DevOpsStreamline Project Documentation

Welcome to the DevOpsStreamline project! This document provides step-by-step instructions on setting up and configuring a DevOps pipeline using AWS EC2 instances, Ansible, Jenkins, Docker, and GitHub. Follow the steps below to create a master-slave architecture and automate the deployment process.

Table of Contents

Definitions of Tools Used

1. AWS EC2 (Amazon Web Services Elastic Compute Cloud)

AWS EC2 is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers by allowing them to create virtual servers, known as instances, in a highly flexible and scalable environment.

2. PuTTY

PuTTY is a free and open-source terminal emulator, serial console, and network file transfer application that supports several network protocols, including SSH (Secure Shell). It is used to connect to remote machines, such as EC2 instances, over the network.

3. Ansible

Ansible is an open-source automation tool used for configuration management, application deployment, and task automation. It uses simple, human-readable YAML syntax to define automation workflows, making it easy to manage complex IT environments.

4. SSH (Secure Shell)

SSH (Secure Shell) is a cryptographic network protocol used for secure data communication, remote command-line login, and other secure network services between two networked computers. SSH provides strong authentication and secure encrypted data communications between two computers.

5. Jenkins

Jenkins is an open-source automation server used to build, test, and deploy software. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery (CI/CD).

6. Docker

Docker is an open-source platform that automates the deployment of applications inside lightweight, portable containers. Containers include everything an application needs to run, including libraries, dependencies, and configuration files, ensuring consistency across multiple environments.

7. GitHub

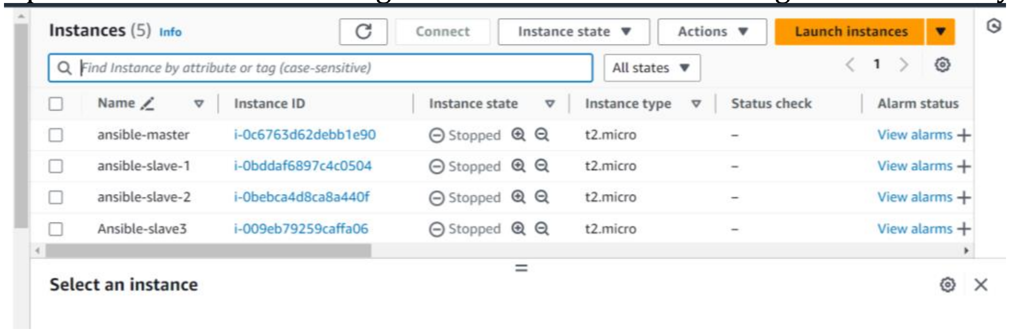
GitHub is a web-based version-control and collaboration platform for software developers. It uses Git, a distributed version control system, to help developers manage and track changes to code. GitHub also facilitates collaboration through features like pull requests, issues, and integrated CI/CD.

---

1. Setup EC2 Instances

Step-by-Step Guide

1. **Sign into AWS Management Console**
   - Open the AWS Management Console and log in with your credentials.

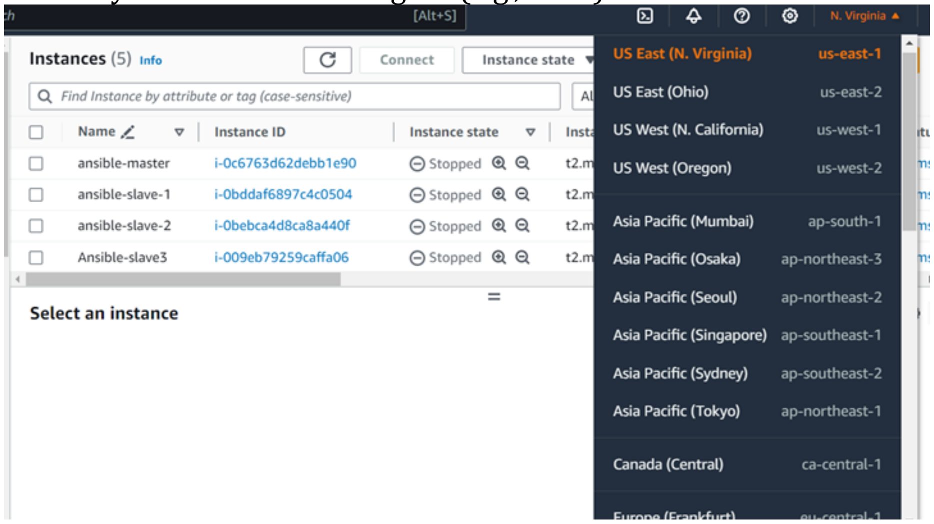


Instances (5) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
☐	ansible-master	i-0c6763d62debb1e90	Stopped	t2.micro	–	View alarms +
☐	ansible-slave-1	i-0bddaf6897c4c0504	Stopped	t2.micro	–	View alarms +
☐	ansible-slave-2	i-0bebca4d8ca8a440f	Stopped	t2.micro	–	View alarms +
☐	Ansible-slave3	i-009eb79259caffa06	Stopped	t2.micro	–	View alarms +

Select an instance

2. **Select a Region**
   - Choose your desired AWS region (e.g., Ohio).

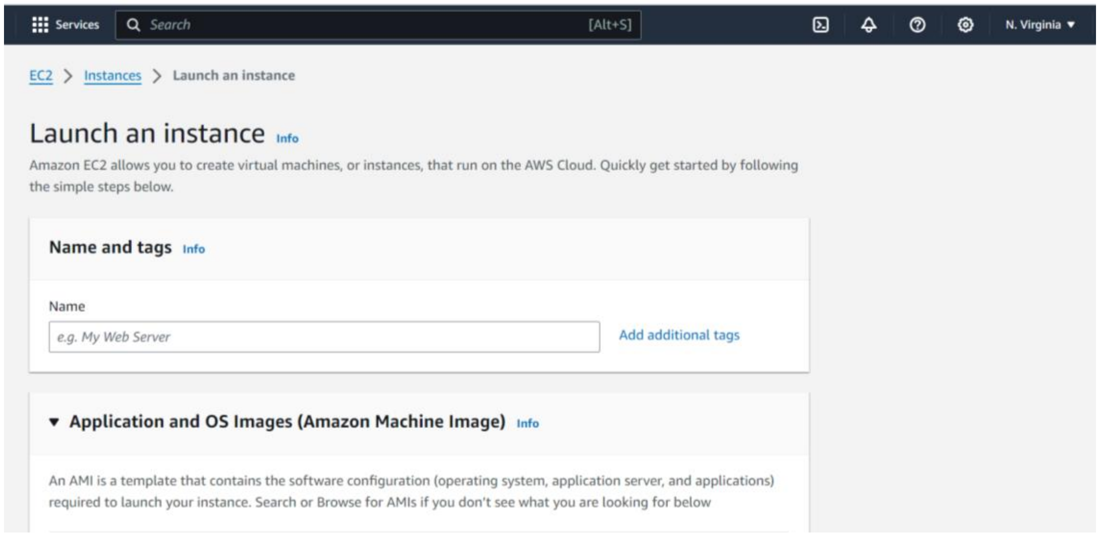


US East (N. Virginia)	us-east-1
US East (Ohio)	us-east-2
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Osaka)	ap-northeast-3
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Frankfurt)	eu-central-1

	Name	Instance ID	Instance state	Insta
☐	ansible-master	i-0c6763d62debb1e90	Stopped	t2.m
☐	ansible-slave-1	i-0bddaf6897c4c0504	Stopped	t2.m
☐	ansible-slave-2	i-0bebca4d8ca8a440f	Stopped	t2.m
☐	Ansible-slave3	i-009eb79259caffa06	Stopped	t2.m

Select an instance

3. **Navigate to EC2 Service**
   - Under the “Compute” section, select **EC2**.

– 
Services Search [Alt+S] N. Virginia

EC2 > Instances > Launch an instance

Launch an instance Info

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.

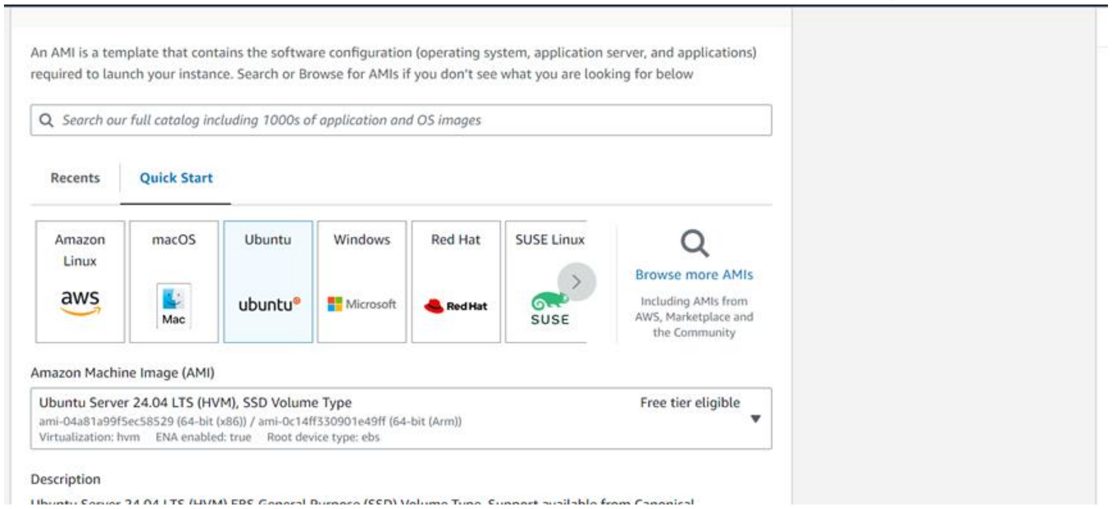
Name and tags Info

Name: *e.g. My Web Server* Add additional tags

▼ Application and OS Images (Amazon Machine Image) Info

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

4. **Create EC2 Instances**
   - Click on **Launch Instance** and create three EC2 instances:
     - **1 Master Node**
     - **2 Slave Nodes**
5. **Select an Amazon Machine Image (AMI)**
   - Choose an AMI, such as an Ubuntu Server 20.04 LTS (Free tier eligible).

– 
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

Amazon Machine Image (AMI)

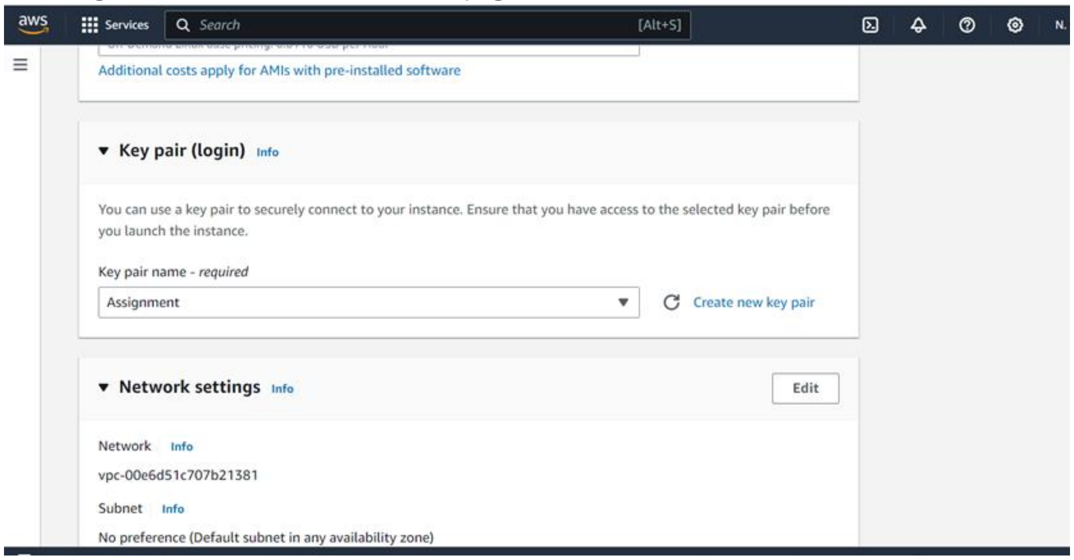
Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-04a81a99f5ec58529 (64-bit (x86)) / ami-0c14ff330901e49ff (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical

6. **Choose Instance Type**
   - Select an instance type (e.g., `t2.micro` for the free tier).
7. **Configure Instance Details**
   - Configure instance details according to your needs.
8. **Add Storage**
   - Specify storage size and type if needed.
9. **Add Tags**

- Add tags for easier identification (e.g., Name: Master, Name: Slave1, Name: Slave2).


Additional costs apply for AMIs with pre-installed software

Key pair (login) Info

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*
Assignment

Create new key pair

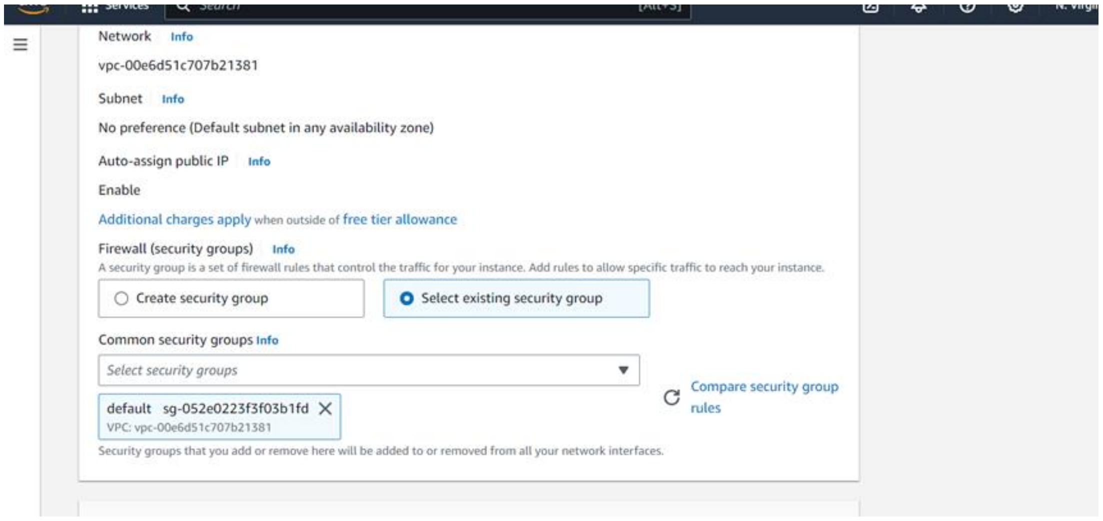
Network settings Info

Network Info
vpc-00e6d51c707b21381

Subnet Info
No preference (Default subnet in any availability zone)

10. **Configure Security Group**

- Configure security groups to allow SSH (port 22) and other necessary ports.


Network Info
vpc-00e6d51c707b21381

Subnet Info
No preference (Default subnet in any availability zone)

Auto-assign public IP Info
Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group [✓ Select existing security group]

Common security groups Info
Select security groups

default sg-052e0223f3f03b1fd
VPC: vpc-00e6d51c707b21381

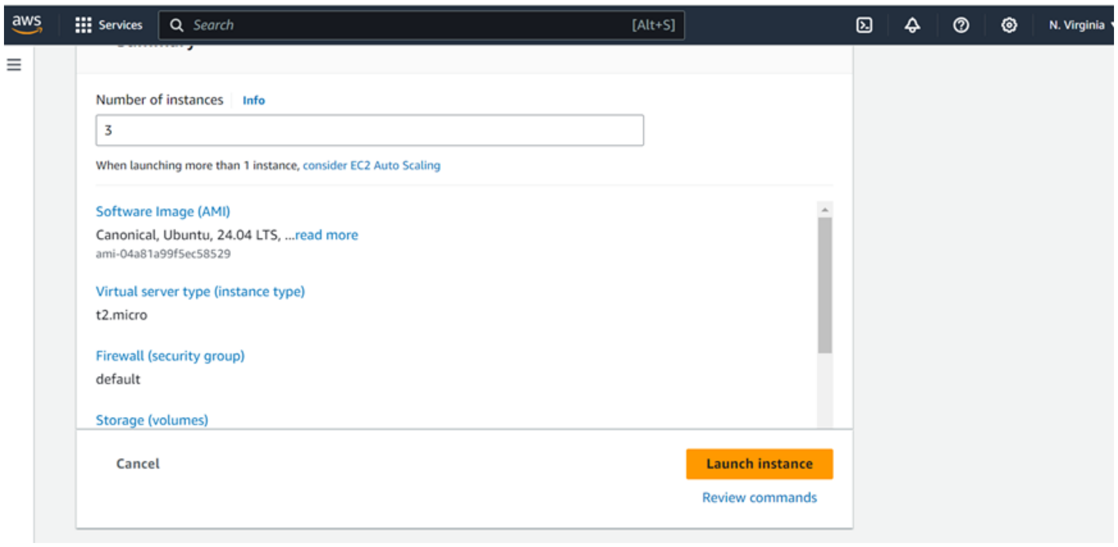
Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

-

11. **Review and Launch**

- Review your instance configuration and click **Launch**.



Number of instances Info

3

When launching more than 1 instance, consider EC2 Auto Scaling

Software Image (AMI)
Canonical, Ubuntu, 24.04 LTS, ...read more
ami-04a81a99f5ec58529

Virtual server type (instance type)
t2.micro

Firewall (security group)
default

Storage (volumes)

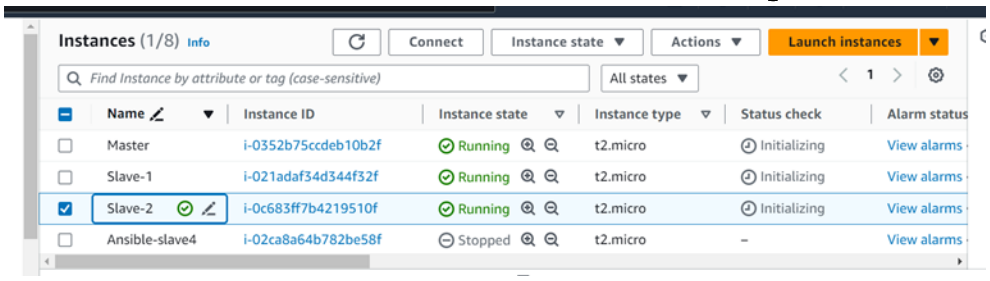
-

12. **Create a Key Pair**

- Create a new key pair, download it (e.g., `devopsstreamline-key.pem`), and launch your instances.

13. **Instance Initialization**

- Wait for the instances to initialize and reach the "running" status.



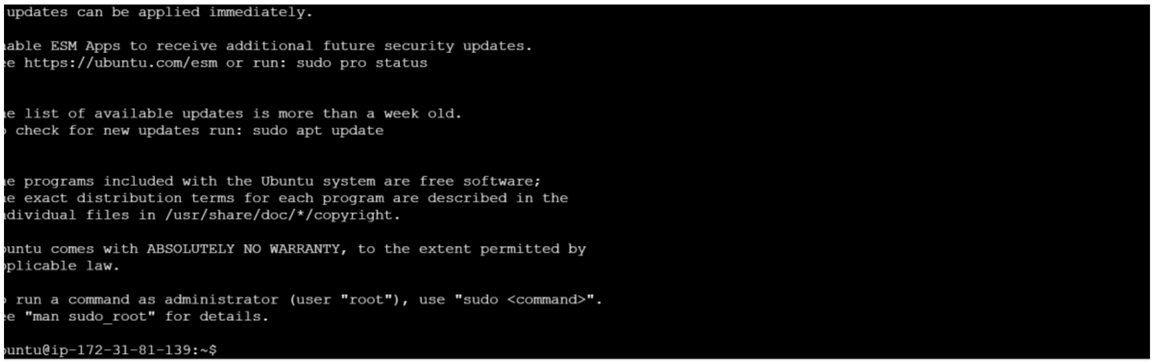
Name	Instance ID	Instance state	Instance type	Status check	Alarm status
Master	i-0352b75ccdeb10b2f	Running	t2.micro	Initializing	View alarms
Slave-1	i-021adaf34d344f32f	Running	t2.micro	Initializing	View alarms
Slave-2	i-0c683ff7b4219510f	Running	t2.micro	Initializing	View alarms
Ansible-slave4	i-02ca8a64b782be58f	Stopped	t2.micro	–	View alarms

-

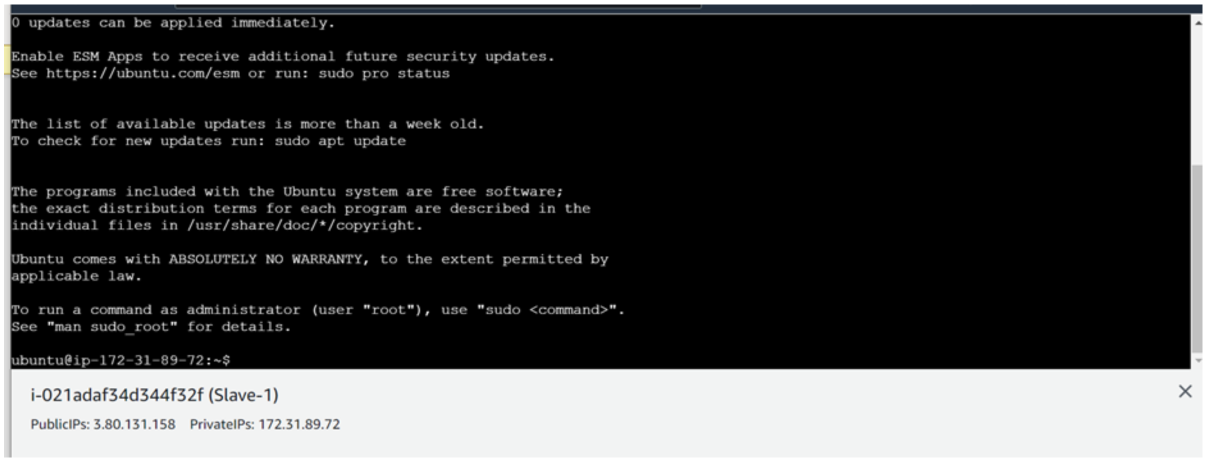
2. Configure Master and Slave Nodes

Connecting via PuTTY

1. **Install PuTTY**
   - Download and install PuTTY on your local system.
2. **Convert PEM to PPK**
   - Use PuTTYgen to convert your `.pem` key file to a `.ppk` format.
3. **Connect to EC2 Instances**
   - Open PuTTY and connect to your EC2 instances using their public IP addresses and the `.ppk` key.
   -



```
updates can be applied immediately.

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


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


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

untu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
plicable law.

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

untu@ip-172-31-81-139:~$
```



```
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-172-31-89-72:~$
```

i-021adaf34d344f32f (Slave-1)

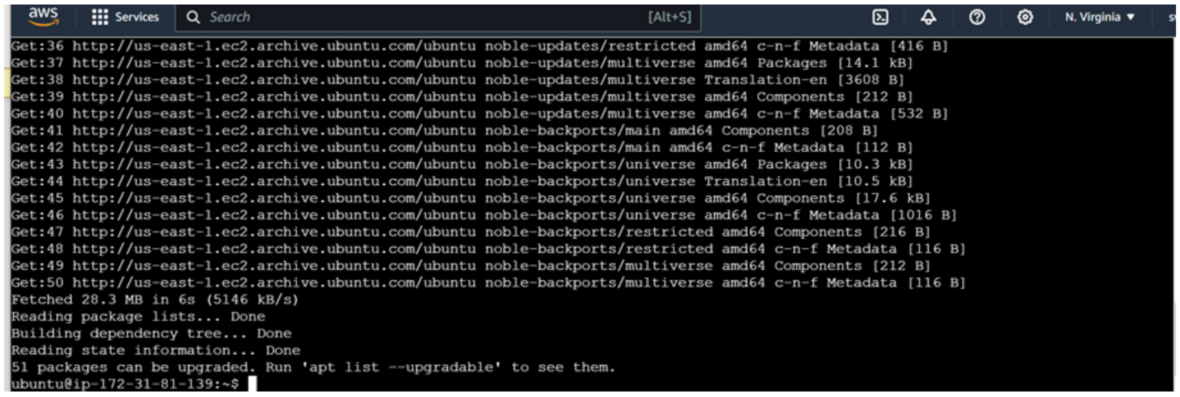
PublicIPs: 3.80.131.158 PrivateIPs: 172.31.89.72

3. Install Ansible and Setup SSH

Install Ansible on Master Node

1. **Update System Packages**

```
sudo apt update
```



```
Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [416 B]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [14.1 kB]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3608 B]
Get:39 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [212 B]
Get:40 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:41 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:42 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:43 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.3 kB]
Get:44 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.5 kB]
Get:45 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:46 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1016 B]
Get:47 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:48 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:49 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:50 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Fetched 28.3 MB in 6s (5146 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
51 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-81-139:~$
```

2. **Install Required Software**

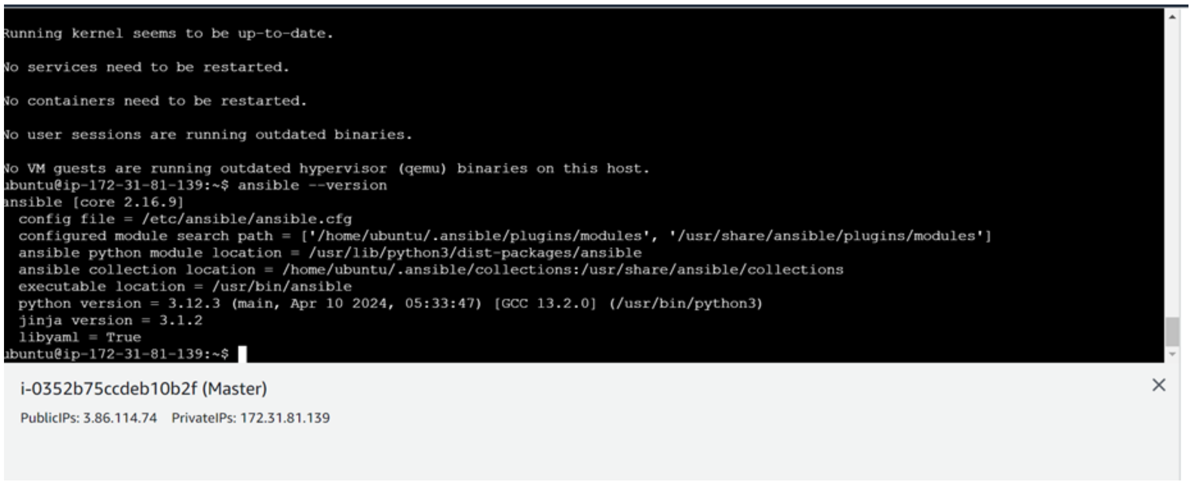
```
sudo apt install software-properties-common
```

3. **Add Ansible Repository**

```
sudo add-apt-repository --yes --update ppa:ansible/ansible
```

4. **Install Ansible**

```
sudo apt install ansible
```



```
Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-81-139:~$ ansible --version
ansible [core 2.16.9]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Apr 10 2024, 05:33:47) [GCC 13.2.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
ubuntu@ip-172-31-81-139:~$
```

i-0352b75ccdeb10b2f (Master)

PublicIPs: 3.86.114.74   PrivateIPs: 172.31.81.139

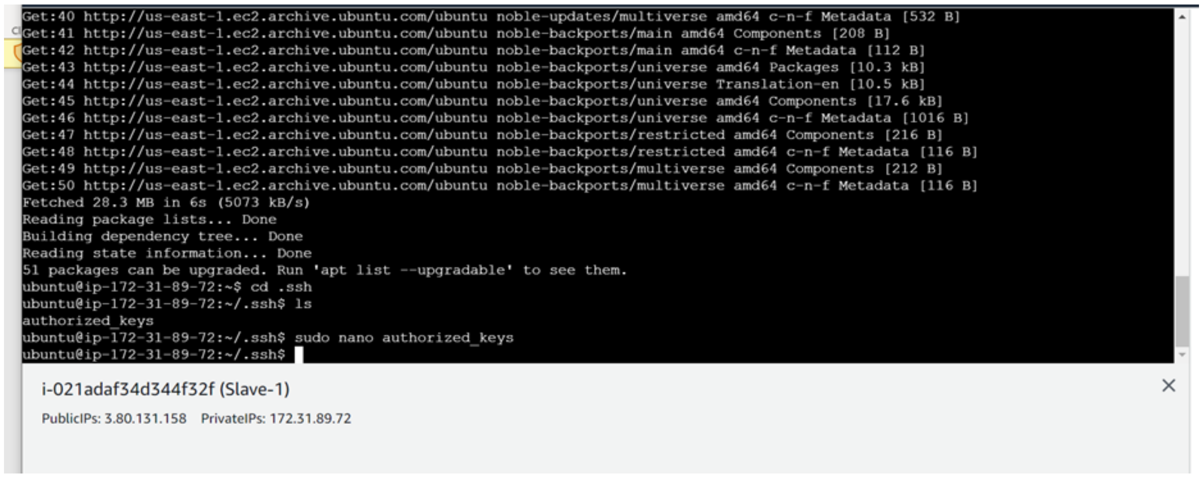
Setup Password-less SSH

1. **Generate SSH Key on Master Node**

```
cd ~/.ssh
ssh-keygen
cat id_rsa.pub
```

2. **Copy Public Key to Slave Nodes**

   – Copy the content of `id_rsa.pub` and paste it into the `authorized_keys` file on both slave nodes (`~/.ssh/authorized_keys`).



```
Get:40 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:41 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:42 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:43 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [10.3 kB]
Get:44 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [10.5 kB]
Get:45 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [17.6 kB]
Get:46 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1016 B]
Get:47 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:48 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:49 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:50 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Fetched 28.3 MB in 6s (5073 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
51 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-89-72:~$ cd .ssh
ubuntu@ip-172-31-89-72:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-89-72:~/.ssh$ sudo nano authorized_keys
ubuntu@ip-172-31-89-72:~/.ssh$
```

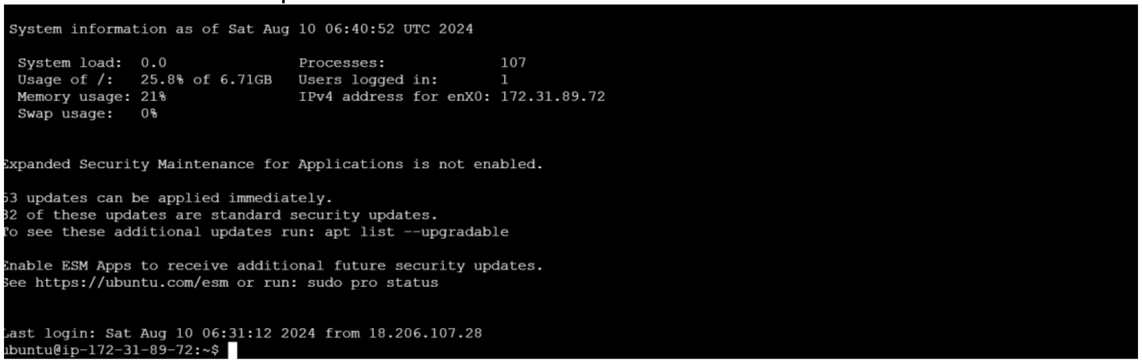
i-021adaf34d344f32f (Slave-1)

PublicIPs: 3.80.131.158   PrivateIPs: 172.31.89.72

3. **Verify SSH Connection**

   – Test the SSH connection from the master node to both slave nodes:

```
ssh <slave-node-ip>
```



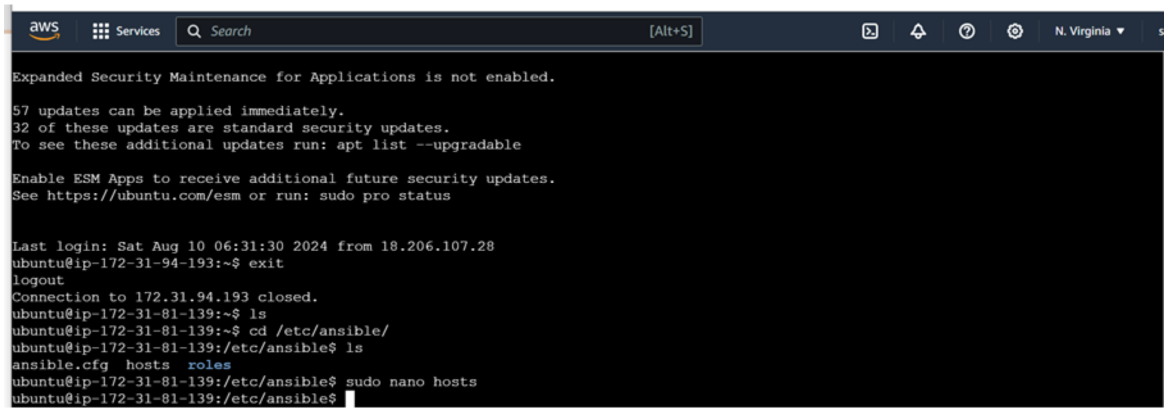
```
System information as of Sat Aug 10 06:40:52 UTC 2024

  System load:  0.0               Processes:             107
  Usage of /:   25.8% of 6.71GB   Users logged in:       1
  Memory usage: 21%               IPv4 address for enX0: 172.31.89.72
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

53 updates can be applied immediately.
32 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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


Last login: Sat Aug 10 06:31:12 2024 from 18.206.107.28
ubuntu@ip-172-31-89-72:~$
```

4. Create and Run Ansible Playbook

Create Ansible Inventory

1. **Edit the Inventory File**
   - Define the IP addresses of your slave nodes in the inventory file:

```
[test]
<slave1-ip>

[prod]
<slave2-ip>
```



```
Expanded Security Maintenance for Applications is not enabled.

57 updates can be applied immediately.
32 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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


Last login: Sat Aug 10 06:31:30 2024 from 18.206.107.28
ubuntu@ip-172-31-94-193:~$ exit
logout
Connection to 172.31.94.193 closed.
ubuntu@ip-172-31-81-139:~$ ls
ubuntu@ip-172-31-81-139:~$ cd /etc/ansible/
ubuntu@ip-172-31-81-139:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-81-139:/etc/ansible$ sudo nano hosts
ubuntu@ip-172-31-81-139:/etc/ansible$
```

Create Ansible Playbook

1. **Create Playbook File**
   - Create a file named `play.yml` with the following content:

```
---
- name: Task for Master
  hosts: localhost
  become: true
  tasks:
    - name: Executing script on master
      script: master.sh

- name: Task for Slave
  hosts: all
  become: true
  tasks:
    - name: Executing script on slave
      script: slave.sh
```

2. **Create Script Files**
   - **Master Script (`master.sh`):**

```
sudo apt update
sudo apt install openjdk-11-jdk -y
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]
https://pkg.jenkins.io/debian-stable binary/" | sudo tee
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

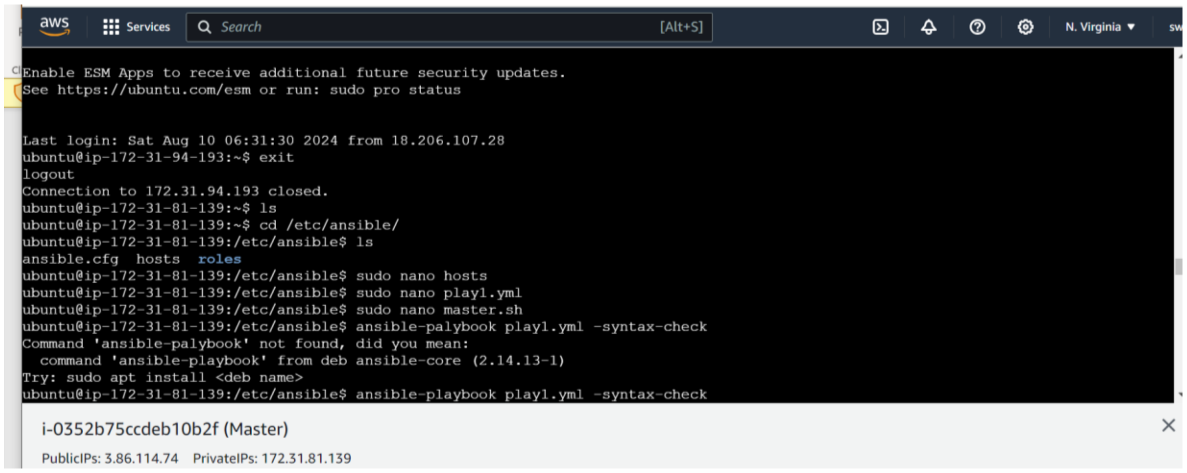
```
sudo apt-get update
sudo apt-get install jenkins -y
```

– **Slave Script (`slave.sh`):**

```
sudo apt update
sudo apt install openjdk-11-jdk -y
sudo apt install docker.io -y
```

3. **Check Playbook Syntax**

```
ansible-playbook play.yml --syntax-check
```

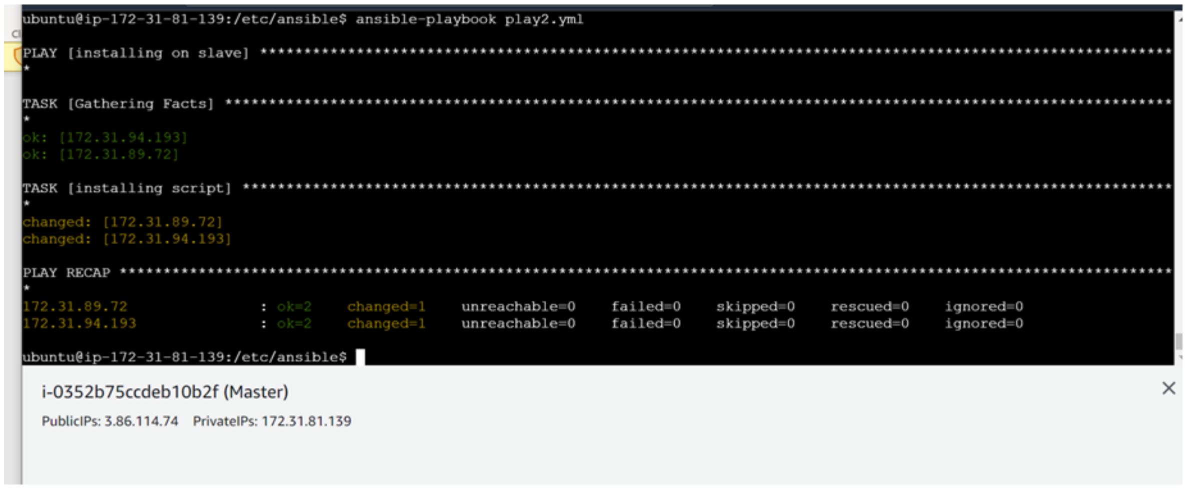


```
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Sat Aug 10 06:31:30 2024 from 18.206.107.28
ubuntu@ip-172-31-94-193:~$ exit
logout
Connection to 172.31.94.193 closed.
ubuntu@ip-172-31-81-139:~$ ls
ubuntu@ip-172-31-81-139:~$ cd /etc/ansible/
ubuntu@ip-172-31-81-139:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-81-139:/etc/ansible$ sudo nano hosts
ubuntu@ip-172-31-81-139:/etc/ansible$ sudo nano play1.yml
ubuntu@ip-172-31-81-139:/etc/ansible$ sudo nano master.sh
ubuntu@ip-172-31-81-139:/etc/ansible$ ansible-palybook play1.yml -syntax-check
Command 'ansible-palybook' not found, did you mean:
  command 'ansible-playbook' from deb ansible-core (2.14.13-1)
Try: sudo apt install <deb name>
ubuntu@ip-172-31-81-139:/etc/ansible$ ansible-playbook play1.yml -syntax-check
```

i-0352b75ccdeb10b2f (Master)
PublicIPs: 3.86.114.74   PrivateIPs: 172.31.81.139

4. **Run Playbook**

```
ansible-playbook play.yml
```



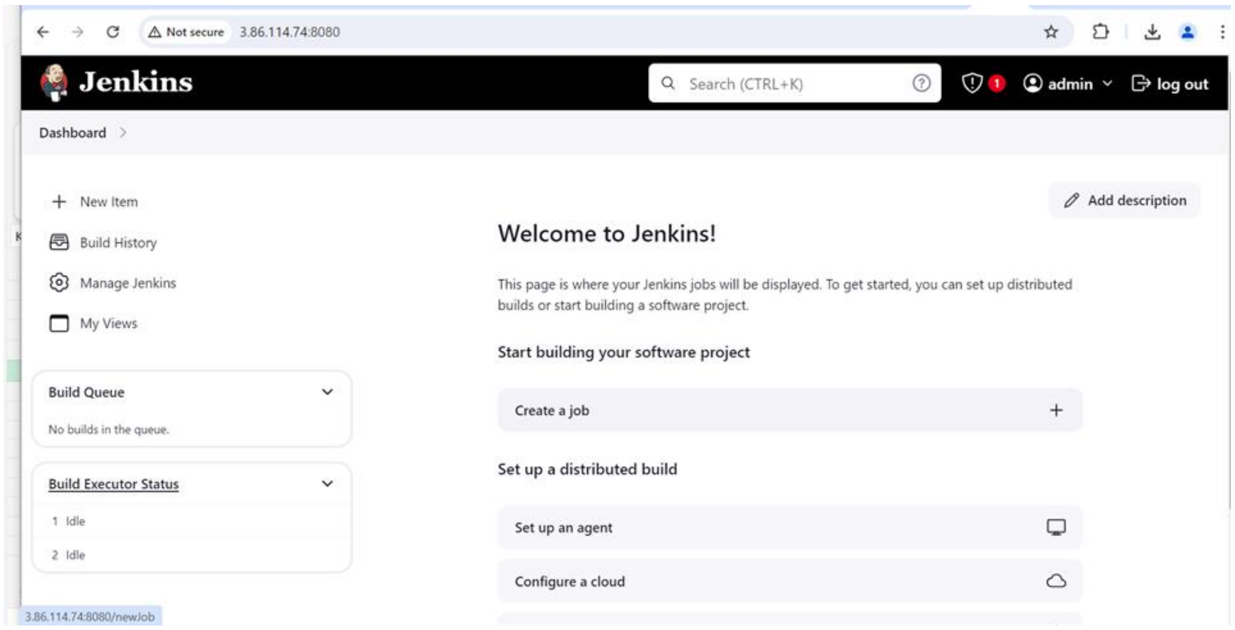
```
ubuntu@ip-172-31-81-139:/etc/ansible$ ansible-playbook play2.yml

PLAY [installing on slave] ******************************************************************************
*

TASK [Gathering Facts] **********************************************************************************
*
ok: [172.31.94.193]
ok: [172.31.89.72]

TASK [installing script] ********************************************************************************
*
changed: [172.31.89.72]
changed: [172.31.94.193]

PLAY RECAP **********************************************************************************************
*
172.31.89.72               : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.94.193              : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-172-31-81-139:/etc/ansible$
```

i-0352b75ccdeb10b2f (Master)
PublicIPs: 3.86.114.74   PrivateIPs: 172.31.81.139

5. Set Up Jenkins and Docker

Access Jenkins Dashboard

1. **Access Jenkins**
   – Open a browser and navigate to `http://<master-ip>:8080`.



Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Set up a distributed build

No builds in the queue.

1 Idle

2 Idle

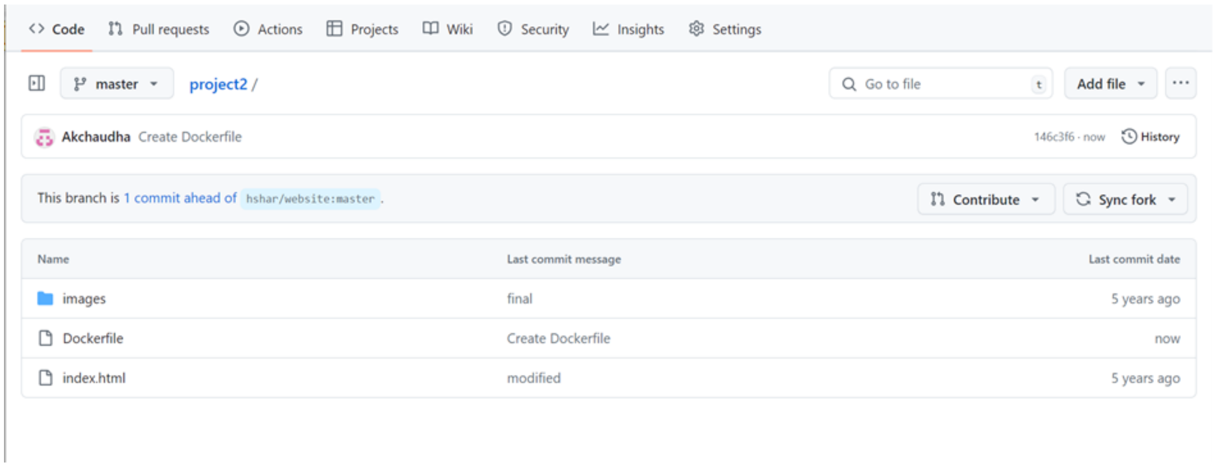
2. **Set Up Jenkins**
   - Follow the setup wizard to complete Jenkins installation and configuration.

Create a Dockerfile on GitHub

1. **Create Dockerfile**
   - Create a `Dockerfile` with the following content:

```
FROM ubuntu
RUN apt update
RUN apt install apache2 -y
ADD . /var/www/html
ENTRYPOINT apachectl -D FOREGROUND
```

2. **Commit Dockerfile to GitHub**
   - Push the `Dockerfile` to your GitHub repository.



Akchaudha Create Dockerfile 146c3f6 · now

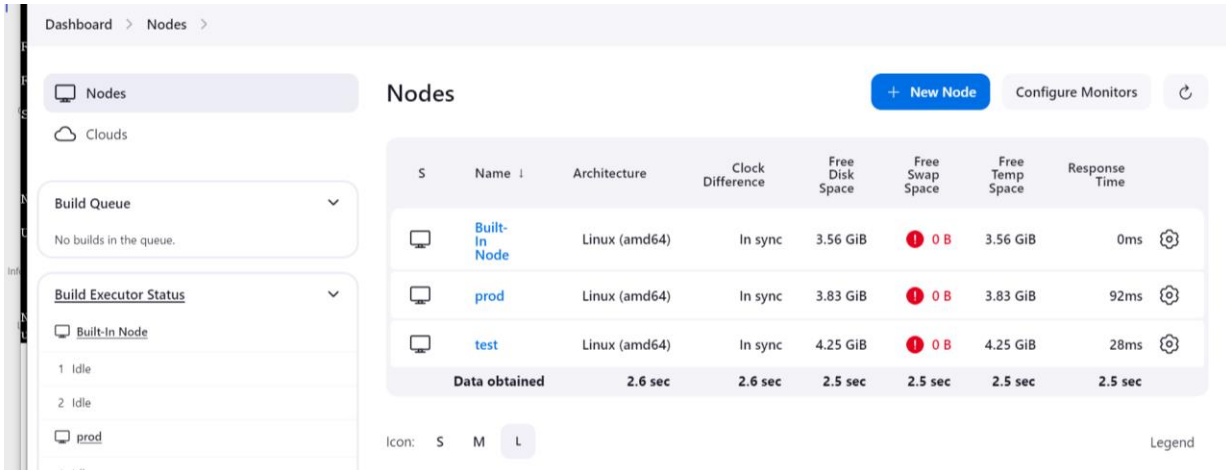
This branch is 1 commit ahead of hshar/website:master .

Name	Last commit message	Last commit date
images	final	5 years ago
Dockerfile	Create Dockerfile	now
index.html	modified	5 years ago

6. Create Jenkins Jobs

Job Configuration

1. **Create Jenkins Nodes**
   - Create nodes named `test` and `prod`:
     - **Test Node:** Slave1, Root directory: `/tmp`, Launch method: Launch agent via SSH.
     - **Prod Node:** Slave2, similar settings as the test node.



Dashboard > Nodes >

Nodes

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (amd64)	In sync	3.56 GiB	0 B	3.56 GiB	0ms
	prod	Linux (amd64)	In sync	3.83 GiB	0 B	3.83 GiB	92ms
	test	Linux (amd64)	In sync	4.25 GiB	0 B	4.25 GiB	28ms
	Data obtained	2.6 sec	2.6 sec	2.5 sec	2.5 sec	2.5 sec	2.5 sec

Build Queue

No builds in the queue.

Build Executor Status

Built-In Node

1 Idle

2 Idle

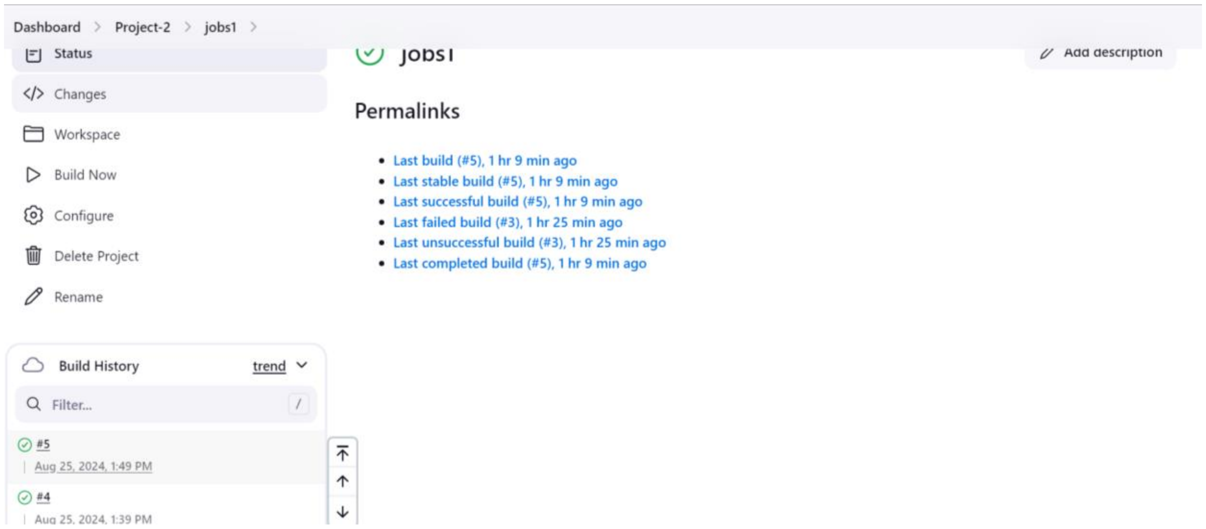
prod

2. **Create Jobs**
   - **Job1:** Build on `test` node using the `develop` branch.
   - **Job2:** Build on `test` node using the master `branch.`
   - **Job3**`: Build on` prod `node using the` master branch.

Job Build Steps

1. **Job1 Build Steps**
   - Restrict where the project can run: `test node`.
   - Source Code Management: GIT, branch `develop`.
   - Build Steps:

```
sudo docker rm -f c1
sudo docker build /home/ubuntu/jenkins/workspace/Job1 -t job1
sudo docker run -itd -p 80:80 --name=c1 job1
```



Dashboard > Project-2 > jobs1 >

jobs1

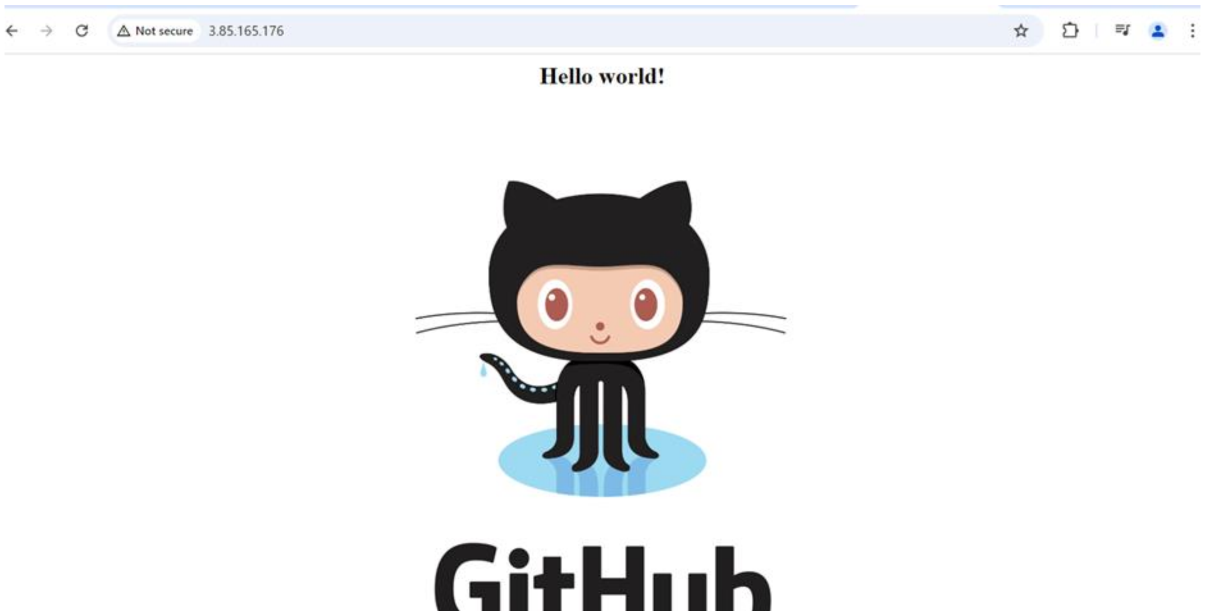
Permalinks

- Last build (#5), 1 hr 9 min ago
- Last stable build (#5), 1 hr 9 min ago
- Last successful build (#5), 1 hr 9 min ago
- Last failed build (#3), 1 hr 25 min ago
- Last unsuccessful build (#3), 1 hr 25 min ago
- Last completed build (#5), 1 hr 9 min ago

Build History

#5 Aug 25, 2024, 1:49 PM

#4 Aug 25, 2024, 1:39 PM

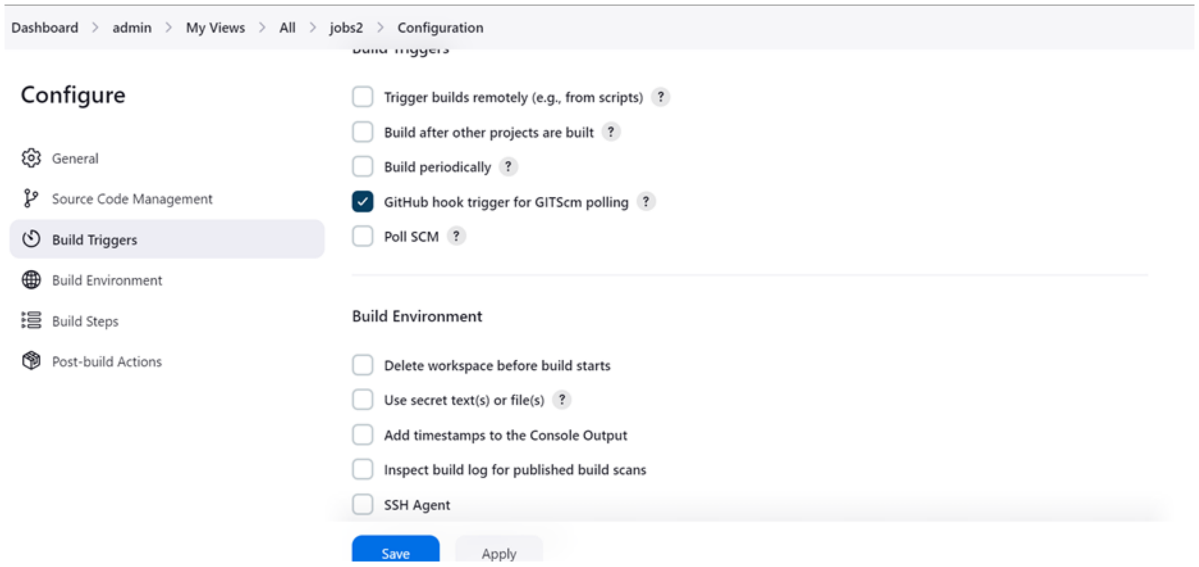


2. **Job2 Build Steps**
   - Similar to Job1, but use `master` branch and different Docker container (`c2`).
3. **Job3 Build Steps**
   - Configure to run on the `prod` node with appropriate build steps.

7. Configure Webhook and Pipeline

Set Up Webhook on GitHub

1. **Add Webhook**
   - In your GitHub repository settings, add a new webhook pointing to your Jenkins URL (e.g., `http://<jenkins-ip>:8080/github-webhook/`).



Dashboard > admin > My Views > All > jobs2 > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Build Triggers

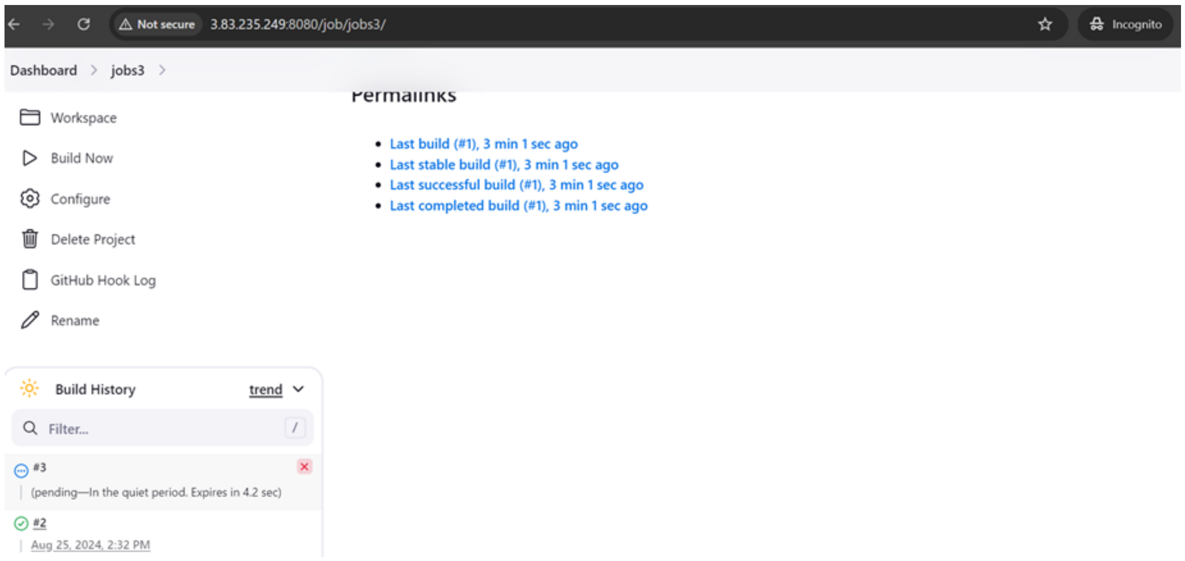
- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build after other projects are built
- [ ] Build periodically
- [x] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Build Environment

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Add timestamps to the Console Output
- [ ] Inspect build log for published build scans
- [ ] SSH Agent

Save Apply

   -
2. **Trigger Builds**
   - Push changes to GitHub to automatically trigger Jenkins jobs.



3.83.235.249:8080/job/jobs3/

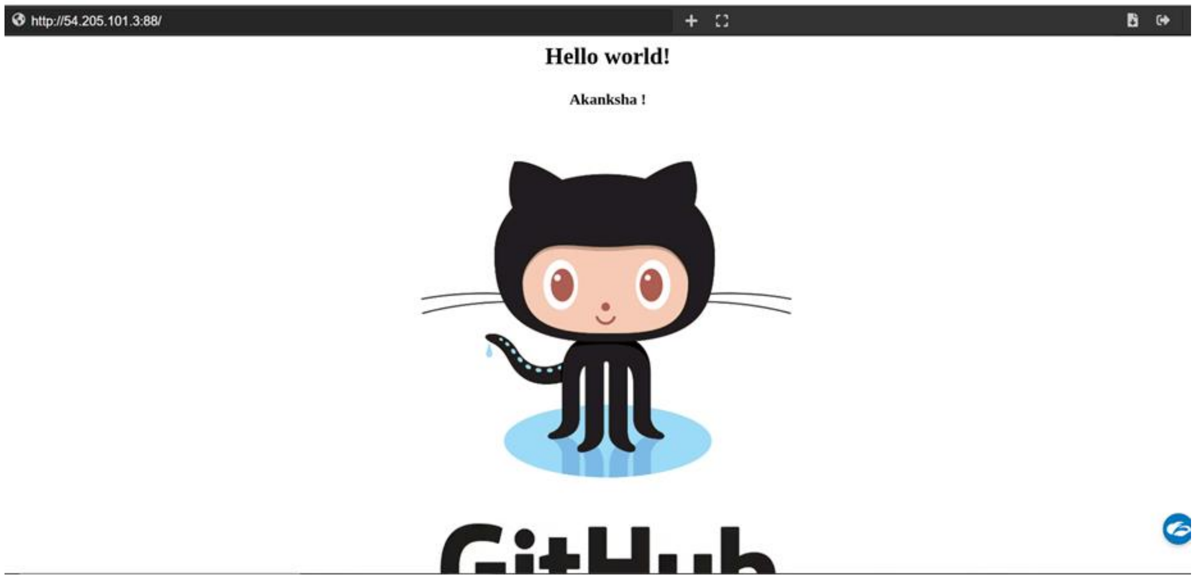
Dashboard > jobs3 >

Permalinks

- Last build (#1), 3 min 1 sec ago
- Last stable build (#1), 3 min 1 sec ago
- Last successful build (#1), 3 min 1 sec ago
- Last completed build (#1), 3 min 1 sec ago

#3
(pending—In the quiet period. Expires in 4.2 sec)

#2
Aug 25, 2024, 2:32 PM



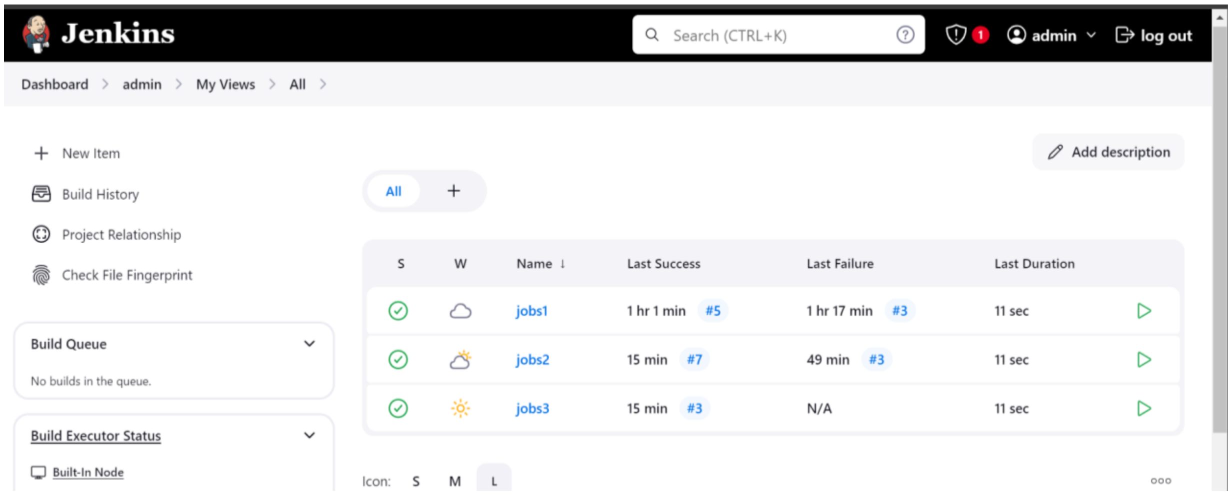
http://54.205.101.3:88/

Hello world!

Akanksha !

Create Jenkins Pipeline

1. **Pipeline Setup**
   – Use Jenkins Pipeline plugin to define and automate your build processes.
2. **Monitor Builds**
   – Monitor the Jenkins console output and ensure successful builds and deployments.



S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☁	jobs1	1 hr 1 min #5	1 hr 17 min #3	11 sec	▷
✓	⛅	jobs2	15 min #7	49 min #3	11 sec	▷
✓	☀	jobs3	15 min #3	N/A	11 sec	▷

By following these steps, you've successfully set up a DevOps pipeline using AWS EC2, Ansible, Jenkins, Docker, and GitHub. This pipeline will help streamline your development and deployment processes, enabling continuous integration and delivery.