS EC2

- Security Group

☒ Create security group ☐ Select existing security group

We'll create a new security group called 'launch-wizard-5' with the following rules:

- ☒ Allow SSH traffic from Anywhere (0.0.0.0/0)
- ☒ Allow HTTPS traffic from the internet
- ☒ Allow HTTP traffic from the internet

When launching more than 1 instance, [consider EC2 Auto Scaling](#)

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd64...[read more](#)
ami-0c1ac8a41498c1a9c

Virtual server type (instance type)
t3.micro

[Cancel](#) [Launch instance](#)

- Launch the instance

- Scroll down to Advanced Details → Paste this in User data:

```
#!/bin/bash
apt update -y
apt install -y apt-transport-https ca-certificates curl software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key add -
add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
apt update -y
apt install -y docker-ce
systemctl enable docker
systemctl start docker
```

When launching more than 1 instance, [consider EC2 Auto Scaling](#)

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd64...[read more](#)
ami-0c1ac8a41498c1a9c

Virtual server type (instance type)
t3.micro

Step 2: Edit Security Group Inbound Rules

[EC2](#) > [Security Groups](#) > [sg-04c931c07e774a059 - launch-wizard-5](#) > Edit inbound rules

Inbound rules

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sg-06fb8654fd7aa048f	Custom TCP	TCP	2377	Cust... 0.0.0.0/0		Delete
sg-09459a5aed2829803	Custom TCP	TCP	7946	Cust... 0.0.0.0/0		Delete
sg-063d1391cdfc20786	SSH	TCP	22	Cust... 0.0.0.0/0		Delete
-	Custom UDP	UDP	7946	Any... 0.0.0.0/0		Delete
-	Custom UDP	UDP	4789	Any... 0.0.0.0/0		Delete

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject: Containers & Orchestration

Name of the Student: Mitali Bhattad

PRN: 20220801063

Title of Practical: High Availability Container Orchestration with Docker
Swarm Master-Worker Setup on AWS EC2

STEP 3: CONNECT TO ALL 4 INSTANCES

1. 1st Instance: Initialize Swarm (MANAGER/LEADER)

On Instance 1 terminal, run:

- docker swarm init

```
ubuntu@ip-172-31-16-191:~$ sudo -i
root@ip-172-31-16-191:~# docker swarm init
Swarm initialized: current node (eb3x90awljdypufik8dpvja) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-ljomzgfnykzm8shqpd57nfoajufnhr6oga2fqm1bkoq7tu73kx-dqq4sihedxklaf8kif34rpt7 172.31.16.191:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

Get The manager token by firing this command on Instance 1 and copy it:

- sudo docker swarm join-token manager

```
root@ip-172-31-16-191:~# sudo docker swarm join-token manager
To add a manager to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-ljomzgfnykzm8shqpd57nfoajufnhr6oga2fqm1bkoq7tu73kx-d7okvhpepeqbu82h4czjghwt1 172.31.16.191:2377
```

2. 2nd Instance: Join INSTANCE 2 AS MANAGER

Paste the Manager token from Instance 1

```
root@ip-172-31-26-210:~# docker swarm join --token SWMTKN-1-ljomzgfnykzm8shqpd57nfoajufnhr6oga2fqm1bkoq7tu73kx-d7okvhpepeqbu82h4czjghwt1 172.31.16.191:2377
This node joined a swarm as a manager.
root@ip-172-31-26-210:~#
```

3. 3rd Instance: Join INSTANCE 3 AS WORKER

Paste the Worker token from Instance 1

```
root@ip-172-31-20-159:~# docker swarm join --token SWMTKN-1-ljomzgfnykzm8shqpd57nfoajufnhr6oga2fqm1bkoq7tu73kx-dqq4sihedxklaf8kif34rpt7 172.31.16.191:2377
This node joined a swarm as a worker.
```

School of Computer Science, Engineering and Applications(SCSEA)

B.C.A. TY (CCSA)

Subject: Containers & Orchestration

Name of the Student: Mitali Bhattad

PRN: 20220801063

Title of Practical: High Availability Container Orchestration with Docker Swarm Master-Worker Setup on AWS EC2

4. 4th Instance: Join INSTANCE 4 AS WORKER

Paste the Worker token from Instance 1

```
root@ip-172-31-27-102:~# docker swarm join --token SWMTKN-1-1jomzgfnykzm8shqpd57nfoajufnhr6oga2fqm1bkoq7tu73kx-dqq4sihecdxklaf8kif34rpt7 172.31.16.191:2377
This node joined a swarm as a worker.
```

STEP 4: Perform Operations on Instance 1:

- View all nodes: `docker node ls`
- Promote Worker Node: `docker node promote <worker-hostname>`
- Demote Manager Node: `docker node demote <manager-hostname>`

```
root@ip-172-31-16-191:~# docker node ls
```

ID	HOSTNAME	STATUS	AVAILABILITY	MANAGER STATUS	ENGINE VERSION
eb3x90awljdwyuipfik8dpvja *	ip-172-31-16-191	Ready	Active	Leader	28.1.1
aune3uhd2k4vimufet1fojbue	ip-172-31-20-159	Ready	Active		28.1.1
g2ml22t7r0h7acq7p6jnvylb	ip-172-31-26-210	Ready	Active	Reachable	28.1.1
lgm0egem9v3fpcjidvyw6y7hi	ip-172-31-26-210	Down	Active		28.1.1
5w21mkuy0j549z7nih5f4xfce	ip-172-31-27-102	Ready	Active		28.1.1

```
ubuntu@ip-172-31-16-191:~$ sudo -i
root@ip-172-31-16-191:~# docker node promote ip-172-31-20-159
Node ip-172-31-20-159 promoted to a manager in the swarm.
root@ip-172-31-16-191:~#
root@ip-172-31-16-191:~# docker node demote ip-172-31-26-210
Manager ip-172-31-26-210 demoted in the swarm.
root@ip-172-31-16-191:~#
```

STEP 5: Now, Check the status of Worker node and Manager node

Manager node should be demoted & Worker node should be promoted

WORKER Node's Terminal:

```
root@ip-172-31-20-159:~# docker node ls
```

ID	HOSTNAME	STATUS	AVAILABILITY	MANAGER STATUS	ENGINE VERSION
eb3x90awljdwyuipfik8dpvja	ip-172-31-16-191	Ready	Active	Leader	28.1.1
aune3uhd2k4vimufet1fojbue *	ip-172-31-20-159	Ready	Active	Reachable	28.1.1
g2ml22t7r0h7acq7p6jnvylb	ip-172-31-26-210	Ready	Active	Reachable	28.1.1
lgm0egem9v3fpcjidvyw6y7hi	ip-172-31-26-210	Down	Active		28.1.1
5w21mkuy0j549z7nih5f4xfce	ip-172-31-27-102	Ready	Active		28.1.1

```
root@ip-172-31-20-159:~#
```

MANAGER Node's Terminal:

```
root@ip-172-31-26-210:~# docker node ls
Error response from daemon: This node is not a swarm manager. Worker nodes can't be used to view or modify cluster state. Please run this command on a manager node or promote the current node to a manager.
root@ip-172-31-26-210:~#
```